MANUEL DE L'UTILISATEUR DE CRIMSON 2

MODULAR CONTROLLER

AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

Rédacteurs : Mike Granby et Jesse Benefiel.

TABLE DES MATIERES

TABLE DES MATIERES I
INTRODUCTION 1
CONFIGURATION SYSTEME REQUISE 1
INSTALLATION DU LOGICIEL 1
VERIFICATION DES MISES A JOUR 1
INSTALLATION DES PILOTES USB 2
DEMARRAGE RAPIDE 3
PRESENTATION 3
MODULES 3
COMMUNICATIONS 4
MAPPING DES DONNEES 5
TELECHARGEMENT 8
CONNEXION DU PERIPHERIQUE ET DU CONTROLEUR MODULAIRE 8
NOTIONS DE BASE SUR CRIMSON 9
ICONES DE L'ECRAN PRINCIPAL 9
MODULES 9
COMMUNICATIONS 9
ETIQUETTES DE DONNEES 10
INTERFACE UTILISATEUR 10
PROGRAMMATION 10
HISTORIQUE DES DONNEES 10
SERVEUR WEB 11
GESTIONNAIRE DE SECURITE 11
SELECTION DU CONTROLEUR MODULAIRE 11
UTILISATION DES INFO-BULLES 12
TRAVAILLER AVEC LES BASES DE DONNEES 12
TELECHARGEMENT VERS LE MAITRE 12
CONFIGURATION DU LIEN 13
VERIFICATION DU LIEN USB 13
DEFINITION DE L'ADRESSE IP 13
ENVOI DE LA BASE DE DONNEES 14
EXTRACTION DES BASES DE DONNEES 14
MONTAGE DE LA CARTE COMPACTFLASH 15
FORMATAGE DE LA CARTE COMPACTFLASH 16
ENVOI DE L'HEURE ET DE LA DATE 16
TELECHARGEMENT DEPUIS LA CARTE COMPACTFLASH 16
CONFIGURATION DES MODULES 18
MODULES 18
AJOUT DE MODULES 18
DEPLACEMENT DE MODULES 18
MODIFICATION DE MODULES 19
CSPID – PROGRAMMATION DU MODULE PID 19
L'ONGLET GENERAL 19

L'ONGLET COMMANDE 21
L'ONGLET PUISSANCE 24
L'ONGLET ALARMES 25
L'ONGLET SORTIES 29
AUTOTUNE 30
DONNEES DISPONIBLES 32
CSSG – PROGRAMMATION DU MODULE PID A ENTREE DE CAPTEUR D'EFFORT 36
L'ONGLET GENERAL 36
L'ONGLET COMMANDE 39
L'ONGLET PUISSANCE 41
L'ONGLET ALARMES 42
L'ONGLET SORTIES 46
AUTOTUNE 47
DONNEES DISPONIBLES 49
CSTC/CSRTD – PROGRAMMATION DU MODULE D'ENTREE DE LA TEMPERATURE 55
L'ONGLET CONFIGURATION 55
DONNEES DISPONIBLES 56
CSINI/CSINV – PROGRAMMATION DU MODULE A ENTREE ANALOGIQUE 57
L'ONGLET CONFIGURATION 57
L'ONGLET LINEARISATION 58
DONNEES DISPONIBLES 59
CSDIO – PROGRAMMATION DU MODULE D'ENTREE-SORTIE NUMERIQUE 60
L'ONGLET CONFIGURATION 60
L'ONGLET EDITEUR LOGIQUE 61
DONNEES DISPONIBLES 63
CONFIGURATION DES COMMUNICATIONS 64
UTILISATION DES PORTS SERIE 64
SELECTION D'UN PROTOCOLE 64
OPTIONS DU PROTOCOLE 65
PERIPHERIQUES 66
CONFIGURATION DU PORT ETHERNET 66
PARAMETRES IP 67
ROUTAGE IP 67
COUCHE PHYSIQUE 67
MISE A JOUR A DISTANCE 67
SÉLECTION DU PROTOCOLE 67
MAPPING DES DONNÉES 69
MAPPING D'ELEMENTS SUR UN BLOC 69
ACCES AUX BITS INDIVIDUELS 70
LECTURE/ECRITURE DE VARIABLES 70
CONVERSION DE PROTOCOLE 71
MAITRE ET ESCLAVE 71
MAITRE ET MAITRE 72
QUELLE SOLUTION ? 72
TRANSFORMATION DES DONNEES 72
COMMUNICATIONS AVANCEES 73
UTILISATION DES CARTES D'EXTENSION 73

PARTAGE DU PORT SERIE 74
ACTIVATION DU PROTOCOLE TCP/IP 74
PARTAGE DU PORT 74
CONNEXION VIA UN AUTRE PORT 75
CONNEXION VIA ETHERNET 75
PORTS VIRTUELS PURS 77
LIMITATIONS 77
UTILISATION DE LA MESSAGERIE ELECTRONIQUE 78
CONFIGURATION DU PROTOCOLE SMTP 78
CONFIGURATION DU PROTOCOLE SMS 79
LE CARNET D'ADRESSES 81
UTILISATION DE LA GESTION DU TEMPS 81
CONFIGURATION DU GESTIONNAIRE DE L'HEURE 82
SELECTION D'UN SERVEUR SNTP 83
CONFIGURATION DES FUSEAUX HORAIRES 84
MODEMS 84
QUELQUES EXEMPLES D'APPLICATIONS 84
AJOUT D'UNE CONNEXION D'APPEL ENTRANT 85
AJOUT D'UNE CONNEXION D'APPEL SORTANT 88
AJOUT D'UNE CONNEXION SMS 90
TRAITEMENT DES MESSAGES SMS 90
UTILISATION DE PLUSIEURS INTERFACES 90
VERIFICATION DU STATUT DU MODEM 92
CONFIGURATION DES ETIQUETTES DE DONNEES 94
TOUT SUR LES ETIQUETTES 94
TYPES D'ETIQUETTES 94
POURQUOI UTILISER DES ETIQUETTES ? 96
CREATION D'ETIQUETTES 97
MODIFICATION D'ETIQUETTES 97
MODIFICATION DE PROPRIETES 98
PROPRIETES EXPRESSION 98
GESTION DES LANGUES 99
PROPRIETES COULEUR* 100
MODIFICATION D'ETIQUETTES DE TYPE BIT 101
L'ONGLET DONNEES (VARIABLES) 101
L'ONGLET DONNEES (FORMULES) 102
L'ONGLET DONNEES (TABLEAUX) 103
L'ONGLET FORMAT 104
L'ONGLET COULEURS* 105
L'ONGLET ALARMES 105
L'ONGLET TRIGGERS 107
MODIFICATION D'ETIQUETTES D'ENTIER 107
L'ONGLET DONNEES (VARIABLES) 108
L'ONGLET DONNEES (FORMULES) 109
L'ONGLET DONNEES (TABLEAUX) 110
L'ONGLET FORMAT 111
L'ONGLET COULEURS* 112
LES ONGLETS ALARMES 113

L'ONGLET TRIGGERS 114
MODIFICATION D'ETIQUETTES MULTI 114
L'ONGLET DONNEES (VARIABLES) 115
L'ONGLET DONNEES (FORMULES) 115
L'ONGLET DONNEES (TABLEAUX) 116
L'ONGLET FORMAT 116
L'ONGLET COULEURS* 118
LES ONGLETS ALARMES 118
L'ONGLET TRIGGERS 119
MODIFICATION D'ETIQUETTES DE REELS 119
MODIFICATION D'ETIQUETTES DE CHAINES 120
L'ONGLET DONNEES (VARIABLES) 120
L'ONGLET DONNEES (FORMULES) 121
L'ONGLET DONNEES (TABLEAUX) 121
L'ONGLET FORMAT 122
L'ONGLET COULEURS* 123
PLUS DE DEUX ALARMES 124
VALIDATION D'ETIQUETTES 124
EXPORTATION DU MAPPING DES ETIQUETTES 124
ENREGISTREMENT DE MESSAGES D'EVENEMENTS* 124
REMARQUES POUR LES UTILISATEURS D'EDICT 125
CONFIGURATION DE L'IHM VIRTUEL OU NON MONOCHROME 126
CONTROLE DE L'AFFICHAGE 126
AUTRES OPTIONS D'AFFICHAGE 126
UTILISATION DE LA LISTE DES PAGES 127
AFFICHAGE DES BOITES A OUTILS DE L'EDITEUR 127
LA BOITE A OUTILS DE DESSIN 127
LA BOITE A OUTILS DU FORMAT DE REMPLISSAGE 127
LA BOITE A OUTILS DU FORMAT DE LIGNE 128
LA BOITE A OUTILS DU FORMAT DE TEXTE 128
LA BOITE A OUTILS DU PREMIER PLAN 128
LA BOITE A OUTILS DE L'ARRIERE-PLAN 128
AJOUT DE PRIMITIVES D'AFFICHAGE 129
ALIGNEMENT INTELLIGENT 129
OPTIONS DU CLAVIER 130
MODE D'INSERTION VERROUILLEE 130
SELECTION DE PRIMITIVES 130
DEPLACEMENT ET REDIMENSIONNEMENT 130
ORGANISATION DES PRIMITIVES 131
MODIFICATION DES PRIMITIVES 132
DESCRIPTIONS DE PRIMITIVES 132
LA PRIMITIVE LIGNE 132
LES PRIMITIVES GEOMETRIQUES SIMPLES 132
LES PRIMITIVES TYPE RESERVOIR 133
LES PRIMITIVES BARRES GRAPH SIMPLES 133
LA PRIMITIVE TEXTE FIXE 134
LA PRIMITIVE ETIQUETTE 135

AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

LES PRIMITIVES TEXTE ETIQUETTE 136
MODIFICATION DE L'ETIQUETTE 138
LA PRIMITIVE HEURE ET DATE 139
LES PRIMITIVES BARRES GRAPH RICHES 141
LES PRIMITIVES SYSTEME 143
DEFINITION DES PROPRIETES DE LA PAGE 143
DEFINITION DES ACTIONS DU SYSTEME 144
DEFINITION DU COMPORTEMENT DES TOUCHES 144
CONDITION D'ACTIONS 145
DESCRIPTIONS D'ACTIONS 146
L'ACTION ALLER PAGE 146
L'ACTION BOUTON BISTABLE POUSSOIR ETC 147
L'ACTION MODIFIER VALEUR D'UN ENTIER 148
L'ACTION MODIFICATION VALEUR ENTIER SUIVANT RAMPE 148
L'ACTION JOUER SONNERIE 149
L'ACTION DEFINIE PAR L'UTILISATEUR 149
BLOQUER ACTION PAR DEFAUT 150
MODIFICATION DE LA LANGUE 150
RUBRIQUES AVANCEES 150
TRAITEMENT DES ACTIONS 151
DISPONIBILITE DES DONNEES 151
CONFIGURATION D'UN IHM VIRTUEL COULEUR 153
CONTROLE DE L'AFFICHAGE 153
AUTRES OPTIONS D'AFFICHAGE 153
UTILISATION DE LA LISTE DES PAGES 154
GRILLE 154
LA BOITE A OUTILS DE DESSIN 155
AJOUT DE PRIMITIVES D'AFFICHAGE 155
ALIGNEMENT INTELLIGENT 155
OPTIONS DU CLAVIER 156
MODE D'INSERTION VERROUILLEE 157
UTILISATION DE LA BIBLIOTHEQUE D'IMAGES 157
SELECTION DE PRIMITIVES 157
DEPLACEMENT ET REDIMENSIONNEMENT 158
ALIGNEMENT DES PRIMITIVES 158
PRIMITIVES D'ESPACEMENT 159
REORGANISATION DE PRIMITIVES 159
GROUPEMENT DE PRIMITIVES 160
MODIFICATION DE PRIMITIVES 160
DEFINITION DE COULEURS 160
DEFINITION DEMOTIFS DE REMPLISSAGE 161
DEFINITION DES ACTIONS 161
CONDITION DES ACTIONS 162
DESCRIPTIONS DES ACTIONS 162
L'ACTION ALLER PAGE 163

L'ACTION BOUTON BISTABLE POUSSOIR 164
L'ACTION MODIFIER VALEUR D'UN ENTIER 165
L'ACTION MODIFICATION VALEUR ENTIER SUIVANT RAMPE 165
L'ACTION JOUER SONNERIE 166
L'ACTION DEFINIE PAR L'UTILISATEUR 166
UTILISATION DE PARAMETRES PAR DEFAUT 167
DESCRIPTIONS DE PRIMITIVES 167
LA PRIMITIVE LIGNE 167
LES PRIMITIVES GEOMETRIQUE SIMPLES 167
LES PRIMITIVES DE RESERVOIR 168
LES PRIMITIVES BARRES GRAPH SIMPLES 168
LES PRIMITIVES GRAPHIQUE A BARRES 169
LA PRIMITIVE GRAPHIQUE A POINTS 170
LES PRIMITIVES ECHELLE 172
LA PRIMITIVE TEXTE FIXE 173
LA PRIMITIVE ETIQUETTE 175
LES PRIMITIVES TEXTE ETIQUETTE 175
MODIFICATION DE L'ETIQUETTE ASSOCIEE 179
LES PRIMITIVES TEXTE MULTI-LIGNE 180
LA PRIMITIVE HEURE ET DATE 180
LES PRIMITIVES BARRES GRAPH RICHES 182
LES PRIMITIVES GLISSEUR RICHES 185
LA PRIMITIVE AFFICHEUR D'ALARME 187
LA PRIMITIVE BANDEAU D'ALARMES 189
LA PRIMITIVE AFFICHEUR D'EVENEMENTS 190
LES PRIMITIVES HISTORIQUE/COURBES DE DONNEES 190
LA PRIMITIVE BOUTON GENERAL 192
LA PRIMITIVE BOUTON A BIT 194
LES PRIMITIVES SELECTEUR 196
LA PRIMITIVE IMAGE 198
LES PRIMITIVES CADRAN 201
DEFINITION DES PROPRIETES DE LA PAGE 204
DEFINITION DES ACTIONS DU SYSTEME 206
AUTRES PROPRIETES DU SYSTEME 206
SELECTION DE LANGUES 207
MODIFICATION DE LA LANGUE 207
DEFINITION DU COMPORTEMENT DE LA TOUCHE 208
BLOCAGE DES ACTIONS PAR DEFAUT 209
DISPONIBILITE DES DONNEES 209
REMARQUES POUR LES UTILISATEURS D'EDICT 209
CONFIGURATION DE PROGRAMMES 210
UTILISATION DE LA LISTE DES PROGRAMMES 210
MODIFICATION DE PROGRAMMES 211
PROPRIETES DU PROGRAMME 211
AJOUT DE COMMENTAIRES 213
RETOUR DE VALEURS 213
ATTENTION ! 213

AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

PASSAGE D'ARGUMENTS....214
CONSEILS DE PROGRAMMATION....214
PLUSIEURS ACTIONS....214
INSTRUCTIONS IF....215
INSTRUCTIONS SWITCH....215
VARIABLES LOCALES....216
CONSTRUCTIONS DE BOUCLES....217
REMARQUES POUR LES UTILISATEURS D'EDICT....219
CONFIGURATION DE L'ENREGISTREMENT DE DONNEES....220
CREATION D'UN HISTORIQUES DE DONNEES....220
UTILISATION DE LA LISTE DES HISTORIQUES....220
PROPRIETES DE L'HISTORIQUE DE DONNEES....221
STOCKAGE DES FICHIERS JOURNAUX....222
LE PROCESSUS D'ENREGISTREMENT....222
ACCES AUX FICHIERS JOURNAUX....223
UTILISATION DE WEBSYNC....223
SYNTAXE DE WEBSYNC....224
SWITCHES FACULTATIFS....224
EXEMPLE....224
REMARQUES POUR LES UTILISATEURS D'EDICT....225
CONFIGURATION DU SERVEUR WEB....226
PROPRIETES DU SERVEUR WEB....226
AJOUT DE PAGES WEB....227
UTILISATION D'UN SITE WEB PERSONNALISE....228
CREATION DU SITE....228
INCORPORATION DE DONNEES....228
DEPLOIEMENT DU SITE....228
ACCES A LA CARTE COMPACTFLASH....229
EXEMPLES DU SERVEUR WEB....229
UTILISATION DU SYSTEME DE SECURITE....233
CONCEPTS DE BASE DE LA SECURITE....233
SECURITE BASEE SUR L'OBJET....233
UTILISATEURS NOMMES....233
DROITS D'UTILISATEUR....234
CONTROLE DE L'ACCES....234
JOURNALISATION D'ECRITURES....235
ACCES PAR DEFAUT....235
CONNEXION A LA DEMANDE....235
ACCES A LA MAINTENANCE....235
PARAMETRES DE SECURITE....236
CREATION DES UTILISATEURS....237
SPECIFICATION DE LA SECURITE DES ETIQUETTES....238
SPECIFICATION DE LA SECURITE DES PAGES....238
LA PRIMITIVE GESTIONNAIRE DE SECURITE....238
FONCTIONS RELATIVES A LA SECURITE....238

ECRITURE D'EXPRESSIONS 239
VALEURS DE DONNEES 239
CONSTANTES 239
VALEURS D'ETIQUETTES 241
REFERENCES DES COMMUNICATIONS 241
MATHEMATIQUES SIMPLES 241
PRIORITE DES OPERATEURS 242
CONVERSION DE TYPE 242
COMPARAISON DES VALEURS 242
TEST DES BITS 243
CONDITIONS MULTIPLES 244
CHOIX DES VALEURS 244
MANIPULATION DE BITS 244
ET (AND), OU (OR), OU EXCLUSIF (XOR) 245
OPERATEURS DE DEPLACEMENT 245
NON DE MANIPULATION DE BITS 245
INDEXATION DE TABLEAUX 246
INDEXATION DE CHAINES 246
AJOUT DE CHAINES 246
APPEL DE PROGRAMMES 246
UTILISATION DE FONCTIONS 246
SYNTHESE DES PRIORITES 246
REMARQUES POUR LES UTILISATEURS D'EDICT 247
ECRITURES D'ACTIONS 248
MODIFICATION DE LA PAGE 248
MODIFICATION DE VALEURS NUMERIQUES 248
ATTRIBUTION SIMPLE 248
ATTRIBUTION COMPOSEE 248
INCREMENTATION ET DECREMENTATION 248
MODIFICATION DES VALEURS DE BITS 249
EXECUTION DE PROGRAMMES 249
UTILISATION DE FONCTIONS 249
PRIORITE DES OPERATEURS 249
REMARQUES POUR LES UTILISATEURS D'EDICT 249
UTILISATION DES PORTS BRUTS 250
CONFIGURATION D'UN PORT SERIE 250
CONFIGURATION D'UN SOCKET TCP/IP 251
LECTURE DE CARACTERES 251
LECTURE DE TRAMES COMPLETES 252
ENVOI DE DONNEES 252
REMARQUES POUR LES UTILISATEURS D'EDICT 252
REFERENCE DES VARIABLES SYSTEME 255
UTILISATION DES VARIABLES SYSTEME 255

AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

ACTIVEALARMS....256
COMMSERROR....257
DISPBRIGHTNESS....258
DISPCONTRAST....259
DISPCOUNT....260
DISPUPDATES....261
PI....262
REFERENCE DES FONCTIONS....263
REMARQUES POUR LES UTILISATEURS D'EDICT....263
ABS(*VALEUR*)....264
ACOS(*VALEUR*)....265
ASIN(*VALEUR*)....266
ATAN(*VALEUR*)....267
ATAN2(*A, B*)....268
BEEP(*FRÉQUENCE*, *PÉRIODE*)....269
CLEAREVENTS()....270
CLOSEFILE(*FICHIER*)....271
COMPACTFLASHEJECT()....272
COMPACTFLASHSTATUS()....273
CONTROLDEVICE(*PÉRIPHERIQUE,ACTIVER*)....274
COPY(*DESTINATION*, *SOURCE*, *NOMBRE*)....275
COS(*THÊTA*)....276
CREATEDIRECTORY(*NOM*)....277
CREATEFILE(*NOM*)....278
DATATOTEXT(*DONNEES, LIMITE*)....279
DATE(*A, M, J*)....280
DECTOTEXT(*DONNEES, SIGNE, AVANT, APRES, NON SIGNIFICATIF, GROUPE*)....281
DEG2RAD(*THÊTA*)....282
DELETEDIRECTORY(*NOM*)....283
DELETEFILE(*FICHIER*)....284
DEVCTRL(*PÉRIPHERIQUE, FONCTION, DONNEES*)....285
DISABLEDEVICE(*PÉRIPHERIQUE*)....286
DISPOFF()....287
DISPON()....288
DRVCTRL(*PORT, FONCTION, DONNEES*)....289
ENABLEDEVICE(*PERIPHERIQUE*)....290
EXP(*VALEUR*)....291
EXP10(*VALEUR*)....292
FILL(*ELEMENT, DONNEES, NOMBRE*)....293
FIND(*CHAINE, CARACTERE, SAUTER*)....294
FINDFILEFIRST(*RÉPERTOIRE*)....295

FINDFILENEXT() 296
FORMATCOMPACTFLASH() 297
GETDATE (*TEMPS*) ET FAMILLE 298
GETINTERFACESTATUS(*PORT*) 299
GETMONTHDAYS(*A, M*) 300
GETNETGATE(*PORT*) 301
GETNETID(*PORT*) 302
GETNETIP(*PORT*) 303
GETNETMASK(*PORT*) 304
GETNOW() 305
GETNOWDATE() 306
GETNOWTIME() 307
GETUPDOWNDATA(*DONNÉES, LIMITE*) 308
GETUPDOWNSTEP(*DONNÉES, LIMITE*) 309
GOTOPAGE(*NOM*) 310
GOTOPREVIOUS() 311
HIDEPOPUP() 312
INTTOTEXT(*DONNEES, BASE, NOMBRE*) 313
ISDEVICEONLINE(*PERIPHERIQUE*) 314
LEFT(*CHAINE, NOMBRE*) 315
LEN(*CHAINE*) 316
LOG(*VALEUR*) 317
LOG10(*VALEUR*) 318
MAKEFLOAT(*VALEUR*) 319
MAKEINT(*VALEUR*) 320
MAX(*A, B*) 321
MEAN(*ELEMENT, NOMBRE*) 322
MID(*CHAINE, POSITION, NOMBRE*) 323
MIN(*A, B*) 324
MULDIV(*A, B, C*) 325
NOP() 326
OPENFILE(*NOM, MODE*) 327
PI() 328
PLAYRTTTL(*MÉLODIE*) 329
POPDEV(*ELEMENT, NOMBRE*) 330
PORTCLOSE(*PORT*) 331
PORTINPUT(*PORT, DEBUT, FIN, DELAI, LONGUEUR*) 332
PORTPRINT(*PORT, CHAINE*) 333
PORTREAD(*PORT, PERIODE*) 334
PORTWRITE(*PORT, DONNEES*) 335
POSTKEY(*CODE, TRANSITION*) 336

AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

POWER(*VALEUR, PUISSANCE*) 338
RAD2DEG(*THÊTA*) 339
RANDOM(*GAMME*) 340
READDATA(*DONNEES, NOMBRE*) 341
READFILELINE(*FICHIER*) 342
RIGHT(*CHAINE, NOMBRE*) 343
SCALE(*DONNEES, R1, R2, E1, E2*) 344
SENDMAIL(*DESTINATAIRE, OBJET, CORPS*) 345
SET(*ETIQUETTE, VALEUR*) 346
SETLANGUAGE(*CODE*) 347
SETNETCONFIG(*PORT, ADRESSE, MASQUE, PASSERELLE*) 348
SETNOW(*HEURE*) 349
SGN(*VALEUR*) 350
SHOWMENU(*NOM*) 351
SHOWPOPUP(*NOM*) 352
SIN(*THÊTA*) 353
SLEEP(*PÉRIODE*) 354
SQRT(*VALEUR*) 355
STDDEV(*ELEMENT, NOMBRE*) 356
STOPSYSTEM() 357
STRIP(*TEXTE, CIBLE*) 358
SUM(*ELEMENT, NOMBRE*) 359
TAN(*THÊTA*) 360
TESTACCESS(*DROITS, INVITE*) 361
TEXTTOADDR(*ADRESSE*) 362
TEXTTOFLOAT(*CHAINE*) 363
TEXTTOINT(*CHAINE*, BASE) 364
TIME(*H, M, S*) 365
USERLOGOFF() 366
USERLOGON() 367
WAITDATA(*DONNEES, NOMBRE, TEMPS*) 368
WRITEFILELINE(*FICHIER, TEXTE*) 369

AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

MANUEL DE L'UTILISATEUR DE CRIMSON 2

CONTROLEUR MODULAIRE

AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

INTRODUCTION

Bienvenue à Crimson 2, le dernier package de configuration de la société Red Lion Controls. Crimson est conçu pour fournir un accès rapide et simple aux fonctionnalités du contrôleur modulaire et de la station de données tout en permettant à l'utilisateur avancé d'utiliser les fonctions avancées telles que le support de programmation unique de Crimson.

CONFIGURATION SYSTÈME REQUISE

Crimson 2 est conçu pour s'exécuter sur des ordinateurs dotés des spécifications suivantes :

- Processeur de type Pentium, tel que requis pour le système d'exploitation sélectionné.
- RAM et espace disque disponible, tels que requis par le système d'exploitation sélectionné.
- 50 Mo d'espace disque supplémentaire pour l'installation logicielle.
- écran d'au moins 800 x 600 pixels, avec 256 ou plus.
- Connecteur RS-232 ou port USB pour le téléchargement.

Crimson 2 est compatible avec toutes les versions de Microsoft Windows, à partir de Windows 95. Si vous souhaitez profiter des fonctions du port USB fournies par le module maître, vous devrez utiliser au minimum Windows 98. Si vous souhaitez utiliser le port USB pour accéder à distance à la carte CompactFlash du maître, nous vous conseillons d'utiliser Windows 2000 ou Windows XP. Même si Windows 98 peut accéder à la carte, les dernières versions du système d'exploitation permettent des opérations plus solides. Elles sont plus adaptées lorsqu'elles choisissent de verrouiller la carte, empêchant par conséquent l'écriture de données lors de l'activation de C2.

INSTALLATION DU LOGICIEL

Si vous avez téléchargé le logiciel Crimson sur le site Web Red Lion, il vous suffit d'exécuter le fichier téléchargé et de suivre les instructions. Si vous avez reçu un exemplaire de Crimson sur un CD, insérez ce CD dans le lecteur CD-ROM de votre ordinateur, puis suivez les instructions qui s'affichent. Si aucune instruction ne s'affiche, c'est peut-être que l'option d'exécution automatique est désactivée. Dans ce cas, sélectionnez l'option Exécuter dans le menu Démarrer, puis entrez **`x:\setup`**, **`x`** désignant votre lecteur CD-ROM. Suivez alors les instructions pour procéder à l'installation du logiciel.

VÉRIFICATION DES MISES A JOUR

Si vous possédez une connexion à Internet, vous pouvez utiliser la commande Vérifier s'il y a une mise à jour dans le menu Aide pour rechercher une nouvelle version de Crimson sur le site Web Red Lion. Si vous trouvez une version ultérieure à celle que vous utilisez, Crimson vous demandera si vous souhaitez télécharger la mise à niveau pour mettre automatiquement à jour votre logiciel. Vous pouvez également télécharger de façon manuelle la mise à niveau sur le site Web Red Lion en visitant la page Téléchargements de la section Support. Quelle

que soit l'option choisie, lorsque le package de mise à niveau s'exécute, assurez-vous que vous avez bien sélectionné l'option *Réparer* pour mettre à jour votre installation.

INSTALLATION DES PILOTES USB

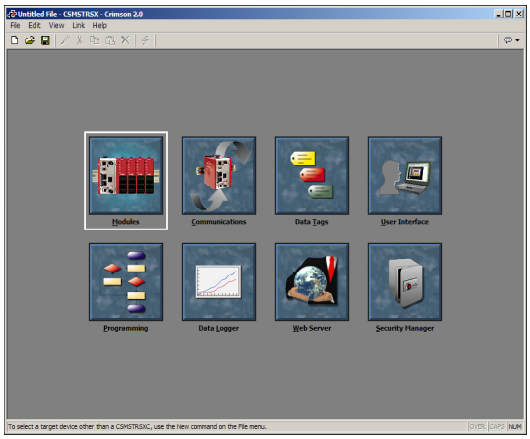
Lorsque vous connectez pour la première fois un module maître sur votre ordinateur à l'aide d'un câble USB, Windows vous demande de définir l'emplacement des pilotes du périphérique. L'emplacement par défaut de ces pilotes est : C:\Program Files\Red Lion Controls\Crimson 2.0\Device. Lorsque l'Assistant Matériel s'affiche, sélectionnez l'option Parcourir, pointez sur l'Assistant situé à cet emplacement ou sur n'importe quel autre emplacement que vous avez défini lors de l'installation du logiciel. Il est important que cette étape soit correctement effectuée, sinon vous devrez peut-être supprimer manuellement les pilotes à l'aide du Gestionnaire de périphériques, puis refaire l'installation. Les utilisateurs de Windows XP doivent noter que Microsoft n'a pas signé numériquement les pilotes USB de Crimson. De ce fait, une boîte de dialogue vous permettra d'interrompre l'installation. Vous devez vous assurer que vous avez bien sélectionné l'option *Continuer* pour indiquer que vous souhaitez effectivement installer les pilotes.

DÉMARRAGE RAPIDE

Le contrôleur modulaire est une plateforme flexible pour laquelle un manuel de plus de 300 pages s'avérait réellement nécessaire. Cependant, vous voudrez sans doute immédiatement passer à la partie programmation avant d'en prendre connaissance dans son intégralité. La section suivante va assez loin pour vous permettre de créer un système opérationnel.

PRÉSENTATION

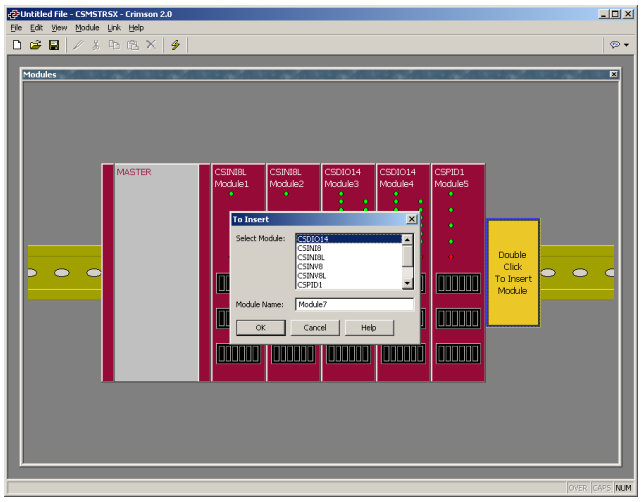
Pour les applications les plus simples, seules les deux premières icônes (Modules et Communications) sont nécessaires. La première permet de configurer le système (ajout, suppression et modification de modules) et la seconde permet de mapper les données de module sur un ou plusieurs périphériques externes.



MODULES

La première étape de la configuration d'une base de données consiste à créer et configurer les différents modules utilisés dans l'application via la fenêtre Modules de Crimson.

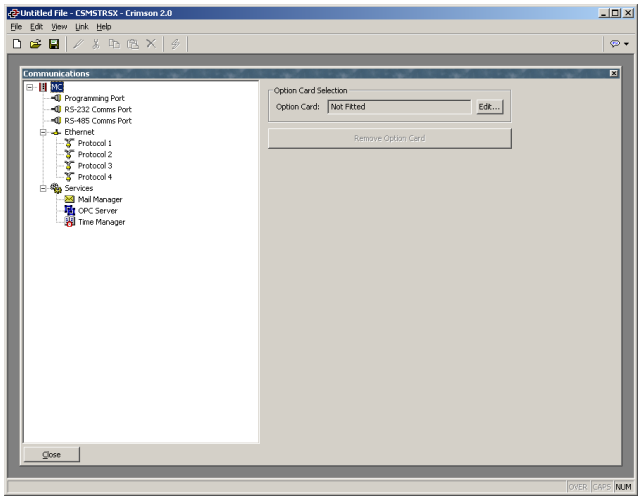
Pour ajouter un module dans le système, double-cliquez sur une base vierge. Vous devez alors sélectionner le type de module qui sera ajouté. Vous pouvez également fournir un nom descriptif du module.



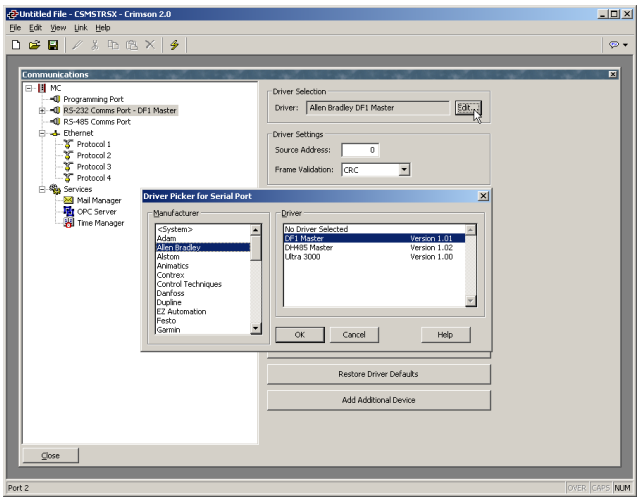
Modifiez les propriétés de chaque module en double-cliquant dessus. Chaque module et ses propriétés sont présentés de façon détaillée dans les sections suivantes.

COMMUNICATIONS

Ensuite, vous devrez configurer un port pour communiquer les données à votre API, PC, etc. Les ports sont configurés dans la fenêtre Communications du logiciel. Cliquez sur le bouton Edition dans la zone de sélection du pilote pour choisir un protocole.



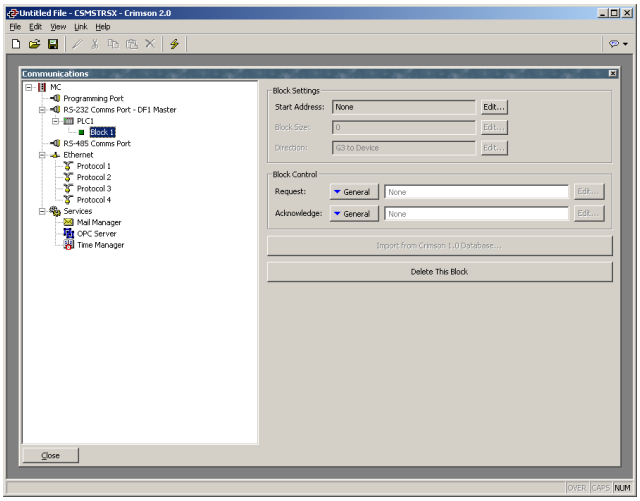
Dans l'exemple ci-dessous, le maître DF1Allen Bradley a été sélectionné comme protocole. Un périphérique appelé PLC1 a été créé. Vous devez vérifier que les propriétés du pilote (par exemple, la parité, la vitesse de transmission, l'adresse, etc.) soient logiques pour votre application.



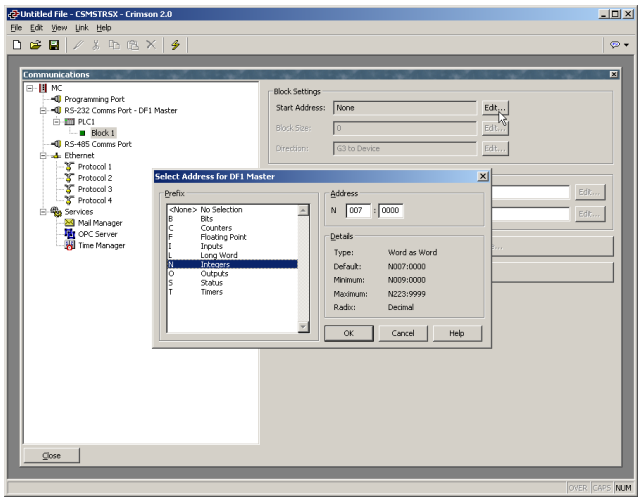
MAPPING DES DONNÉES

Pour mapper des données entre le contrôleur modulaire et un périphérique externe, vous devez créer deux « blocs de passerelle » : le premier permet de déplacer les données de module *dans* votre périphérique et le second d'obtenir les données de module *à partir de* votre périphérique.

Pour créer un bloc de passerelle, cliquez sur *PLC1*, puis sur le bouton Ajouter bloc de passerelle situé dans le volet droit. Un bloc de passerelle unique est ainsi créé pour mapper les données…

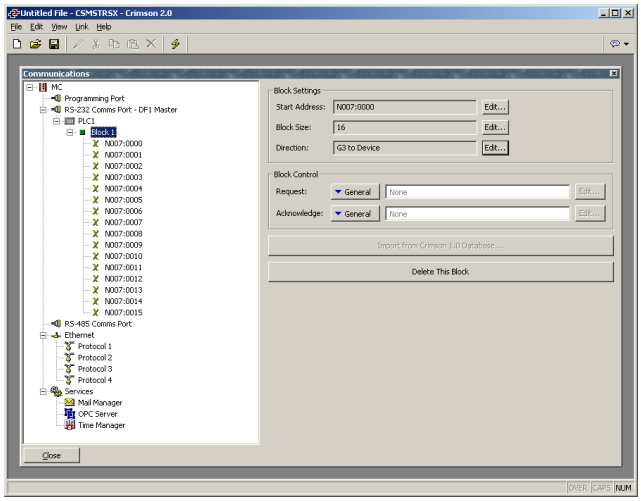


Vous devez ensuite identifier les emplacements des registres vers ou à partir desquels vous souhaitez mapper les données, en commençant par l'adresse de registre. Cliquez sur le bouton Edition pour connaître les registres gérés par Crimson pour le protocole sélectionné…



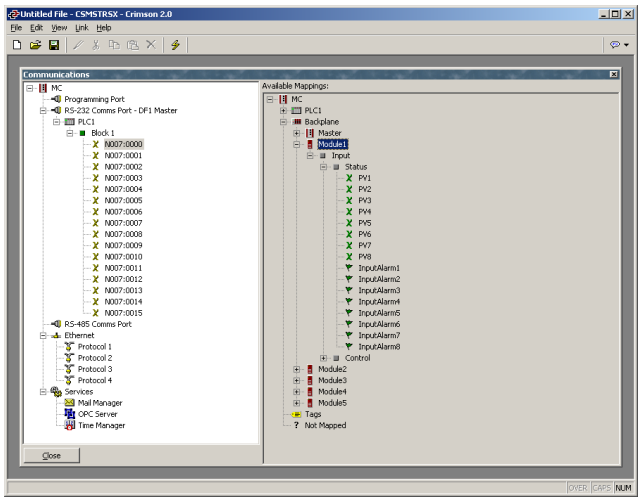
Saisissez le nombre souhaité de registres dans la propriété *Taille du bloc* ainsi que la direction souhaitée à l'aide de la propriété *Direction*.

Dans notre exemple ci-dessous, les registres N7:0 à N7:16 ont été attribués.

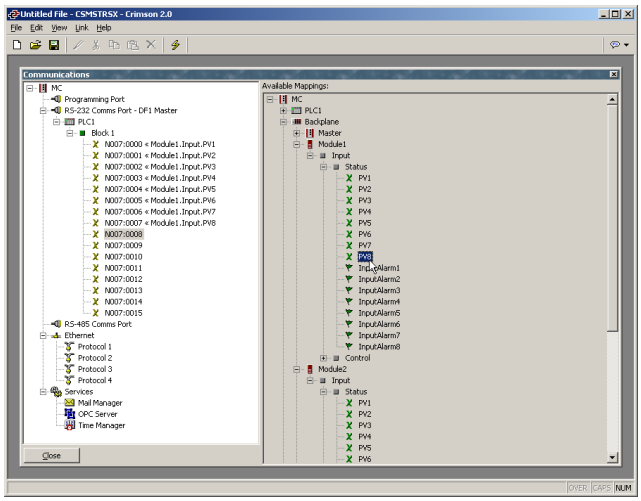


Pour mapper des données de module, sélectionnez l'un des registres. Une liste des données disponibles s'affiche dans le volet droit…

AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

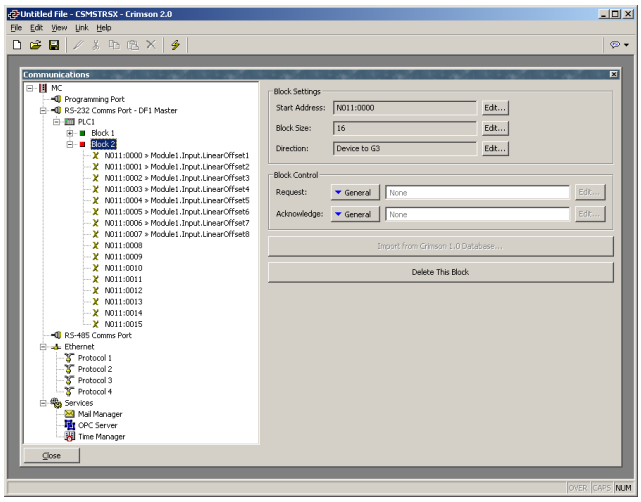


Pour mapper des éléments à partir du volet droit sur les éléments situés à gauche, il vous suffit de double-cliquer dessus, un par un. Le curseur se placera automatiquement sur le registre suivant, ce qui vous permettra de mapper rapidement tous les éléments. Au lieu de cliquer sur les données dans le volet droit, vous pouvez également effectuer un glisser-déplacer des données dans des registres spécifiques.



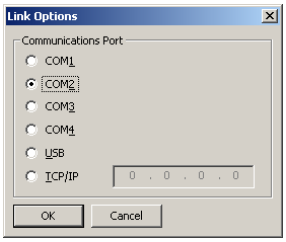
Dans l'exemple ci-dessus, chaque valeur d'entrée du premier module a été mappée sur les registres N7:0 à N7:7. La direction des flèches indique que le contrôleur modulaire écrit ces valeurs dans l'API.

Ensuite, un autre bloc est créé pour déplacer les données à partir de l'API vers le contrôleur modulaire. Cette opération est effectuée en modifiant la propriété *Direction* du bloc sur le périphérique sur le contrôleur modulaire. Dans l'exemple suivant, les valeurs de décalage de Module1 sont mappées sur les registres N11:0 à N11:7. La propriété *Direction* indique que le contrôleur modulaire obtient sa valeur à partir des registres de l'API et les écrit dans les valeurs de décalage du module.



TÉLÉCHARGEMENT

Le téléchargement dans le maître via un port série est facile. Il suffit de vous assurer que vous avez sélectionné le port COM correct sous Options de lien.



Si vous voulez utiliser le port USB pour télécharger dans le maître, vous devez prendre le temps de lire la section intitulée Installation des pilotes USB. Le port Ethernet n'est pas configuré par le fabricant. Il ne pourra donc pas être utilisé pour télécharger une base de données initiale.

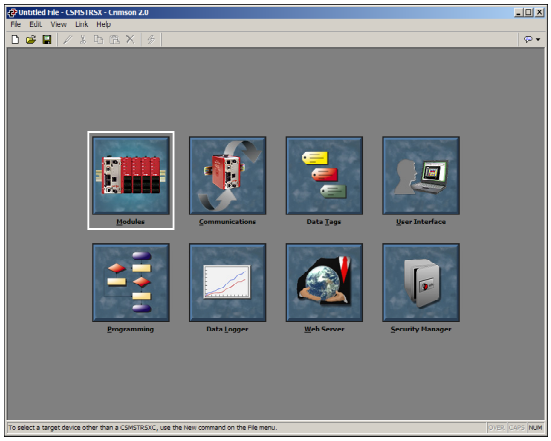
CONNEXION DU PÉRIPHÉRIQUE ET DU CONTRÔLEUR MODULAIRE

Vous y êtes ! Connectez le périphérique au contrôleur modulaire et si vous avez sélectionné un protocole maître, le contrôleur modulaire fera le reste. Examinez simplement les registres de destination : ils doivent contenir les données écrites. Essayez de modifier les registres qui ont été définis pour le périphérique sur le contrôleur modulaire et les modules devraient répondre en conséquence.

Si vous avez configuré le contrôleur modulaire pour qu'il agisse en esclave, vous devrez initialiser l'élément (quel qu'il soit) que vous utilisez pour communiquer avec le contrôleur modulaire, puis lire et écrire les données en conséquence.

NOTIONS DE BASE SUR CRIMSON

Pour exécuter Crimson, sélectionnez l'icône Crimson dans le répertoire Red Lion Controls au niveau des Programmes du menu Démarrer. L'écran principal C2 s'affiche, indiquant les icônes qui sont utilisées pour configurer les différents aspects du comportement du contrôleur modulaire…



Le logiciel est conçu de telle façon que les trois premières icônes sont les seules qui sont nécessaires pour la plupart des applications simples. Les icônes restantes permettent d'accéder aux fonctionnalités les plus avancées du maître comme la programmation, l'enregistrement de données et le serveur Web.

ICÔNES DE L'ECRAN PRINCIPAL

Les sections ci-dessous décrivent les différentes icônes…

MODULES

Cette icône est utilisée pour configurer les modules dans le système. Les modules peuvent être ajoutés, supprimés ou leur ordre sur le rail peut être modifié. Vous pouvez alors accéder à chaque module pour configurer ses propriétés spécifiques.

COMMUNICATIONS

Cette icône sert à indiquer les protocoles qui sont utilisés sur les ports série du maître et sur le port Ethernet. Lorsque les protocoles maîtres sont utilisés (c'est-à-dire des protocoles par lesquels le module maître initie le transfert de données vers et à partir d'un périphérique distant), vous pouvez également utiliser cette icône pour spécifier un ou plusieurs périphériques qui seront accédés. Lorsque les protocoles esclaves sont utilisés (c'est-à-dire les protocoles par lesquels le maître reçoit et répond aux demandes des périphériques ou des systèmes informatiques distants), vous pouvez choisir les éléments de données qui seront exposés pour un accès en lecture ou en écriture. Cette icône sert également à déplacer des données d'un périphérique distant à un autre via le convertisseur de protocole de Crimson.

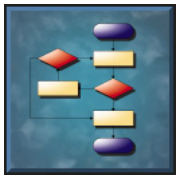
ETIQUETTES DE DONNÉES

Cette icône est utilisée pour renommer les éléments de données à partir des modules, mais aussi pour définir les éléments auxquels vous accèderez dans les périphériques distants. Elle permet également de créer des éléments de données internes pour enregistrer des informations dans le maître lui-même. Chaque étiquette possède différentes propriétés qui lui sont associées. La propriété de base est la mise en forme des données, utilisée pour définir la façon dont les données contenues dans une étiquette s'affichent dans l' IHM virtuel du maître (Web Serveur) et sur les pages Web. En précisant ces informations dans l'étiquette, Crimson supprime le besoin de ressaisir les données de mise en forme à chaque affichage d'une étiquette. Les propriétés d'étiquette plus avancées comprennent des alarmes qui peuvent s'activer lorsque différentes conditions relatives à l'étiquette se produisent, ou des déclencheurs qui effectuent des actions programmables dans des conditions identiques.

INTERFACE UTILISATEUR

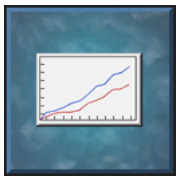
Cette icône est utilisée pour créer et modifier les pages d'affichage du HMI virtuel ainsi que pour spécifier les mesures qui doivent être prises lorsque vous appuyez, relâchez ou maintenez enfoncées les touches de l' IHM. L'éditeur de page vous permet d'afficher différents éléments graphiques connus comme étant des primitives. Ils peuvent varier des éléments simples comme les rectangles et les lignes aux éléments plus complexes qui peuvent être liés à la valeur d'une étiquette ou expression particulière. Par défaut, ces primitives utilisent les informations de mise en forme définies lors de la création de l'étiquette, mais ces informations peuvent être écrasées, si nécessaire.

PROGRAMMATION



Cette icône sert à créer et modifier des programmes grâce au langage de programmation unique de type C de Crimson 2. Ces programmes peuvent prendre des décisions complexes ou effectuer des opérations de manipulation de données selon les éléments de données présents dans le système. Ils permettent d'étendre les fonctionnalités de Crimson au-delà des fonctions standard comprises dans le logiciel, garantissant ainsi que même les applications les plus complexes peuvent être abordées en toute simplicité.

HISTORIQUE DES DONNÉES



Cette icône est utilisée pour créer et gérer les historiques de données. Chacun d'entre eux peut enregistrer n'importe quel nombre de variables sur la carte CompactFlash du maître. Les données peuvent être enregistrées très rapidement : une donnée par seconde. Les valeurs enregistrées seront stockées dans des fichiers CSV (variable séparée par une virgule) qui peuvent facilement être importés dans des applications comme Microsoft Excel. Vous pouvez accéder aux fichiers en lisant la carte CompactFlash directement par un lecteur de CompactFlash, par le port USB d'un PC connecté au port USB du maître ou en y accédant via le serveur Web de Crimson par le biais du port Ethernet.

SERVEUR WEB

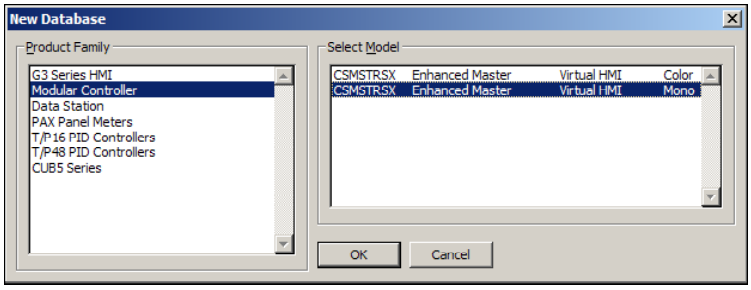
Cette icône sert à configurer le serveur Web de Crimson, à créer et à modifier des pages Web. Le serveur Web peut créer un accès distant au maître via plusieurs mécanismes. Tout d'abord, vous pouvez utiliser Crimson pour créer des pages Web automatiques contenant des listes d'étiquettes formatées selon les propriétés des étiquettes. Ensuite, vous pouvez créer un site personnalisé à l'aide d'un éditeur HTML tiers comme Microsoft FrontPage, puis ajouter du texte spécifique pour demander à Crimson d'insérer des valeurs d'étiquette en direct. Enfin, vous pouvez activer les fonctionnalités uniques de contrôle/commande et d'accès à distance de Crimson 2, qui permettent au navigateur Web d'afficher l' IHM virtuel du maître et de contrôler son clavier. Le serveur Web peut également être utilisé pour accéder aux fichiers CSV à partir de l'historique de données.

GESTIONNAIRE DE SÉCURITÉ

Cette icône sert à créer et gérer les différents utilisateurs de l'écran ainsi que leurs droits d'accès. Vous pouvez aussi leur attribuer des noms complets qui permettent à l'historique de données d'enregistrer non seulement les données qui ont été modifiées et leur date de modification, mais également l'utilisateur qui les a modifiées. Les droits requis pour modifier une étiquette particulière ou pour accéder à une page, sont définis via les propriétés de sécurité de chaque élément.

SÉLECTION DU CONTRÔLEUR MODULAIRE

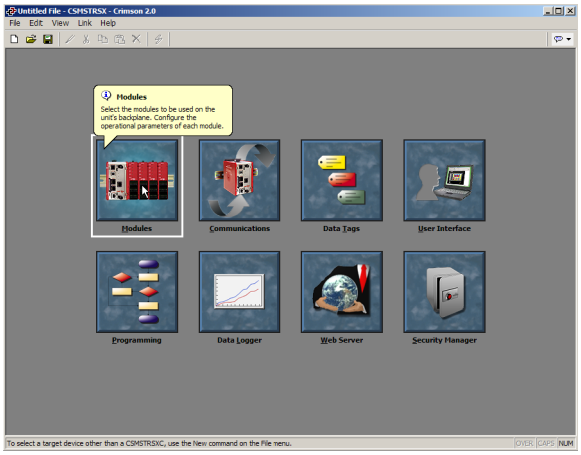
Au premier démarrage de Crimson, il suppose que vous continuez à travailler avec le même périphérique Red Lion qui était utilisé par la dernière base de données chargée. Si Crimson n'a pas déjà été exécuté, il suppose que vous travaillez avec un HMI G303. Pour configurer le contrôleur modulaire, sélectionnez la commande Nouveau dans le menu Fichier. La boîte de dialogue suivante s'affiche…



Cette boîte de dialogue répertorie les périphériques pris en charge par la version actuelle du logiciel, fournissant une description de chacun d'entre eux. La sélection d'un périphérique crée une base de données vierge et configure à nouveau Crimson pour qu'il fonctionne avec ce produit spécifique.

UTILISATION DES INFO-BULLES

Crimson propose une fonctionnalité pratique appelée les infos-bulles...



Cette fonctionnalité vous permet de visualiser les informations d'aide pour chaque icône du menu principal ou pour chaque champ d'une boîte de dialogue ou d'une fenêtre. Elle est contrôlée via l'icône située dans la partie inférieure droite de la barre d'outils et peut être configurée en trois modes : « Ne pas afficher » (les infos-bulles sont désactivées), « Lorsque la souris est dessus » (l'aide s'affiche lorsque le pointeur de la souris est placé sur un champ spécifique pendant quelques instants) ou « Lorsque l'objet est sélectionné » (l'aide du champ sélectionné s'affiche toujours).

TRAVAILLER AVEC LES BASES DE DONNÉES

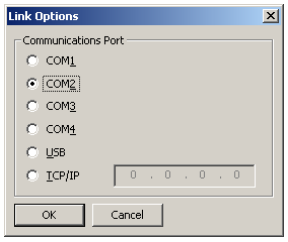
Crimson enregistre toutes les informations sur la configuration d'un périphérique particulier dans un fichier de base de données. Ces fichiers ont pour extension **CD2** bien que l'Explorateur Windows masque cette extension si elle est laissée dans sa configuration par défaut. Les fichiers de base de données de Crimson diffèrent des fichiers utilisés par les anciens produits Red Lion car ce sont des fichiers texte qui sont donc plus faciles à récupérer en cas de corruption accidentelle. Les bases de données sont manipulées via les commandes présentes dans le menu Fichier. Ces commandes sont standards pour toutes les applications Windows et ne nécessitent aucune explication complémentaire. L'exception est la commande Enregistrer le fichier image, que nous aborderons ultérieurement.

TÉLÉCHARGEMENT VERS LE MAÎTRE

Les fichiers de base de données Crimson sont téléchargés sur le maître à l'aide du menu Lien. Le processus de téléchargement ne dure en général que quelques secondes, mais il peut durer davantage lors du premier téléchargement si Crimson doit effectuer la mise à jour du firmware sur le maître ou si ce dernier ne contient pas une version plus ancienne de la base de données actuelle. Toutefois, après ce premier téléchargement, Crimson utilise un processus appelé le téléchargement incrémental pour garantir que seules les modifications apportées à la base de données sont transférées. Ainsi, les modifications peuvent être apportées en quelques secondes, réduisant de ce fait la durée de votre cycle de développement et simplifiant le processus de déboguage.

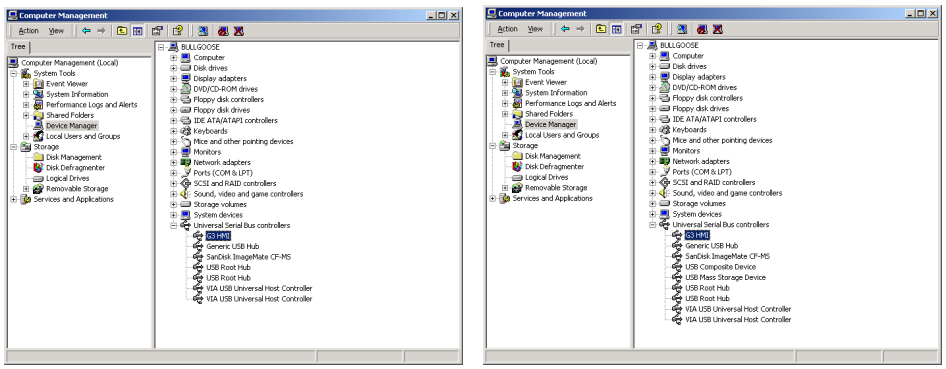
CONFIGURATION DU LIEN

Le lien de programmation entre le PC et le maître s'effectue au moyen d'un port série RS-232, d'un port USB ou d'une connexion TCP/IP. Bien que les connexions TCP/IP sont en règle générale établies via le port Ethernet du maître, elles peuvent également l'être via un lien d'appel entrant. Avant tout téléchargement, vous devez utiliser la commande Options de lien pour vous assurer que vous avez sélectionné la méthode…



VÉRIFICATION DU LIEN USB

Si vous utilisez un port USB, assurez-vous également que les pilotes USB du maître ont correctement été installés. Pour cela, connectez le module maître et, si les pilotes n'ont pas encore été installés, suivez les instructions indiquées au début de ce guide. Ouvrez le Gestionnaire de périphériques de votre système d'exploitation, puis développez l'icône USB pour afficher l'icône du périphérique du module maître. Assurez-vous que cette icône n'affiche pas un symbole d'avertissement. Si tel est le cas, retirez le périphérique, débranchez et rebranchez le module maître, puis vérifiez que vous avez suivi à la lettre la procédure d'installation du pilote. Les écrans ci-dessous montrent des vues générales du Gestionnaire de périphériques avec la carte CompactFlash respectivement retirée, puis reconnectée….

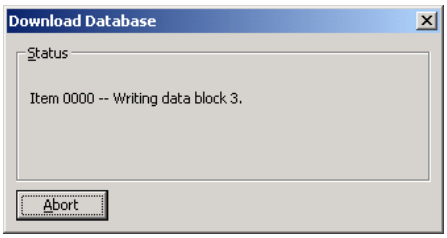


DEFINITION DE L'ADRESSE IP

Si vous utilisez une connexion TCP/IP, vous devez entrer l'adresse IP du périphérique de destination dans le champ approprié de la boîte de dialogue. Si vous laissez l'adresse IP sous la forme 0.0.0.0, Crimson examinera la base de données actuellement chargée pour vérifier que l'adresse du maître peut être déterminée à partir des informations de configuration. Cette fonctionnalité permet de ne pas modifier les adresses IP lors du basculement entre plusieurs bases de données conçues pour des terminaux différents.

ENVOI DE LA BASE DE DONNÉES

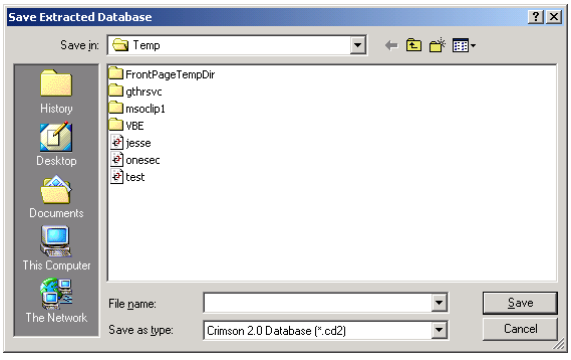
Une fois que le lien est configuré, la base de données peut être téléchargée à l'aide des commandes Lien-Envoyer ou Lien-Actualiser. La première commande envoie la base de données dans son intégralité, que des objets individuels aient été modifiés ou non dans le fichier. La seconde commande envoie uniquement les modifications et elle dure bien moins longtemps. La commande Actualiser est en général la seule commande dont vous aurez besoin car Crimson a automatiquement recours à un envoi complet si le téléchargement incrémental échoue pour quelque raison que ce soit. Pour un raccourci, notez que vous pouvez accéder à la commande Lien-Actualiser via le symbole de l'éclair situé dans la barre d'outils ou en appuyant sur la touche **F9** de votre clavier.



Notez que le téléchargement via le protocole TCP/IP utilise sur une carte CompactFlash installée dans le maître si le firmware du périphérique doit être mis à niveau. Comme vous devrez à un moment ou à un autre effectuer ces mises à niveau, nous vous conseillons vivement d'installer une carte CompactFlash sur tout périphérique sur lequel les téléchargements TCP/IP sont susceptibles d'être effectués.

EXTRACTION DES BASES DE DONNÉES

Vous pouvez utiliser la commande Lien-Supporter Upload pour demander à Crimson si les informations nécessaires à la prise en charge du téléchargement de la base de données doivent être ajoutées lors de l'envoi d'une base de données à un module maître. La prise en charge du téléchargement ralentit quelque peu le processus de téléchargement et peut échouer avec les très grandes bases de données contenant de nombreuses images incorporées. Toutefois, si vous perdez votre fichier de base de données, elle garantit que vous pouvez extraire une image modifiable à partir du terminal.



N.B : si vous perdez votre fichier de base de données et que la prise en charge du téléchargement n'est pas activée, vous devrez recréer votre fichier en totalité. Pour extraire une base de données à partir d'un maître, utilisez la commande Lien-Extraire. Cette

AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

commande télécharge la base de données et vous invite ensuite à saisir un nom pour enregistrer le fichier. Le fichier est ouvert pour être modifié.

MONTAGE DE LA CARTE COMPACTFLASH

Si vous êtes connecté à un module maître via le port USB, vous pouvez demander à Crimson de « monter » la carte CompactFlash du maître sous forme de lecteur dans l'Explorateur Windows. Vous pouvez utiliser cette fonctionnalité pour enregistrer des fichiers sur la carte ou pour lire des informations à partir de l'historique de données. Le lecteur est « monté » et « démonté » en envoyant des commandes à l'aide des options Monter flash et Démonter flash présentes dans le menu Lien. Une fois une commande envoyée, le module maître est réinitialisé et Windows rafraîchit les fenêtres de l'Explorateur appropriées pour afficher ou masquer le lecteur CompactFlash.

Notez que vous devez faire preuve de prudence lorsque vous « montez » la carte CompactFlash...

- Une fois la carte « montée », le maître signale périodiquement au PC les modifications de données apportées à la carte. C'est-à-dire que le PC et le maître subissent des gains de performances si la carte est montée lors d'opérations d'enregistrement de données qui durent plus longtemps que prévu.

- Si vous écrivez sur la carte CompactFlash à partir de votre PC, le maître ne peut pas y accéder si Windows n'ouvre pas son « verrou » sur le contenu de la carte. Cela peut durer une minute, limiter les opérations d'enregistrement de données sur cette période et empêcher d'accéder aux pages Web personnalisées. Crimson utilise la RAM du maître pour garantir qu'aucune donnée n'est perdue, mais si trop d'écritures sont effectuées, qui verrouillent la carte pendant quatre minutes ou plus, les données peuvent être ignorées. Notez que Windows 98 n'est pas particulièrement performant pour conserver la carte verrouillée lorsqu'elle n'est pas utile. Windows 2000 ou Windows XP sont les systèmes d'exploitation idéaux lorsque vous utilisez cette fonctionnalité.

- Vous ne devez jamais essayer d'utiliser Windows pour formater une carte CompactFlash que vous avez « montée » via le maître, que ce soit par le biais de l'Explorateur ou à partir de l'invite de commande. Windows ne verrouille pas correctement la carte lors des opérations de formatage ce qui pourrait entraîner

un manque de fiabilité du format ainsi qu'une perte de données. Reportez-vous aux détails ci-dessous décrivant le formatage d'une carte de façon fiable.

FORMATAGE DE LA CARTE COMPACTFLASH

La meilleure méthode pour formater une carte consiste à utiliser la commande Formater Flash du menu Lien. Cette commande vous explique que le processus de formatage détruit toutes les données enregistrées sur la carte CompactFlash et vous permet d'annuler l'opération. Si vous choisissez de poursuivre, il sera demandé au maître de formater la carte. Notez que ce processus peut prendre quelques minutes pour une carte de grande capacité. Les formatages lents sur des maîtres qui effectuent une opération d'enregistrement de données peuvent provoquer des vides dans les données enregistrées.

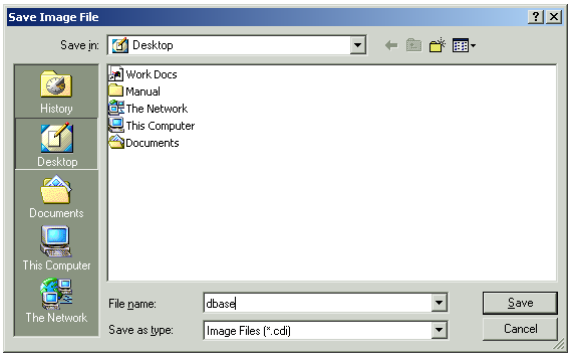
Une méthode moins fiable de formatage de votre carte consiste à utiliser un lecteur CompactFlash dédié connecté à votre PC. Si vous appliquez cette méthode, assurez-vous de demander à Windows de formater la carte à l'aide d'un volume FAT16. Pour les cartes de très petite ou très grande capacité, Windows risque de choisir le mauvais format par défaut. Pire encore, certaines versions de l'Explorateur Windows ne vous permettent pas de remplacer le format par défaut, vous forçant à utiliser à la place la version de ligne de commande **FORMAT**.

ENVOI DE L'HEURE ET DE LA DATE

Vous pouvez utiliser la commande Lien-Envoyer heure pour définir l'heure du maître afin qu'elle corresponde à celle du PC sur lequel Crimson est en cours d'exécution. Assurez-vous évidemment que vous disposez de la bonne heure avant d'effectuer cette opération !

TÉLÉCHARGEMENT DEPUIS LA CARTE COMPACTFLASH

Si vous devez mettre à jour la base de données dans une unité qui est déjà installée sur le site du client, Crimson vous permet d'enregistrer une copie de la base de données sur une carte CompactFlash, d'envoyer cette carte à votre client et de faire charger la base de données à partir de cette carte par le G3. Ce processus s'effectue via la commande Enregistrer le fichier image du menu Fichier.

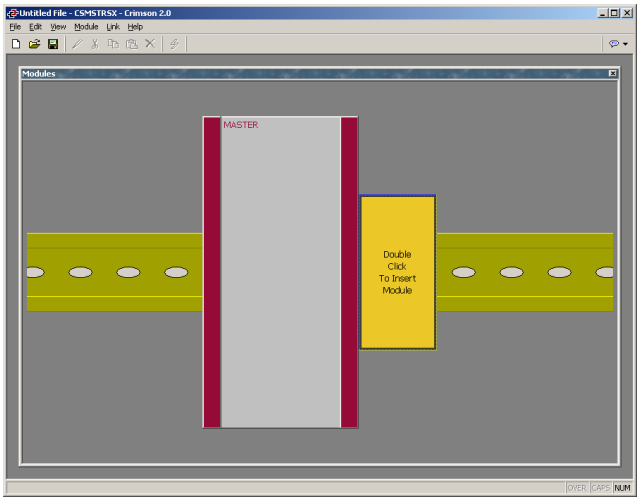


La commande Enregistrer le fichier image crée un fichier image de la base de données Crimson avec une extension **CDI**. Elle crée également trois autres fichiers avec des extensions **BIN**, **LDR** et **ROM**. Le fichier image doit être nommé **DBASE.CDI** et avec les fichiers **BIN**, **LDR** et **ROM**, il doit être placé dans le répertoire racine d'une carte CompactFlash. Pour mettre à jour un module maître, éteignez l'unité, insérez la carte CompactFlash contenant les quatre fichiers

et allumez l'unité. Le chargeur de démarrage du maître vérifie d'abord s'il doit mettre à niveau le firmware de l'unité et une fois ce processus terminé, l'application d'exécution de Crimson charge la base de données enregistrée sur la carte. Vous pouvez ensuite retirer la carte CompactFlash ou la laisser en place.

CONFIGURATION DES MODULES

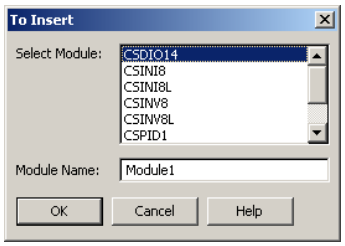
La première étape de la configuration d'une base de données du contrôleur modulaire consiste à créer et configurer les différents modules utilisés dans l'application via la fenêtre Modules de Crimson.



MODULES

AJOUT DE MODULES

Pour ajouter un module dans le système, double-cliquez sur une base vierge. Vous êtes invité à sélectionner le type de module qui sera ajouté. Vous pouvez également fournir un nom descriptif du module.



Si vous devez programmer plusieurs modules, programmez-en seulement un, puis utilisez les fonctions de copier et coller pour créer des doublons. Vous pouvez effectuer cette opération en cliquant avec le bouton droit de la souris sur un module ou sur la base, ou en utilisant les commandes Windows standard comme **`Ctrl+C`** et **`Ctrl+V`**.

DEPLACEMENT DE MODULES

Vous pouvez supprimer ou déplacer des modules à l'aide du menu déroulant Module. Il vous suffit de sélectionner un module, puis à partir du menu, de sélectionner Supprimer, Déplacer à gauche ou Déplacer à droite.

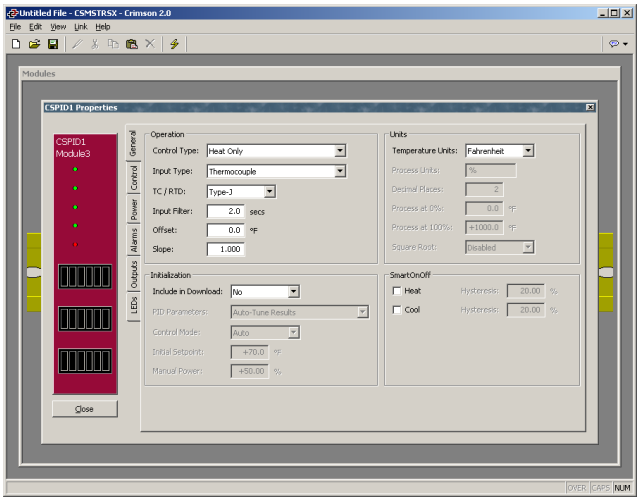
MODIFICATION DE MODULES

Pour modifier les propriétés d'un module spécifique, double-cliquez dessus. Pour obtenir une description des différents modules et de leurs propriétés, reportez-vous aux sections suivantes.

CSPID – PROGRAMMATION DU MODULE PID

Les paramètres du module CSPID sont divisés en groupes et chacun d'entre eux possède sa propre page. Le module CSPID2 dispose de plusieurs onglets supplémentaires permettant de configurer la deuxième boucle. Utilisez les onglets situés dans la partie gauche de la fenêtre pour afficher les différentes pages.

L'ONGLET GENERAL



FONCTIONNEMENT

- La propriété *Type de commande* vous permet de choisir entre Chaud, Froid ou Chaud et froid, en fonction du type de processus à contrôler. Pour les processus qui ne sont pas des applications thermiques, sélectionnez Chaud pour les applications inversées et Froid pour les applications directes.
- La propriété *Type* d'entrée est utilisée pour configurer la puissance de la RTD, du thermocouple ou le type d'entrée du processus correct.
- La propriété *TC/RTD* est utilisée pour sélectionner le capteur standard utilisé.
- La propriété *Filtre d'entrée* est une constante de temps utilisée pour stabiliser les signaux d'entrée variables.

UNITES

- La propriété *Unités de température* est utilisée pour sélectionner l'une des échelles de température suivantes : Kelvin, Fahrenheit ou Celsius.

- La propriété *Unités du process* vous permet d'entrer les unités techniques du processus. Elle est uniquement utilisée pour identifier les champs appropriés dans le logiciel. Ce paramètre est enregistré dans le fichier Crimson, mais il n'est pas utilisé dans le module.

- La propriété *Décimales* est utilisée pour permettre à Crimson d'afficher les unités techniques avec la bonne résolution. Elle ne sert qu'à afficher la résolution appropriée dans le logiciel, mais elle n'est pas utilisée dans le module.

- Les propriétés *Signal à 0 %* et *Signal à 100 %* sont utilisées pour mettre à l'échelle les signaux d'entrée du courant continu. Entrez la lecture de la valeur de processus souhaitée des niveaux de signal d'entrée minimums et maximums. Par exemple, si l'application implique un capteur de flux avec une sortie de 4 à 20 mA, proportionnelle à 5 à 105 litres par minute, sélectionnez Process 4-20 mA comme propriété *Type d'entrée*, saisissez 5 comme paramètre *Signal à 0 %* et saisissez 105 comme paramètre *Signal à 100 %.*

INITIALISATION

Les paramètres d'initialisation fournissent des valeurs initiales aux paramètres habituellement contrôlés par un PC ou une API. Dans les applications générales, ces paramètres ne sont utilisés que jusqu'à ce que les communications soient établies pour la première fois.

- La propriété *Inclure au téléchargement* est utilisée pour déterminer si les valeurs d'initialisation sont téléchargées dans le module. En choisissant l'option « Non », vous autorisez la modification et le téléchargement des bases de données à volonté, sans écrasement accidentel des paramètres du processus définis comme le point de consigne, les valeurs PID, etc.

- La propriété *Paramètres PID* dicte le chargement de tel ou tel paramètre PID par le module ainsi que son utilisation afin de contrôler le processus. Le module contrôle le processus à l'aide des valeurs PID actif et Filtre de puissance actif. (Reportez-vous aux variables **`ActConstP`**, **`ActConstI`**, **`ActConstD`** et **`ActFilter`** du tableau Données disponibles qui se trouve à la fin de cette section.) Le groupe actif est chargé avec les paramètres PID utilisateur ou les valeurs des résultats de l'autotune, selon l'état du bit **`ReqUserPID`**. Si le bit est vrai, le groupe actif est chargé avec les variables de l'utilisateur. Si le bit est faux, les valeurs qui ont été définies par l'autotune sont chargées. La modification de la propriété *Paramètres PID* écrit correctement le bit **`ReqUserPID`** lors de l'initialisation.

- La propriété *Mode de contrôle* dicte le mode (automatique ou manuel) du module lors de l'initialisation. En mode automatique, le contrôleur calcule la sortie requise pour atteindre et conserver le point de consigne, puis agit en conséquence. En mode manuel, la sortie peut être directement contrôlée en écrivant dans la variable **`power`**.

- La propriété *Point de consigne initial* est utilisée comme valeur du point de consigne lors de l'initialisation.

- *Puissance manuelle* est le niveau du module PID en mode manuel. Les valeurs au-delà de 100 % et –100 % peuvent être saisies pour compenser les limitations provoquées par les valeurs du transfert de puissance comme les gains et les décalages qui limiteraient les sorties à des valeurs inférieures à leur maximum.

SMARTONOFF

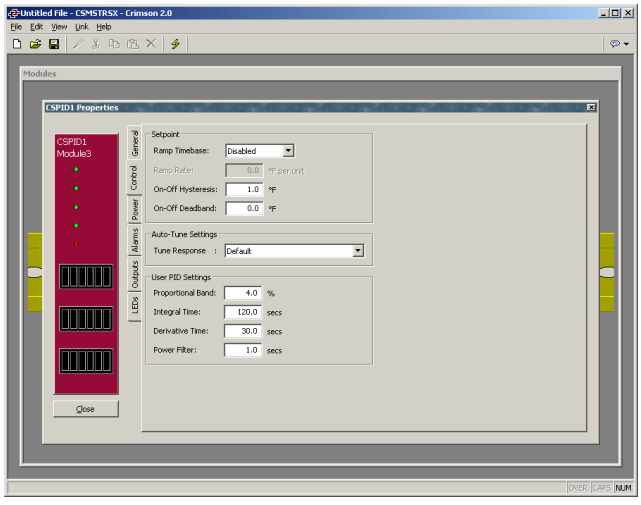
SmartOnOff est conçu pour les situations où une commande d'activation/de désactivation serait normalement utilisée, mais où les avantages du PID sont également nécessaires. Lorsque du chaud ou du froid est placé dans ce mode, la sortie de commande est activée ou désactivée, sans aucune valeur intermédiaire ou proportion temporelle.

Toutefois, au lieu d'utiliser la valeur du processus pour décider quand activer la sortie, SmartOnOff examine la sortie du calcul PID et active la sortie lorsqu'elle dépasse la moitié du gain défini pour ce canal.

Par exemple, avec les paramètres par défaut, SmartOnOff activerait pour le chaud la sortie de chaleur une fois que l'algorithme PID a requis 50 % de puissance ou plus, avec la valeur d'hystérésis qui est utilisée pour garantir que les petites modifications dans le calcul PID ne provoquent aucun battement du relais de sortie.

- La propriété *Hystérésis* est utilisée pour supprimer le battement du relais de sortie en séparant les points d'activation et de désactivation de la ou des sorties lors de la commande de SmartOnOff. La valeur *Hystérésis* est centrée sur le point de consigne, c'est-à-dire que les points de transition de la sortie sont compensés au-dessous et au-dessus du point de consigne par la moitié de la valeur *Hystérésis*.

L'ONGLET COMMANDE



CONSIGNE

- La propriété *Base de temps rampe* sélectionne les secondes, minutes ou heures comme unité de temps pour allonger ou raccourcir le processus.

- La propriété *Vitesse rampe* est utilisée pour réduire le choc soudain sur un processus lors des modifications du point de consigne et au démarrage du système. Une rampe d'accélération et de décélération du point de consigne peut être utilisée pour augmenter ou diminuer le point de consigne réel sur une vitesse contrôlée. La valeur est saisie en unités/temps. Une valeur de 0 désactive la rampe du point de consigne. Si la rampe d'accélération et de décélération du point de consigne est une valeur autre que zéro et si le point de consigne requis est modifié ou que le module est activé, le contrôleur définit le point de consigne réel sur la mesure du processus en cours et utilise cette valeur comme son point de consigne. Il ajuste ensuite le point de consigne réel à la rampe d'accélération et de décélération du point de consigne. Lorsque le point de consigne réel recherche le point de consigne requis, le contrôleur reprend l'utilisation de la valeur du point de consigne requis. (Dans un système configuré et qui fonctionne de manière correcte, le processus suit la valeur du point de consigne réel sur la valeur du point de consigne requis.)

- La propriété *On/Off Hystérésis* est utilisée pour supprimer le battement du relais de sortie en séparant les points d'activation et de désactivation de la ou des sorties lors de la commande d'activation/de désactivation. La valeur d'hystérésis est centrée sur le point de consigne, c'est-à-dire que les points de transition de la sortie sont compensés au-dessous et au-dessus du point de consigne par la moitié de la valeur *On/Off Hystérésis*. Cette valeur affecte les sorties programmées sur Chaud ou Froid. Lors de l'autotune, le contrôleur gère le processus par le biais de 4 cycles d'activation/de désactivation. Il est donc important de définir *On/Off Hystérésis* sur une valeur appropriée avant d'initialiser l'autotune.

- La propriété *On/Off Bande Morte* permet de compenser les points d'activation des sorties de chaud et froid programmées pour une opération d'activation/de désactivation. On obtiendra une bande morte si la valeur est positive et un chevauchement si la valeur est négative. Lorsque vous déterminez les points de transition réels des sorties, la valeur *On/Off Hystérésis* doit également être prise en compte.

PARAMETRES DE L'AUTOTUNE

- La propriété *Réponse Autotune* est utilisée pour garantir qu'un autotune produit les valeurs P, I et D optimales pour différentes applications. Un paramètre Très agressive entraîne la définition du PID qui atteint le point de consigne aussi vite que possible, sans aucune préoccupation pour le dépassement alors qu'un paramètre Très conservatrice sacrifie la vitesse pour empêcher tout dépassement.

- Remarque : si la propriété *Réponse Autotune* est modifiée, l'autotune doit être réinitialisé pour que ces modifications puissent influer sur les paramètres PID. Pour en savoir plus, reportez-vous à la section Autotune.

PARAMETRES PID UTILISATEUR

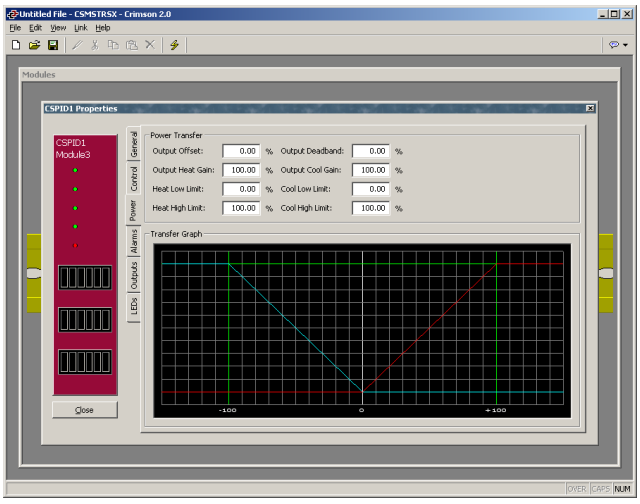
- La propriété *Bande Proportionnelle*, saisie comme un pourcentage de toute la plage d'entrée, est la quantité de modifications d'entrées nécessaires pour varier la plage de sortie complète. Pour les entrées de température, la plage d'entrée est fixée par le thermocouple ou le type de RTD saisi. Pour les entrées, la plage d'entrée est la différence entre les valeurs Signal à 0 % et Signal à 100 %. La propriété *Bande Proportionnelle* peut être réglée de 0,0 % à 1 000,0 % et doit être définie sur une valeur qui fournit la meilleure réponse à une perturbation du processus tout en réduisant le dépassement. Une propriété *Bande Proportionnelle* de 0,0 % force le contrôleur sur la commande d'activation/de désactivation avec son cycle caractéristique sur le point de consigne. La valeur optimale peut être définie en appelant l'autotune.

- La propriété *Temps Intégral* est le temps en secondes qui est nécessaire à l'action intégrale pour égaler l'action proportionnelle lors d'une erreur de processus constante. Aussi longtemps que l'erreur sera présente, l'action intégrale sera répétée à chaque *Temps Intégral*. Plus la valeur est importante, plus la réponse est lente. La valeur optimale peut être définie en appelant l'autotune. La propriété *Temps Intégral* peut être réglée de 0 à 6 000,0 secondes.

- La propriété *Temps Dérivé* représente les secondes par répétition que le contrôleur envisage à l'erreur de rampe pour voir ce que sera la contribution proportionnelle et elle correspond à cette valeur à chaque *Temps dérivé*. Aussi longtemps que l'erreur de rampe sera présente, la contribution dérivée est répétée à chaque heure dérivée. L'augmentation de la valeur permet de stabiliser la réponse, mais si cette valeur est trop élevée et que les processus des signaux sont trop bruyants, la sortie pourra trop varier, provoquant une commande médiocre. La définition de l'heure sur zéro désactive l'action dérivée. La propriété optimale *Temps Dérivé* peut être définie en appelant l'autotune. La valeur peut être réglée de 0 à 600,0 secondes.

- La propriété *Filtre de puissance* est une constante de temps, entrée en secondes, qui bloque la puissance de sortie calculée. L'augmentation de la valeur optimise l'effet de blocage. En règle générale, une propriété *Filtre de puissance* dans la gamme d'un vingtième à un cinquantième du temps intégral du contrôleur (ou de la constante de temps du processus) est effective. Les valeurs supérieures à celles-ci peuvent provoquer l'instabilité du contrôleur du fait de l'ajout de l'effet de décalage.

L'ONGLET PUISSANCE



TRANSFERT DE PUISSANCE

- La valeur *Décalage de la sortie* (Offset) modifie le point de sortie zéro du calcul de la puissance de sortie du module. Cette fonctionnalité est plus couramment utilisée dans les applications uniquement proportionnelles afin de supprimer l'erreur permanente.

- La propriété *Bande Morte de la sortie* définit la zone dans laquelle les sorties chaudes et froides sont inactives (plus connue sous le nom de bande morte) ou la zone dans laquelle elles sont actives (plus connue sous le nom de chevauchement). Une valeur positive provoque une bande morte alors qu'une valeur négative provoque un chevauchement.

- La propriété *Gain Sortie de Chauffage* définit le gain de la sortie chaude relative au gain défini par la propriété *Bande Proportionnelle*. Avec une valeur de 100 %, le gain de chaleur imite le gain déterminé par la bande proportionnelle. Une valeur inférieure à 100 % peut être utilisée dans les applications où la chaleur est trop puissante alors qu'une valeur supérieure à 100 % peut être utilisée lorsque la chaleur est insuffisante. Pour la majorité des applications, la valeur par défaut de 100 % convient et des réglages seront nécessaires uniquement si le processus le nécessite.

- La propriété *Gain Sortie de Refroidissement* définit le gain de la sortie froide relative au gain défini par la propriété Bande Proportionnelle. Avec une valeur de 100 %, le gain de fraîcheur imite le gain déterminé par la bande proportionnelle. Une valeur inférieure à 100 % peut être utilisée dans les applications où le dispositif de refroidissement est trop puissant alors qu'une valeur supérieure à 100 % peut être utilisée lorsque le dispositif de refroidissement est insuffisant. Pour la majorité des applications, la valeur par défaut de 100 % convient et les réglages doivent uniquement être effectués si le processus le nécessite.

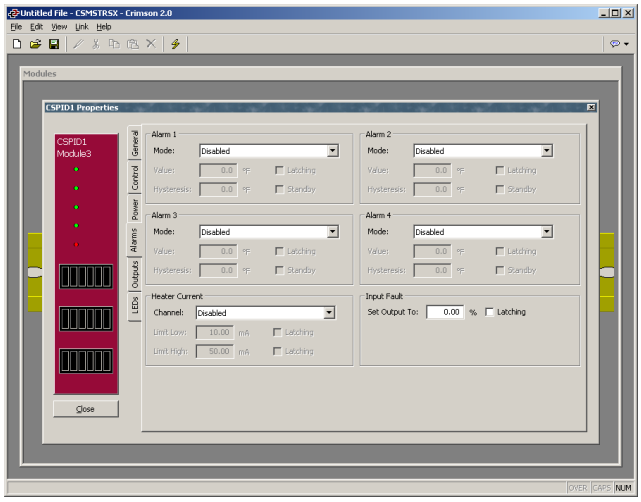
- Les propriétés *Minimum de Chauffage* et *Maximum de Chauffage* peuvent être utilisées pour limiter la puissance du contrôleur lors de perturbations du processus ou de modifications du point de consigne. Entrez les limites de puissance de sortie correctes du processus. Vous pouvez saisir des valeurs supérieures à 100 % et –100 % pour surmonter les limitations provoquées par les valeurs du transfert de puissance comme les gains et les décalages qui limiteraient la sortie à des valeurs inférieures à leur maximum.

- Les propriétés *Minimum de Refroidissement* et *Maximum de Refroidissement* peuvent être utilisées pour limiter la puissance du contrôleur lors de perturbations du processus ou de modifications du point de consigne. Entez les limites de puissance de sortie correctes du processus. Vous pouvez saisir des valeurs supérieures à 100 % et –100 % pour surmonter les limitations provoquées par les valeurs du transfert de puissance comme les gains et les décalages qui limiteraient la sortie à des valeurs inférieures à leur maximum.

GRAPHIQUE DE TRANSFERT

Le graphique de transfert de puissance représente les résultats des modifications des paramètres de la puissance. La ligne bleue représente le dispositif de refroidissement alors que la ligne rouge représente le chauffage.

L'ONGLET ALARMES



Vous pouvez utiliser les quatre alarmes du processus « flexibles » pour contrôler le statut du processus et pour mettre en marche les sorties physiques du module. Sinon, vous pouvez contrôler seulement le bit via le système lui-même ou via des périphériques externes.

- La propriété *Mode* détermine le comportement de l'alarme. Le tableau ci-dessous décrit les différentes sélections.

MODE	**DESCRIPTION**
Inférieur à	L'alarme s'active lorsque la valeur du processus est inférieure à la valeur de l'alarme. L'alarme se désactive lorsque la valeur du processus est supérieure à la valeur de l'alarme + l'hystérésis.
Supérieur à	L'alarme s'active lorsque la valeur du processus est supérieure à la valeur de l'alarme. L'alarme se désactive lorsque la valeur du processus est inférieure à la valeur de l'alarme – l'hystérésis.
Déviation basse	Si la valeur du processus est inférieure à la valeur du point de consigne par la quantité de la valeur de l'alarme, l'alarme s'active. Dans ce mode, le point de l'alarme suit la valeur du point de consigne.
Déviation haute	Si la valeur du processus est supérieure à la valeur du point de consigne par la quantité de la valeur de l'alarme, l'alarme s'active. Dans ce mode, le point de l'alarme suit la valeur du point de consigne.
Dans la bande	Si la différence entre la valeur du point de consigne et la valeur du processus est inférieure à la valeur de l'alarme, l'alarme s'active.
Hors de la bande	Si la valeur du processus est supérieure ou inférieure à la valeur du point de consigne par une quantité égale à la valeur de l'alarme, l'alarme s'active. Dans ce mode, le point de l'alarme suit la valeur du point de consigne.

- La propriété *Valeur* représente le point où l'alarme s'active. Les valeurs de l'alarme sont saisies en tant qu'unité ou degrés du processus.

- La valeur *Hystérésis* sépare les points d'activation et de désactivation de l'alarme, c'est-à-dire qu'une alarme à puissance élevée, programmée pour se mettre en marche à 500 avec une *Hystérésis* de 10 s'éteint lorsque la valeur du processus est inférieure à 490.

- La propriété *Maintenu* dicte la façon dont l'alarme se comporte une fois qu'elle est activée. Pour en savoir plus, reportez-vous à la section ci-dessous, intitulée Tableau du comportement de l'alarme.

- La propriété *Inhibition sur : Seuil/Alim* permet d'empêcher l'apparition de ce qu'on appelle les alarmes de nuisance lors de la mise en marche. Pour en savoir plus, reportez-vous à la section ci-dessous, intitulée Tableau du comportement de l'alarme.

TABLEAU DU COMPORTEMENT DE L'ALARME

MAINTENU	INHIBITION SUR : SEUIL/ALIM	COMPORTEMENT DE L'ALARME
		L'alarme est activée ou désactivée automatiquement lorsque la valeur du processus croise la région de l'alarme. Le bit AlarmAccept désactive l'alarme, quel que soit l'état du processus. Si la condition d'alarme existe et si le bit est écrit sur « 0 », l'alarme s'active.
☑		Une fois que l'alarme est activée, elle reste active jusqu'à ce qu'elle soit acceptée. Si la condition d'alarme n'existe plus, le fait d'écrire le bit AlarmAccept sur « 1 » réinitialise la condition d'alarme. Tant que le bit AlarmAccept est « 1 », l'alarme est activée et désactivée automatiquement lorsque la valeur du processus croise la région de l'alarme.
	☑	L'alarme est activée ou désactivée automatiquement lorsque la valeur du processus croise la région de l'alarme. L'alarme est automatiquement désactivée lors d'une modification du point de consigne ou lors de la première mise en marche du module. Ainsi, aucune alarme de nuisance ne se produit. L'alarme reste désactivée jusqu'à ce que le processus entre dans un état de non-alarme. La prochaine fois qu'une valeur de processus entrera dans une condition d'alarme, l'alarme s'activera en conséquence. Le bit AlarmAccept désactive l'alarme, quel que soit l'état. Si la condition d'alarme existe et si le bit est écrit sur « 0 », l'alarme s'active.
☑	☑	Une fois l'alarme activée, elle reste active jusqu'à ce qu'elle soit acceptée. L'alarme est automatiquement désactivée lorsque le point de consigne est modifié ou lorsque le module est mis en marche pour la première fois. Ainsi, aucune alarme de nuisance ne se produit. L'alarme reste désactivée jusqu'à ce que le processus entre dans un état de non-alarme. La prochaine fois qu'une valeur de processus entrera dans une condition d'alarme, l'alarme s'activera en conséquence. L'écriture momentanée du bit AlarmAccept sur « 1 » désactive une alarme active. Si la condition d'alarme existe encore, l'alarme reste désactivée et elle est mise en mode Inhibition sur : Seuil/Alim. Ainsi, l'alarme reste désactivée jusqu'à la fin de la condition d'alarme, puis elle est réentrée. Si le bit AlarmAccept reste à « 1 », l'alarme est désactivée et ne fonctionne pas.

COURANT DE CHAUFFE

L'alarme du courant de chauffe permet de contrôler la condition du circuit de commande en courant alternatif externe via l'entrée du moniteur du courant de chauffe.

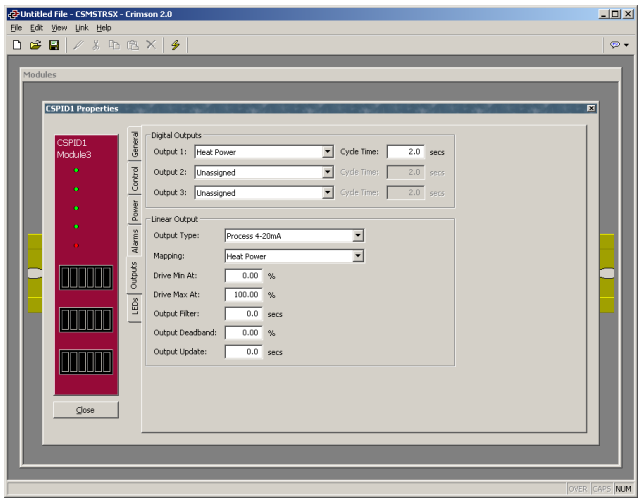
- La propriété *Canal* permet de sélectionner laquelle des trois sorties sera contrôlée.
- La propriété *Minimum* est la valeur mA souhaitée (de 0 à 100,00 mA) qui représente le courant du circuit bloqué permis. Si l'entrée du moniteur du courant de chauffe mesure une valeur de courant supérieure à la valeur *Minimum* lors de l'état bloqué de la sortie, l'alarme s'active.
- La propriété *Minimum* est la valeur mA souhaitée (de 0 à 100,00 mA) qui représente le courant du circuit fermé permis. Si l'entrée du moniteur du courant de chauffe mesure une valeur de courant inférieure à la valeur *Maximum* lors de l'état passant de la sortie, l'alarme s'active.
- La propriété *Maintenu* détermine si une alarme activée reste active jusqu'à ce qu'elle soit acceptée. Pour accepter une alarme maintenue, son bit **`AlarmAccept`** doit être écrit sur 1. Si Maintenu n'est pas sélectionné, l'alarme se désactive automatiquement lorsque la condition d'alarme n'existe plus et le bit **`AlarmAccept`** peut être utilisé comme un moyen de désactiver l'alarme.

ERREUR D'ENTREE

La section Erreur d'entrée permet de définir la réponse des sorties de commande du module CSPID en cas d'échec de l'entrée. L'alarme de l'erreur d'entrée est considérée comme une alarme de processus pour les éléments qui peuvent être mappés sur « n'importe quelle alarme de processus ».

- La propriété *Puissance* représente la valeur de sortie du contrôleur en cas d'un échec du capteur d'entrée. Les valeurs au-delà de 100 % et –100 % peuvent être saisies pour compenser les limitations provoquées par les valeurs du transfert de puissance comme les gains et les décalages qui limiteraient les sorties à des valeurs inférieures à leur maximum.
- Si la propriété *Maintenu* est activée, elle permettra au bit de l'erreur d'entrée de rester actif jusqu'à ce qu'il soit accepté, quel que soit l'état de l'entrée. Pour réinitialiser l'erreur, le bit **`InputAccept`** doit être écrit sur 1. Si Maintenu n'est pas sélectionné, l'erreur se désactive alors immédiatement lorsque l'échec d'entrée est corrigé.

L'ONGLET SORTIES



SORTIES NUMERIQUES

- Les propriétés *Sortie n* permettent d'attribuer les sorties physiques du module aux différentes propriétés ou valeurs internes. La liste du CSPID2 a été étendue pour inclure le numéro de canal, par exemple Puissance de chauffe du canal 1 ou Puissance de chauffe du canal 2.

- La valeur *Temps de cycle* est la somme des cycles d'activation/de désactivation d'une sortie proportionnelle au temps. Avec des sorties proportionnelles au temps, le pourcentage de la puissance de sortie est converti en sortie en temps de la valeur temporelle du cycle. Par exemple : si l'algorithme du contrôleur nécessite une puissance de 65 % et possède une propriété Temps de cycle de 10 secondes, la sortie sera activée pendant 6,5 secondes et désactivée pendant 3,5 secondes. Une propriété Temps de cycle égale ou inférieure à un dixième de la constante de temps du processus est conseillée.

SORTIE LINEAIRE - (UNIQUEMENT CSPID1)

- La propriété *Type de sortie* permet de sélectionner une sortie 0 à 10 V, 0 à 20 mA ou 4 à 20 mA. Assurez-vous que les bretelles de sortie, situées sur le côté du module, sont définies pour le même type de sortie.

- La propriété *Mapping* permet d'attribuer la sortie analogique du module à l'une des différentes propriétés ou valeurs internes. Par exemple : la propriété *Mapping* doit être définie sur Puissance de chauffe pour contrôler un processus de chauffe uniquement via la sortie analogique.

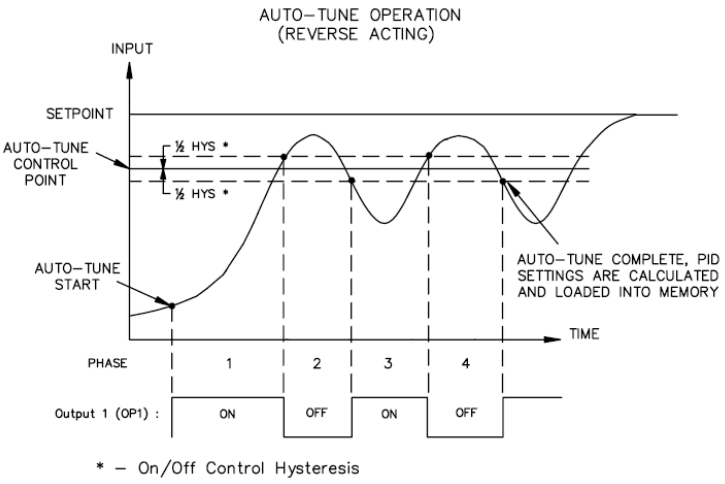
- Les valeurs *Minimum* à et *Maximum* à permettent de mettre à l'échelle la sortie analogique. Les unités exprimées sont identiques à celles de la valeur Mapping. Par conséquent, les limites numériques varient. Exemple : les valeurs de 0 % et 100 % sont généralement utilisées pour contrôler la position d'une valve dans les applications de chauffe uniquement.

- La propriété *Filtre de sortie* est une constante de temps, entrée en secondes, qui bloque la réponse de la sortie analogique. L'augmentation de la valeur augmente l'effet de blocage.
- La valeur *Bande Morte de la sortie* permet d'empêcher la modification de la sortie analogique lorsque de simples petits réglages sont nécessaires. Cela permet de prévenir l'usure mécanique lors de la manipulation d'une valve d'entrée linéaire. La sortie analogique ne se règle pas sauf si la modification nécessaire est supérieure de moitié à la valeur de la bande morte, c'est-à-dire qu'avec une bande morte de 10 % et une valeur de sortie de 50 %, la sortie n'est pas modifiée tant que les valeurs 45 % ou 55 % ne sont pas appelées. Les unités exprimées sont identiques à celles de la valeur Mapping.
- La temporisation *Mise à jour de la sortie* permet de diminuer la fréquence de mise à jour de la sortie analogique. Lorsque la temporisation Mise à jour de la sortie expire, la sortie analogique vérifie que la modification requise est supérieure à la valeur Bande Morte de la sortie. Le cas échéant, la sortie renvoie la nouvelle valeur. Dans le cas contraire, la sortie est identique et le minuteur redémarre.

AUTOTUNE

PRESENTATION

L'autotune permet de définir les valeurs optimales P, I, D, et Filtre de puissance. En gérant le processus par le biais de 4 cycles d'activation/de désactivation, le module obtient des informations sur le processus et détermine les meilleures valeurs.



Comme la figure ci-dessus le montre, le point de consigne utilisé lors de l'autotune représente la valeur 75 % au-dessus de la différence entre la valeur actuelle du processus et le point de consigne. Cela permet aux oscillations de se produire près du point de consigne tout en évitant des dépassements excessifs. Comme le module commande d'activation/de désactivation lors de l'autotune, il est important de définir une valeur d'hystérésis d'activation/de désactivation adaptée avant d'invoquer l'autotune.

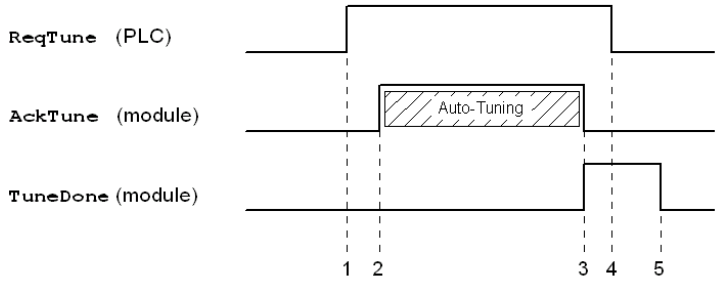
La personnalisation du PID pour définir l'activation de l'autotune est possible en réglant le paramètre Réponse Autotune. Ce paramètre peut être défini sur Très agressive, Agressive, Par défaut, Conservatrice ou Très conservatrice. De plus, il peut être réglé en écrivant 0-4 respectivement sur le registre Réponse Autotune.

RECOURS À L'AUTOTUNE

La séquence de l'autotune utilise un protocole de communication de requête/accusé de réception. Pour appeler l'autotune, écrivez le bit **ReqTune** sur 1. Le module indique que l'autotune s'exécute en définissant le bit **AckTune** sur élevé. Une fois l'autotune terminé, le bit **TuneDone** augmente. La logique externe devrait être écrite pour désactiver le bit de requête de l'autotune lorsque le bit **TuneDone** augmente. à ce niveau, le module redéfinit le bit **AckTune** sur 0.

Si, pour quelque raison que ce soit, l'autotune échoue, les bits **TuneDone** et **TuneFail** augmentent tous les deux. Cette situation peut se produire si, par exemple, une erreur d'entrée a eu lieu. Elle nécessitera alors la réinitialisation de l'autotune.

Une requête d'autotune ressemble à ceci …



1. L'API définit le bit **ReqTune** sur élevé.
2. Le module démarre l'autotune et définit le bit **AckTune** sur élevé.
3. L'autotune est terminé. Le bit **AckTune** diminue et le bit **TuneDone** augmente.
4. L'API voit le bit **TuneDone** augmenter et définit le bit **ReqTune** sur bas.
5. Le module voit le bit **ReqTune** diminuer et définit le bit **TuneDone** sur bas.

DONNEES DISPONIBLES

Les données suivantes représentent les valeurs de données disponibles pour le maître et par conséquent, elles peuvent être mappées sur les registres de l'API. Les décimales sont utilisées pour indiquer uniquement la résolution et ne sont ni lues, ni écrites. Par exemple, une valeur **`Power`** de 10 000 est interprétée comme 100,00 %.

Remarque : le tableau suivant affiche les données disponibles pour le module CSPID1. Dans la plupart des cas, le module CSPID2 contient les mêmes données pour Boucle 1 et Boucle 2. Ainsi, au lieu de répertorier uniquement **`Module1.PV`**, le CSPID2 répertorie **`Module1.Loop1.PV`** et **`Module1.Loop2.PV`**.

BOUCLE/STATUT

DONNEES	DESCRIPTION	GAMME	ACCES
PV	Valeur du processus – La valeur de l'entrée	*	L (Lecture)
Output	Puissance de sortie – Puissance de sortie calculée de la boucle PID avant le gain, les décalages et les limites	+/- 200,00 %	L
HeatPower	Sortie appliquée aux canaux auxquels est attribuée la propriété Chaud	0 – 100,00 %	L
CoolPower	Sortie appliquée aux canaux auxquels est attribuée la propriété Froid	0 – 100,00 %	L
ActSP	Point de consigne réel	*	L
Error	Différence entre la valeur du processus et la valeur du point de consigne requis	*	L
ColdJunc	Valeur de calibration du raccordement de froid	N/A	L
HCMValue	Valeur d'entrée en mA du courant de chauffe	0 – 100,00 mA	L
AckManual	Accusé de réception du mode manuel	0 ou 1 (bit)	L
AckTune	Accusé de réception de la requête de l'autotune	0 ou 1 (bit)	L
TuneDone	Fin de l'autotune	0 ou 1 (bit)	L
TuneFail	Echec de la fin de l'autotune	0 ou 1 (bit)	L
Alarm1	Statut de l'alarme 1	0 ou 1 (bit)	L
Alarm2	Statut de l'alarme 2	0 ou 1 (bit)	L
Alarm3	Statut de l'alarme 3	0 ou 1 (bit)	L
Alarm4	Statut de l'alarme 4	0 ou 1 (bit)	L
HCMAlarmLo	Alarme minimale du moniteur du courant de chauffe	0 ou 1 (bit)	L
HCMAlarmHi	Alarme maximale du moniteur du courant de chauffe	0 ou 1 (bit)	L
InputAlarm	Entrée hors de la plage	0 ou 1 (bit)	L

* Dépend du type de capteur sélectionné.

AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

BOUCLE/COMMANDE

DONNEES	DESCRIPTION	GAMME	ACCES
ReqSP	Point de consigne requis – La valeur du point de consigne écrite dans le contrôleur. Cette valeur peut être différente par rapport au point de consigne réel dans des applications utilisant la rampe du point de consigne.	*	L/E (Lecture/ Ecriture)
Power	Paramètre de la puissance de sortie manuelle	+/- 200,00 %	L/E
SetHyst	Hystérésis du point de consigne de la commande d'activation/de désactivation	*	L/E
SetDead	Bande morte du point de consigne de la commande d'activation/de désactivation	*	L/E
SetRamp	Rampe d'accélération et de décélération du point de consigne	*	L/E
InputFilter	Filtre d'entrée	0 – 60,0 sec.	L/E
ReqManual	Demande du mode manuel – Ecrivez ce bit sur 1 pour appeler le mode manuel. En mode manuel, l'écriture sur le registre Puissance contrôle la puissance de sortie.	0 ou 1 (bit)	L/E
ReqTune	Demande d'autotune – Ecrivez ce bit sur 1 pour appeler l'autotune.	0 ou 1 (bit)	L/E
ReqUserPID	Demande de définition du PID utilisateur – Elevée charge les valeurs d'utilisateur dans le groupe Actif et Basse charge les résultats de l'autotune dans le groupe Actif.	0 ou 1 (bit)	L/E

BOUCLE/ALARMES

DONNEES	DESCRIPTION	GAMME	ACCES
AlarmData1	Valeur de l'alarme 1	*	L/E
AlarmData2	Valeur de l'alarme 2	*	L/E
AlarmData3	Valeur de l'alarme 3	*	L/E
AlarmData4	Valeur de l'alarme 4	*	L/E
AlarmHyst1	Valeur de l'hystérésis de l'alarme 1	*	L/E
AlarmHyst2	Valeur de l'hystérésis de l'alarme 2	*	L/E
AlarmHyst3	Valeur de l'hystérésis de l'alarme 3	*	L/E
AlarmHyst4	Valeur de l'hystérésis de l'alarme 4	*	L/E
AlarmAccept1	Bit d'acquittement de l'alarme 1	0 ou 1 (bit)	L/E
AlarmAccept2	Bit d'acquittement de l'alarme 2	0 ou 1 (bit)	L/E
AlarmAccept3	Bit d'acquittement de l'alarme 3	0 ou 1 (bit)	L/E
AlarmAccept4	Bit d'acquittement de l'alarme 4	0 ou 1 (bit)	L/E
HCMLimitLo	Valeur de l'alarme maximale du courant de chauffe	0 – 100,00 mA	L/E

DONNEES	DESCRIPTION	GAMME	ACCES
HCMLimitHi	Valeur de l'alarme minimale du courant de chauffe	0 – 100,00 mA	L/E
HCMAcceptLo	Acquittement de l'alarme minimale du courant de chauffe	0 ou 1 (bit)	L/E
HCMAcceptHi	Acquittement de l'alarme maximale du courant de chauffe	0 ou 1 (bit)	L/E
InputAccept	Acquittement de l'alarme d'entrée hors de la plage	0 ou 1 (bit)	L/E

BOUCLE/PID

DONNEES	DESCRIPTION	GAMME	ACCES
TuneCode	Code de réponse de l'autotune	0-4	L/E
UserConstP	Valeur proportionnelle de l'utilisateur	0 – 1 000,0 %	L/E
UserConstI	Valeur intégrale de l'utilisateur	0 – 6 000,0 sec.	L/E
UserConstD	Valeur dérivée de l'utilisateur	0 – 600,0 sec.	L/E
UserFilter	Valeur du filtre de puissance de l'utilisateur	0 – 60,0 sec.	L/E
AutoConstP	Valeur proportionnelle réglée automatiquement	0 – 1 000,0 %	L
AutoConstI	Valeur intégrale réglée automatiquement	0 – 6 000,0 sec.	L
AutoConstD	Valeur dérivée réglée automatiquement	0 – 600,0 sec.	L
AutoFilter	Valeur du filtre de puissance réglée automatiquement	0 – 60,0 sec.	L
ActConstP	Valeur proportionnelle active	0 – 1 000,0 %	L
ActConstI	Valeur intégrale active	0 – 6 000,0 sec.	L
ActConstD	Valeur dérivée active	0 – 600,0 sec.	L
ActFilter	Valeur du filtre de puissance active	0 – 60,0 sec.	L

BOUCLE/PUISSANCE

DONNEES	DESCRIPTION	GAMME	ACCES
PowerFault	Puissance de sortie pour l'erreur d'entrée	+/- 200,00 %	L/E
PowerOffset	Valeur du décalage de la puissance de sortie	+/- 100,00 %	L/E
PowerDead	Valeur de la bande morte de la puissance de sortie	+/- 100,00 %	L/E
PowerHeatGain	Valeur du gain de chauffe de la puissance de sortie	0 – 500,00 %	L/E
PowerCoolGain	Valeur du gain de refroidissement de la puissance de sortie	0 – 500,00 %	L/E
PowerHeatHyst	Hystérésis de chauffe de SmartOnOff de la puissance de sortie	0 – 50,00 %	L/E
PowerCoolHyst	Hystérésis de refroidissement de SmartOnOff de la puissance de sortie	0 – 50,00 %	L/E
HeatLimitLo	Minimum de chauffe	0 – 200,00 %	L/E
HeatLimitHi	Maximum de chauffe	0 – 200,00 %	L/E

AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

DONNEES	DESCRIPTION	GAMME	ACCES
CoolLimitLo	Minimum de refroidissement	0 – 200,00 %	L/E
CoolLimitHi	Maximum de refroidissement	0 – 200,00 %	L/E

SORTIES/TEMPS DE CYCLE

DONNEES	DESCRIPTION	GAMME	ACCES
CycleTime1	Temps de cycle pour la sortie 1	0,1 – 60,0 sec.	L/E
CycleTime2	Temps de cycle pour la sortie 2	0,1 – 60,0 sec.	L/E
CycleTime3	Temps de cycle pour la sortie 3	0,1 – 60,0 sec.	L/E
CycleTime4	Temps de cycle pour la sortie 4 (uniquement CSPID2)	0,1 – 60,0 sec.	L/E

SORTIES/DONNEES A DISTANCE

DONNEES	DESCRIPTION	GAMME	ACCES
DigRemote1	Valeur numérique à distance 1 – Les sorties attribuées à la valeur numérique à distance peuvent être contrôlées en écrivant le bit DigRemote sur 1 ou 0.	0 ou 1 (bit)	L/E
DigRemote2	Valeur numérique à distance 2 – Voir DigRemote1	0 ou 1 (bit)	L/E
DigRemote3	Valeur numérique à distance 3 – Voir DigRemote1	0 ou 1 (bit)	L/E
DigRemote4	Valeur numérique à distance 4 – Voir DigRemote1	0 ou 1 (bit)	L/E
AnlRemote1	Valeur analogique à distance 1 – Les sorties attribuées à la valeur analogique peuvent être contrôlées en écrivant un chiffre à ce mot.	*	L/E
AnlRemote2	Valeur analogique à distance 2 – Voir AnlRemote1	*	L/E
AnlRemote3	Valeur analogique à distance 3 – Voir AnlRemote1	*	L/E
AnlRemote4	Valeur analogique à distance 4 – Voir AnlRemote1	*	L/E

SORTIES/INFORMATIONS

DONNEES	DESCRIPTION	GAMME	ACCES
OP1State	Etat de la sortie 1	0 ou 1 (bit)	L
OP2State	Etat de la sortie 2	0 ou 1 (bit)	L
OP3State	Etat de la sortie 3	0 ou 1 (bit)	L

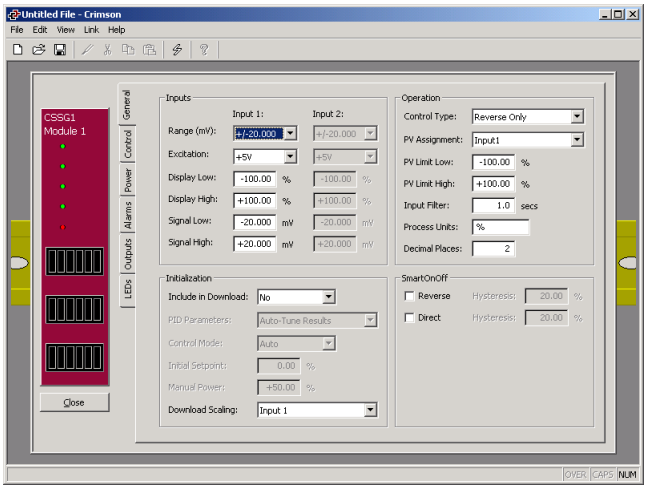
* Dépend de la configuration. A l'exception de l'application détaillée ci-dessous, ces chiffres peuvent être traités comme des entiers signés. Toutes les entrées de température sont mesurées à un dixième de degré de résolution. Pour les entrées, la résolution dépend des valeurs d'échelle de l'utilisateur.

Exception de l'application : si l'application implique une mesure en Fahrenheit supérieure à 3 000 degrés avec un thermocouple de type C, les valeurs suivantes doivent être traitées comme non signées : PV, ReqSP, ActSP et AlarmData 1-4 (lorsqu'elles sont configurées pour une opération absolue).

CSSG – PROGRAMMATION DU MODULE PID À ENTRÉE DE CAPTEUR D'EFFORT

Les paramètres du module CSSG sont divisés en groupes et chacun d'entre eux possède sa propre page. Utilisez les onglets situés dans la partie gauche de la fenêtre pour afficher les différentes pages.

L'ONGLET GENERAL



ENTREES

La section des paramètres d'entrée contient les paramètres des deux entrées. Si le module n'est pas doté de l'entrée secondaire en option, les paramètres de cette dernière sont ignorés.

- La propriété *Gamme* permet de configurer l'entrée de différents niveaux de signal. Elle permet au module de reconfigurer son convertisseur A/D pour une résolution optimale.
- La propriété *Excitation* configure la tension d'excitation de la sortie sur 5 ou 10 Volts.
- Les propriétés *Affichage Min.* et *Affichage Max.* sont utilisées pour mettre à l'échelle l'entrée afin d'indiquer les unités de mesure. Entrez les lectures de valeur du processus souhaitées, correspondant respectivement aux valeurs Signal Min. et Signal Max. Ainsi, si l'application implique un capteur d'effort qui produit une sortie de 0 à 21,00 mV proportionnelle à 0 à 1 000 livres, entrez 0 comme propriété *Affichage Min. et* 1 000 comme propriété *Affichage Max*.
- Les propriétés *Signal Min.* et *Signal Max.* permettent de mettre à l'échelle l'entrée. Saisissez les niveaux de signal qui correspondent respectivement aux propriétés *Affichage Min.* et *Affichage Max*. Ainsi, si l'application implique un capteur d'effort qui produit une sortie de 0 à 21,00 mV proportionnelle à 0 à 1 000 livres, entrez 0 comme propriété *Affichage Min. et* +21,00 comme propriété *Affichage Max*.

FONCTIONNEMENT

- La propriété *Type de commande* vous permet de choisir entre Inversé, Direct ou Inversé et direct, en fonction du type de processus à contrôler.

- La propriété *Val. Process lié à* permet de sélectionner la façon dont le module définit sa valeur de processus mesurée. La valeur PV représente la valeur que l'algorithme PID du module tente de contrôler. Il peut simplement s'agir de la valeur de l'entrée 1 ou l'un des résultats mathématiques de l'entrée 1 et de l'entrée 2.

- Les propriétés *Val. Process Min.* et *Val. Process Max.* permettent de définir la plage de fonctionnement de la valeur du processus et, par conséquent, la plage sur laquelle le module peut effectuer des contrôles. La valeur du processus signalée reste figée sur l'une des limites alors que le processus continue de se déplacer en dehors de ces limites. Le dépassement de l'une de ces limites de plus de 5 % de la plage totale fait supposer au module qu'une erreur de processus s'est produite au moment où la valeur du processus signalée est égale à la valeur du Val. Process Max. (réponse du lecteur supérieure).

- La propriété *Filtre d'entrée* est une constante de temps utilisée pour stabiliser les signaux d'entrée variables.

- La propriété *Unités du process* vous permet d'entrer les unités techniques du processus. Elle est uniquement utilisée pour identifier les champs appropriés dans le logiciel. Ce paramètre est enregistré dans le fichier Crimson, mais il n'est pas utilisé dans le module.

- La propriété *Décimales* est utilisée pour permettre à Crimson d'afficher les unités techniques dans la bonne résolution. Elle ne sert qu'à afficher la résolution appropriée dans le logiciel, mais elle n'est pas utilisée dans le module.

INITIALISATION

Les paramètres d'initialisation fournissent des valeurs initiales aux paramètres habituellement contrôlés par un PC ou une API. Dans les applications générales, ces paramètres ne sont utilisés que jusqu'à ce que les communications soient établies pour la première fois.

- La propriété *Inclure au téléchargement* est utilisée pour déterminer si les valeurs d'initialisation sont téléchargées dans le module. En choisissant l'option « Non », vous autorisez la modification et le téléchargement des bases de données à volonté, sans écrasement accidentel des paramètres du processus définis comme le point de consigne, les valeurs PID, l'augmentation de l'entrée, etc.

- La propriété *Paramètres PID* dicte le chargement de tel ou tel paramètre PID par le module ainsi que son utilisation afin de contrôler le processus. Le module contrôle le processus à l'aide des valeurs PID actif et Filtre de puissance actif. (Reportez-vous aux variables **`ActConstP`**, **`ActConstI`**, **`ActConstD`** et **`ActFilter`** du tableau Données disponibles qui se trouve à la fin de cette

section.) Le groupe actif est chargé avec les paramètres PID utilisateur ou les valeurs des résultats de l'autotune, selon l'état du bit **`ReqUserPID`**. Si le bit est vrai, le groupe actif est chargé avec les variables de l'utilisateur. Si le bit est faux, les valeurs qui ont été définies par l'autotune sont chargées. La modification de la propriété *Paramètres PID* écrit correctement le bit **`ReqUserPID`** lors de l'initialisation.

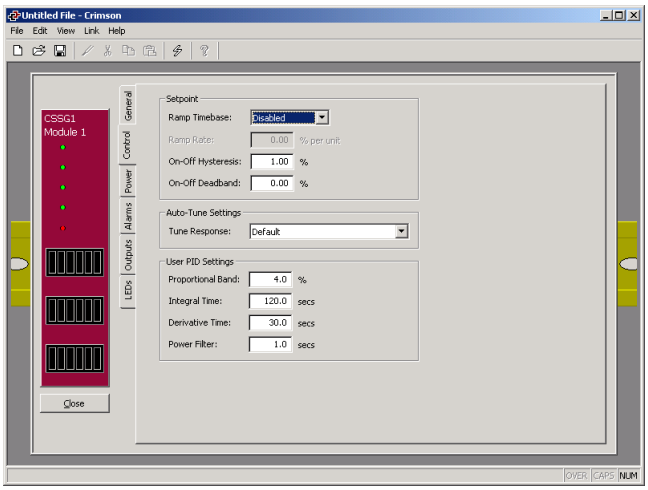
- La propriété *Mode de contrôle* dicte le mode (automatique ou manuel) du module lors de l'initialisation. En mode automatique, le contrôleur calcule la sortie requise pour atteindre et conserver le point de consigne, puis agit en conséquence. En mode manuel, la sortie peut être directement contrôlée en écrivant dans la variable **`power`**.
- La propriété *Point de consigne initial* est utilisée comme valeur du point de consigne lors de l'initialisation.
- *Puissance manuelle* est le niveau du module PID en mode manuel. Les valeurs au-delà de 100 % et –100 % peuvent être saisies pour compenser les limitations provoquées par les valeurs du transfert de puissance comme les gains et les décalages qui limiteraient les sorties à des valeurs inférieures à leur maximum.
- La propriété *Chargé Echelle* définit les valeurs d'échelle, le cas échéant, qui seront téléchargées dans le module. Pour éviter d'écraser les valeurs d'échelle d'un processus calibré, mettez Non.

SMARTONOFF

SmartOnOff est conçu pour les situations où une commande d'activation/de désactivation serait normalement utilisée, mais où les avantages du PID s'avèrerait également nécessaires. Lorsque la sortie inversée ou directe est placée dans ce mode, la sortie de commande est activée ou désactivée, sans aucune valeur intermédiaire ou proportion temporelle. Toutefois, au lieu d'utiliser la valeur du processus pour décider quand activer la sortie, SmartOnOff examine la sortie du calcul PID et active la sortie lorsqu'elle dépasse la moitié du gain défini pour ce canal. Par exemple, avec les paramètres par défaut, pour la sortie inversée, SmartOnOff activerait la sortie une fois que l'algorithme PID a requis 50 % de puissance ou plus, avec la valeur d'hystérésis qui est utilisée pour garantir que les petites modifications dans le calcul PID ne provoquent aucun battement du relais de sortie.

- La propriété *Hystérésis* est utilisée pour supprimer le battement du relais de sortie en séparant les points d'activation et de désactivation de la ou des sorties lors de la commande de SmartOnOff. La valeur *Hystérésis* est centrée sur le point de consigne, c'est-à-dire que les points de transition de la sortie sont compensés au-dessous et au-dessus du point de consigne par la moitié de la valeur *Hystérésis*.

L'ONGLET COMMANDE



POINT DE CONSIGNE

- La propriété *Base de temps rampe* sélectionne les secondes, minutes ou heures comme unité de temps pour allonger ou réduire la durée du processus.

- La propriété *Vitesse rampe* est utilisée pour réduire le choc soudain sur un processus lors des modifications du point de consigne et au démarrage du système. Une rampe d'accélération et de décélération du point de consigne peut être utilisée pour augmenter ou diminuer le point de consigne réel sur une vitesse contrôlée. La valeur est saisie en unités/temps. La valeur 0 désactive la rampe du point de consigne. Si la rampe d'accélération et de décélération du point de consigne est une valeur autre que zéro et si le point de consigne requis est modifié ou que le module est activé, le contrôleur définit le point de consigne réel sur la mesure du processus en cours et utilise cette valeur comme son point de consigne. Il ajuste ensuite le point de consigne réel à la rampe d'accélération et de décélération du point de consigne. Lorsque le point de consigne réel recherche le point de consigne requis, le contrôleur reprend l'utilisation de la valeur du point de consigne requis. (Dans un système configuré et qui fonctionne de manière correcte, le processus suit la valeur du point de consigne réel sur la valeur du point de consigne requis.)

- La propriété *On/Off Hystérésis* sert à supprimer le battement du relais de sortie en séparant les points d'activation et de désactivation de la ou des sorties lors de la commande d'activation/de désactivation. La valeur d'hystérésis est centrée sur le point de consigne, c'est-à-dire que les points de transition de la sortie sont compensés au-dessous et au-dessus du point de consigne par la moitié de la valeur *On/Off Hystérésis*. Cette valeur affecte les sorties programmées sur Inversé ou Direct. Lors de l'autotune, le contrôleur gère le processus par le biais de 4 cycles d'activation/de désactivation. Il est donc important de définir *On/Off Hystérésis* sur une valeur appropriée avant d'initialiser l'autotune.

- La propriété *On/Off Bande morte* permet de compenser les points d'activation des sorties inversées et directes programmées pour une opération d'activation/de désactivation. Cela provoque une bande morte si la valeur est positive et un chevauchement si la valeur est négative. Lorsque vous déterminez les points de transition réels des sorties, la valeur *On/Off Hystérésis* doit également être prise en compte.

PARAMETRES DE L'AUTOTUNE

- La propriété *Réponse Autotune* est utilisée pour garantir qu'un autotune produit les valeurs P, I et D optimales pour différentes applications. Un paramètre Très agressive entraîne la définition du PID qui atteint le point de consigne aussi vite que possible, sans aucune préoccupation pour le dépassement alors qu'un paramètre Très conservatrice sacrifie la vitesse pour empêcher tout dépassement.

  Remarque : si la propriété *Réponse Autotune* est modifiée, l'autotune doit être réinitialisé pour que ces modifications puissent influer sur les paramètres PID. Pour en savoir plus, reportez-vous à la section Autotune.

PARAMETRES PID UTILISATEUR

- La propriété *Bande Proportionnelle*, saisie comme un pourcentage de toute la plage d'entrée, est la quantité de modifications d'entrées nécessaires pour varier la plage de sortie complète. La plage d'entrée représente la différence entre les valeurs Limite Val. Process Min. et Val. Process Max. La propriété *Bande Proportionnelle* peut être réglée de 0,0 % à 1 000,0 % et doit être définie sur une valeur qui fournit la meilleure réponse à une perturbation du processus tout en réduisant le dépassement. Une propriété *Bande Proportionnelle* de 0,0 % force le contrôleur sur la commande d'activation/de désactivation avec son cycle caractéristique sur le point de consigne. La valeur optimale peut être définie en appelant l'autotune.

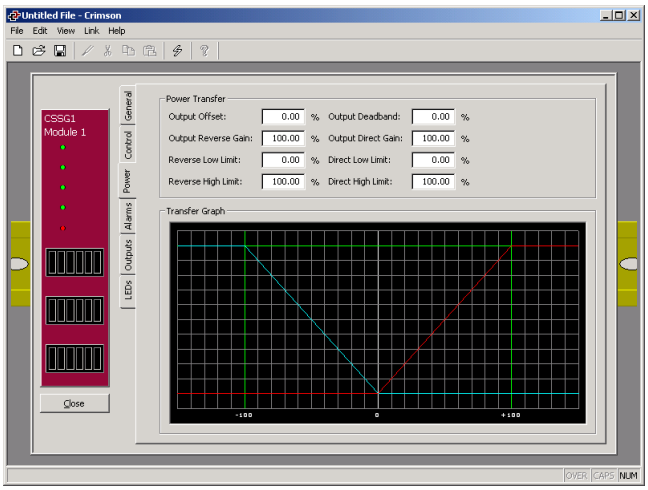
- La propriété *Temps Intégral* est le temps en secondes qui est nécessaire à l'action intégrale pour égaler l'action proportionnelle lors d'une erreur de processus constante. Aussi longtemps que l'erreur sera présente, l'action intégrale sera répétée à chaque *Temps Intégral*. Plus la valeur est importante, plus la réponse est lente. La valeur optimale peut être définie en appelant l'autotune. La propriété *Temps Intégral* peut être réglée de 0 à 6 000,0 secondes.

- La propriété *Temps Dérivé* représente les secondes par répétition que le contrôleur envisage à l'erreur de rampe pour voir ce que sera la contribution proportionnelle et elle correspond à cette valeur à chaque *Temps dérivé*. Aussi longtemps que l'erreur de rampe sera présente, la contribution dérivée est répétée à chaque heure dérivée. L'augmentation de la valeur permet de stabiliser la réponse, mais si cette valeur est trop élevée et que les processus des signaux sont trop bruyants, la sortie pourra trop varier, provoquant une commande médiocre. La définition de l'heure sur zéro désactive l'action dérivée. La propriété optimale

Temps Dérivé peut être définie en appelant l'autotune. La valeur peut être réglée de 0 à 600,0 secondes.

- La propriété *Filtre de puissance* est une constante de temps, entrée en secondes, qui bloque la puissance de sortie calculée. L'augmentation de la valeur optimise l'effet de blocage. En règle générale, une propriété *Filtre de puissance* dans la plage d'un vingtième à un cinquantième du temps intégral du contrôleur (ou de la constante de temps du processus) est effective. Les valeurs supérieures à celles-ci peuvent provoquer l'instabilité du contrôleur du fait de l'ajout de l'effet de décalage.

L'ONGLET PUISSANCE



TRANSFERT DE PUISSANCE

- La valeur *Décalage de la sortie (Offset)* modifie le point de sortie zéro du calcul de la puissance de sortie du module. Cette fonctionnalité est plus couramment utilisée dans les applications uniquement proportionnelles afin de supprimer l'erreur permanente.

- La propriété *Bande Morte de la sortie* définit la zone dans laquelle les sorties inversées et directes sont inactives (plus connue sous le nom de bande morte) ou la zone dans laquelle elles sont actives (plus connue sous le nom de chevauchement). Une valeur positive entraînera une bande morte alors qu'une valeur négative donnera un chevauchement.

- La propriété *Gain en Sortie Inversé* définit le gain de la sortie inversée relatif au gain défini par la propriété *Bande Proportionnelle*. Avec une valeur de 100 %, le gain inversé imite le gain déterminé par la bande proportionnelle. Une valeur inférieure à 100 % peut être utilisée dans les applications où le dispositif de sortie est trop puissant alors qu'une valeur supérieure à 100 % peut être utilisée lorsque le dispositif de sortie est insuffisant. Pour la majorité des applications, la valeur par défaut de 100 % est adaptée et les réglages doivent uniquement être effectués si le processus le nécessite.

- La propriété *Gain en Sortie Direct* définit le gain de la sortie directe relative au gain défini par la propriété *Bande Proportionnelle*. Avec une valeur de 100 %, le gain direct imite le gain déterminé par la bande proportionnelle. Une valeur inférieure à 100 % peut être utilisée dans les applications où le dispositif de sortie est trop puissant alors qu'une valeur supérieure à 100 % peut être utilisée lorsque le dispositif de sortie est insuffisant. Pour la majorité des applications, la valeur par défaut de 100 % est adaptée et les réglages doivent uniquement être effectués si le processus le nécessite.

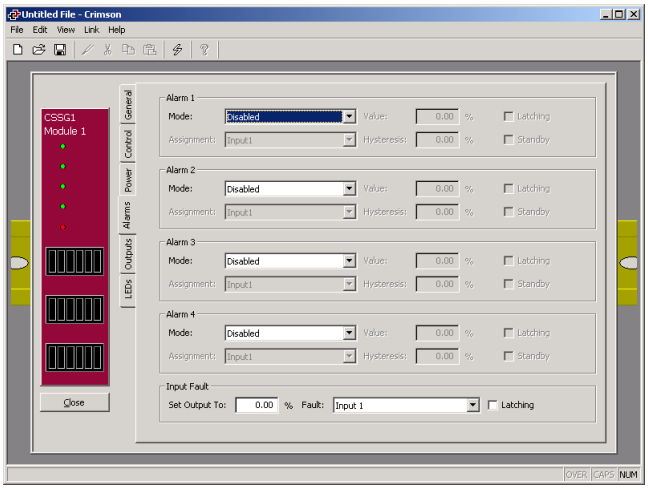
- Les propriétés *Limite Min. Inversée* et *Limite Max. Inversée* peuvent être utilisées pour limiter la puissance du contrôleur lors de perturbations du processus ou de modifications du point de consigne. Saisissez les limites de puissance de sortie correctes du processus. Vous pouvez saisir des valeurs supérieures à 100 % et –100 % pour surmonter les limitations provoquées par les valeurs du transfert de puissance comme les gains et les décalages qui limiteraient la sortie à des valeurs inférieures à leur maximum.

- Les propriétés *Limite Min. Directe* et *Limite Max. Directe* peuvent être utilisées pour limiter la puissance du contrôleur lors de perturbations du processus ou de modifications du point de consigne. Saisissez les limites de puissance de sortie correctes du processus. Vous pouvez saisir des valeurs supérieures à 100 % et – 100 % pour surmonter les limitations provoquées par les valeurs du transfert de puissance comme les gains et les décalages qui limiteraient la sortie à des valeurs inférieures à leur maximum.

GRAPHIQUE DE TRANSFERT

Le graphique de transfert de puissance illustre les résultats des modifications apportées aux paramètres de la puissance. La ligne bleue représente la sortie directe alors que la ligne rouge représente la sortie inversée.

L'ONGLET ALARMES



AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

Vous pouvez utiliser les quatre alarmes du processus « flexibles » pour contrôler l'état du processus et pour mettre en marche les sorties physiques du module. Sinon, vous pouvez contrôler seulement le bit via le système lui-même ou via des périphériques externes.

- La propriété *Mode* détermine le comportement de l'alarme. Le tableau ci-dessous décrit les différentes sélections.

MODE	**DESCRIPTION**
Inférieur à	L'alarme s'active lorsque la valeur mesurée est inférieure à la valeur de l'alarme. L'alarme se désactive lorsque la valeur mesurée est supérieure à la valeur de l'alarme + l'hystérésis.
Supérieur à	L'alarme s'active lorsque la valeur mesurée est supérieure à la valeur de l'alarme. L'alarme se désactive lorsque la valeur mesurée est inférieure à la valeur de l'alarme – l'hystérésis.
Déviation basse	Si la valeur mesurée est inférieure à la valeur du point de consigne par la quantité de la valeur de l'alarme, l'alarme s'active. Dans ce mode, le point de l'alarme suit la valeur du point de consigne.
Déviation haute	Si la valeur mesurée est supérieure à la valeur du point de consigne par la quantité de la valeur de l'alarme, l'alarme s'active. Dans ce mode, le point de l'alarme suit la valeur du point de consigne.
Dans la bande	Si la différence entre la valeur du point de consigne et la valeur mesurée est inférieure à la valeur de l'alarme, l'alarme s'active.
Hors de la bande	Si la valeur mesurée est supérieure ou inférieure à la valeur du point de consigne par une quantité égale à la valeur de l'alarme, l'alarme s'active. Dans ce mode, le point de l'alarme suit la valeur du point de consigne.

- La propriété *Attribution* modifie la valeur que les alarmes contrôlent et à laquelle elles réagissent.
- La propriété *Valeur* représente le point où l'alarme s'active. Les valeurs de l'alarme sont entrées dans les mêmes unités de mesure que celles qui sont utilisées pour mettre à l'échelle la variable choisie dans la propriété Assignment.
- La valeur *Hystérésis* sépare les points d'activation et de désactivation de l'alarme, c'est-à-dire qu'une alarme à puissance élevée, programmée pour se mettre en marche à 500 avec une *Hystérésis* de 10 s'éteint lorsque la valeur du processus est inférieure à 490.

- La propriété *Maintenu* dicte la façon dont l'alarme se comporte une fois qu'elle est activée. Pour en savoir plus, reportez-vous à la section ci-dessous, intitulée Tableau du comportement de l'alarme.
- La propriété *Inhibition sur : Seuil/Alim* permet d'empêcher l'apparition de ce qu'on appelle les alarmes de nuisance lors de la mise en marche. Pour en savoir plus, reportez-vous à la section ci-dessous, intitulée Tableau du comportement de l'alarme.

TABLEAU DU COMPORTEMENT DE L'ALARME

MAINTENU	**INHIBITION SUR : SEUIL/ALIM**	**COMPORTEMENT DE L'ALARME**
		L'alarme est activée ou désactivée automatiquement lorsque la valeur mesurée croise la région de l'alarme. Le bit AlarmAccept désactive l'alarme, quel que soit l'état du processus. Si la condition d'alarme existe et si le bit est écrit sur « 0 », l'alarme s'active.
☑		Une fois que l'alarme est activée, elle reste active jusqu'à ce qu'elle soit acceptée. Si la condition d'alarme n'existe plus, le fait d'écrire le bit AlarmAccept sur « 1 » réinitialise la condition d'alarme. Tant que le bit AlarmAccept est « 1 », l'alarme est activée ou désactivée automatiquement lorsque la valeur mesurée croise la région de l'alarme.
	☑	L'alarme est activée ou désactivée automatiquement lorsque la valeur mesurée croise la région de l'alarme. L'alarme est automatiquement désactivée lors d'une modification du point de consigne ou lors de la première mise en marche du module. Ainsi, aucune alarme de nuisance ne se produit. L'alarme reste désactivée jusqu'à ce que le processus entre dans un état de non-alarme. La prochaine fois qu'une valeur mesurée entrera dans une condition d'alarme, l'alarme s'activera en conséquence. Le bit AlarmAccept désactive l'alarme, quel que soit l'état. Si la condition d'alarme existe et si le bit est écrit sur « 0 », l'alarme s'active.

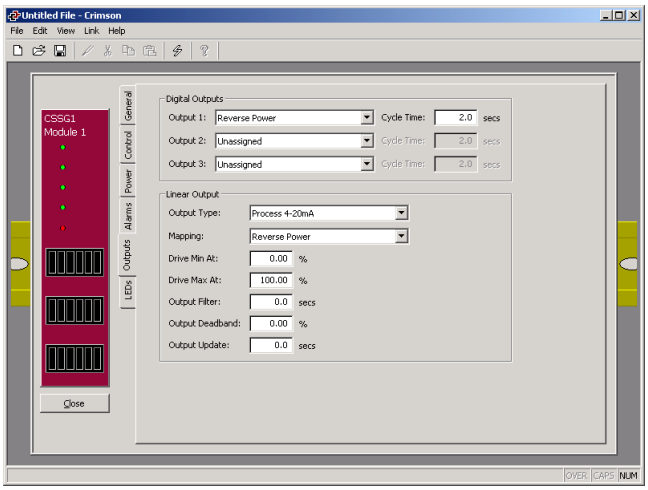
MAINTENU	INHIBITION SUR : SEUIL/ALIM	COMPORTEMENT DE L'ALARME
☑	☑	Une fois l'alarme activée, elle reste active jusqu'à ce qu'elle soit acceptée. L'alarme est automatiquement désactivée lors d'une modification du point de consigne ou lors de la première mise en marche du module. Ainsi, aucune alarme de nuisance ne se produit. L'alarme reste désactivée jusqu'à ce que le processus entre dans un état de non-alarme. La prochaine fois qu'une valeur mesurée entrera dans une condition d'alarme, l'alarme s'activera en conséquence. L'écriture momentanée du bit AlarmAccept sur « 1 » désactive une alarme active. Si la condition d'alarme existe encore, l'alarme reste désactivée et elle est mise en mode Inhibition sur : Seuil/Alim. Ainsi, l'alarme reste désactivée jusqu'à la fin de la condition d'alarme, puis elle est réentrée. Si le bit AlarmAccept reste à « 1 », l'alarme est désactivée et ne fonctionne pas.

ERREUR D'ENTRÉE

La section Erreur d'entrée permet de définir la réponse des sorties de commande du module CSSG en cas d'erreur d'entrée et/ou lorsque la valeur du processus dépasse les valeurs Val. Process Min. ou Val. Process Max. L'alarme de l'erreur d'entrée est considérée comme une alarme de processus pour les éléments qui peuvent être mappés sur « n'importe quelle alarme de process ».

- La propriété *Puissance* représente la valeur de sortie du contrôleur en cas d'un échec du capteur d'entrée. Les valeurs au-delà de 100 % et –100 % peuvent être saisies pour compenser les limitations provoquées par les valeurs du transfert de puissance comme les gains et les décalages qui limiteraient les sorties à des valeurs inférieures à leur maximum.
- La propriété *Défaut* dicte si un échec de capteur sur l'entrée 1 ou sur toute entrée est requis pour attribuer à la sortie la valeur Puissance.
- Si la propriété *Maintenu* est activée, elle permettra aux bits de l'erreur d'entrée de rester actifs jusqu'à ce qu'elle soit acceptée, quel que soit l'état de l'entrée ou des entrées. Pour accepter l'erreur, le bit Accept doit être écrit sur 1. Si *Maintenu* n'est pas sélectionné, l'erreur ou les erreurs se désactivent dès la correction de l'échec ou des échecs d'entrée.

L'ONGLET SORTIES



SORTIES NUMERIQUES

- Les propriétés *Sortie n* permettent d'attribuer les sorties physiques du module aux différentes propriétés ou valeurs internes.
- La valeur *Temps de cycle* est la somme des cycles d'activation/de désactivation d'une sortie proportionnelle au temps. Avec des sorties proportionnelles au temps, le pourcentage de la puissance de sortie est converti en sortie en temps de la valeur temporelle du cycle. Par exemple : si l'algorithme du contrôleur nécessite une puissance de 65 % et possède une propriété *Temps de cycle* de 10 secondes, la sortie sera activée pendant 6,5 secondes et désactivée pendant 3,5 secondes. Une propriété *Temps de cycle* égale ou inférieure à un dixième de la constante de temps du processus est conseillée.

SORTIE LINEAIRE

- La propriété *Type de sortie* permet de sélectionner une sortie 0 à 10 V, 0 à 20 mA ou 4 à 20 mA. Assurez-vous que les bretelles de sortie, situées sur le côté du module, sont définies pour le même type de sortie.
- La propriété *Mapping* permet d'attribuer la sortie analogique du module à l'une des différentes propriétés ou valeurs internes.
- Les valeurs *Minimum à* et *Maximum à* permettent de mettre à l'échelle la sortie analogique. Les unités exprimées sont identiques à celles de la valeur *Mapping*. Par conséquent, les limites numériques varient. Exemple : les valeurs de 0 % et 100 % sont généralement utilisées pour contrôler un processus.
- La propriété *Filtre de sortie* est une constante de temps, saisie en secondes, qui bloque la réponse de la sortie analogique. L'augmentation de la valeur augmente l'effet de blocage.
- La valeur *Bande Morte de la sortie* permet d'empêcher la modification de la sortie analogique lorsque de simples petits réglages sont nécessaires. Cela

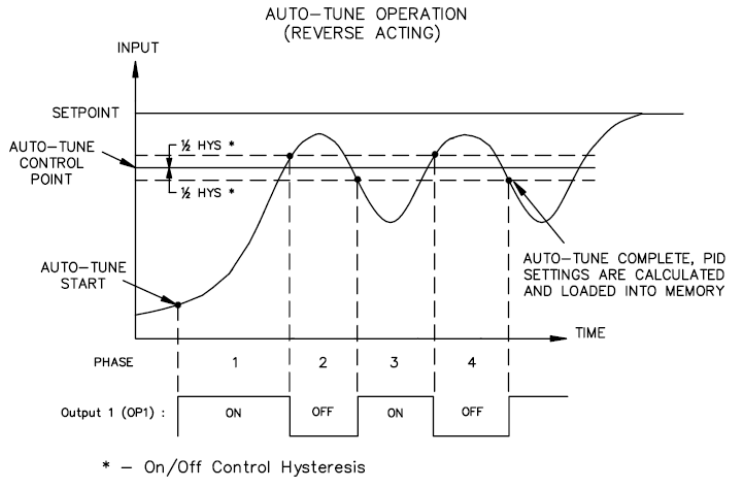
permet de prévenir l'usure mécanique lors de la manipulation d'une valve d'entrée linéaire. La sortie analogique ne se règle pas sauf si la modification nécessaire est supérieure de moitié à la valeur de la bande morte, c'est-à-dire qu'avec une bande morte de 10 % et une valeur de sortie de 50 %, la sortie n'est pas modifiée tant que les valeurs 45 % ou 55 % ne sont pas appelées. Les unités exprimées sont identiques à celles de la valeur Mapping.

- La temporisation *Mise à jour* de la sortie permet de diminuer la fréquence de mise à jour de la sortie analogique.
- Lorsque la temporisation *Mise à jour de la sortie* expire, la sortie analogique vérifie que la modification requise est supérieure à la valeur *Bande Morte de la sortie*. Si tel est le cas, la sortie renvoie la nouvelle valeur. Dans le cas contraire, la sortie est identique et le minuteur redémarre.

AUTOTUNE

PRESENTATION

L'autotune permet de définir les valeurs optimales P, I, D, et Filtre de puissance. En gérant le processus par le biais de 4 cycles d'activation/de désactivation, le module obtient des informations sur le processus et détermine les meilleures valeurs.



Comme la figure ci-dessus le montre, le point de consigne utilisé lors de l'autotune représente la valeur 75 % au-dessus de la différence entre la valeur du processus actuelle et le point de consigne. Cela permet aux oscillations de se produire près du point de consigne tout en évitant des dépassements excessifs. Comme le module commande d'activation/de désactivation lors de l'autotune, il est important de définir une valeur d'hystérésis d'activation/de désactivation adaptée avant d'invoquer l'autotune.

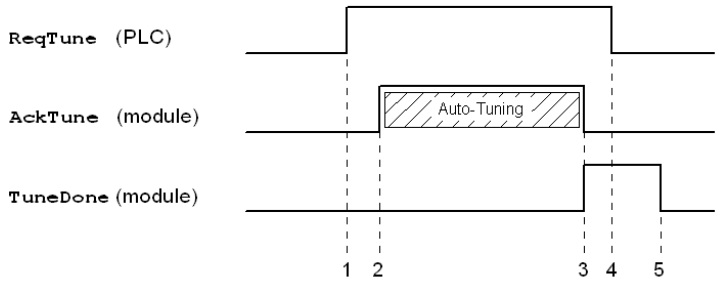
La personnalisation du PID pour définir l'activation de l'autotune est possible en réglant le paramètre Réponse Autotune. Ce paramètre peut être défini sur Très agressive, Agressive, Par défaut, Conservatrice ou Très conservatrice. De plus, il peut être réglé en écrivant 0-4 respectivement sur le registre Réponse Autotune.

RECOURS A L'AUTOTUNE

La séquence de l'autotune utilise un protocole de communication de requête/accusé de réception. Pour appeler l'autotune, écrivez le bit **ReqTune** sur 1. Le module indique que l'autotune s'exécute en définissant le bit **AckTune** sur élevé. Une fois l'autotune terminé, le bit **TuneDone** augmente. La logique externe devrait être écrite pour désactiver le bit de requête de l'autotune lorsque le bit **TuneDone** augmente. à ce niveau, le module redéfinit le bit **AckTune** sur 0.

Si, pour quelque raison que ce soit, l'autotune échoue, les bits **TuneDone** et **TuneFail** augmentent tous les deux. Cette situation peut se produire si, par exemple, une erreur d'entrée a eu lieu. Elle nécessitera alors la réinitialisation de l'autotune.

Une requête d'autotune ressemble à ce qui suit…



1. L'API définit le bit **ReqTune** sur élevé.
2. Le module démarre l'autotune et définit le bit **AckTune** sur élevé.
3. L'autotune est terminé. Le bit **AckTune** diminue et le bit **TuneDone** augmente.
4. L'API voit le bit **TuneDone** augmenter et définit le bit **ReqTune** sur bas.
5. Le module voit le bit **ReqTune** diminuer et définit le bit **TuneDone** sur bas.

DONNEES DISPONIBLES

Les données suivantes représentent les valeurs de données disponibles pour le maître et par conséquent, elles peuvent être mappées sur les registres de l'API. Les décimales sont utilisées pour indiquer uniquement la résolution et ne sont ni lues, ni écrites. Par exemple, une valeur **`Power`** de 10 000 est interprétée comme 100,00 %.

BOUCLE/STATUT

DONNEES	DESCRIPTION	GAMME	ACCES
PV	Valeur du processus – La valeur contrôlée par la boucle PID.	*	L
Input1	Valeur mise à l'échelle de l'entrée 1	*	L
Input2	Valeur mise à l'échelle de l'entrée 2	*	L
Output	Puissance de sortie – Puissance de sortie calculée de la boucle PID avant le gain, les décalages et les limites	+/- 200,00 %	L
RevPower	Sortie appliquée aux canaux auxquels est attribuée la propriété Inversé	0 – 100,00 %	L
DirPower	Sortie appliquée aux canaux auxquels est attribuée la propriété Direct	0 – 100,00 %	L
ActSP	Point de consigne réel	*	L
Error	Différence entre la valeur du processus et la valeur du point de consigne requis	*	L
AckManual	Accusé de réception du mode manuel	0 ou 1 (bit)	L
AckTune	Accusé de réception de la requête de l'autotune	0 ou 1 (bit)	L
TuneDone	Fin de l'autotune	0 ou 1 (bit)	L
TuneFail	Echec de la fin de l'autotune	0 ou 1 (bit)	L
Alarm1	Statut de l'alarme 1	0 ou 1 (bit)	L
Alarm2	Statut de l'alarme 2	0 ou 1 (bit)	L
Alarm3	Statut de l'alarme 3	0 ou 1 (bit)	L
Alarm4	Statut de l'alarme 4	0 ou 1 (bit)	L
PVAlarm	Valeur du processus hors de la plage	0 ou 1 (bit)	L
Inp1Alarm	Entrée 1 hors de la plage	0 ou 1 (bit)	L
Inp2Alarm	Entrée 2 hors de la plage	0 ou 1 (bit)	L

BOUCLE/COMMANDE

DONNEES	DESCRIPTION	GAMME	ACCES
ReqSP	Point de consigne requis – La valeur du point de consigne écrite dans le contrôleur. Cette valeur peut être différente par rapport au point de consigne réel dans des applications utilisant la rampe du point de consigne.	*	L/E
Power	Paramètre de la puissance de sortie manuelle	+/- 200,00 %	L/E
SetHyst	Hystérésis du point de consigne de la commande d'activation/de désactivation	*	L/E
SetDead	Bande morte du point de consigne de la commande d'activation/de désactivation	*	L/E
SetRamp	Rampe d'accélération et de décélération du point de consigne	*	L/E
InputFilter	Filtre d'entrée	0 – 60,0 sec.	L/E
ReqManual	Demande du mode manuel – écrivez ce bit sur 1 pour appeler le mode manuel. En mode manuel, l'écriture sur le registre Puissance contrôle la puissance de sortie.	0 ou 1 (bit)	L/E
ReqTune	Demande d'autotune – écrivez ce bit sur 1 pour appeler l'autotune.	0 ou 1 (bit)	L/E
ReqUserPID	Demande de définition du PID utilisateur – Elevée charge les valeurs d'utilisateur dans le groupe Actif et Basse charge les résultats de l'autotune dans le groupe Actif.	0 ou 1 (bit)	L/E

BOUCLE/ALARMES

DONNEES	DESCRIPTION	GAMME	ACCES
AlarmData1	Valeur de l'alarme 1	*	L/E
AlarmData2	Valeur de l'alarme 2	*	L/E
AlarmData3	Valeur de l'alarme 3	*	L/E
AlarmData4	Valeur de l'alarme 4	*	L/E
AlarmHyst1	Valeur de l'hystérésis de l'alarme 1	*	L/E
AlarmHyst2	Valeur de l'hystérésis de l'alarme 2	*	L/E
AlarmHyst3	Valeur de l'hystérésis de l'alarme 3	*	L/E
AlarmHyst4	Valeur de l'hystérésis de l'alarme 4	*	L/E
AlarmAccept1	Bit d'acquittement de l'alarme 1	0 ou 1 (bit)	L/E
AlarmAccept2	Bit d'acquittement de l'alarme 2	0 ou 1 (bit)	L/E
AlarmAccept3	Bit d'acquittement de l'alarme 3	0 ou 1 (bit)	L/E
AlarmAccept4	Bit d'acquittement de l'alarme 4	0 ou 1 (bit)	L/E

DONNEES	DESCRIPTION	GAMME	ACCES
InputAccept	Acquittement de l'alarme d'entrée(s) hors de la plage	0 ou 1 (bit)	L/E

BOUCLE/PID

DONNEES	DESCRIPTION	GAMME	ACCES
TuneCode	Code de réponse de l'autotune	0-4	L/E
UserConstP	Valeur proportionnelle de l'utilisateur	0 – 1 000,0 %	L/E
UserConstI	Valeur intégrale de l'utilisateur	0 – 6 000,0 sec.	L/E
UserConstD	Valeur dérivée de l'utilisateur	0 – 600,0 sec.	L/E
UserFilter	Valeur du filtre de puissance de l'utilisateur	0 – 60,0 sec.	L/E
AutoConstP	Valeur proportionnelle réglée automatiquement	0 – 1 000,0 %	L
AutoConstI	Valeur intégrale réglée automatiquement	0 – 6 000,0 sec.	L
AutoConstD	Valeur dérivée réglée automatiquement	0 – 600,0 sec.	L
AutoFilter	Valeur du filtre de puissance réglée automatiquement	0 – 60,0 sec.	L
ActConstP	Valeur proportionnelle active	0 – 1 000,0 %	L
ActConstI	Valeur intégrale active	0 – 6 000,0 sec.	L
ActConstD	Valeur dérivée active	0 – 600,0 sec.	L
ActFilter	Valeur du filtre de puissance active	0 – 60,0 sec.	L

BOUCLE/PUISSANCE

DONNEES	DESCRIPTION	GAMME	ACCES
PowerFault	Puissance de sortie pour l'erreur d'entrée	+/- 200,00 %	L/E
PowerOffset	Valeur du décalage de la puissance de sortie	+/- 100,00 %	L/E
PowerDead	Valeur de la bande morte de la puissance de sortie	+/- 100,00 %	L/E
PowerRevGain	Valeur du gain inversé de la puissance de sortie	0 – 500,00 %	L/E
PowerDirGain	Valeur du gain direct de la puissance de sortie	0 – 500,00 %	L/E
PowerRevHyst	Hystérésis inversée de SmartOnOff de la puissance de sortie	0 – 50,00 %	L/E
PowerDirHyst	Hystérésis directe de SmartOnOff de la puissance de sortie	0 – 50,00 %	L/E
RevLimitLo	Limite minimale inversée	0 – 200,00 %	L/E
RevLimitHi	Limite maximale inversée	0 – 200,00 %	L/E
DirLimitLo	Limite minimale directe	0 – 200,00 %	L/E
DirLimitHi	Limite maximale directe	0 – 200,00 %	L/E

BOUCLE/SCALEINPUT1

DONNEES	DESCRIPTION	GAMME	ACCES
DispLo1	Valeur inférieure de l'affichage de l'entrée 1	+/-30 000	L/E
DispHi1	Valeur supérieure de l'affichage de l'entrée 1	+/-30 000	L/E
SiglLoKey1	Signal min. de l'entrée 1 – valeur entrée (saisie)	+/-30 000	L/E
SigHiKey1	Signal maximum de l'entrée 1 – valeur entrée (saisie)	+/-30 000	L/E
SigLoApp1	Signal minimum de l'entrée 1 – valeur appliquée	+/-30 000	L
SigHiApp1	Signal maximum de l'entrée 1 – valeur appliquée	+/-30 000	L
StoreSigLo1	Enregistrement du signal minimum de l'entrée 1 (déclenchement par front d'impulsion) – sur le front positif, le signal en millivolts appliqué à l'entrée 1 est enregistré sous SigLoApp1	0 ou 1 (bit)	L/E
StoreSigHi1	Enregistrement du signal maximum de l'entrée 1 (déclenchement par front d'impulsion) – sur le front positif, le signal en millivolts appliqué à l'entrée 1 est enregistré sous SigHiApp1	0 ou 1 (bit)	L/E
SelectScaling1	Sélection des signaux appliqués à l'entrée 1 (déclenchement par le niveau d'impulsion) – lorsque les valeurs du signal appliqué sont élevées, elles sont actives ; lorsqu'elles sont faibles, les valeurs saisies du signal sont actives	0 ou 1 (bit)	L/E

BOUCLE/SCALEINPUT2

Les données disponibles pour ScaleInput2 sont identiques à celles de ScaleInput1.

BOUCLE/PEAKVALLEY

DONNÉES	DESCRIPTION	GAMME	ACCES
PVPeak	La valeur du processus maximale, mesurée depuis la dernière réinitialisation de Peak	+/-30 000	L
PVVall	La valeur du processus minimale, mesurée depuis la dernière réinitialisation de Valley	+/-30 000	L
ResetPkVall	Réinitialisation de Peak et Valley – Ecrivez ce bit sur 1 (déclenchement par le niveau d'impulsion) pour réinitialiser les registres Peak et Valley sur leur valeur de processus existante.	0 ou 1 (bit)	L/E
ResetPeak	Réinitialisation de Peak – Ecrivez ce bit sur 1 (déclenchement par le niveau d'impulsion) pour réinitialiser le registre Peak sur sa valeur de processus existante.	0 ou 1 (bit)	L/E

DONNÉES	DESCRIPTION	GAMME	ACCES
ResetVall	Réinitialisation de Valley – Ecrivez ce bit sur 1 (déclenchement par le niveau d'impulsion) pour réinitialiser le registre Valley sur sa valeur de processus existante.	0 ou 1 (bit)	L/E

BOUCLE/TARE

DONNÉES	DESCRIPTION	GAMME	ACCES
PVGross	Valeur brute du processus ; résultat direct de l'affectation mathématique de la valeur du processus avant la tare.	+/-30 000	L
Inp1Gross	Valeur brute de l'entrée 1 ; résultat direct de la mise à l'échelle de l'entrée avant la tare.	+/-30 000	L
Inp2Gross	Valeur brute de l'entrée 2 ; résultat direct de la mise à l'échelle de l'entrée avant la tare.	+/-30 000	L
PVTareTot	Total de la tare de la valeur du processus – La somme de la tare des valeurs du processus depuis la dernière réinitialisation du total de la tare de la valeur du processus.	+/-30 000	L
Inp1TareTot	Total de la tare de l'entrée 1 – La somme de la tare de l'entrée 1 depuis la dernière réinitialisation du total de la tare de l'entrée 1.	+/-30 000	L
Inp2TareTot	Total de la tare de l'entrée 2 – La somme de la tare de l'entrée 2 depuis la dernière réinitialisation du total de la tare de l'entrée 2.	+/-30 000	L
TarePV	Valeur du processus de la tare – Ecrivez ce bit sur 1 (déclenchement par front d'impulsion) pour mettre la tare (réinitialisation) de la valeur du processus sur 0. La tare de la valeur est ajoutée au total de la tare de la valeur du processus.	0 ou 1 (bit)	L/E
TareInp1	écrivez ce bit sur 1 (déclenchement par front d'impulsion) pour mettre la tare (réinitialisation) de l'entrée 1 sur 0. La tare de l'entrée 1 est ajoutée au total de la tare de l'entrée 1.	0 ou 1 (bit)	L/E
TareInp2	écrivez ce bit sur 1 (déclenchement par front d'impulsion) pour mettre la tare (réinitialisation) de l'entrée 2 sur 0. La tare de l'entrée 2 est ajoutée au total de la tare de l'entrée 2.	0 ou 1 (bit)	L/E
RstPVTareTot	écrivez ce bit sur 1 (déclenchement par front d'impulsion) pour réinitialiser le total de la tare de la valeur du processus sur 0.	0 ou 1 (bit)	L/E
RstIn1TareTot	écrivez ce bit sur 1 (déclenchement par front d'impulsion) pour réinitialiser le total de la tare de l'entrée 1 sur 0.	0 ou 1 (bit)	L/E

DONNÉES	DESCRIPTION	GAMME	ACCES
RstIn2TareTot	Ecrivez ce bit sur 1 (déclenchement par front d'impulsion) pour réinitialiser le total de la tare de l'entrée 2 sur 0.	0 ou 1 (bit)	L/E

SORTIES/TEMPS DE CYCLE

DONNÉES	DESCRIPTION	GAMME	ACCES
CycleTime1	Temps de cycle pour la sortie 1	0,1– 60,0 sec.	L/E
CycleTime2	Temps de cycle pour la sortie 2	0,1– 60,0 sec.	L/E
CycleTime3	Temps de cycle pour la sortie 3	0,1– 60,0 sec.	L/E

SORTIES/DONNEES A DISTANCE

DONNÉES	DESCRIPTION	GAMME	ACCES
DigRemote1	Valeur numérique à distance 1 – Les sorties attribuées à la valeur numérique à distance peuvent être contrôlées en écrivant le bit DigRemote sur 1 ou 0.	0 ou 1 (bit)	L/E
DigRemote2	Valeur numérique à distance 2 – Voir DigRemote1	0 ou 1 (bit)	L/E
DigRemote3	Valeur numérique à distance 3 – Voir DigRemote1	0 ou 1 (bit)	L/E
DigRemote4	Valeur numérique à distance 4 – Voir DigRemote1	0 ou 1 (bit)	L/E
AnlRemote1	Valeur analogique à distance 1 – Les sorties attribuées à la valeur analogique peuvent être contrôlées en écrivant un chiffre à ce mot.	*	L/E
AnlRemote2	Valeur analogique à distance 2 – Voir AnlRemote1	*	L/E
AnlRemote3	Valeur analogique à distance 3 – Voir AnlRemote1	*	L/E
AnlRemote4	Valeur analogique à distance 4 – Voir AnlRemote1	*	L/E

SORTIES/INFORMATIONS

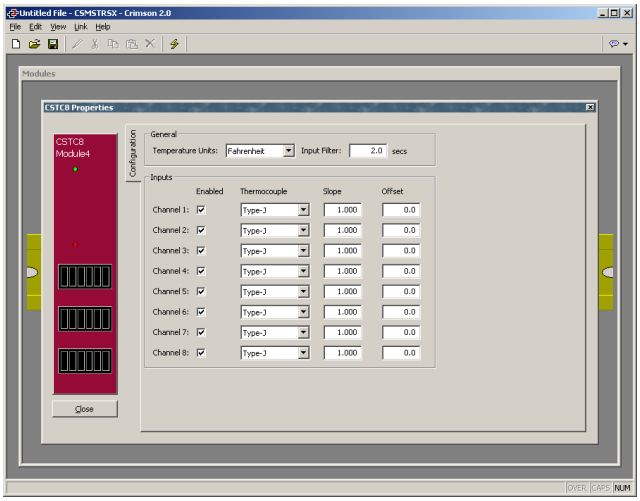
DONNÉES	DESCRIPTION	GAMME	ACCES
OP1State	état de la sortie 1	0 ou 1 (bit)	L
OP2State	état de la sortie 2	0 ou 1 (bit)	L
OP3State	Etat de la sortie 3	0 ou 1 (bit)	L

* Dépend de la configuration. La résolution dépend des valeurs d'échelle de l'utilisateur.

CSTC/CSRTD – PROGRAMMATION DU MODULE D'ENTRÉE DE LA TEMPÉRATURE

Pour accéder à la configuration d'un module, double-cliquez dessus. Tous les paramètres du module sont visibles sur une page unique.

L'ONGLET CONFIGURATION



GENERAL

Les paramètres s'appliquent à toutes les entrées du module.

- La propriété *Unités de température* est utilisée pour sélectionner l'une des échelles de température suivantes : Kelvin, Fahrenheit ou Celsius.
- La propriété *Filtre d'entrée* est une constante de temps utilisée pour stabiliser les signaux d'entrée variables.

ENTREES

Ces paramètres permettent de personnaliser de façon indépendante tous les paramètres de l'entrée.

- La propriété *Activé* permet de désactiver les entrées inutilisées, augmentant de ce fait la vitesse de lecture des entrées restantes. Pour en savoir plus sur les vitesses de lecture, reportez-vous au bulletin du matériel.
- La propriété *Thermocouple* ou *RTD (selon le modèle)* est utilisée pour sélectionner le capteur standard utilisé.
- La propriété *Pente* peut être utilisée pour modifier la proportion des valeurs du processus se rapportant à la lecture du capteur. Elle est utile dans les applications où l'erreur de capteur est non linéaire. Reportez-vous à l'exemple d'application ci-dessous.
- La propriété *Décalage* peut être utilisée pour compenser ou modifier la valeur du processus. Elle permet de personnaliser chaque entrée selon une erreur de

capteur donnée. Elle permet également de corriger la valeur du processus dans les applications où le capteur ne mesure pas directement le processus, provoquant donc une erreur. Reportez-vous à l'exemple d'application ci-dessous.

EXEMPLE D'APPLICATION

La lecture de la valeur du processus à partir d'un thermocouple est de 3 degrés inférieure à la température réelle lorsque le processus se produit à 200 degrés. Il lit seulement 1 degré de moins que la température réelle lorsque le processus se produit à 300 degrés.

Valeur du processus souhaitée = (Valeur du processus x Coefficient angulaire) + Décalage

Coefficient angulaire = $\frac{300-200}{299-197} = 0,980$

Décalage = 200 – (0,980 x 197) = 6,940

Une valeur *Pente* de 0,980 et une valeur *Décalage* de 6,940 corrigent l'erreur du capteur.

DONNEES DISPONIBLES

Les données suivantes représentent les valeurs de données disponibles pour le maître et par conséquent, elles peuvent être mappées sur les registres de l'API.

ENTREE/STATUT

DONNEES	DESCRIPTION	GAMME	ACCES
PV1 – 8	Valeur du processus – après le calcul du coefficient angulaire et du décalage	*	L
ColdJunc**	Valeur de calibration du raccordement de froid	N/A	L
InputAlarm1 – 8	Entrée hors de la plage	0 ou 1 (bit)	L

* Dépend du type de capteur sélectionné.

** Uniquement pour le modèle CSTC.

ENTREE/COMMANDE

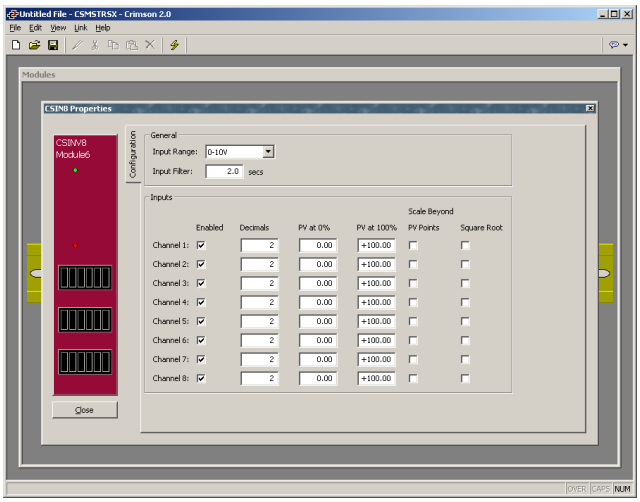
DONNEES	DESCRIPTION	GAMME	ACCES
InputFilter	Filtre d'entrée	0– 60,0 sec.	L/E
InputOffset1 – 8	Valeur de décalage ajoutée à la valeur du processus	+/- 100,0 degrés	L/E
InputSlope1 – 8	Valeur du coefficient angulaire appliquée à la valeur du processus	0,001 – 10 000	L/E

AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

CSINI/CSINV – PROGRAMMATION DU MODULE A ENTRÉE ANALOGIQUE

Pour accéder à la configuration d'un module, double-cliquez dessus.

L'ONGLET CONFIGURATION



GENERAL

Les paramètres s'appliquent à toutes les entrées du module.

- La propriété *Plage d'entrée* permet de sélectionner la plage entre 0 et 20 mA et 4 à 20 mA pour le module CSINI et entre 0 à 10 V et +/-10 V pour le module CSINV.
- La propriété *Filtre d'entrée* est une constante de temps qui sert à stabiliser les signaux d'entrée variables.

ENTREES

Ces paramètres permettent de personnaliser de façon indépendante tous les paramètres de l'entrée.

- La propriété *Activé* permet de désactiver les entrées inutilisées, augmentant de ce fait la vitesse de lecture des entrées restantes. Pour en savoir plus sur les vitesses de lecture, reportez-vous au bulletin du matériel.
- La propriété *Décimales* est utilisée pour permettre à Crimson d'afficher les unités techniques dans la bonne résolution. Elle ne sert qu'à afficher la résolution appropriée dans le logiciel, mais elle n'est pas utilisée dans le module.
- Les propriétés *Valeur Process à 0 %* et *Valeur Process à 100 %* sont utilisées pour mettre à l'échelle les signaux d'entrée. Entrez la lecture de la valeur du processus souhaitée des niveaux de signaux d'entrée minimums et maximums. Par exemple, si l'application implique un capteur de flux avec une sortie de 4 à 20 mA, proportionnelle à 5 à 105 litres par minute, sélectionnez Process 4-

20 mA comme propriété *Type d'entrée*, entrez 5 comme paramètre *Valeur Process à 0 %* et entrez 105 comme paramètre *Valeur Process à 100 %*.

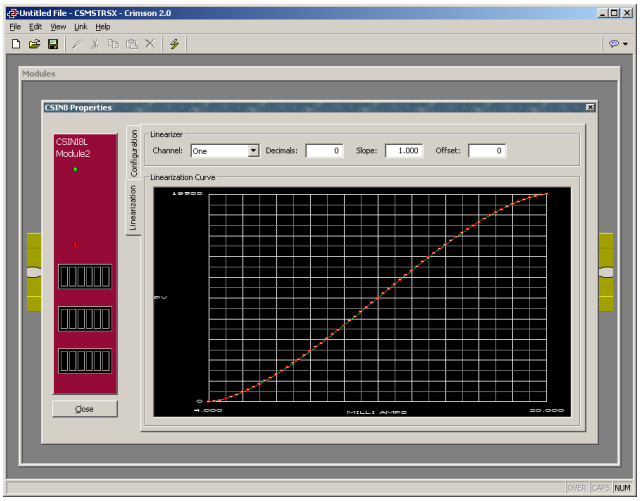
- La propriété *Mise à l'échelle au-delà de la Valeur Process* permet à la valeur du processus de continuer l'extrapolation au-delà des valeurs *Valeur Process à 0 %* et *Valeur Process à 100 %* si l'entrée dépasse ses limites habituelles. Si la propriété *Mise à l'échelle au-delà de la Valeur Process* n'est pas cochée, la valeur du processus mesure jusqu'à ses limites, même si le signal est légèrement en dehors de la plage de mesure.

- La propriété *Racine Carrée* permet d'utiliser l'unité dans des applications où le signal de mesure est le carré de la valeur du processus. Elle est utile dans les applications comme la mesure du flux avec un capteur de pression différentiel.

- La propriété *Linéarisateur* (uniquement dans les modèles de linéarisateur) permet d'utiliser l'unité dans les applications où le signal mesuré est non linéaire en ce qui concerne la valeur du processus souhaitée. Elle est utile dans les applications comme la mesure du volume matériel dans un réservoir ou un récipient circulaire et monté horizontalement, avec une sonde de niveau.

L'ONGLET LINEARISATION

Les modules CSINI8L et CSINV8L offrent une fonction de linéarisation à 100 points par canal.



LINEARISATEUR

Pour utiliser la fonction de linéarisateur pour une entrée donnée, la propriété *Linéarisateur* doit être cochée sous l'onglet Configuration.

- La valeur *Canal* sélectionne l'entrée avec laquelle vous souhaitez travailler, mais ce n'est pas vraiment une propriété.

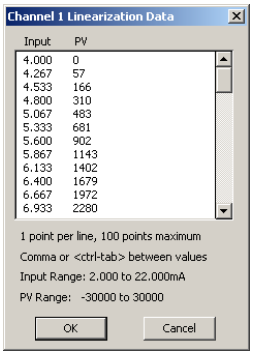
- La propriété *Décimales* est utilisée pour permettre à Crimson d'afficher les unités techniques dans la bonne résolution. Elle ne sert qu'à afficher la résolution appropriée dans le logiciel, mais elle n'est pas utilisée dans le module.

- La propriété *Pente* peut être utilisée pour modifier la proportion des valeurs du processus se rapportant à la lecture du capteur. Elle s'avère utile dans les applications où l'erreur de capteur est non linéaire, mais elle peut encore être mesurée avec une échelle simple à deux points.
- La propriété *Décalage* peut être utilisée pour compenser ou modifier la valeur du processus. Elle est utile dans les applications où l'erreur de capteur est non linéaire, mais elle peut encore être mesurée avec une échelle simple à deux points.

COURBE DE LINEARISATION

La propriété *Courbe de Linéarisation* définit la valeur du processus sur les points d'entrée. Pour entrer de nouvelles valeurs, cliquez n'importe où sur le graphique.

DONNEES DE LINEARISATION DU CANAL N



Entrez jusqu'à 100 entrées/paires de valeurs du processus pour le canal d'entrée sélectionné. Utilisez une virgule ou <Ctrl+Tabulation> pour séparer les paires, puis appuyez sur le bouton Entrée pour saisir la paire suivante. Le graphique se met à jour pour afficher le résultat des données. Les données peuvent également être copiées à partir de programmes comme MS Excel, puis collées dans la boîte de dialogue.

DONNEES DISPONIBLES

Les données suivantes représentent les valeurs de données disponibles pour le maître et par conséquent, elles peuvent être mappées sur les registres de l'API.

ENTREE/STATUT

DONNEES	DESCRIPTION	GAMME	ACCES
PV1 – 8	Valeur du processus – Mise à l'échelle selon les valeurs Valeur Process Min. et Valeur Process Max.	*	L
InputAlarm1 – 8	Entrée hors de la plage	0 ou 1 (bit)	L

* Dépend de la configuration.

ENTREE/COMMANDE

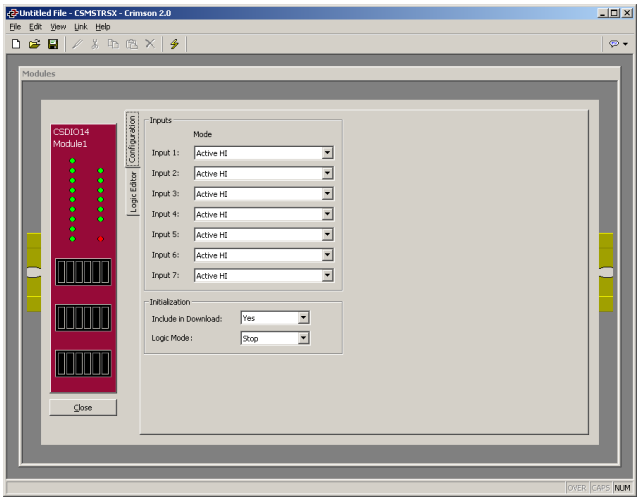
DONNEES	DESCRIPTION	GAMME	ACCES
InputFilter	Filtre d'entrée	0– 60,0 sec.	L/E
ProcessMin1 – 8	Valeur du processus souhaitée au niveau du signal d'entrée minimum	+/- 30 000	L/E
ProcessMax1 – 8	Valeur du processus souhaitée au niveau du signal d'entrée maximum	+/- 30 000	L/E
LinearSlope1 – 8**	Valeur du coefficient angulaire appliquée à la valeur du processus	0,001 – 10 000	L/E
LinearOffset1 – 8**	Valeur de décalage appliquée à la valeur du processus	+/- 30 000	L/E

** Modèles uniquement avec le linéarisateur.

CSDIO – PROGRAMMATION DU MODULE D'ENTRÉE-SORTIE NUMÉRIQUE

Pour accéder à la configuration d'un module, double-cliquez dessus.

L'ONGLET CONFIGURATION



ENTRÉES

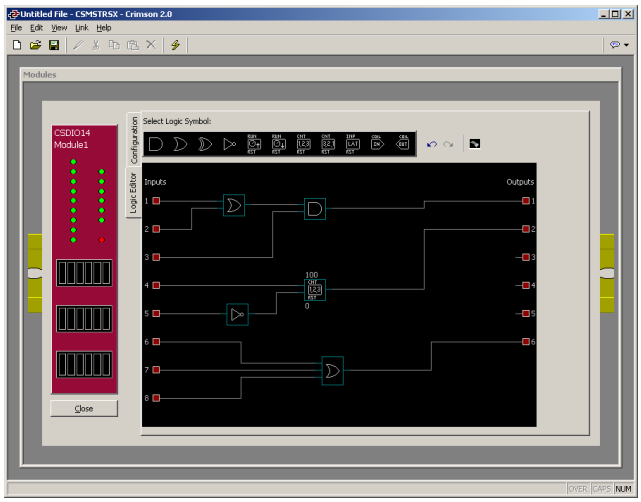
- La propriété *Mode* détermine si un état supérieur ou inférieur doit être considéré comme actif pour une entrée donnée.

INITIALISATION

- La propriété *Inclure au téléchargement* est utilisée pour déterminer si les valeurs d'initialisation sont téléchargées. En choisissant l'option « Non », vous autorisez la modification et le téléchargement des bases de données à volonté, sans écrasement accidentel de la propriété *Mode Logique*.
- La propriété *Mode Logique* permet de forcer l'état du moteur logique à s'arrêter ou à s'exécuter lors d'un téléchargement.

L'ONGLET EDITEUR LOGIQUE

Le module CSDIO peut effectuer des opérations de logique et fournir des minuteurs et compteurs aux processus nécessitant une entrée-sortie limitée. L'interface utilisateur logique est graphique, elle utilise des symboles de portes standard ainsi que des icônes uniques pour une fonctionnalité optimale. L'interface utilisateur permet également d'effectuer des simulations logiques. Pour simuler l'entrée-sortie, cliquez sur le canal d'entrée souhaité pour l'activer ou le désactiver.



SYMBOLES

Pour sélectionner un symbole, il vous suffit de cliquer dessus dans le haut de la page. Pour placer le ou les symboles sur l'écran, cliquez sur n'importe quelle partie de l'espace de travail. Une fois que vous avez terminé de disposer un symbole, cliquez dessus avec le bouton droit de la souris pour désactiver le mode de placement. Vous pouvez déplacer les symboles en les sélectionnant par un clic de souris, puis en les faisant glisser vers leur nouvel emplacement. Ensuite, en cliquant à nouveau sur la souris, vous disposez le symbole à son nouvel emplacement. Vous pouvez régler un ou plusieurs paramètres de la plupart des symboles. Double-cliquez sur le symbole en question pour accéder à ses paramètres. Pour supprimer un symbole, il vous suffit de cliquer avec le bouton droit de la souris.

FILS

Pour dessiner les fils, utilisez le bouton gauche de la souris. Il vous suffit de cliquer sur un point de raccordement, puis sur le suivant pour dessiner le raccordement. Les points de raccordement seront alors automatiquement reliés, mais parfois, vous aurez à déplacer les fils pour éviter les chevauchements. Vous pouvez déplacer les sections horizontales en repositionnant le symbole et les sections verticales en cliquant dessus. Déplacez la section vers l'endroit souhaité, puis cliquez dessus pour la positionner.

DESCRIPTIONS DES SYMBOLES

SYMBOLE	DESCRIPTION
	Le *ET* logique nécessite que toutes les entrées soient actives pour une sortie de 1. Pour ajouter d'autres entrées au symbole, double-cliquez dessus, puis entrez un nouveau chiffre.
	Le *OU* logique nécessite qu'une ou plusieurs entrées soient actives pour une sortie de 1. Pour ajouter d'autres entrées au symbole, double-cliquez dessus, puis entrez un nouveau chiffre.
	Le *OU exclusif* logique nécessite qu'une seule entrée, mais pas toutes les entrées, soit active pour une sortie de 1. Pour ajouter d'autres entrées au symbole, double-cliquez dessus, puis entrez un nouveau chiffre.
	L'*inverseur* inverse simplement l'état de l'entrée.
	Le symbole du *minuteur* fournit une entrée d'activation ainsi qu'une entrée de réinitialisation. Lorsque l'entrée d'activation est vraie, le minuteur augmente vers le point de consigne pour les minuteurs supérieurs et vers zéro pour les minuteurs inférieurs. Lorsque l'entrée d'activation est fausse, le minuteur interrompt l'augmentation. Une fois que le point de consigne ou que zéro est atteint, la sortie devient vraie. Selon le type de minuteur, une entrée de réinitialisation active provoque la réinitialisation sur zéro ou sur la valeur prédéfinie du minuteur. Pour accéder à la valeur prédéfinie du minuteur, double-cliquez sur le symbole du minuteur. Lorsque vous utilisez des minuteurs, vous devez attribuer à chacun d'entre eux un chiffre d'ID de mapping unique (1-8). Lorsque vous mappez la valeur réelle du minuteur et la valeur prédéfinie du minuteur via les communications, le chiffre d'ID de mapping correspond aux nombres variables. (Pour en savoir plus, reportez-vous au tableau Données disponibles à la fin de cette section.)
	Le symbole du *compteur* fournit une entrée de comptage ainsi qu'une entrée de réinitialisation. Le compteur augmente vers la valeur prédéfinie (compteur supérieur) ou diminue vers zéro (compteur inférieur) chaque fois que l'entrée de comptage est active. Une fois que le point de consigne ou que zéro est atteint, la sortie devient vraie. Selon le type de compteur, une entrée de réinitialisation active provoque la réinitialisation sur zéro ou sur la valeur prédéfinie du compteur. Pour accéder à la valeur prédéfinie du compteur, double-cliquez sur le symbole du compteur. Lorsque vous utilisez des compteurs, vous devez attribuer à chacun d'entre eux un chiffre d'ID de mapping unique (1-8). Lorsque vous mappez la valeur de comptage réelle et la valeur prédéfinie du compteur via les communications, le chiffre d'ID de mapping correspond aux nombres variables. (Pour en savoir plus, reportez-vous au tableau Données disponibles à la fin de cette section.)
	Le symbole du *verrou* permet de convertir une entrée par impulsion en une sortie maintenue. La sortie reste active tant que l'entrée de réinitialisation est activée.

AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - http://www.redlion.net/controller
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

SYMBOLE	DESCRIPTION
	Les bobines d'entrée et de sortie fonctionnent comme des entrées et des sorties « flexibles ». Elles permettent de fournir des signaux qui sont mappés sur des périphériques externes via les communications, sans forcément nécessiter une entrée ou une sortie physique. Vous devez attribuer à chaque bobine un chiffre d'ID de mapping unique (de 1 à 8). Lorsque vous mappez les bobines via les communications, le numéro d'ID de mapping correspond au nombre de variable. (Pour en savoir plus, reportez-vous au tableau Données disponibles à la fin de cette section.)

DONNEES DISPONIBLES

Les données suivantes représentent les valeurs de données disponibles pour le maître et par conséquent, elles peuvent être mappées sur les registres de l'API.

VARIABLES/ENTRÉES

DONNEES	DESCRIPTION	GAMME	ACCES
Input1...8	état d'entrée<0}	0 ou 1 (bit)	L
InputCoil1...8	état de la bobine d'entrée<0}	0 ou 1 (bit)	L/E

VARIABLES/SORTIES

DONNEES	DESCRIPTION	GAMME	ACCES
Output1...6	état de sortie<0}	0 ou 1 (bit)	L/E
OutputCoil1...8	état de la bobine de sortie<0}	0 ou 1 (bit)	L/E

VARIABLES/VALEURS PREDEFINIES

DONNEES	DESCRIPTION	GAMME	ACCES
CounterPreset1...8	Valeur prédéfinie du compteur	0 – 65 535	L/E
TimerPreset1...8	Valeur prédéfinie du minuteur	0– 6 553,5 sec.	L/E

VARIABLES/VALEURS

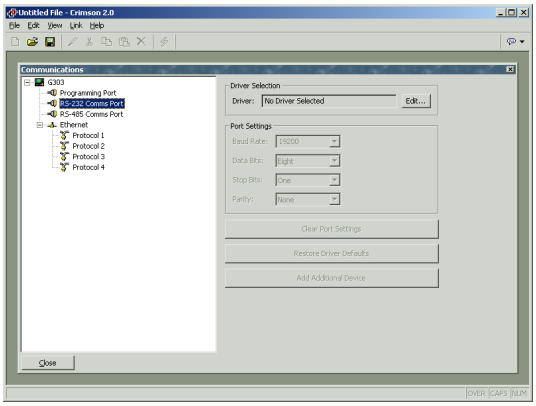
DONNEES	DESCRIPTION	GAMME	ACCES
CounterValue1...8	Valeur du compteur actuel	0 – 65 535	L
TimerValue1...8	Valeur du minuteur actuel	0– 6 553,5 sec.	L

VARIABLES/COMMANDE

DONNEES	DESCRIPTION	GAMME	ACCES
LogicHalt	Démarre/Arrête le moteur logique	0 ou 1 (bit)	L/E

CONFIGURATION DES COMMUNICATIONS

L'étape suivante est celle de la configuration des ports des communications du module maître afin d'indiquer les protocoles que vous souhaitez utiliser et les périphériques distants auxquels vous voulez accéder. Ces opérations s'effectuent à partir de la fenêtre Communications, qui s'ouvre lorsque vous sélectionnez la première icône de l'écran Crimson.



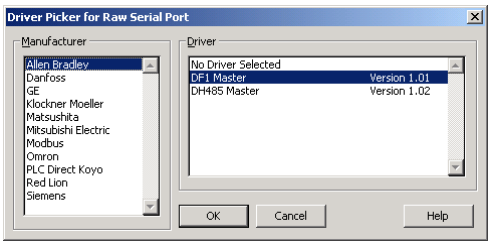
Comme vous pouvez le voir, la fenêtre Communications répertorie les ports de l'unité disponibles sous la forme d'une arborescence. Les modules maîtres sont dotés de trois ports série principaux et vous pouvez ajouter deux ports supplémentaires sous la forme d'une carte d'extension. Ils disposent également d'un port Ethernet unique qui peut exécuter simultanément quatre protocoles de communication.

UTILISATION DES PORTS SERIE

Lorsque vous choisissez les ports série du maître que vous utiliserez pour les communications, souvenez-vous que le port de programmation de l'unité peut être utilisé comme un port de communication supplémentaire et qu'il ne sera donc pas disponible pour le téléchargement si vous l'utilisez de cette façon. Cela ne représente pas un problème si le port USB est utilisé ainsi et nous vous recommandons fortement d'utiliser cette méthode de téléchargement si vous souhaitez connecter des périphériques série via le port de programmation.

SELECTION D'UN PROTOCOLE

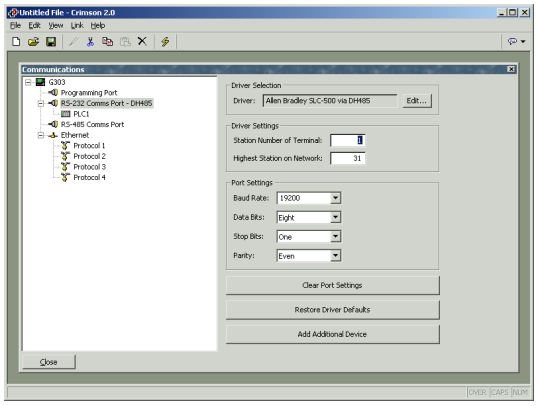
Pour sélectionner le protocole d'un port spécifique, cliquez sur l'icône de ce port dans le volet gauche de la fenêtre Communications, puis appuyez sur le bouton Edition situé en regard du champ Pilote dans le volet droit. La boîte de dialogue suivante s'affiche…



Sélectionnez le fabricant et le pilote appropriés et appuyez sur le bouton OK pour fermer la boîte de dialogue. Le port est alors configuré pour utiliser le protocole approprié et une seule icône de périphérique est créée dans le volet gauche. Si vous configurez un port série, les différents champs Paramètres du port (Débit en bauds, Bits de données, Bits d'arrêt et Parité) sont définis sur les valeurs correspondant au protocole en question. Vous devez évidemment vérifier ces paramètres pour vous assurer qu'ils correspondent aux paramètres du périphérique en question.

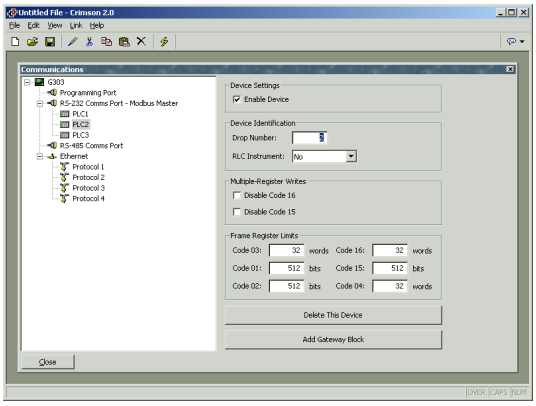
OPTIONS DU PROTOCOLE

Certains protocoles nécessitent une configuration supplémentaire des paramètres qui leur sont spécifiques. Ils s'affichent dans le volet droit de la fenêtre Communications lorsque vous sélectionnez l'icône du port correspondant. L'exemple ci-dessous montre les autres paramètres du pilote Allen-Bradley DH-485, qui s'affichent dans la fenêtre, sous la section Paramètres du pilote.



PERIPHERIQUES

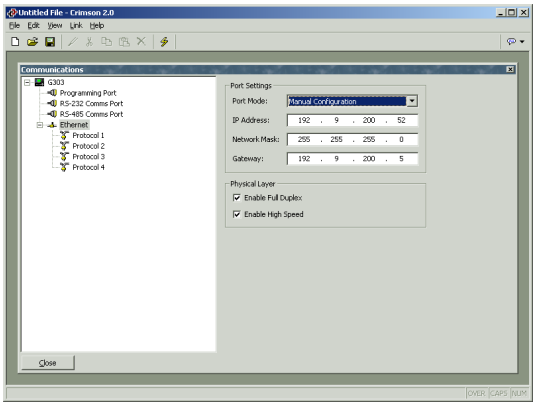
Comme nous venons de l'expliquer, lorsque vous sélectionnez un protocole de communication, un seul périphérique est créé sous l'icône du port correspondante. Dans le cas d'un protocole maître, cela représente le périphérique à distance initial qui est adressé via le protocole. Si le protocole prend en charge l'accès à plusieurs périphériques, vous pouvez utiliser le bouton Ajouter un périphérique supplémentaire qui est compris dans les propriétés de l'icône du port afin d'ajouter d'autres périphériques cibles. Chaque périphérique est représenté via une icône dans le volet gauche de la fenêtre Communications et, selon le protocole en question, peut posséder plusieurs propriétés qui doivent être configurées…



Dans l'exemple ci-dessus, le protocole Modbus Universal Master a été sélectionné et deux autres périphériques ont été créés, ce qui indique qu'il est possible d'accéder à un total de trois périphériques distants. Le volet droit de la fenêtre affiche les propriétés d'un périphérique. La propriété Activer périphérique s'affiche pour les périphériques de tous les protocoles alors que l'équilibre des champs est spécifique au protocole qui a été sélectionné. Notez que Crimson a attribué aux périphériques des noms par défaut lors de leur création. Vous pouvez modifier ces noms. Pour cela, il vous suffit de sélectionner l'icône appropriée dans le volet gauche et de taper le nouveau nom du périphérique.

CONFIGURATION DU PORT ETHERNET

Le port Ethernet du maître est configuré via l'icône Ethernet située dans le volet gauche de la fenêtre Communications. Lorsque vous sélectionnez cette icône, les paramètres suivants s'affichent…



PARAMETRES IP

Le champ Mode du Port contrôle si le port est activé ou désactivé ainsi que la méthode par laquelle le port obtient sa configuration IP. Si le mode DHCP est sélectionné, le maître tente d'obtenir une adresse IP et les paramètres associés à partir d'un serveur DHCP sur le réseau local. Si l'unité est configurée pour utiliser des protocoles esclaves ou pour servir des pages Web, cette option ne sera utile que si le serveur DHCP est configuré pour attribuer une adresse IP connue à l'adresse MAC associée à l'unité, sinon les utilisateurs ne sauront pas comment s'adresser au maître !

Si le mode plus courant Configuration manuelle est sélectionné, les champs Adresse IP, Masque réseau et Passerelle doivent être remplis avec les informations appropriées. Les valeurs par défaut fournies pour ces champs ne seront presque jamais adaptées à votre application ! Consultez votre administrateur réseau pour sélectionner les valeurs appropriées et assurez-vous que vous saisissez et téléchargez ces valeurs avant de connecter le maître à votre réseau. Si vous ne suivez pas ces conseils, il est possible (bien qu'improbable) que vous soyez à l'origine de problèmes sur le réseau.

ROUTAGE IP

Vous pouvez utiliser l'option Routage IP pour activer ou désactiver le routage des paquets IP entre le port Ethernet et toutes les connexions PPP effectuées vers ou par le maître. Vous ne devez pas activer cette option si vous n'avez pas saisi les implications liées à l'activation d'un tel routage. Pour en savoir plus, reportez-vous au chapitre Communications avancées.

COUCHE PHYSIQUE

Les options de la couche physique contrôlent le type de connexion que le maître essaie de négocier avec le hub auquel il est connecté. En règle générale, vous pouvez laisser ces options à leur état par défaut, mais si vous rencontrez des problèmes lors de la création d'une connexion fiable (principalement lorsque vous vous connectez directement à un PC sans l'intervention d'un hub ou d'un switch), essayez de désactiver les opérations Full Duplex et 10/100 Mbps pour voir si elles résolvent le problème.

MISE A JOUR A DISTANCE

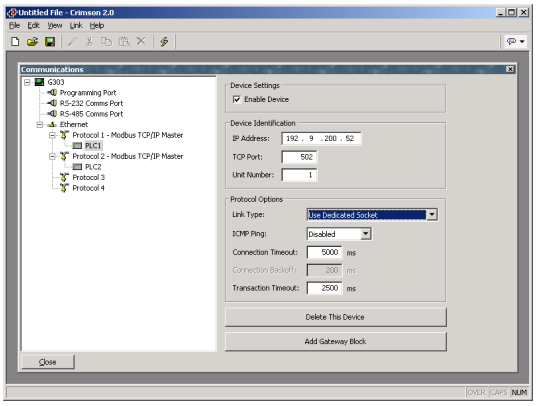
Vous pouvez utiliser l'option Programmation à Distance pour activer ou désactiver la possibilité de reprogrammer à distance l'unité (firmware y compris) et la configuration via le protocole TCP/IP. Comme nous l'avons indiqué précédemment, les mises à jour distantes du firmware sur TCP/IP nécessitent que les unités soient dotées d'une carte CompactFlash. Comme les téléchargements impliqueront certainement une mise à jour de firmware à un moment ou un autre, une telle carte est fortement conseillée lorsque vous utilisez cette fonctionnalité.

SELECTION DU PROTOCOLE

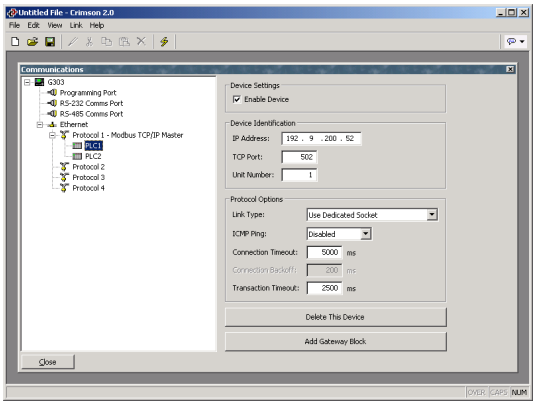
Une fois que vous avez configuré le port Ethernet, vous pouvez sélectionner les protocoles que vous souhaitez utiliser pour vos communications (Protocole1, 2, 3 ou 4). Vous pouvez utiliser simultanément jusqu'à quatre protocoles et de nombreux protocoles prennent en

charge plusieurs périphériques distants. C'est-à-dire que plusieurs options s'offrent à vous lorsque vous décidez de la façon d'associer les protocoles et les périphériques afin d'obtenir les résultats que vous souhaitez.

Supposez par exemple que vous vouliez vous connecter à deux périphériques esclaves à l'aide de Modbus sur TCP/IP. Votre première option consiste à utiliser deux sockets (protocole1 et protocole2) du port Ethernet et à les configurer comme deux maîtres TCP/IP Modbus, avec un seul périphérique connecté à chaque protocole...



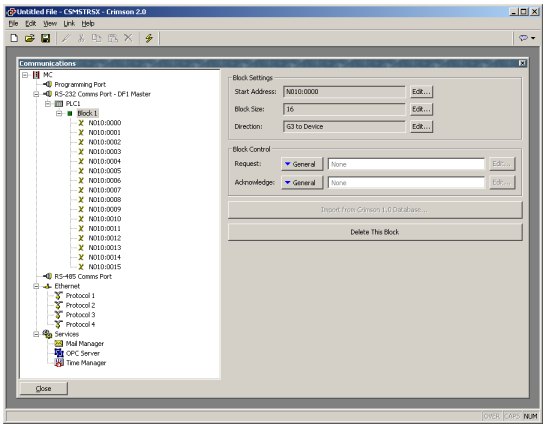
Pour la plupart des protocoles, cette option permettra d'obtenir des performances supérieures car les communications simultanées seront autorisées avec les deux périphériques. Toutefois, cela consommera deux des quatre protocoles, limitant ainsi votre capacité à vous connecter via d'autres protocoles dans des applications complexes. La seconde option consiste donc à utiliser un seul socket (Protocole1) configuré comme un maître TCP/IP Modbus et à ajouter un autre périphérique de façon à ce que vous puissiez accéder aux deux esclaves via le même pilote…



Cette option offrira des performances légèrement réduites car Crimson interrogera chaque périphérique l'un après l'autre, au lieu de communiquer avec les deux simultanément. Cependant, elle permettra de conserver les sockets Ethernet, autorisant d'autres applications complexes sans manquer de ressources.

MAPPING DES DONNEES

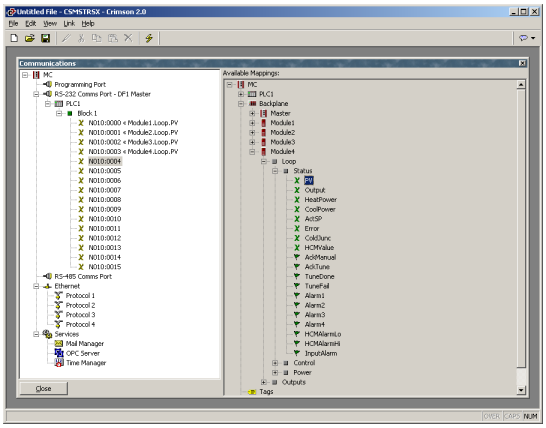
Une fois que vous avez configuré le protocole, vous devez choisir la plage d'adresses que vous souhaitez utiliser dans le périphérique de destination. Dans cet exemple, vous souhaitez que le maître lise et écrive des données provenant des cartes du Modular Controller dans un API Allen Bradley via DF1. Commencez par sélectionner l'icône de périphérique PLC1 (ou le nom de votre système) dans le volet gauche de la fenêtre Communications, puis cliquez sur le bouton Ajouter bloc de passerelle dans le volet droit. Une icône représentant le bloc 1 s'affiche. Si vous la sélectionnez, les paramètres suivants s'affichent…



Dans l'exemple ci-dessus, « Adresse de départ » a été configurée sur **`N010:0000`** pour indiquer le début du bloc et la taille de 16 indique le nombre de registres sur lesquels nous voulons mapper les données. Enfin, nous avons configuré la propriété « Direction » comme Contrôleur Modulaire à Périphérique pour indiquer que nous souhaitons que le Modular Controller écrive les données de cartes dans les registres de l'automate (API).

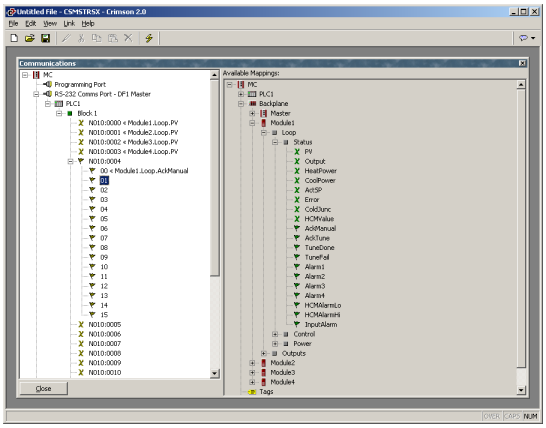
MAPPING D'ELEMENTS SUR UN BLOC

Une fois que le bloc a été créé et que sa taille a été définie, des entrées s'affichent dans le volet gauche de la fenêtre et représentent chacun des registres que le bloc contient. Lorsque vous sélectionnez l'un des registres, le volet droit affiche une liste d'éléments de données disponibles provenant des cartes ainsi que toutes les autres étiquettes que vous avez configurées.



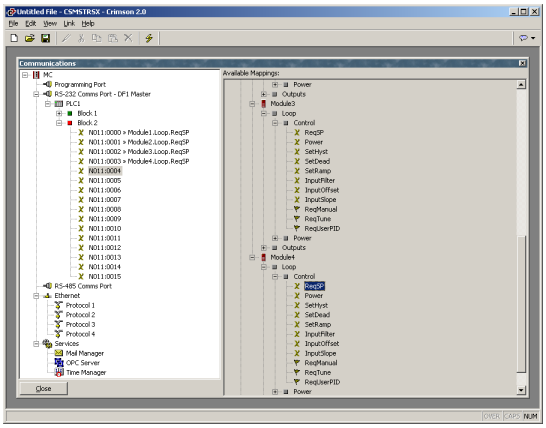
Pour faire correspondre et mapper une variable du Modular Controller avec un registre de l'API, il vous suffit de cliquer et glisser cet élément du volet droit vers le volet gauche et de la placer sur le registre API que vous souhaitez. L'exemple ci-dessus montre les variables de la valeur du processus (PV) de quatre modules PID mappés sur les registres de l'API `N010:0000` par le biais de `N010:0003`.

ACCES AUX BITS INDIVIDUELS



Pour travailler avec des bits individuels dans un mot, il vous suffit de glisser et déplacer le bit d'une carte dans un registre. Après vous avoir demandé l'autorisation d'afficher le mot en tant que bits individuels, Crimson étend le mot en bits de mot. La variable bit est placé dans la première position du bit. Vous pouvez mapper les autres bits dans les bits du mot restants.

LECTURE/ECRITURE DE VARIABLES



Certaines données des cartes sont des variables d'écriture/lecture, ce qui leur permet d'être contrôlées par l'API ou le HMI sur lequel le Modular Controller est connecté. Pour mapper des variables accessibles en écriture, appliquez la même procédure telle que décrite ci-dessus, mais cette fois, définissez la propriété « Direction » Périphérique à Modular Controller. Notez qu'avec la propriété « Direction » définie comme Périphérique à Modular Controller, seules les variables accessibles en écriture s'affichent dans le volet droit.

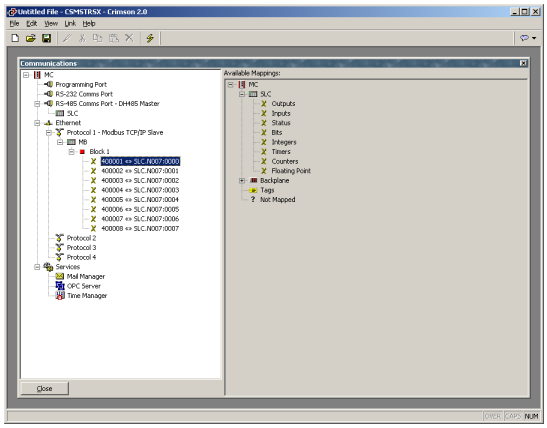
AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

Dans l'exemple ci-dessus, les modules 1 à 4 ont leurs points de consigne mappés sur les registres de l'API **N011:0000** à **N011:0003**. Par conséquent, le maître interroge le contenu des registres de l'API et les écrits dans les variables des cartes respectives.

CONVERSION DE PROTOCOLE

Outre le mapping des données des cartes, les blocs de passerelle peuvent être utilisés pour exposer des données qui sont obtenues à partir d'autres périphériques distants ou pour déplacer des données entre ces deux périphériques. Cette fonctionnalité unique de conversion de protocole permet une intégration plus étroite entre les éléments de votre système de commande, même lors de l'utilisation de périphériques simples et peu onéreux.

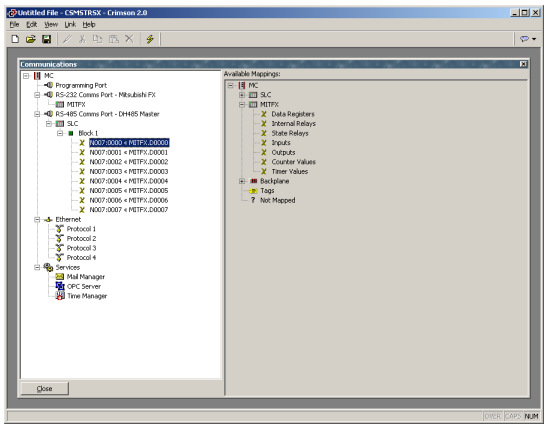
MAITRE ET ESCLAVE



L'exposition de données à partir d'autres périphériques sur un protocole esclave est simplement une extension du processus de mapping décrit ci-dessus, sauf que cette fois, au lieu de glisser une étiquette à partir du volet droit, vous devez étendre le périphérique maître approprié et le glisser sur l'icône qui représente les registres que vous souhaitez exposer. Il vous est alors demandé d'entrer une adresse de départ dans le périphérique maître ainsi que le nombre de registres à mapper pour que les mappings soient créés comme dans l'illustration.

Dans cet exemple, les registres **N7:0** à **N7:7** d'un automate Allen-Bradley ont été exposés à l'accès via le protocole TCP/IP Modbus sur les registres **40001** à **40008**. Crimson garantit automatiquement que ces données sont lues à partir de l'API Allen-Bradley afin de répondre aux requêtes Modbus et convertir automatiquement les écritures vers les registres Modbus en écritures vers l'API. Ce mécanisme permet même de connecter de simples API sur un réseau Ethernet.

MAITRE ET MAITRE



Pour déplacer les données entre deux périphériques maîtres, il vous suffit de sélectionner l'un d'entre eux et de créer un bloc de passerelle pour ce périphérique. Vous pouvez ajouter des mnémoniques aux autres registres du périphérique comme vous le feriez lors de l'exposition des données sur un protocole esclave. Encore une fois, C2 lit ou écrit automatiquement les données comme requis, en déplaçant de façon transparente les données entre les périphériques. L'exemple ci-dessus montre la façon de déplacer les données à partir d'un périphérique Mitsubishi FX dans un périphérique SLC-500.

QUELLE SOLUTION ?

Vous pouvez vous demander si vous devez créer le bloc de passerelle dans le périphérique Allen-Bradley (comme dans cet exemple) ou dans le périphérique Mitsubishi. La première chose à noter est que vous n'avez pas besoin de créer plusieurs blocs pour effectuer des transferts dans une seule direction. Si vous créez un bloc dans AB pour lire à partir de MITFX et un bloc dans MITFX pour écrire dans AB, vous effectuerez simplement deux fois le transfert et vous ralentirez tous les processus ! La deuxième observation est que la décision concernant la « possession » du bloc de passerelle par le périphérique est arbitraire. En règle générale, vous devez créer vos blocs afin de réduire le nombre de blocs dans la base de données. C'est-à-dire que si les registres du périphérique Allen-Bradley se trouvent dans une seule plage, mais si les registres du périphérique Mitsubishi sont dispersés dans l'API, le bloc de passerelle doit être créé dans le périphérique Allen-Bradley afin de supprimer le besoin de créer plusieurs blocs pour accéder aux différentes plages du périphérique Mitsubishi.

TRANSFORMATION DES DONNEES

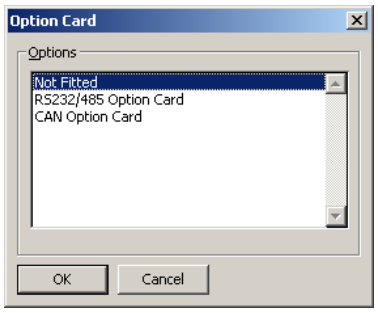
Vous pouvez également utiliser les blocs de passerelle pour effectuer des opérations mathématiques que votre API n'est pas en mesure de gérer. Par exemple, vous devrez peut-être lire un registre à partir de l'API, le mettre à l'échelle, en extraire la racine carrée et la réécrire sur un autre registre API. Pour cela, reportez-vous à la section « Etiquettes de données » et créez une variable mappée pour représenter la valeur d'entrée qui sera lue à partir du périphérique API. Créez ensuite une formule pour représenter la valeur de sortie en définissant l'expression mathématique (le calcul) de façon à effectuer le calcul requis. Vous pouvez enfin créer un bloc de passerelle à destination du registre de sortie, puis glisser la formule dessus pour demander à Crimson de réécrire cette valeur recalculée sur l'API.

COMMUNICATIONS AVANCEES

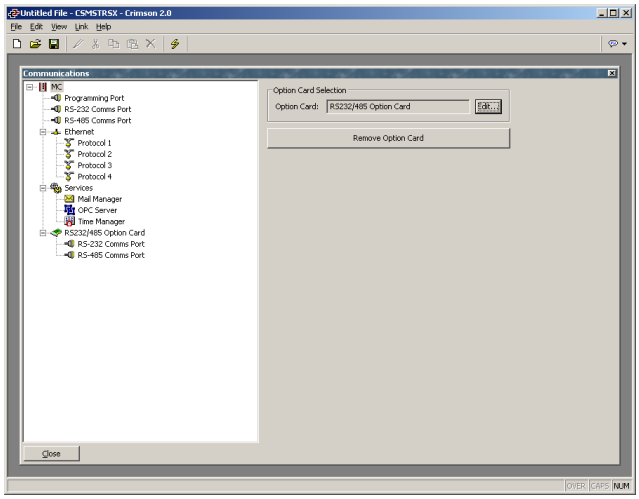
Ce chapitre explique l'utilisation de certaines des fonctionnalités de communication les plus avancées, prises en charge par Crimson. Les applications simples peuvent ne pas nécessiter ces fonctionnalités et vous pouvez par conséquent choisir de passer ce chapitre et de le consulter ultérieurement.

UTILISATION DES CARTES D'EXTENSION

Le module maître du Modular Controller Enhanced type CSMSTRSX ou GT peut héberger une carte d'extension afin de fournir des services de communication supplémentaires. Ces cartes, disponibles depuis le début de l'année 2006, proposent d'autres ports série et une prise en charge de Profibus, DeviceNet et CANOpen. D'autres cartes seront disponibles dès que la gamme sera développée. Les instructions relatives à l'installation matérielle sont fournies avec chacune des cartes. Veuillez donc consulter la fiche technique de la carte en question pour obtenir des informations sur son montage. Une fois que la carte est installée, vous pouvez effectuer sa configuration en sélectionnant l'icône du Modular Controller dans le volet gauche de la fenêtre Communications et en cliquant sur le bouton Edition situé en regard de la propriété Carte optionnelle…



Une fois la carte sélectionnée, une icône appropriée est ajoutée dans l'arborescence qui s'affiche dans le volet gauche de la fenêtre. Cette icône contient des icônes de port(s) supplémentaires () que la carte a rendues disponibles. L'exemple ci-dessous affiche un maître avec une carte d'extension installée…



Vous pouvez configurer les autres ports en suivant les instructions présentées dans le chapitre précédent. Notez que les pilotes disponibles pour un port dépendent du type de connexion qu'il prend en charge. Par exemple, la carte d'extension CANOpen affiche un port qui prend uniquement en charge les pilotes conçus pour la norme de communication CAN.

PARTAGE DU PORT SERIE

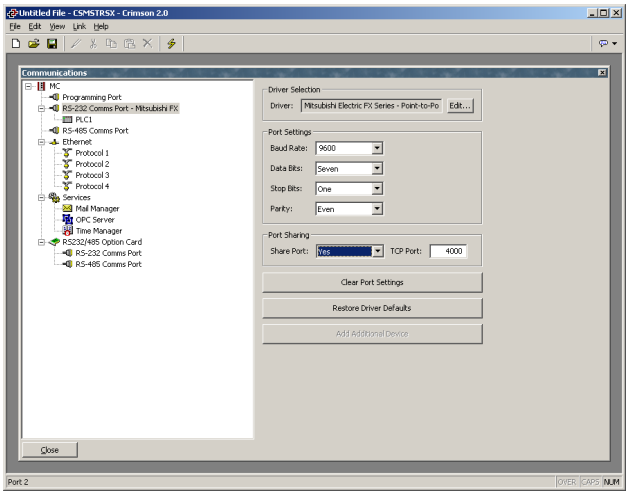
Le maître fournit ce qu'on appelle une fonction de « partage de port » qui permet d'établir des connexions série physiques ou virtuelles sur n'importe quel périphérique connecté dessus. Par exemple, vous pouvez utiliser le maître avec un petit automate programmable, mais comme celui-ci ne dispose que d'un seul port série, il se peut que vous deviez en permanence échanger les câbles lorsque vous modifiez le programme de l'API. En partageant le port de communication du module maître, vous pouvez reprogrammer votre l'API en utilisant simplement le soft de mise en œuvre de l'API, soit à partir d'un port série sur le maître, soit au moyen d'une connexion série virtuelle établie Ethernet.

ACTIVATION DU PROTOCOLE TCP/IP

La première étape de la configuration lors de l'utilisation du partage des ports consiste à activer le port Ethernet du maître, tel que décrit dans le chapitre précédent. Même si vous choisissez de ne pas utiliser la fonction Port série virtuel, le partage local des ports repose sur le protocole TCP/IP qui n'est disponible que si l'Ethernet est activé. Pour activer l'Ethernet, sélectionnez l'icône Ethernet dans la fenêtre Communications à droite, puis le mode de configuration requis. Pour les installations où l'Ethernet n'est pas utilisé, vous pouvez sélectionner la configuration manuelle et laisser le reste des options sur leurs paramètres par défaut.

PARTAGE DU PORT

La prochaine étape consiste à partager le port, ce que vous pouvez effectuer en sélectionnant Oui dans la propriété « Partage du port » et en entrant un numéro de port TCP/IP adapté (en option). Ce numéro représente le port virtuel que vous utilisez pour exposer le port série à un accès via le protocole sur TCP/IP.

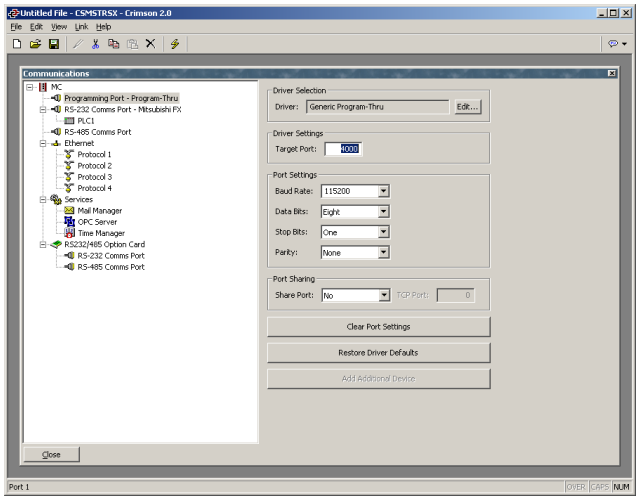


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

Si vous laissez le paramètre du port sur zéro, un numéro de 4 000 plus l'index logique du port sont utilisés. (Pour obtenir l'index logique du port, comptez la position du port dans la liste en notant que le port de programmation est toujours le port logique 1.) Vous pouvez utiliser tout numéro qui n'est pas déjà utilisé par un autre protocole TCP/IP. Si vous êtes à court d'idées, nous vous conseillons des nombres entre 4 000 et 4 099.

CONNEXION VIA UN AUTRE PORT

Si vous souhaitez utiliser un autre port sur le maître pour router les données vers le port partagé, vous devez sélectionner le pilote Generic Program Thru pour ce port, puis configurer ce pilote avec le numéro de port TCP/IP du port série que vous avez partagé. Dans l'exemple ci-dessous, nous routons les données à partir du port de programmation vers un API qui est connectée via le port de communication RS-232…



Notez que Débit en bauds et les autres paramètres de port ne doivent pas forcément être identiques à ceux du port que nous partageons sauf si l'unité programmée est un indicateur PAX. Dans ce cas, le PAX et le port G3 **DOIVENT** être définis sur 9 600 8 N 1. Dans la configuration ci-dessus, les données à partir de et vers le logiciel de programmation sont envoyées à une vitesse de transmission supérieure aux données de l'API, et le module maître effectue la mise en mémoire tampon et la conversion appropriées.

Dans cet exemple, pour utiliser le port partagé, vous devez connecter un port série disponible de votre PC au port de programmation du maître, puis configurer le logiciel de programmation de l'API pour qu'il communique avec le port COM de votre PC. Dès que le PC commence à communiquer avec l'API, les communications entre le maître et l'API sont suspendues et les deux ports du maître (celui du PC et de l'API) sont « connectés » comme si le PC communiquait directement avec l'API. Si aucune donnée n'est transférée pendant plus d'une minute, les communications entre le maître et l'API reprennent.

CONNEXION VIA ETHERNET

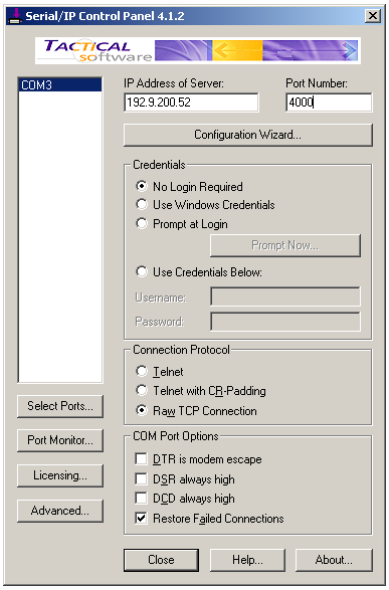
Au lieu d'utiliser un port série sur votre PC et sur le maître, vous pouvez utiliser un utilitaire tiers pour créer ce qu'on appelle des ports série virtuels sur votre ordinateur. Ils apparaissent comme des ports COM physiques aux applications, mais en fait, ils envoient et reçoivent des

données sur un périphérique distant par le protocole TCP/IP. En installant l'un de ces utilitaires et en le configurant pour qu'il s'adresse au maître, vous pouvez disposer d'un accès série sur tous les périphériques connectés au maître sans câblage supplémentaire. En effet, il est complètement inutile d'avoir des ports série disponibles sur le PC, ce qui est très précieux lorsque vous travaillez avec des ordinateurs portables modernes sur lesquels un port COM série est souvent une option onéreuse.

Plusieurs utilitaires de port série virtuels tiers sont disponibles. Côté freeware, une société nommée HW Group (http://www.hw-group.com) propose un utilitaire appelé HW Virtual Serial Port. Il existe également plusieurs autres freeware mais la plupart d'entre eux semblent provenir de la même base source. Côté commercial, une société appelée Tactical Software (http://www.tacticalsoftware.com) propose la solution Serial/IP pour environ 100 $ le port.

Même si les différents freeware ont sans aucun doute de nombreux clients satisfaits, nous avons découvert que ces pilotes présentaient des problèmes de stabilité occasionnels sur certains PC. La solution Serial/IP de Tactical Software est ainsi le seul package que nous sommes en mesure de prendre en charge et les informations suivantes supposent que vous utilisez ce package.

Pour créer un port série virtuel, ouvrez l'écran de configuration Serial/IP, puis sélectionnez le nom du port COM série que vous souhaitez définir. En général, c'est le premier port série libre après ceux qui ont été alloués aux ports physiques et aux modems etc … installés sur votre PC. Saisissez ensuite l'adresse IP du maître (CSMSTRSX ou GT ou DSP ou G3), puis le numéro de port TCP/IP que vous avez alloué lors du partage du port. L'exemple ci-dessous est configuré comme décrit ci-dessus dans ce document. Pour finir, assurez-vous que Connexion TCP Raw est sélectionnée, puis fermez la boîte de dialogue Serial/IP.



Vous pouvez maintenant configurer n'importe quels logiciels Windows pour qu'il utilise le port COM série virtuel qui vient d'être créé pour effectuer un téléchargement via l'unitée (CSMSTRSX ou GT ou Data Station Plus ou G3). Lorsque le logiciel ouvre la connexion, le maître (CSMSTRSX ou GT ou Data Station Plus ou G3) suspend les communications sur le

port partagé, puis les données sont ensuite échangées entre le logiciel de programmation de l'API distant, comme s'ils étaient directement connectés ! Une fois que le port est fermé ou si aucune donnée n'est transférée pendant une minute, les communications reprennent.

Notez qu'en supposant que vous avez acheté le nombre de licences approprié pour la solution Serial/IP, vous pouvez créer autant de ports virtuels que vous le souhaitez. Ainsi, vous pouvez être connecté à plusieurs périphériques à partir du même PC en téléchargeant sur chacun par le biais de son package de programmation respectif, le tout sans connecter ou déconnecter un seul câble. Cette fonctionnalité est très utile lorsque vous disposez de plusieurs périphériques dans un système complexe.

PORTS VIRTUELS PURS

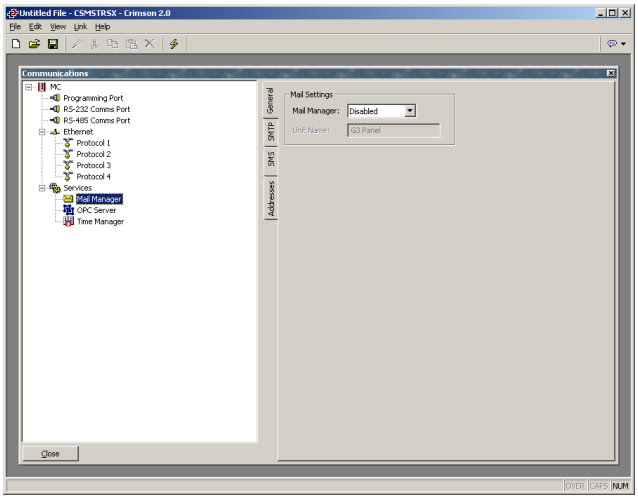
Parfois, vous pouvez vouloir utiliser un port série disponible sur un maître (CSMSTRSX ou GT ou Data Station Plus ou G3) pour fournir un accès à un périphérique distant qui n'est pas connecté dessus. Ou vous pouvez vouloir utiliser un tel port pour effectuer une connexion sur un port de programmation dédié sur un périphérique même si le maître utilise un autre port pour établir des communications avec ce périphérique. Par exemple, si votre API est dotée d'un port HMI dédié ainsi qu'un port de programmation dédié, vous pouvez communiquer dessus via le premier et télécharger dessus via le second. Pour cela, configurez le port qui sera utilisé pour la programmation de façon classique, en sélectionnant le pilote du port série virtuel. Puis partagez le port tel que décrit ci-dessus. Le pilote du port série virtuel n'effectue aucune activité de communication, mais il autorise le partage pour l'accès à distance.

LIMITATIONS

Notez que certains logiciels de programmation d'API peuvent ne pas fonctionner avec des ports partagés de façon virtuelle ou physique. Les problèmes à surveiller sont : les délais d'attente court qui ne permettent pas au maître de relayer les données sur l'API ; une dépendance à l'envoi de signaux d'interruption ou à la manipulation de lignes d'établissement de liaison du matériel ; ou un accès au port de type DOS tel que le logiciel ne peut pas « voir » les ports série virtuels. Heureusement, ces problèmes sont rares et la majorité des logiciels communiquent sans problème comme s'ils étaient directement connectés à l'API en question.

UTILISATION DE LA MESSAGERIE ELECTRONIQUE

Crimson peut être configuré pour envoyer des messages électroniques lorsque les conditions d'alarme sont présentes ou lorsque des notifications doivent être fournies pour d'autres événements du système. Les méthodes utilisées pour envoyer le courrier électronique sont configurées via l'icône Mail dans la fenêtre Communications…

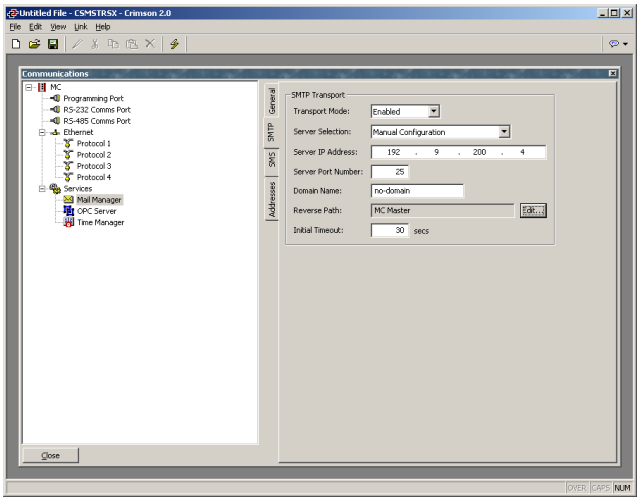


Les propriétés de l'onglet Général sont utilisées pour activer ou désactiver le Gestionnaire de Messages et pour fournir un nom au maître. Ce nom est utilisé dans les messages électroniques pour identifier l'auteur du message. Les applications utilisent généralement le nom de la machine sur laquelle le maître est installé ou le nom du site qu'il contrôle.

CONFIGURATION DU PROTOCOLE SMTP

L'onglet SMTP permet de configurer le protocole SMTP (Simple Mail Transport Protocol). C'est le protocole standard utilisé pour envoyer des messages sur Internet ou sur les autres réseaux TCP/IP. Les adresses SMTP suivent la même norme habituelle **`name@domain`**.

Les options de configuration du transport SMTP sont affichées ci-dessous…



AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

- La propriété *Mode de transport* permet d'activer ou de désactiver le transport. Notez que le Gestionnaire de Messages doit être activé via l'onglet Général avant que le transport SMTP ne puisse être activé. Notez aussi que le protocole SMTP ou SMS doit être activé pour que le Gestionnaire de Messages puisse envoyer des messages.

- La propriété *Sélection Serveur* permet de définir la façon dont le transport localise un serveur SMTP. Si Sélection Manuelle est utilisée, la propriété *Adresse IP Serveur* doit être utilisée pour créer manuellement un serveur. Si *Configuré via DHCP* est sélectionné, le port Ethernet de l'unité doit être configuré pour utiliser le protocole DHCP et le serveur DHCP du réseau doit être configuré pour créer un serveur SMTP via l'option 69.

- La propriété *Adresse IP Serveur* permet de créer un serveur SMTP lorsque la sélection du serveur manuel est activée. Le serveur doit être configuré pour accepter les messages électroniques à partir du maître (CSMSTRSX ou GT ou Data Station Plus ou G3) et pour les relayer si l'application le nécessite.

- La propriété *Numéro de port Serveur* permet de définir le numéro de port TCP qui est utilisé lors des sessions SMTP. La valeur par défaut est 25. Cette valeur est adaptée pour la majorité des applications et ne nécessite qu'un réglage si le serveur SMTP a été reconfiguré pour utiliser un autre port.

- La propriété *Nom de domaine* permet de spécifier le nom de domaine qui sera passé au serveur SMTP dans la commande HELO. La grande majorité des serveurs SMTP ignorent cette chaîne. Dans le cas improbable où votre serveur SMTP tente de rechercher un DNS pour confirmer l'identité de son client, vous devrez peut-être entrer des données appropriées à votre configuration DNS.

- La propriété *E-mail de retour* permet de spécifier l'adresse électronique qui est fournie pour l'auteur des messages envoyés par le maître(CSMSTRSX ou GT ou Data Station Plus ou G3). Cette propriété comprend un nom complet et une adresse électronique. Puisque le maître (CSMSTRSX ou GT ou Data Station Plus ou G3) n'est pas en mesure de recevoir des messages, l'adresse électronique est souvent définie sur quelque chose qui renvoie un message « Non remis » si une réponse est envoyée.

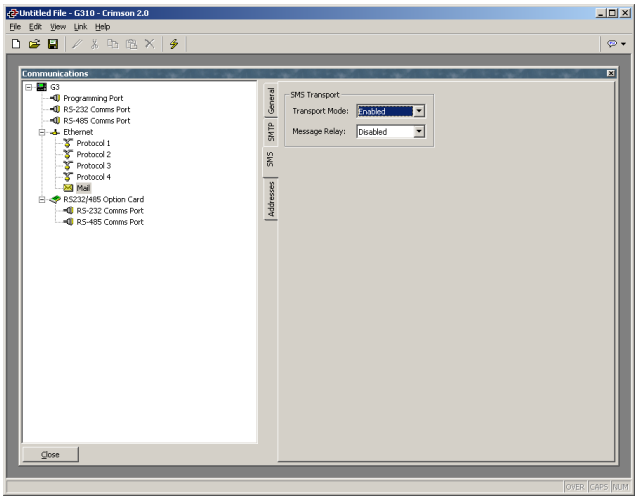
- La propriété *Temps Max Initial* permet de spécifier le temps d'attente où le client de la messagerie attend que le serveur SMTP envoie sa bannière de bienvenue. Certains serveurs Microsoft essaient de négocier une authentification spécifique à Microsoft avec les clients de la messagerie, retardant ainsi le point sur lequel la bannière s'affiche. Vous devrez peut-être rallonger ce délai de 2 minutes ou plus lorsque vous travaillerez avec de tels serveurs.

CONFIGURATION DU PROTOCOLE SMS

L'onglet SMS permet de configurer le service SMS (Short Messaging Service). Ce transport est utilisé pour envoyer des messages texte aux téléphones portables via un modem GSM. Les adresses électroniques SMS comprennent un numéro de téléphone au format international,

mais sans le signe plus. Un exemple d'adresse aux états-Unis serait 17175551234 et un exemple d'adresse au Royaume-Uni 441246555555. Dans ces deux cas, l'adresse contient le code du pays suivi du code de la région et du numéro de l'abonné.

Les options de configuration du service SMS sont affichées ci-dessous…



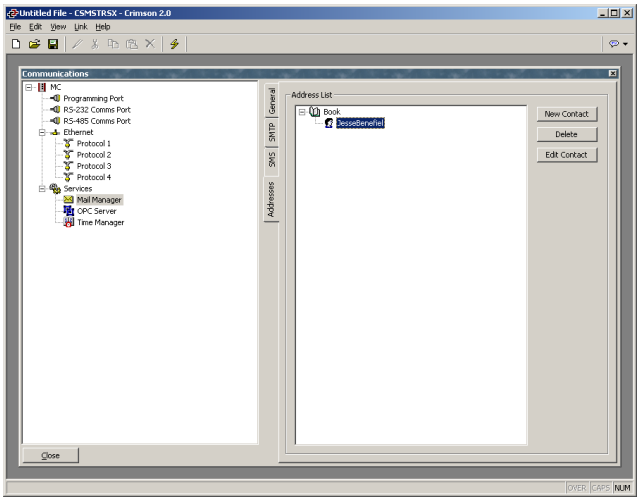
- La propriété *Mode de transport* permet d'activer ou de désactiver le transport. Notez que le Gestionnaire de Messages doit être activé via l'onglet Général avant que le transport SMS ne puisse être activé. Notez aussi que le protocole SMTP ou SMS doit être activé pour que le Gestionnaire de Messages puisse envoyer des messages.

- La propriété *Relais des Messages* permet d'activer ou de désactiver la fonctionnalité de relais SMS du maître(CSMSTRSX ou GT ou Data Station Plus ou G3). Si cette fonctionnalité est activée, un utilisateur qui reçoit un message SMS envoyé à plusieurs destinataires peut répondre à ce message et le maître relaie le message aux autres destinataires. Ainsi, un service de conférence simple est offert entre les destinataires des messages.

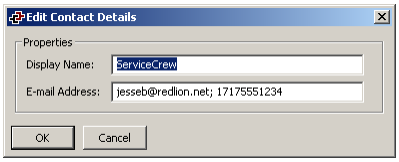
Notez que pour que le transport SMS fonctionne, un modem GSM doit avoir été installé sur l'un des ports série de l'unité. Reportez-vous aux sections ultérieures de ce chapitre pour obtenir des détails sur la configuration d'un tel modem et sur l'interaction de plusieurs modems.

LE CARNET D'ADRESSES

L'onglet Adresses permet de définir les destinataires des messages électroniques…



Vous pouvez ajouter, modifier ou supprimer un nombre illimité d'entrées au carnet d'adresses à l'aide des boutons situés dans le volet droit. Chaque entrée peut se rapporter à un ou plusieurs destinataires à partir de n'importe quel transport activé dans la base de données. Les destinataires de plusieurs transports peuvent être ajoutés dans la même entrée. La boîte de dialogue utilisée pour définir les propriétés de chaque destinataire est affichée ci-dessous…



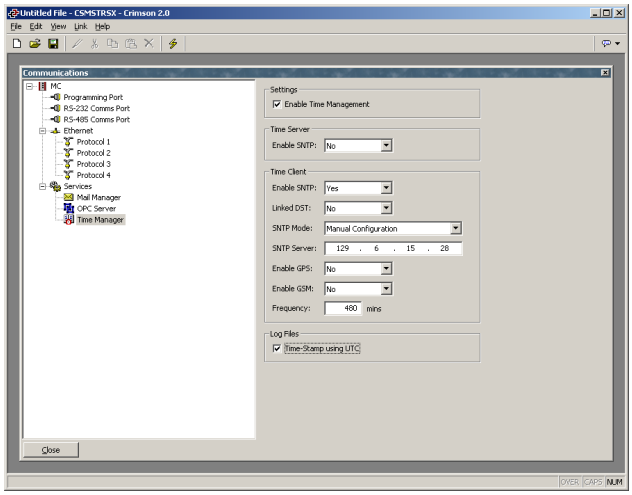
- La propriété *Nom* permet de définir le nom de la personne de l'entrée du carnet d'adresses. C'est le nom qui est utilisé comme nom complet des destinataires SMTP et pour sélectionner une entrée du carnet d'adresses de Crimson.
- La propriété *E-mail* permet de définir un ou plusieurs destinataires pour cette entrée du carnet d'adresses. Lorsqu'il y a plusieurs destinataires, ils doivent être séparés par un point-virgule. Le format de chaque destinataire dépend du transport prévu pour envoyer le message. Dans l'exemple ci-dessus, l'entrée du carnet d'adresses se rapporte à un destinataire SMTP et à un destinataire SMS.

UTILISATION DE LA GESTION DU TEMPS

Crimson propose des fonctions qui vous permettent de synchroniser l'heure et la date dans CSMSTRSX ou GT ou Data Station Plus ou G3) avec différentes sources. Le Gestionnaire de l'heure peut également conserver des informations sur le fuseau horaire actuel du maître et si l'heure d'été est actuellement activée. En effet, les informations précises sur le fuseau horaire représentent une synchronisation vitale et correcte car les différentes méthodes de synchronisation sont toutes conçues pour fonctionner avec l'heure UCT (également appelée heure de Greenwich).

CONFIGURATION DU GESTIONNAIRE DE L'HEURE

Vous pouvez accéder aux différentes propriétés associées à la configuration des fonctions du Gestionnaire de l'heure via l'icône Gestionnaire de l'heure de la fenêtre Communications…



Les propriétés sont détaillées ci-dessous…

- La propriété *Activer le Gestionnaire de l'heure* permet de contrôler l'accès aux autres fonctions. Si elle n'est pas activée, Crimson fonctionne à l'heure locale et ne reconnaît pas les fuseaux horaires ou les autres informations de gestion de l'heure.

- La propriété *Activer le SNTP* dans le groupe *En tant que Serveur* est utilisée pour activer le serveur SNTP (Simple Network Time Service) dans Crimson. Ce serveur rend disponible l'heure actuelle et la date du maître aux autres produits Red Lion via un lien TCP/IP. Notez que la mise en œuvre du protocole SNTP de Crimson n'est pas complètement conforme à RFC et n'est pas prise en charge en tant que source de synchronisation des clients non-Red Lion.

- La propriété *Activer le SNTP* du groupe *En tant que Client* permet de demander au maître de synchroniser son horloge sur le serveur SNTP spécifié. Le serveur peut être un maître, un écran G3 ou un serveur tiers tel que fourni par la famille Windows 2000 Server des systèmes d'exploitation.

- La propriété *DST liée* permet de demander au client SNTP de tenter de lire le paramètre actuel Heure d'été à partir du serveur SNTP. Comme cette fonction ne fait pas partie standard du protocole SNTP, elle ne fonctionne que si un autre maître ou un écran opérateur G3 est défini comme serveur. Cette fonction est utile dans le sens où elle permet de régler l'heure d'été via un seul périphérique sur le réseau de l'usine et les autres périphériques suivent le paramètre central.

- Les propriétés *Mode du SNTP* et *Serveur SNTP* permettent de configurer l'adresse IP du serveur SNTP (Simple Network Time Service). Si *Configuré via DHCP* est sélectionné, le port Ethernet de l'unité doit être configuré pour utiliser

le protocole DHCP et le serveur DHCP du réseau doit être configuré pour créer un serveur via l'option 42.

- La propriété *Sync GPS* permet de demander au client de temps d'utiliser une unité GPS connectée via NMEA-0183 comme autre méthode d'obtention de l'heure actuelle. L'unité peut être connectée sur n'importe quel port série à l'aide du pilote approprié.

- La propriété *Fréquence* permet de spécifier la fréquence à laquelle le maître doit tenter de synchroniser son heure par les méthodes activées ci-dessus. Le maître tente de synchroniser vingt secondes après la mise sous tension, puis tel que spécifié par cette propriété. Si une tentative donnée de synchroniser échoue, l'unité retente toutes les 30 secondes jusqu'à ce que l'opération réussisse. Si les synchronisations du GPS et du SNTP sont activées, le SNTP est uniquement utilisé si un GPS n'est pas disponible.

- La propriété *Horodatage à l'aide de l'UTC* permet de demander à Crimson de baser son enregistrement d'événements et de données sur l'UTC plutôt que sur l'heure locale. Ainsi, les fichiers journaux qui sont facilement transférables sur des fuseaux horaires et qui ne rencontrent pas de discontinuités lors de l'entrée ou de la sortie de l'heure d'été sont créés. Ce paramètre est global et a des conséquences sur tous les fichiers journaux du système.

SELECTION D'UN SERVEUR SNTP

Lorsque vous configurez le client SNTP, plusieurs options s'offrent à vous lors de la sélection d'un serveur.

Si vous disposez d'un serveur de temps Windows ou Unix dans le cadre de votre infrastructure réseau, vous devez finir par synchroniser sur cette source pour garantir une synchronisation sur l'ensemble de l'entreprise. Si plusieurs maîtres sont synchronisés sur le même réseau, il vous est plus simple de nommer l'un d'entre eux en tant que périphérique maître afin de définir l'heure d'été et de faire synchroniser ce maître seul sur la source de l'heure de l'entreprise. Vous pouvez ensuite configurer les autres périphériques pour qu'ils synchronisent sur le périphérique maître, puis activer la fonction DST liée afin de diffuser le paramètre Heure d'été au sein de votre usine.

Si aucune source de l'heure d'entreprise n'est disponible, vous pouvez choisir de nommer un seul maître en tant que point là où un opérateur définit l'heure, puis faire synchroniser les autres maîtres sur cette source. Sinon, si votre installation fournit un accès TCP/IP sur Internet via l'Ethernet ou une connexion par modem, vous pouvez configurer le client SNTP pour qu'il synchronise sur un serveur de temps public. Exemple : 192.6.15.28, qui est l'adresse IP actuelle d'un serveur de temps public fourni par NIST. Une liste des autres serveurs se trouvent à l'adresse :

http://support.microsoft.com/kb/262680

Notez que comme Crimson utilise une adresse IP et pas un nom d'hôte pour désigner le serveur SNTP, il perd la connexion avec tous les serveurs qui sont déplacés sur une nouvelle

adresse réseau. Même si ces déplacements sont très rares, ils dépassent votre contrôle et celui de Red Lion. L'utilisation d'une source de l'heure d'entreprise qui accède à sa propre source via le serveur DNS est par conséquent préférable !

CONFIGURATION DES FUSEAUX HORAIRES

Comme nous venons de l'expliquer, le CSMSTRSX ou GT ou Data Station Plus ou G3) doit reconnaître le fuseau horaire actuel s'il doit utiliser la gestion de l'heure avancée. Ces informations peuvent être fournies au maître de deux façons : La méthode la plus simple consiste à utiliser la commande Envoyer l'heure du menu Lien du logiciel de configuration de Crimson. Outre la configuration de l'horloge du maître, cette commande envoie également le fuseau horaire actuel du maître et le statut de l'heure d'été. Le maître enregistre ces données dans une mémoire non volatile et les utilise à partir de ce point. Vous devez vous assurer que le PC contient des informations valides sur l'heure et la date avant de les envoyer vers l'unité !

- L'autre méthode consiste à utiliser les variables système **`TimeZone`** et **`UseDST`**. La première variable contient le nombre d'heures selon lequel le fuseau horaire local diffère de l'UTC et peut être négatif ou positif. Par exemple, un paramètre de –5 correspond à l'heure de l'Est des Etats-Unis. La deuxième variable contient soit 0, soit 1, selon si l'heure d'été est active. La modification de l'une de ces variables via l'interface utilisateur entraîne le changement d'heure de l'unité pour prendre en compte les nouveaux paramètres. Par exemple, l'activation de l'heure d'été avance l'heure d'une heure alors que la désactivation la recule. Une base de données classique doit uniquement exposer **`UseDST`** pour la modification par l'utilisateur et même cela peut ne pas être nécessaire si la fonction DST liée décrite ci-dessus est en cours d'utilisation.

MODEMS

Cette section décrit la configuration du module maître (CSMSTRSX ou GT ou Data Station Plus ou G3) pour qu'il fonctionne avec des modems ou des connexions série directes sur des ordinateurs dotés du système d'exploitation Windows. Notez que la prise en charge du modem de Crimson est très différente de celle qui est fournie par les produits antérieurs de Red Lion dans le sens où elle est complètement basée sur le protocole PPP (Point-To-Point Protocol). Alors que les protocoles comme Modbus permettent une seule conversation entre les deux périphériques, le protocole PPP est plus semblable à une connexion Ethernet car elle permet un nombre illimité de connexions logiques sur un seul lien physique. Une seule connexion PPP peut ainsi permettre un accès simultané au service de téléchargement TCP/IP du maître, à son serveur Web, à ses ports série partagés et à tous les protocoles TCP/IP qui ont été sélectionnés par le biais de la fenêtre Communications.

QUELQUES EXEMPLES D'APPLICATIONS

Les sections ci-dessous répertorient certaines applications typiques de la technologie de modem…

- Vous souhaitez qu'un maître (CSMSTRSX ou GT ou Data Station Plus ou G3) situé à un emplacement distant envoie un message électronique à un ingénieur de

la maintenance pour l'informer d'une condition d'erreur. En configurant une connexion à la demande sur un fournisseur de services Internet, il est demandé au maître (CSMSTRSX ou GT ou Data Station Plus ou G3) de se connecter automatiquement lorsqu'un message électronique doit être envoyé, puis de raccrocher lorsque le message a été transféré.

- Vous souhaitez qu'un maître (CSMSTRSX ou GT ou Data Station Plus ou G3) situé à un emplacement distant envoie directement des messages aux téléphones portables d'un groupe d'ingénieurs de la maintenance pour les informer d'une condition d'erreur. En configurant un modem GSM avec la prise en charge SMS, il est demandé au maître(CSMSTRSX ou GT ou Data Station Plus ou G3) de notifier les ingénieurs du défaut par le biais de messages texte courts. D'autre part, lorsqu'un ingénieur répond au message afin d'indiquer qu'il se charge du problème, le maître (CSMSTRSX ou GT ou Data Station Plus ou G3)peut, le cas échéant, transférer la réponse à tous les autres ingénieurs pour leur faire savoir qu'il assume la responsabilité du problème.

- Un maître (CSMSTRSX ou GT ou Data Station Plus ou G3)situé à un emplacement distant est configuré pour accepter les connexions entrantes à partir d'un PC basé dans un bureau central. Une fois la connexion établie, la base de données du maître (CSMSTRSX ou GT ou Data Station Plus ou G3) peut être mise à niveau à distance en demandant à la configuration de Crimson de télécharger via le lien TCP/IP. Si telle est la configuration, il est possible d'accéder au serveur Web du maître (CSMSTRSX ou GT ou Data Station Plus ou G3) afin de fournir des fonctions de commande à distance. Mieux, en installant le logiciel de port série virtuel sur le PC et en activant le partage de port sur le maître, (CSMSTRSX ou GT ou Data Station Plus ou G3) un logiciel de programmation d'API peut être utilisé pour télécharger via le CSMSTRSX ou GT ou Data Station Plus ou G3 une modification du programme de l'API.

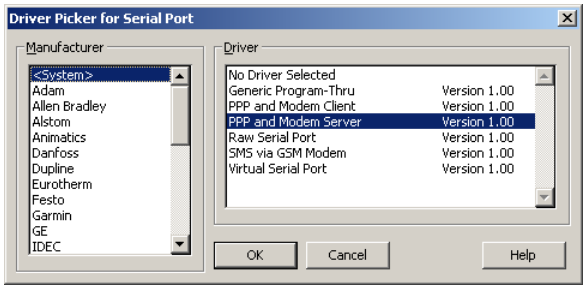
- Un maître (CSMSTRSX ou GT ou Data Station Plus ou G3) situé à un emplacement distant est configuré pour accepter les connexions entrantes à partir d'un système SCADA (Supervision) basé dans un bureau central. Le logiciel SCADA peut utiliser le protocole TCP/IP Modbus pour accéder aux blocs de passerelle dans le maître(CSMSTRSX ou GT ou Data Station Plus ou G3), en lisant et écrivant donc les données collectées à partir des périphériques connectés à ses ports série. Le package SCADA peut également établir un contact direct avec les périphériques connectés au maître par le biais de la fonctionnalité de routage IP du maître.

Il existe bien entendu de nombreux autres exemples d'applications.

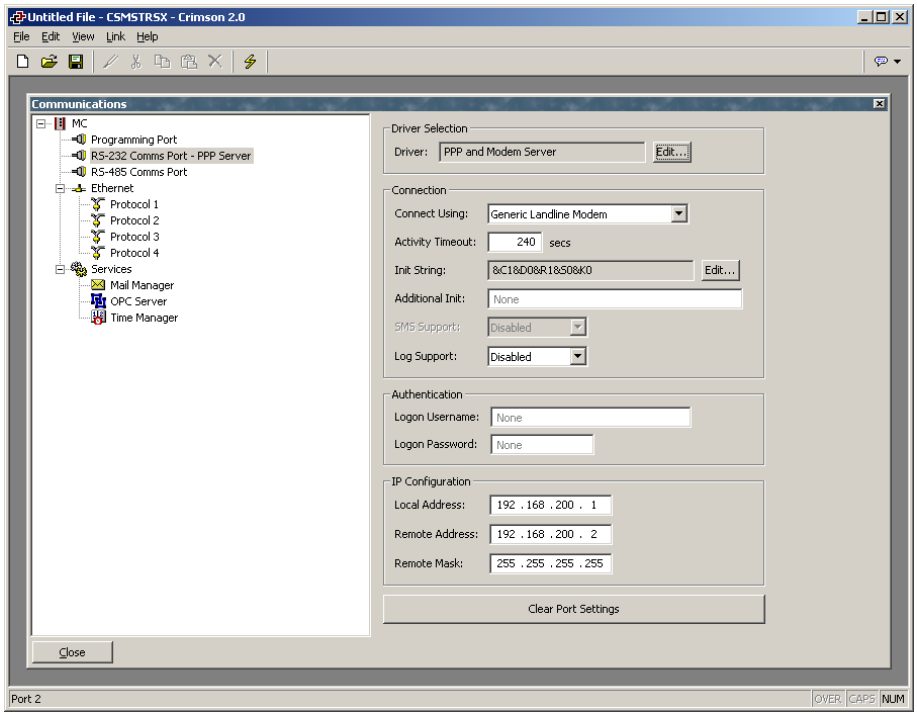
AJOUT D'UNE CONNEXION D'APPEL ENTRANT

Pour ajouter une connexion d'appel entrant à votre base de données (Vous souhaitez contacter le CSMSTRSX ou GT ou Data Station Plus ou G3), ouvrez la fenêtre Communications et sélectionnez le port série sur lequel la connexion sera établie et le modem sera connecté.

Cliquez sur le bouton Edition dans le champ Pilote situé dans le volet droit, puis sélectionnez le pilote *Modem Serveur PPP* dans la section Système de la boîte de dialogue…



Le volet droit affiche alors la configuration du modem…



Le modem est doté des options de configuration suivantes…

- La propriété *Se connecter via* permet de sélectionner le périphérique physique qui est utilisé pour établir la connexion. Les périphériques pris en charge à ce niveau sont des connexions série directes sur des ordinateurs dotés d'un système d'exploitation Microsoft Windows, des modems à ligne terrestre générique qui implémentent le jeu de commandes Hayes ainsi que le mode GSM Telit GM-862. Pour les connexions d'appel entrant, le périphérique Telit doit être configuré en mode Données basculées sur le circuit.

- La propriété *Arrêt après inactivité de* permet de définir la durée où le maître n'envoie aucun paquet sur le lien PPP pour que la connexion soit terminée. Pour les connexions d'appel entrant, on suppose que le périphérique de connexion est convivial. Par conséquent, aucun effort n'est fourni pour filtrer les paquets facultatifs qui peuvent avoir pour conséquence que le lien reste actif sur de

longues périodes. Notez que même si vous voulez une connexion permanente, vous devez entrer un délai d'attente approprié pour permettre de détecter les liens morts. Cela implique que les « connexions permanentes » peuvent être perdues à certains moments, mais comme le client rétablit immédiatement le lien, ce n'est pas un problème.

- La chaîne *Init Supplémentaire* est utilisée avec les liens non directs et fournit une série de commandes AT qui servent à initialiser le modem. Le préfixe initial AT n'est pas requis. Vous pouvez associer plusieurs commandes simplement en les plaçant l'une après l'autre. La chaîne exacte requise pour votre modem dépend de son logiciel interne. Ainsi, si vous contactez le support technique pour obtenir une assistance, assurez-vous que vous disposez des informations exactes du modèle et de la marque.

- La propriété *Support SMS* permet d'activer la messagerie SMS (Short Message Service) lors de l'utilisation d'un modem GSM. Pour que la messagerie SMS fonctionne correctement, vous devez activer le transport SMS à l'aide de l'icône Mail qui se trouve dans la fenêtre Communications, tel que décrit ci-dessus.

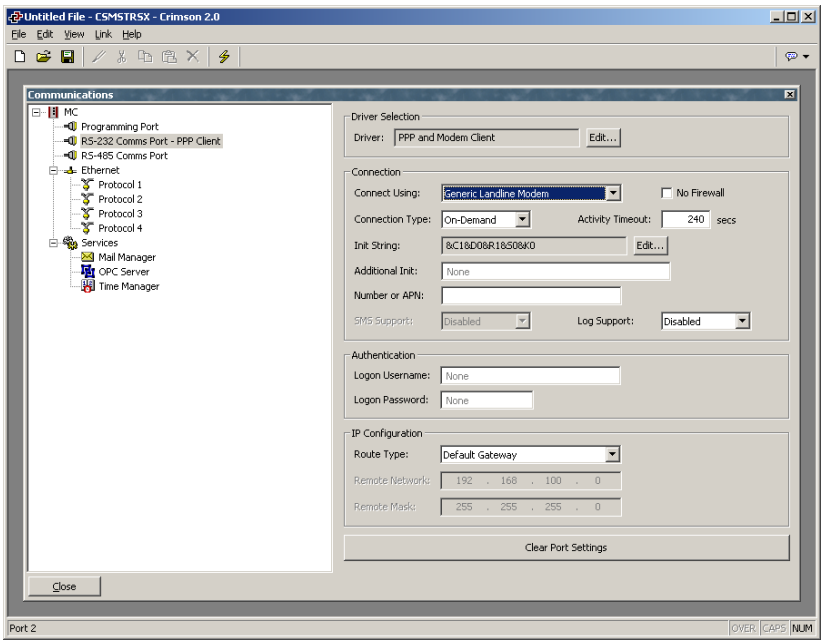
- Les propriétés *Nom d'Utilisateur* et *Mot de Passe* permettent de définir les informations d'identification que le client distant doit fournir afin d'être autorisé à se connecter à ce périphérique. Le nom d'utilisateur ne respecte pas la casse, mais le mot de passe la respecte. L'implémentation du protocole PPP de Crimson demande à son homologue d'utiliser l'authentification CHAP pour éviter la transmission ou la réception des mots de passe en clair, mais a recours au protocole PAP si le client distant ne prend pas en charge l'authentification CHAP.

- La propriété *Adresse locale* permet de définir l'adresse IP attribuée à des fins locale de la connexion. Ainsi, vous utiliserez cette adresse pour rentrer en communication par Ethernet ou Internet ou par un modem avec les CSMSTRSX ou GT ou Data Station Plus ou G3. Veuillez noter qu'elle ne doit pas être identique à l'adresse IP du port Ethernet du maître (CSMSTRSX ou GT ou Data Station Plus ou G3) car chaque interface IP physique doit posséder une adresse IP distincte. La valeur par défaut fonctionne dans la plupart des cas sauf si la conception de votre réseau exige que vous utilisiez un paramètre différent.

- La propriété *Adresse distante* permet de définir l'adresse IP attribuée à la connexion distante. Elle est utilisée avec la propriété *Masque distant* pour déterminer les paquets qui seront routés vers cette connexion. Pour la plupart des applications, un masque de 255.255.255.255 est utilisé, demandant alors à Crimson d'envoyer uniquement via cette interface les paquets à destination directe du client distant. En revanche, un masque de 0.0.0.0 permet de transférer au client distant tous les paquets qui ne correspondent pas expressément à une autre interface, vraisemblablement pour un transfert ultérieur vers l'hôte visé. Les masques intermédiaires peuvent être utilisés pour contrôler l'envoi exact des paquets.

AJOUT D'UNE CONNEXION D'APPEL SORTANT

Les connexions d'appel sortant sont configurées de la même façon que ce que nous venons de décrire ci-dessus sauf que le pilote *Modem Client PPP* doit être sélectionné pour le port requis. Les options de configuration de ce modem sont affichées ci-dessous…



Le modem est doté des propriétés suivantes qui sont distinctes des propriétés des connexions d'appel entrant…

- La propriété *Se connecter via* s'adresse aux connexions d'appel entrant avec en plus, la prise en charge des connexions GPRS via un modem GSM. Ces connexions diffèrent des connexions CSD car elles atteignent des vitesses bien supérieures et elles sont généralement chargées sur la base de la quantité de données transférées plutôt que sur la durée de la connexion. Ainsi, les connexions GPRS peuvent être configurées pour une connexion permanente sauf si un temps d'arrêt est nécessaire pour permettre le transfert des messages SMS.

- La propriété *Aucun pare-feu* permet de désactiver la protection par pare-feu qui est par ailleurs fournie par les connexions d'appel sortant. Cette protection empêche les connexions entrantes d'être établies sur cette interface et empêche le maître (CSMSTRSX ou GT ou Data Station Plus ou G3) d'envoyer certains paquets de diagnostic qui pourraient fournir aux pirates des informations sur le système ou être utilisés par les attaquants pour maintenir une connexion active en l'absence du transfert de données réel. Si vous êtes directement connecté à Internet par le biais de cette connexion, vous ne devez normalement pas désactiver le pare-feu. Le pare-feu doit être désactivé uniquement pour les connexions sur les réseaux d'entreprise ou sur les autres environnements contrôlés.

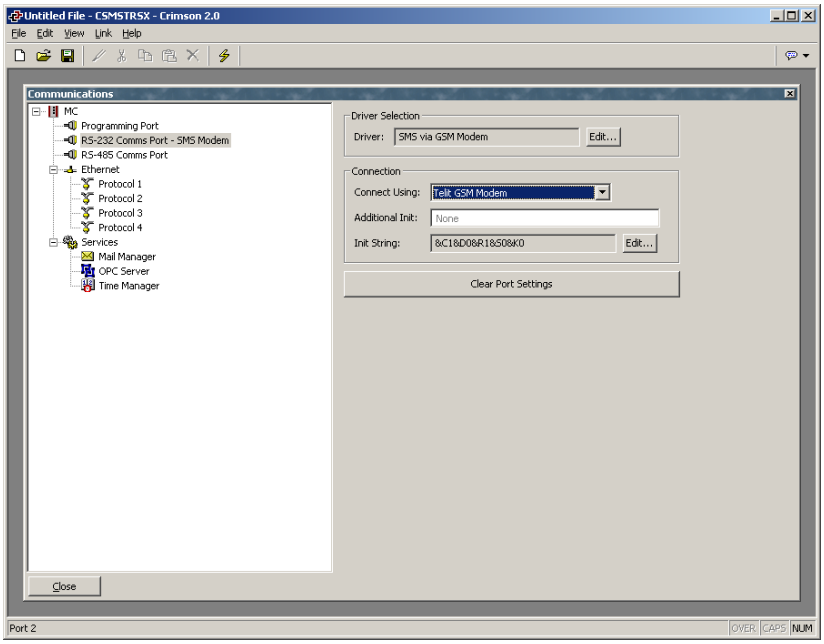
- La propriété *Type de connexion* permet d'indiquer si vous souhaitez que cette connexion soit conservée en permanence ou si vous souhaitez qu'elle soit établie automatiquement lorsqu'une tentative de transférer les données vers des hôtes qui peuvent être atteints via cette interface est effectuée. Si vous sélectionnez la connexion à la demande, vous devez spécifier le délai d'attente précédant la fin du lien si aucun paquet n'a été transmis par le maître.

- Les propriétés *Nom d'Utilisateur* et *Mot de Passe* permettent de définir les informations d'identification qui sont envoyées au serveur distant lors de la tentative d'initialisation de cette connexion. Le nom d'utilisateur ne respecte pas la casse, mais le mot de passe la respecte. L'implémentation du protocole PPP de Crimson demande à son homologue d'utiliser l'authentification CHAP pour éviter la transmission ou la réception des mots de passe en clair, mais a recours au protocole PAP si le serveur distant ne prend pas en charge l'authentification CHAP.

- La propriété *Type de Routage* permet de définir les données qui sont transférées via cette interface. Pour les connexions à la demande, elle définit efficacement l'activation de la connexion. Si Passerelle par défaut est sélectionnée, tous les paquets qui ne correspondent pas à l'adresse et au masque réseau de la connexion Ethernet sont envoyés vers cette interface. Notez que dans ce mode, le port Ethernet doit disposer d'un paramètre de passerelle de 0.0.0.0, sinon il prendra tous les paquets et n'en laissera aucun afin d'activer le modem ! Si Réseau spécifique est sélectionné, vous devez fournir l'adresse et le masque réseau qui définissent le réseau sur lequel les paquets sont routés.

AJOUT D'UNE CONNEXION SMS

Les connexions SMS sont utilisées lorsque la fonctionnalité de la messagerie écrite est requise, mais lorsque aucune connexion PPP d'appel entrant ou sortant n'est établie. Elles sont configurées comme nous venons de le décrire ci-dessus sauf que le périphérique *SMS via modem GSM* doit être sélectionné pour le port requis. Les options de configuration de ce modem sont affichées ci-dessous…



Les propriétés du périphérique sont un sous-ensemble de celles qui sont fournies pour les connexions d'appel entrant. La prise en charge de SMS est toujours active avec ce pilote, mais une fois encore, pour que la messagerie SMS fonctionne correctement, vous devez activer le transport SMS à l'aide de l'icône Mail qui se trouve dans la fenêtre Communications.

TRAITEMENT DES MESSAGES SMS

Lorsque la messagerie SMS est activée, le maître demande au modem GSM de vérifier la présence de messages entrants ou sortants toutes les cinq secondes. Les messages entrants sont transférés au Gestionnaire de Messages qui les transférera le cas échéant aux autres utilisateurs en fonction de sa configuration. Notez que vous pouvez vérifier la présence de messages lorsque le modem est connecté sur une session CSD ou GPRS, ce qui vous permet d'éviter d'utiliser les connexions permanentes lorsque vous travaillez avec le service SMS. Notez également que si plusieurs modems GSM sont configurés, ils peuvent tous recevoir des messages, mais seul le deuxième modem est utilisé pour l'envoi.

UTILISATION DE PLUSIEURS INTERFACES

Chaque module maître peut prendre en charge jusqu'à deux connexions indépendantes par modem. Associé avec le port Ethernet, cela fournit un total maximum de trois interfaces IP distinctes, le tout fonctionnant conformément aux paramètres de configuration définis pour

chaque connexion. Cette section décrit la façon dont plusieurs interfaces interagissent et la façon dont le maître décide de l'endroit où envoyer chaque paquet de données.

SELECTION DE L'INTERFACE

Chaque interface dispose d'une adresse IP et d'un masque réseau qui sont utilisés pour décider si les paquets doivent être transférés à cette interface. Par exemple, si l'interface Ethernet est configurée avec une adresse IP de 192.168.1.0 et avec un masque réseau de 255.255.255.0, tous les paquets des adresses IP commençant par 192.168.1 sont envoyés vers cette interface. De la même façon, si une connexion par modem à la demande dispose d'une adresse IP distante de 192.168.2.2 et d'un masque réseau de 255.255.255.255, l'envoi d'un paquet pour adresser 192.168.2.2 permettra d'établir la connexion.

ITINERAIRE PAR DEFAUT

De plus, une seule interface peut également définir un itinéraire par défaut qui est utilisé pour gérer les paquets qui ne correspondent pas expressément à toute autre interface. La méthode utilisée pour configurer l'itinéraire varie en fonction du type d'interface, comme le tableau ci-dessous le montre…

INTERFACE	POUR DEFINIR L'ITINERAIRE PAR DEFAUT
Ethernet	Saisissez une valeur autre que zéro pour la propriété *Passerelle*.
Appel entrant	Saisissez 0.0.0.0 pour la propriété *Masque distant*.
Appel sortant	Sélectionnez Passerelle par défaut pour la propriété *Type de Routage*.

Notez une nouvelle fois que seule une interface unique peut définir un itinéraire par défaut. Par exemple, un module maître peut être connecté à plusieurs périphériques Ethernet à l'aide d'une adresse IP de 192.168.1.0 et d'un masque réseau de 255.255.255.0, sans aucune définition de passerelle. Une connexion par modem à la demande peut être configurée pour accéder à un fournisseur de services Internet afin d'envoyer des messages électroniques (E-mail) d'alarme. Son *Type de Routage* est défini sur Passerelle par défaut, ce qui en fait l'itinéraire pour tous les paquets des adresses IP qui ne correspondent pas au réseau défini du port Ethernet. Le serveur SMTP est configuré en tant que 24.104.0.39, entraînant une connexion d'appel sortant lorsque vous tentez d'envoyer un message.

ROUTAGE IP

L'icône Ethernet de la fenêtre Communications contient une propriété appelée *Routage IP*. Si cette fonction est activée, les paquets entrants à partir des interfaces de modem non pare-feu sont comparés à l'adresse IP et au masque réseau de l'interface Ethernet et sont transférés à cette interface en cas de correspondance. Cette fonction est la plus souvent utilisée avec les connexions d'appel entrant et permet un accès IP à tous les périphériques connectés au port Ethernet sous réserve qu'un itinéraire adapté soit défini par le client.

VERIFICATION DU STATUT DU MODEM

Pour déboguer les connexions par modem, Crimson propose la fonction **`GetInterfaceStatus`**. Cette fonction prend un seul argument qui est l'index numérique de l'interface requise. L'interface zéro est toujours l'interface de bouclage du maître. Ensuite vient l'interface Ethernet, si elle est activée, de telle façon que la première interface PPP est numérotée 1 lorsque l'Ethernet est désactivé et 2 lorsqu'il est activé.

La fonction renvoie une chaîne qui peut être interprétée selon le tableau suivant…

STATUT	SIGNIFICATION
CLOSED (Fermé)	L'interface n'a pas encore été initialisée. Cet état se produit uniquement sur une courte période lors du démarrage du système.
INIT (Initialisation)	Le modem est en cours d'initialisation. Si la connexion reste sur cet état, c'est qu'il existe probablement des erreurs dans les chaînes d'initialisation qui sont envoyées au modem.
IDLE (Inactif)	Le lien est inactif. Les modems GSM renvoient un nombre à la fin de la chaîne pour indiquer la force du signal. Le tableau suivant décrit la façon d'interpréter ces valeurs.
SMS	Le modem envoie des messages SMS ou demande au modem de vérifier si de nouveaux messages SMS sont disponibles. Si la messagerie SMS est activée pour un modem, cet état s'affiche pendant une courte période toutes les cinq secondes.
CONNECTING (Connexion)	Le modem est en train d'établir une connexion. Cet état s'affiche en général uniquement pour les connexions clientes et indique qu'un appel est effectué.
LISTENING (Ecoute)	Le modem attend un appel. Cet état s'affiche uniquement pour les connexions serveur. Notez que les modems GSM renvoient également un état INACTIF lorsqu'ils attendent un appel afin d'indiquer la force du signal.
ANSWER (Réponse)	Le modem répond à un appel et essaie de négocier le débit en bauds de la connexion. Cet état s'affiche uniquement pour les connexions serveur. Si la connexion est établie, le modem entre l'état CONNECTE.
CONNECTED (Connecté)	Le modem a établi une connexion. Cet état persiste pendant une courte durée, le temps que le processus de négociation LCP commence après un petit délai.
NEG LCP	La connexion est en train de négocier les options LCP. Ce processus décide d'un groupe de paramètres du protocole de lien qui sont acceptables pour le client et le serveur.
AUTH	La connexion effectue le processus d'authentification pour garantir que les informations d'identification appropriées de l'utilisateur sont utilisées.

STATUT	SIGNIFICATION
NEG IPCP	La connexion est en train de négocier les options IPCP. Ce processus décide d'un groupe de paramètres du protocole réseau qui sont acceptables pour le client et le serveur.
UP (Active)	La connexion est active et les données IP peuvent être échangées.
HANG UP (Fin de la communication)	Le modem est en cours de déconnexion. Cet état existe uniquement pendant une courte durée avant que le modem ne revienne sur INACTIF.

Les valeurs de la force du signal renvoyées par les modems GSM ont la signification suivante…

VALEUR	FORCE DU SIGNAL
0	–113 dBm ou moins.
1	–111 dBm.
2-30	–109 dBm a –52 dBm par étapes de 2dBm.
31	–51 dBm ou plus.
99	La force du signal ne peut pas être déterminé.

Les téléphones portables interprètent généralement ces valeurs, comme décrit ci-dessous lors de l'affichage de la force du signal…

VALEUR	FORCE	NOMBRE DE BARS
5 ou moins.	–103 dBm ou moins.	Aucun
6 à 9.	–101 dBm à –95 dBm	Deux
10 à 14.	–93 dBm à –85 dBm	Trois
15 ou plus.	–83 dBm ou plus.	Quatre

CONFIGURATION DES ETIQUETTES DE DONNEES

Pour les applications qui nécessitent plus que la fonctionnalité de mapping de données vers et à partir des cartes du Modular Controller, Crimson permet de créer des étiquettes.

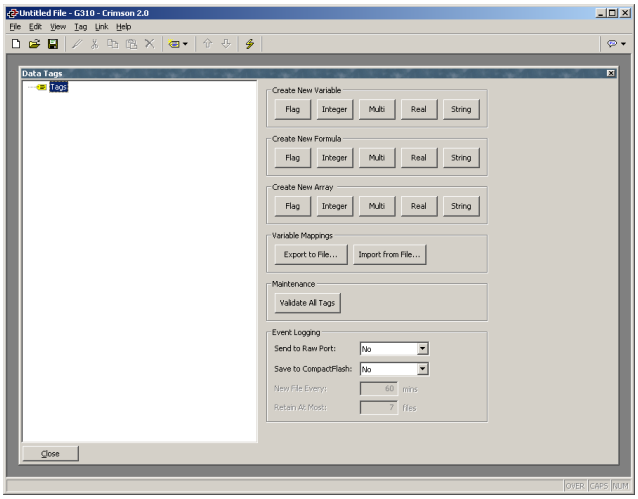
Notez que ce chapitre est écrit en supposant que vous configurez un maître avec un HMI virtuel couleur. Si vous travaillez avec le maître configuré pour avoir un HMI virtuel monochrome, vous remarquerez que les étiquettes ne disposent pas de l'onglet Couleurs et que certaines autres fonctions peuvent ne pas être présentes. Les fonctionnalités qui ne sont pas prises en charge sur la version mono sont marquées d'un astérisque.

TOUT SUR LES ETIQUETTES

Les étiquettes de données sont des entités nommées (Mnémoniques) qui représentent des éléments de données dans le maître (CSMSTRSX ou GT ou Data Station Plus ou G3). Les étiquettes peuvent être « mappées » sur les variables dans les modules attachés ainsi que sur les registres des périphériques distants (API). De plus, Crimson lit automatiquement les valeurs correspondantes lorsque l'étiquette est référencée ou affichée. De la même façon, si vous modifiez une étiquette mappée, Crimson écrit automatiquement la nouvelle valeur sur le périphérique distant.

TYPES D'ETIQUETTES

Lorsque vous ouvrez la fenêtre Etiquettes de données pour la première fois, le volet droit contient un nombre apparemment important de boutons qui peuvent être utilisés pour créer différents types d'étiquettes de données. Alors que tous ces boutons peuvent tout d'abord sembler un peu intimidants, les quinze différents types d'étiquettes peuvent être répartis en trois familles, chacune contenant cinq membres.



FAMILLES D'ETIQUETTES

Les trois familles d'étiquettes sont répertoriées ci-dessous…

- *Variables* représente un seul élément de données dans le maître. Les variables peuvent être mappées aux registres de l'API et configurées sur Avec rétention,

auquel cas leurs valeurs sont conservées dans la mémoire même lorsque le maître (CSMSTRSX ou GT ou Data Station Plus ou G3) est hors tension. La définition d'une variable se caractérise par la présence d'un seul élément et par le fait qu'il est *en théorie* possible d'écrire sur cet élément même si en pratique, la variable est configurée sur lecture seule. (Si cela ne vous semble pas très clair, continuez à prendre connaissance de ce qui suit et vous verrez la façon dont cela contraste avec une formule, qui n'est pas dotée de cette propriété.)

- *Formules* représente des valeurs dérivées. Ce sont une combinaison des autres éléments de données, généralement associés à l'aide d'une ou plusieurs opérations mathématiques. Par exemple, une formule peut représenter la somme de deux niveaux de réservoirs. Même si une formule peut être définie pour être égale au contenu d'un registre de module, elle n'est pas vraiment mappée sur ce registre dans le sens où elle ne peut *jamais* être écrite et ne peut donc pas être considérée comme étant équivalente à ce registre. Le besoin de cette restriction est évident si vous considérez une formule telle que **`Temperature1+Temperature2`**. Que pourrait signifier l'écriture sur cette expression ?

- *Tableaux* représente un ensemble d'éléments de données dans le maître (CSMSTRSX ou GT ou Data Station Plus ou G3). Ils sont généralement utilisés pour enregistrer les données de recette ou pour créer des groupes de données en vue d'une analyse statistique. Ils ne sont pas utilisés dans la majorité des applications simples, mais fournissent un outil puissant pour la plupart des projets complexes.

TYPES D'ETIQUETTES

Chaque famille contient cinq types d'étiquettes et chacun d'entre eux possède un type de données différent…

- Les étiquettes *de type bit* représentent une seule condition vraie ou fausse. Lorsqu'elles sont mappées sur un registre dans un périphérique distant, c'est qu'elles correspondent en général à une bobine interne ou à un seul point d'E/S numérique. Les formules *bit* représentent généralement des combinaisons de ces éléments ou des comparaisons des valeurs numériques.

- Les étiquettes *Entier* représentent des nombres signés sur 32 bits. Ces étiquettes peuvent enregistrer des valeurs entre –2 147 483 648 et +2 147 483 647. Même si une étiquette est mappée sur un registre de l'API contenant uniquement 16 bits de données, Crimson effectue ses opérations internes à un niveau de précision supérieur pour garantir que les grandes valeurs intermédiaires peuvent être gérées en toute simplicité.

- Les étiquettes *Multi* représentent des valeurs numériques qui correspondent à plusieurs états distincts. Ainsi, même si un entier peut représenter un niveau de réservoir, une étiquette Multi représente l'un des trois états d'une machine comme Arrêté, En cours d'exécution ou En Pause. La distinction est évidente si

vous considérez qu'une étiquette Multi s'affiche comme une chaîne d'un jeu de chaînes alors qu'un entier s'affiche comme un nombre.

- Les étiquettes *Réel* représentent des nombres en virgule flottante simple précision à 32 bits. Ces étiquettes peuvent enregistrer des valeurs entre $\pm 10^{-38}$ et $\pm 10^{+38}$ avec une précision d'environ 7 chiffres significatifs. Même s'il est rarement nécessaire d'utiliser des étiquettes Réel pour représenter des quantités physiques (qui ont généralement plus de plages définies), elles sont utiles pour effectuer des opérations statistiques ou d'autres fonctions mathématiques.

- Les étiquettes *Chaîne* représentent un élément de texte constitué de plusieurs caractères (Chaîne ASCII). Elles sont utilisées pour enregistrer des éléments comme des noms de recettes ou pour traiter les données reçues à l'aide des pilotes du périphérique Raw port. Les chaînes ne peuvent pas être mappées sur les registres de l'API, mais peuvent être utilisées pour enregistrer ces données dans le maître (CSMSTRSX ou GT ou Data Station Plus ou G3) lui-même.

POURQUOI UTILISER DES ETIQUETTES ?

Etant donné toutes ces différentes options, vous pouvez vous demander pourquoi vous devriez utiliser des étiquettes ? Après tout, Crimson vous permet de placer directement un module ou un registre d'API sur une page d'affichage de façon à ce que vous puissiez configurer une base de données simple sans ne jamais ouvrir l'icône Etiquettes de données. Les réponses de base se présentent comme suit…

- Les étiquettes vous permettent de nommer des éléments de données (Mnémonique). Ainsi, vous pouvez savoir à quels éléments de données de l'API vous vous référez. D'autre part, si les données de l'API changent ou si vous décidez de basculer dans une famille d'API complètement différente, vous pouvez simplement mapper à nouveau les étiquettes et éviter de devoir apporter d'autres modifications à votre base de données.

- Les étiquettes vous permettent d'éviter de devoir rentrer les mêmes informations plusieurs fois. Lorsque vous créez une étiquette, vous spécifiez la façon dont l'étiquette s'affiche. Dans le cas d'une étiquette Entier, cela signifie que vous demandez à Crimson combien de décimales seront utilisées et les unités, le cas échéant, qui seront ajoutées à la valeur. Lorsque vous placez une étiquette sur une page d'affichage, Crimson sait comment la formater sans que vous ne deviez rien faire d'autre. De la même façon, si vous décidez de modifier la mise en forme et, peut-être, de basculer d'un jeu d'unités vers un autre, vous pouvez le faire en un endroit sans devoir modifier les pages d'affichage les unes après les autres.

- Sur les maîtres (CSMSTRSX ou GT ou Data Station Plus ou G3) configurés pour avoir un HMI virtuel couleur, les étiquettes sont utilisées comme base de l'animation couleur. Les différentes couleurs d'une étiquette peuvent être utilisées pour spécifier les autres primitives d'animation qui sont affichées. Sans les étiquettes, vous ne pouvez absolument pas modifier la couleur de toute autre chose que les champs de données basés sur le texte.

- Les étiquettes représentent la clé de la mise en œuvre des protocoles esclaves. Crimson considère ces protocoles comme des mécanismes d'exposition des éléments de données dans le maître (CSMSTRSX ou GT ou Data Station Plus ou G3). Cela permet d'accéder aux mêmes données via plusieurs ports de façon que, par exemple, une configuration de machine pourrait être modifiée par un package SCADA local et un package semblable fonctionnant sur Ethernet à partir d'un site distant. Sans les étiquettes, rien ne serait exposé et ce mécanisme ne pourrait pas être mise en œuvre !

- Les étiquettes sont utilisées chez Crimson pour implémenter de nombreuses fonctionnalités avancées. Si vous souhaitez utiliser des fonctionnalités comme les alarmes, les Triggers, l'enregistrement de données ou le serveur Web, vous devrez utiliser des étiquettes. Les données de mise en forme à partir de la définition de l'étiquette sont généralement requises par toutes ces fonctionnalités, ainsi les étiquettes sont obligatoires pour garantir le fonctionnement.

Pour résumer, les étiquettes automatisent de nombreuses tâches lors de la programmation, vous permettant de gagner du temps. Même si vous décidez de ne pas utiliser les étiquettes, la majorité des chapitres ultérieurs de ce manuel se rapportent aux concepts abordés dans ce chapitre. Vous devez donc le lire soigneusement avant de continuer.

CREATION D'ETIQUETTES

Pour créer une étiquette, il vous suffit de cliquer sur l'un des boutons qui s'affichent lorsque vous sélectionnez l'icône Etiquettes dans le volet gauche de la fenêtre Etiquettes de données ou d'utiliser les nouveaux boutons d'étiquettes de la barre d'outils. Quelle que soit la façon que vous choisissez, une nouvelle étiquette est ajoutée dans la liste d'étiquettes. Pour ajouter le nom d'étiquette, sélectionnez l'étiquette dans le volet gauche, puis tapez le nouveau nom.

Les noms d'étiquettes doivent être conformes aux règles suivantes…

- Les noms d'étiquettes ne peuvent pas contenir d'espaces ou de signes de ponctuation.

- Les noms d'étiquettes doivent démarrer avec une lettre de l'alphabet ou un trait de soulignement (Under score).

- Les caractères suivants doivent être des chiffres, des lettres ou des traits de soulignement (Under score).

- Les noms d'étiquettes ne doivent pas dépasser 24 caractères de longueur.

MODIFICATION D'ETIQUETTES

Lorsque vous sélectionnez une étiquette dans le volet gauche, le volet droit affiche plusieurs onglets et chacun d'entre eux indique certaines propriétés de l'étiquette. En fonction de la famille et du type de l'étiquette, différents onglets peuvent être présents et chaque étiquette peut contenir différents champs.

Quel que soit le type d'étiquette sélectionné, **le premier onglet du volet droit est toujours l'onglet Données**. Cet onglet contient des champs qui indiquent les données que l'étiquette doit représenter et la façon dont les données sont enregistrées et peut-être transférées vers et à partir d'un périphérique distant. Les contenu exact de l'onglet varie selon la famille et le type d'étiquette concerné.

Le deuxième onglet est toujours l'onglet Format. Il indique la façon dont les données de l'étiquette sont formatées tel qu'indiqué sur l' HMI virtuel ou tel que présenté à un utilisateur par le biais de tout autre mécanisme comme une page Web. L'onglet Format a la même forme pour toutes les onglets de telle façon que toutes les étiquettes d'entier par exemple partagent le même ensemble de propriétés.

Le reste des onglets définit les alarmes et les Triggers de l'étiquette. Ils ne sont pas compris dans les étiquettes de chaînes ou pour les tableaux**. Les alarmes** sont utilisées pour détecter une condition qui doit être portée à l'attention de l'opérateur ou simplement pour enregistrer le fait que la condition s'est produite. **Les Triggers** fonctionnent de façon semblable, mais au lieu d'enregistrer la condition, ils sont utilisés pour exécuter une action.

MODIFICATION DE PROPRIETES

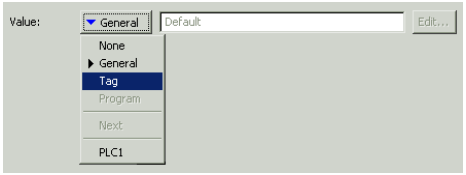
La majorité des propriétés sont modifiées d'une façon qui est évidente à quiconque a utilisé un système d'exploitation Windows. Par exemple, vous devrez peut-être saisir une valeur numérique ou sélectionner un élément à partir d'une liste déroulante. Toutefois, certains types de propriétés fournissent des options de modification plus complexes qui sont décrites ci-dessous.

PROPRIETES EXPRESSION

Les propriétés Expression peuvent être définies sur…

- Une valeur constante.
- Le contenu de l'étiquette de données.
- Le contenu d'un registre dans un périphérique de communication distant.
- Une combinaison de ces éléments liés les uns aux autres à l'aide de différentes opérateurs mathématiques.

Dans son état par défaut, le bouton fléché situé immédiatement après le nom de la propriété indique que le champ est en mode Général et la zone de modification située à droite du bouton indique une chaîne grisée montrant le comportement par défaut de la propriété…



Si vous maîtrisez la syntaxe d'expression de Crimson (vous pouvez en trouver une description complète dans la section Ecriture d'expressions), vous pouvez modifier la

AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

propriété en tapant directement une expression dans la zone de modification. Pourtant, la plupart des utilisateurs préfèrent appuyer sur le bouton fléché et faire leur choix dans le menu d'options qui s'affiche...

- L'option *Etiquette* affiche une boîte de dialogue qui contient une liste d'étiquettes de données. Vous pouvez sélectionner l'étiquette que vous souhaitez utiliser pour contrôler cette propriété. Dans certains cas, vous pouvez également créer une nouvelle étiquette et définir ses propriétés de base. Cette option n'est pas disponible lors de la modification des propriétés qui appartiennent directement aux autres étiquettes de données car il est très facile d'oublier l'étiquette que vous modifiez !

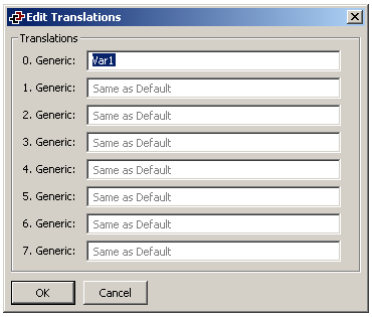
- L'option *Nom de périphérique* affiche une boîte de dialogue qui vous permet de choisir un registre dans le périphérique de communication distant. Les différents périphériques de communication sont répertoriés à la fin du menu selon leur ordre de création.

- L'option *Suivant* définit que la propriété est égale au registre qui suit le dernier registre sélectionné dans le dernier périphérique sélectionné. Par exemple, si vous avez utilisé l'option *Nom de périphérique* pour définir une propriété précédente sur **`N7:10`** de l'API1, en choisissant l'option *Suivant*, la propriété actuelle du même périphérique est définie sur **`N7:11`**.

GESTION DES LANGUES

Les bases de données Crimson sont conçues pour prendre en charge un fonctionnement multilingue grâce à laquelle toute chaîne qui est présentée à l'utilisateur de l' HMI virtuel peut être affichée dans l'une des différentes langues. Pour vous permettre de définir ces traductions, les propriétés qui contiennent ces chaînes disposent d'un bouton nommé Traduction, situé à leur droite.

Pour entrer les traductions, cliquez sur le bouton et la boîte de dialogue suivante s'affiche…

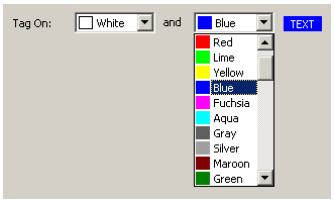


Si vous n'entrez pas de texte pour une langue particulière et que l'opérateur choisit ensuite cette langue, Crimson utilise alors la langue par défaut. Notez que Crimson configure Windows pour qu'il utilise l'éditeur de méthode d'entrée lorsqu'une langue complexe (par exemple, une langue basée sur l'Unicode) est modifiée sur un maître (CSMSTRSX ou GT ou Data Station Plus ou G3) configuré avec un HMI virtuel couleur. Pour de plus amples

informations sur la sélection des langues de la base de données et sur la configuration d'une touche ou d'un menu pour sélectionner une langue différente, reportez-vous à la section Interface utilisateur du présent manuel.

PROPRIETES COULEUR*

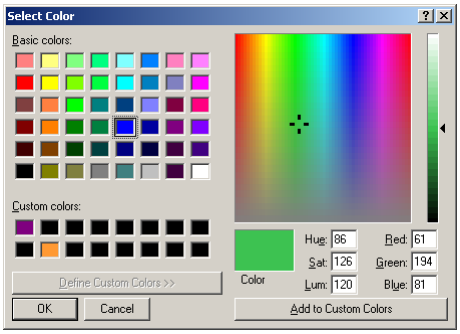
Les propriétés Couleur de l'étiquette représentent une couleur de premier plan et d'arrière-plan, qui est utilisée pour afficher l'état de l'étiquette en forme textuelle. Ces couleurs peuvent être utilisées pour définir la couleur des autres primitives d'animation. L'exemple ci-dessous présente un couple de couleurs en cours de modification…



La liste déroulante contient les couleurs suivantes…

- Les seize couleurs VGA standard.
- Les seize couleurs personnalisées, définies par l'utilisateur.
- Quatorze tons de gris qui varient entre le noir et blanc.

L'option Autres couleurs, située en bas de la liste, peut être utilisée pour appeler la boîte de dialogue de définition des couleurs…



Grâce à cette boîte de dialogue, vous pouvez définir une couleur de plusieurs façons. Vous pouvez les choisir dans la palette, dans la fenêtre de « l'arc en ciel » ou entrer les paramètres TSL ou RVB explicites. La boîte de dialogue vous permet également d'ajouter des couleurs personnalisées à la palette. Elles s'affichent dès que la boîte de dialogue est appelée ainsi que dans la liste déroulante décrite ci-dessus. Notez que les couleurs qui s'affichent dans « l'arc-en-ciel » ne peuvent pas toutes être rendues sur l'écran 256 couleurs du HMI virtuel. Crimson choisit la couleur la plus proche dans les capacités du périphérique.

* Non présentes sur le HMI virtuel monochrome.

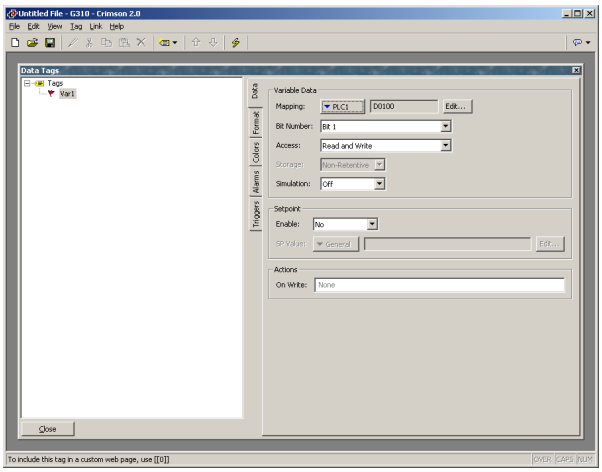
AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - http://www.redlion.net/controller
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

MODIFICATION D'ETIQUETTES DE TYPE BIT

Rappelez-vous que les étiquettes de type bit représentent une valeur vraie ou fausse. Les sections suivantes décrivent les différents onglets qui s'affichent dans la partie droite de la fenêtre Etiquettes de données lors de la modification de l'un des différents types d'étiquettes de type bit.

L'ONGLET DONNEES (VARIABLES)

L'onglet Données d'une variable *bit* contient les propriétés suivantes...



- La propriété *Mapping* permet de spécifier si la variable doit être mappée sur un registre dans un périphérique distant ou si elle existe uniquement dans le maître (CSMSTRSX ou GT ou Data Station Plus ou G3). Si vous appuyez sur le bouton fléché pour sélectionner un nom de périphérique dans le menu, une boîte de dialogue s'affiche, permettant se sélectionner le registre de l'API.

- La propriété *Numéro de bit* est utilisée lorsqu'une variable indicateur est mappée sur un registre d'API qui contient plusieurs bits d'informations (Bit de mot). La propriété est alors utilisée pour indiquer le bit du registre auquel l'étiquette doit accéder.

- La propriété *Accès* permet de spécifier quelle sorte de transfert de données doit être effectuée pour une variable mappée. Vous pouvez indiquer que les données sont lues et écrites ou simplement lues ou écrites, le cas échéant. Les étiquettes en écriture seule peuvent être utilisées pour éviter les opérations de lecture inutiles sur des données qui ne peuvent être modifiées que par le maître (CSMSTRSX ou GT ou Data Station Plus ou G3). Elles sont généralement définies sur Avec rétention car leur valeur ne peut pas être obtenue à partir d'un API et doit donc être enregistrée par le maître (CSMSTRSX ou GT ou Data Station Plus ou G3).

- La propriété *Mémoire* est utilisée pour indiquer si Crimson doit attribuer de la mémoire FLASH dans le maître (CSMSTRSX ou GT ou Data Station Plus ou G3) pour conserver la valeur de l'étiquette lors de sa mise hors tension. Les étiquettes mappées qui ne sont pas en écriture seule ne peuvent pas être définies

sur Avec rétention car leurs valeurs sont lues à partir de l'API. Par conséquent, il n'est pas logique de gâcher de la mémoire locale pour conserver des données qui sont écrasées.

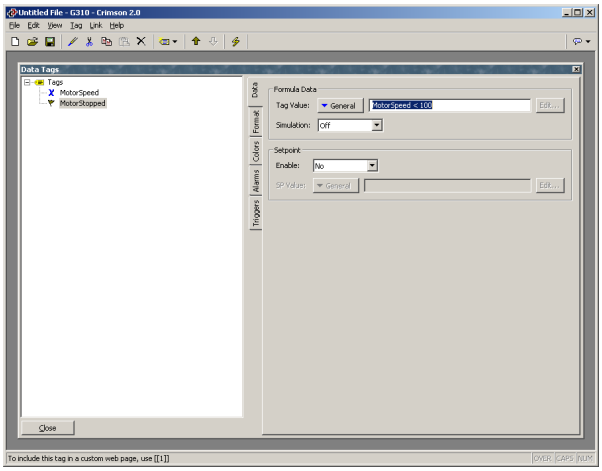
- La propriété *Simulation* permet de sélectionner la valeur que Crimson attribue à cette étiquette lorsqu'elle est affichée dans l'écran de l'éditeur de texte. Cette fonction peut être utile pour documenter les bases de données car elle permet de configurer une page d'affichage pour qu'elle représente un état de machine particulier de telle façon qu'une capture d'écran puisse alors être collée dans le manuel d'opérateur ou dans d'autres documentations.

- La propriété *Point de consigne* permet d'indiquer si un point de consigne est spécifié pour cette étiquette et se que ce point de consigne fera. Les points de consigne sont utilisés par certains modes d'alarme et permettent de comparer l'état réel d'une étiquette à son état voulu. Par exemple, une étiquette qui représente l'état d'une entrée d'un variateur de vitesse peut avoir la sortie de commande du moteur spécifiée comme point de consigne. Ainsi, une alarme peut être programmée pour s'activer si le démarrage du moteur échoue.

- La propriété *Sur Ecriture* permet de définir une action qui est exécutée lorsqu'une modification est apportée à l'étiquette. Vous pouvez utiliser cette action pour mettre à jour des valeurs dépendantes ou pour effectuer d'autres actions spécifiques à la base de données. Soyez attentif à ne pas effectuer des actions qui sont trop complexes ou les performances du système pourraient être réduites.

L'ONGLET DONNEES (FORMULES)

L'onglet Données d'une formule indicateur contient les propriétés suivantes...

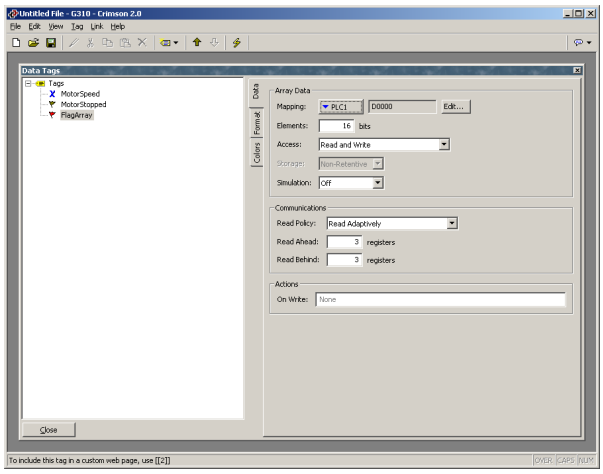


- La propriété *Valeur d'étiquette* permet de spécifier la valeur qui est représentée par cette étiquette. Elle est généralement définie sur une combinaison logique d'autres étiquettes ou de registres d'API ou sur une comparaison entre des valeurs numériques. Dans l'exemple ci-dessus, l'étiquette est configurée pour être vraie lorsque la vitesse d'un moteur dépasse une certaine valeur.

- Les propriétés *Point de consigne* et *Simulation* sont identiques à celles qui sont décrites pour les variables *bits*.

L'ONGLET DONNEES (TABLEAUX)

L'onglet Données d'un tableau de *bits* contient les propriétés suivantes...



- La propriété *Eléments* permet d'indiquer le nombre d'éléments de données que le tableau doit contenir. Il est fait référence aux éléments du tableau à l'aide de crochets. Ainsi **`Array[0]`** est le premier élément, **`Array[n-1]`** le dernier et **`n`** est égal à la valeur saisie pour cette propriété.

- Les propriétés *Accès* et *Mémoire* sont identiques à celles qui sont décrites pour les variables *bits*.

- La propriété *Simulation* est identique à celle qui est décrite pour les variables *bits*. Notez que la valeur à simuler s'applique à tous les éléments du tableau. Si vous devez effectuer une simulation sur une base par élément, utilisez plusieurs formules pour ajouter un alias dans les éléments du tableau.

- La propriété *Méthode de lecture* permet de définir la façon dont Crimson lit les données des tableaux qui sont mappés sur des éléments de données distants. Le tableau ci-dessous répertorie les différentes stratégies qui peuvent être configurées, puis décrit leur mode de fonctionnement…

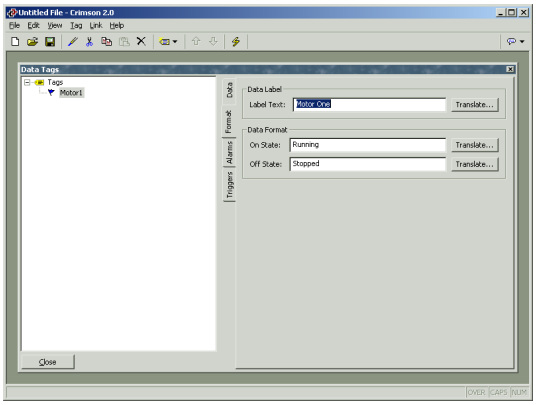
MODE	DESCRIPTION
Lecture adaptive	Tous les éléments référencés du tableau sont ajoutés à l'analyse des communications. Les données provenant d'un côté ou l'autre d'un élément référencé, tel que défini par les propriétés *Lire devant* et *Lire derrière*, sont également lues. Les anciennes données peuvent être affichées momentanément lorsqu'un élément provenant d'un tableau adaptatif est d'abord affiché sur l' HMI virtuel.

MODE	DESCRIPTION
Lecture manuelle	Le tableau est lu si et uniquement si la fonction **`ReadData`** est appelée. Ce mode est utile pour les éléments qui sont rarement en lecture seule ou qui sont connus pour ne pas changer dans le périphérique distant.
Lire tout le tableau	L'intégralité du tableau est ajouté à l'analyse des communications si tout élément du tableau est référencé. Ce mode garantit que tous les éléments de données sont disponibles avant d'être référencés, mais il peut réduire les performances du système.

- Les propriétés *Lire devant* et *Lire derrière* modifient le comportement de la méthode de lecture adaptive en contrôlant le nombre de registres adjacents qui sont lus lors du référencement d'un élément de tableau spécifique. Les lectures adjacentes permettent d'augmenter la possibilité que les données soient disponibles lorsque l'indirection est utilisée pour faire défiler un tableau vers le haut ou vers le bas. Les valeurs par défaut doivent être adaptées à la plupart des applications.
- La propriété *Sur Ecriture* est identique à celle qui est décrite pour les variables *bits*.

L'ONGLET FORMAT

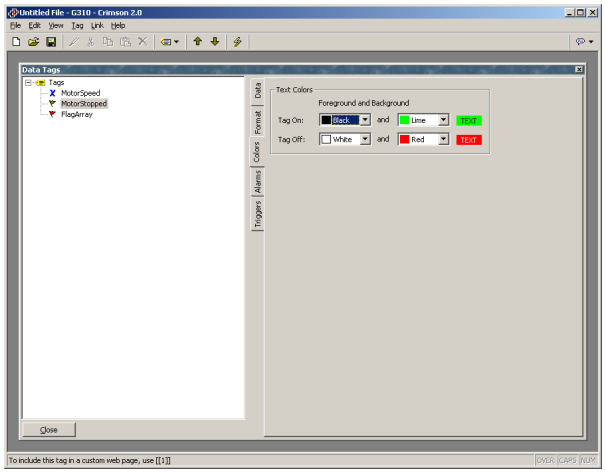
L'onglet Format d'une étiquette de type *bit* contient les propriétés suivantes...



- La propriété *Texte d'étiquette* permet de spécifier le nom qui est affiché en regard de cette étiquette lors de l'ajout de l'étiquette sur une page d'affichage. Le nom diffère du nom d'étiquette car le premier peut être traduit pour les applications internationales alors que le second reste inchangé et ne s'affiche jamais à l'utilisateur de l' HMI virtuel.
- Les propriétés *Etat activé* et *Etat désactivé* permettent de spécifier le texte qui est affiché lorsque l'étiquette contient respectivement une valeur à 1 et une valeur 0. Lorsque vous entrez le texte de la propriété Etat activé, Crimson essaie de générer le texte correspondant pour l'état désactivé en se référant aux étiquettes de type bit déjà créées et à sa liste interne de paires antonymiques courantes.

L'ONGLET COULEURS*

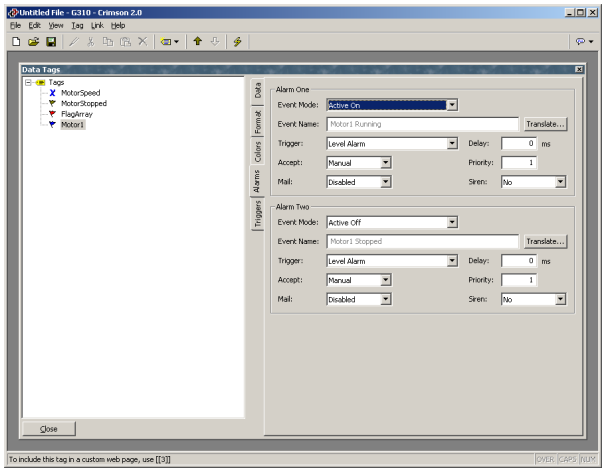
L'onglet Couleurs d'une étiquette de type bit contient les propriétés suivantes...



- La propriété *Etiquette activée* permet de définir la paire de couleurs utilisée pour afficher l'étiquette lorsqu'elle est sur l'état activé.
- La propriété *Etiquette désactivée* permet de définir la paire de couleurs utilisée pour afficher l'étiquette lorsqu'elle est sur l'état désactivé.

L'ONGLET ALARMES

L'onglet Alarmes d'une variable ou d'une formule bit contient les propriétés suivantes...



- La propriété *Condition d'alarme* permet d'indiquer la logique qui est utilisée pour décider si l'alarme doit s'activer. Les tableaux ci-dessous répertorient les modes disponibles.

* Non présentes sur le HMI virtuel monochrome.

MODE	ACTIVATION DE L'ALARME LORSQUE…
Actif ON	L'étiquette est vraie.
Actif OFF	L'étiquette est fausse.

Les modes suivants sont uniquement disponibles lorsqu'un point de consigne est défini…

MODE	ACTIVATION DE L'ALARME LORSQUE…
Pas égal à SP	L'étiquette n'est pas égale à son point de consigne.
Bit OFF et SP ON	L'étiquette ne répond pas à un point de consigne ON.
Bit ON et SP OFF	L'étiquette ne répond pas à un point de consigne OFF.
Pas égal à SP	L'étiquette est égale à son point de consigne.

- La propriété *Texte de l'alarme* permet de définir le nom qui s'affiche dans l'afficheur d'alarme ou dans l'afficheur d'événements, le cas échéant. Crimson propose un nom par défaut en fonction du nom de l'étiquette et la condition d'alarme qui a été sélectionnée.

- La propriété *Déclenchement* permet d'indiquer si l'alarme doit être déclenchée sur un front ou un niveau. Dans le premier cas (sur front), l'alarme se déclenche lorsque la condition spécifiée par la condition d'alarme devient vraie pour la première fois. Dans le second cas (de niveau), l'alarme continue dans son état actif alors que la condition persiste. Cette propriété permet également d'indiquer que cette alarme doit être utilisée uniquement en tant qu'événement. Dans ce cas, l'alarme est déclenchée sur front, mais elle n'entraîne pas une condition d'alarme. Un événement est alors enregistré dans la mémoire interne du maître (CSMSTRSX ou GT ou Data Station Plus ou G3).

- La propriété *Retard* permet d'indiquer la durée de la condition d'alarme avant l'activation de l'alarme. Dans le cas d'une alarme ou d'un événement déclenché(e) sur front, cette propriété spécifie également la durée pendant laquelle la condition d'alarme n'existe plus avant que les réactivations ultérieures provoquent le signalement d'une autre alarme. Exemple : si une alarme est définie pour s'activer lorsqu'un variateur de vitesse indique qu'un moteur ne fonctionne pas même lorsque son démarrage a été requis, cette propriété peut être utilisée pour fournir au moteur le temps nécessaire pour démarrer avant l'activation de l'alarme.

- La propriété *Acquittement* permet d'indiquer si l'utilisateur doit explicitement accepter une alarme avant que son affichage ne soit terminé. Les alarmes déclenchées sur front doivent toujours être acceptées manuellement.

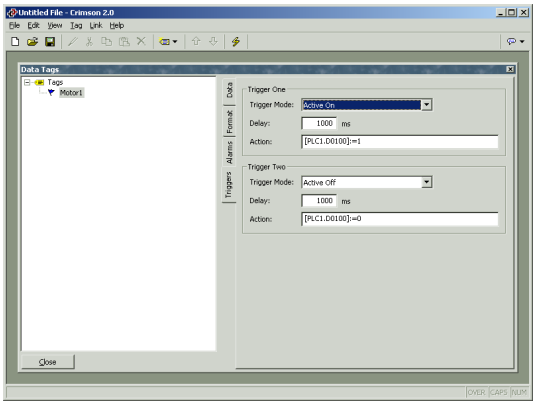
- La propriété *Priorité* permet de contrôler l'ordre dans lequel les alarmes sont affichées par l'afficheur d'alarme de Crimson. Plus la valeur numérique du champ Priorité est faible, plus l'alarme est affichée dans la partie supérieure.

- La propriété *Message* permet de spécifier l'entrée du carnet d'adresses à laquelle un message doit être envoyé lors de l'activation de cette alarme. Reportez-vous

au chapitre Communications avancées pour obtenir des informations sur la configuration du courrier électronique.

L'ONGLET TRIGGERS

L'onglet Triggers d'une variable ou d'une formule bits contient les propriétés suivantes...



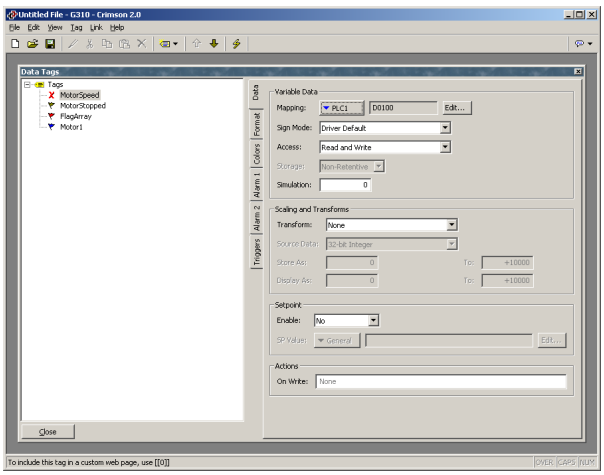
- La propriété *Condition d'alarme* est identique à celle de l'onglet Alarmes.
- La propriété *Retard* est identique à celle de l'onglet Alarmes.
- La propriété *Action* permet d'indiquer l'action qui doit être effectuée lors de l'activation du déclencheur. Reportez-vous à la section Ecriture d'actions pour obtenir une description de la syntaxe utilisée pour définir les différentes actions prises en charge par Crimson.

MODIFICATION D'ETIQUETTES D'ENTIER

Rappelez-vous que les étiquettes d'entier représentent une valeur signée à 32 bits. Les sections suivantes décrivent les différents onglets qui s'affichent dans la partie droite de la fenêtre Etiquettes de données lors de la modification de l'un des différents types d'étiquettes d'entier.

L'ONGLET DONNEES (VARIABLES)

L'onglet Données d'une variable entière contient les propriétés suivantes...



- La propriété *Mapping* permet de spécifier si la variable doit être mappée sur un registre dans un périphérique distant ou si elle existe uniquement dans le maître (CSMSTRSX ou GT ou Data Station Plus ou G3). Si vous appuyez sur le bouton fléché pour sélectionner un nom de périphérique dans le menu, une boîte de dialogue s'affiche, permettant se sélectionner le registre de l'API.

- La propriété *Signe* permet de remplacer le comportement par défaut du pilote de communication lors de la lecture des valeurs 16 bits à partir d'un périphérique distant. Le pilote décide normalement s'il doit traiter ces valeurs comme signées ou non signées en fonction de la quantité de données qui est normalement utilisée dans le périphérique. Si vous souhaitez passer outre cette décision, définissez la propriété comme requis.

- La propriété *Accès* est identique à celle qui est décrite pour les variables bit.

- La propriété *Mémoire* est identique à celle qui est décrite pour les variables bit.

- La propriété *Simulation* est identique à celle qui est décrite pour les variables bit.

- La propriété *Mise à l'échelle* permet de modifier la valeur de données car elle est lue et écrite à partir du périphérique distant. Lorsque vous sélectionnez le mode d'échelle linéaire, la gamme *Valeur registre* indique les limites supérieures et inférieures de la variable dans l'API alors que la gamme *Valeur affichée* indique les valeurs correspondantes telles qu'elles seront présentées à l'opérateur. La propriété *Données d'origine* peut également être utilisée dans ce mode pour forcer Crimson à traiter les données sous-jacentes comme valeur en virgule flottante avant d'effectuer la mise à l'échelle. Les autres modes se présentent comme suit…

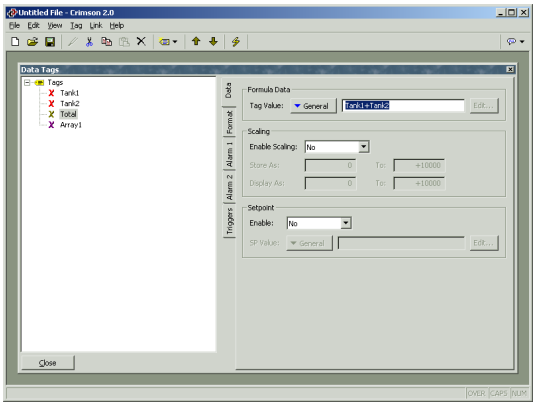
MODE	DESCRIPTION
BCD Vers Binaire	La valeur BCD est convertie en binaire.
Binaire vers BCD	La valeur binaire est convertie en BCD.
Inverser Octets dans Mots	Les deux octets inférieurs de la valeur sont inversés.
Inverser Octets dans Long	Les quatre octets de la valeur sont inversés.
Inverser Mots	Les mots supérieurs et inférieurs de la valeur sont inversés.
Miroir Bits dans Octets	Les octets 0 à 7 de la valeur sont rétablis.
Miroir Bits dans Mots	Les octets 0 à 15 de la valeur sont rétablis.
Miroir Bits dans Long	Les octets 0 à 31 de la valeur sont rétablis.
Inverser Bit dans Octets	Les octets 0 à 7 de la valeur sont inversés.
Inverser Bits dans Mots	Les octets 0 à 15 de la valeur sont inversés.
Inverser Bits dans Long	Les octets 0 à 31 de la valeur sont inversés.

- La propriété *Point de consigne* permet d'indiquer si un point de consigne est spécifié pour cette étiquette et ce à quoi il ressemble. Les points de consigne sont utilisés par certains modes d'alarme et permettent de comparer l'état réel d'une étiquette à son état voulu. Par exemple, une étiquette qui représente la température d'un navire peut disposer d'un point de consigne qui indique la température requise. Elle permet d'activer une alarme si le navire s'éloigne d'une certaine distance de sa cible.

- La propriété *Sur Ecriture* est identique à celle qui est décrite pour les variables bit.

L'ONGLET DONNEES (FORMULES)

L'onglet Données d'une formule entière contient les propriétés suivantes...



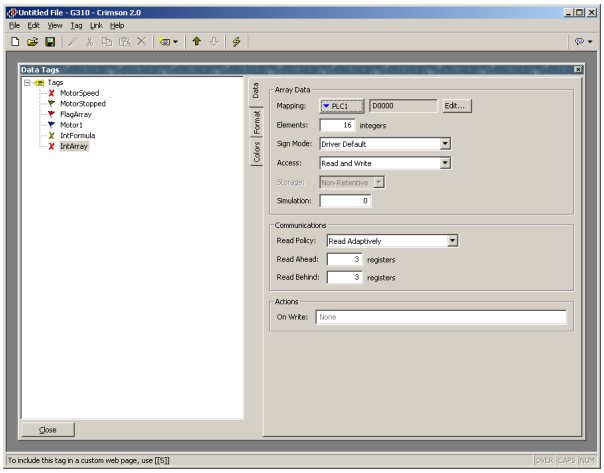
- La propriété *Valeur d'étiquette* permet de spécifier la valeur qui est représentée par cette étiquette. Elle est généralement définie sur une association de plusieurs autres étiquettes, liées les unes aux autres à l'aide d'opérateurs mathématiques. Dans l'exemple ci-dessus, l'étiquette est définie pour être égale à la somme des

deux niveaux de réservoir, indiquant de ce fait la quantité totale de matières premières disponibles.

- La propriété *Mise à l'échelle* est identique à celle qui est décrite pour les variables entières.
- La propriété *Point de consigne* est identique à celle qui est décrite pour les variables entières.

L'ONGLET DONNEES (TABLEAUX)

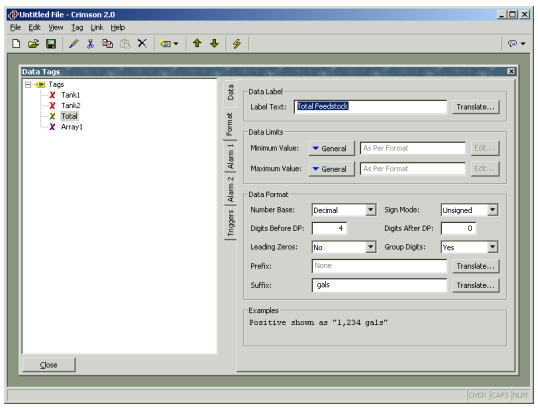
L'onglet Données d'un tableau d'entiers contient les propriétés suivantes...



- La propriété *Mapping* permet de spécifier si la variable doit être mappée sur un registre dans un périphérique distant ou si elle existe uniquement dans le maître. Si vous appuyez sur le bouton fléché pour sélectionner un nom de périphérique dans le menu, une boîte de dialogue s'affiche, permettant se sélectionner le registre de l'API.
- La propriété *Eléments* permet d'indiquer le nombre d'éléments de données que le tableau doit contenir. Il est fait référence aux éléments du tableau à l'aide de crochets. Ainsi **`Array[0]`** est le premier élément, **`Array[n-1]`** le dernier et **`n`** est égal à la valeur saisie pour cette propriété.
- La propriété *Signe* permet de remplacer le comportement par défaut du pilote de communication lors de la lecture des valeurs 16 bits à partir d'un périphérique distant. Le pilote décide normalement s'il doit traiter ces valeurs comme signées ou non signées en fonction de la quantité de données qui est normalement utilisée dans le périphérique. Si vous souhaitez passer outre cette décision, définissez la propriété comme requis.
- Les propriétés restantes sont identiques à celles qui sont décrites dans l'onglet Données des étiquettes de type bit.

L'ONGLET FORMAT

L'onglet Format d'une étiquette d'entiers contient les propriétés suivantes...



- La propriété *Texte d'étiquette* permet de spécifier le nom qui est affiché en regard de cette étiquette lors de l'ajout de l'étiquette sur une page d'affichage. Le nom diffère du nom d'étiquette car le premier peut être traduit pour les applications internationales alors que le second reste inchangé et ne s'affiche jamais à l'utilisateur du HMI virtuel.

- Les propriétés *Valeur minimum* et *Valeur maximum* permettent de définir les limites utilisées pour la saisie des données et de fournir des limites semblables aux différentes primitives graphiques qui doivent connaître les limites de variation de l'étiquette comme lors de la mise à l'échelle d'une valeur d'étiquette pour l'afficher en tant que graphique à barres.
- La propriété *Exprimer en* permet d'indiquer si l'étiquette doit être affichée en décimale, hexadécimale, binaire, octal ou par mot de passe. Les valeurs décimales peuvent être signées ou non signées alors que les autres bases numériques impliquent une opération non signée. En sélectionnant le mode Mot de passe, les astérisques s'affichent pour les valeurs saisies et ce mode est prévu à des fins de sécurité.

- La propriété *Signe* permet d'indiquer si un signe doit être préfixé ou non à la valeur de l'étiquette. Si vous sélectionnez un signe forcé, un signe positif ou négatif est préfixé le cas échéant. Si vous sélectionnez un signe facultatif, aucun signe positif ne s'affiche, mais un espace est préfixé.
- La propriété *Chiffres avant Virg* permet d'indiquer le nombre de chiffres à afficher avant la décimale ou, si aucun chiffre ne doit être affiché après la décimale, elle permet d'indiquer le nombre de chiffres à afficher au total.

- La propriété *Chiffres après Virg* permet d'indiquer le nombre de chiffres à afficher après la décimale. A l'évidence, les décimales ne sont pas prises en charge si vous avez sélectionné une base numérique autre que la décimale !

- La propriété *Zéros non significatifs* permet d'indiquer si des zéros doivent être affichés au début d'un chiffre ou s'ils doivent être remplacés par des espaces.

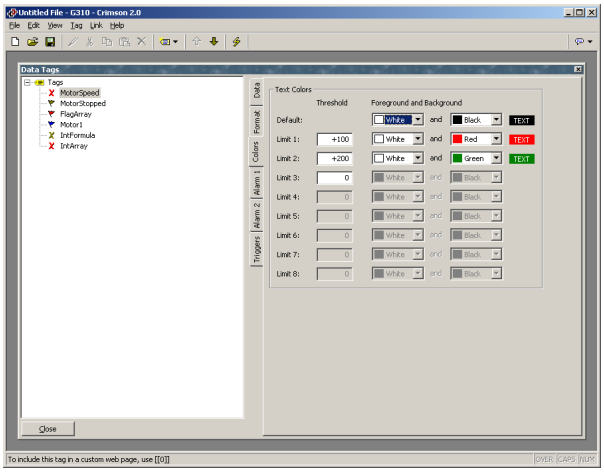
- La propriété *Grouper les chiffres* permet d'indiquer si les valeurs décimales doivent avoir des chiffres avant la décimale, groupés par trois et séparés par des

virgules. Une séparation semblable est effectuée sur les autres bases numériques à l'aide de groupements et de séparateurs qui sont adaptés à la base sélectionnée.

- La propriété *Préfixe* permet de spécifier une chaîne traduisible qui est affichée avant la valeur numérique. Elle est généralement utilisée pour indiquer des unités de mesure.
- La propriété *Suffixe* permet de spécifier une chaîne traduisible qui est affichée après la valeur numérique. Elle est aussi utilisée en règle générale pour indiquer des unités de mesure.

L'ONGLET COULEURS*

L'onglet Couleurs d'une étiquette d'entiers contient les propriétés suivantes...

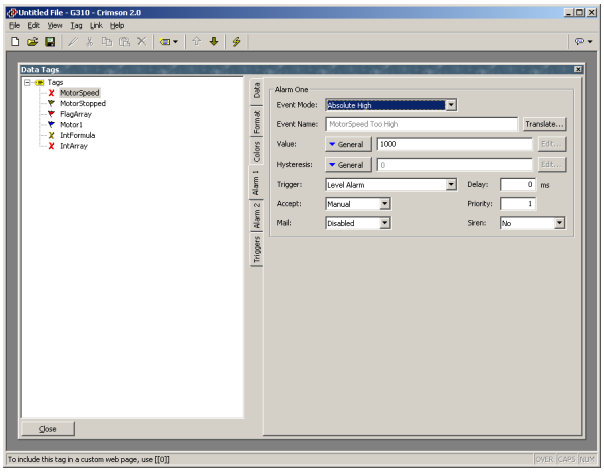


- La propriété *Par défaut* permet de définir la paire de couleurs utilisée pour afficher l'étiquette lorsque sa valeur est inférieure à la propriété *Limite 1*.
- Les propriétés restantes définissent des limites et les paires de couleurs qui sont utilisées pour afficher l'étiquette lorsque sa valeur est supérieure à la limite correspondante et inférieure à la limite suivante. Si la limite suivante est égale à zéro, la paire de couleurs est utilisée lorsque la valeur de l'étiquette dépasse la limite spécifiée.

* Non présentes sur le HMI virtuel monochrome.

LES ONGLETS ALARMES

Chaque onglet Alarme d'une variable ou d'une formule entière contient les propriétés suivantes...



- La propriété *Condition d'alarme* permet d'indiquer la logique qui est utilisée pour décider si l'alarme doit s'activer. Les tableaux ci-dessous répertorient les modes disponibles.

MODE	ACTIVATION DE L'ALARME LORSQUE...
Egal à	La valeur de l'étiquette est égale à la *valeur* de l'alarme.
Différent de	La valeur de l'étiquette n'est pas égale à la *valeur* de l'alarme.
Supérieur à	La valeur de l'étiquette est supérieure à la *valeur* de l'alarme.
Inférieur à	La valeur de l'étiquette est inférieure à la *valeur* de l'alarme.

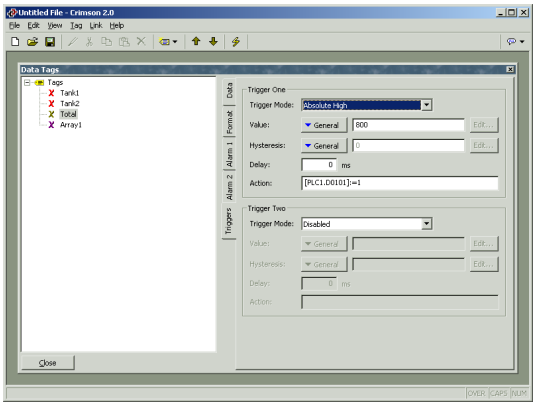
Les modes suivants sont uniquement disponibles lorsqu'un point de consigne est défini…

MODE	ACTIVATION DE L'ALARME LORSQUE...
Déviation haute	La valeur de l'étiquette est supérieure à la valeur du *point de consigne* de l'étiquette d'un montant égal à ou supérieur à la *valeur* de l'alarme.
Déviation basse	La valeur de l'étiquette est inférieure à la valeur du *point de consigne* d'un montant supérieur ou égal à la *valeur* de l'alarme.
Hors de la bande	L'étiquette sort d'une bande égale en largeur à deux fois la *valeur* de l'alarme et centrée sur le *point de consigne* de l'étiquette.
Dans la bande	L'étiquette entre dans une bande égale en largeur à deux fois la *valeur* de l'alarme et centrée sur le *point de consigne* de l'étiquette.

- La propriété *Valeur* permet de définir la valeur absolue à laquelle l'alarme est activée ou la déviation à partir de la valeur du point de consigne. L'interprétation exacte dépend de la condition d'alarme telle que décrite ci-dessus.
- La propriété *Hystérésis* empêche l'oscillation d'une alarme entre les états activé et désactivé lorsque le processus est proche de la condition d'alarme. Par exemple, pour une alarme supérieure, l'alarme est activée lorsque l'étiquette dépasse la valeur de l'alarme, mais elle se désactive uniquement lorsque l'étiquette est inférieure à la valeur par un montant supérieur ou égal à l'hystérésis de l'alarme. Gardez à l'esprit que la propriété agit toujours pour maintenir une alarme une fois qu'elle est activée et non pas pour modifier le point sur lequel l'activation se produit.
- Les propriétés restantes sont identiques à celles qui sont décrites dans l'onglet Alarmes des étiquettes de type bit.

L'ONGLET TRIGGERS

L'onglet Triggers d'une variable ou d'une formule entière contient les propriétés suivantes...



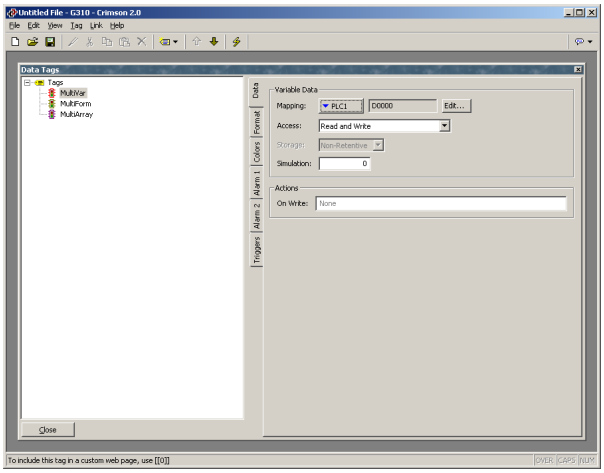
- La propriété *Condition d'alarme* est identique à celle de l'onglet Alarme.
- La propriété *Retard* est identique à celle de l'onglet Alarmes d'une étiquette de type bit.
- La propriété *Action* permet d'indiquer l'action qui doit être effectuée lors de l'activation du Trigger. Reportez-vous à la section écriture d'actions pour obtenir une description de la syntaxe utilisée pour définir les différentes actions prises en charge par Crimson.

MODIFICATION D'ETIQUETTES MULTI

Rappelez-vous que les étiquettes Multi représentent une valeur signée à 32 bits et qu'elles sont utilisées pour sélectionner l'une des chaînes de texte. Les sections suivantes décrivent les différents onglets qui s'affichent dans la partie droite de la fenêtre Etiquettes de données lors de la modification d'une étiquette Multi.

L'ONGLET DONNEES (VARIABLES)

L'onglet Données d'une multi variable contient les propriétés suivantes...

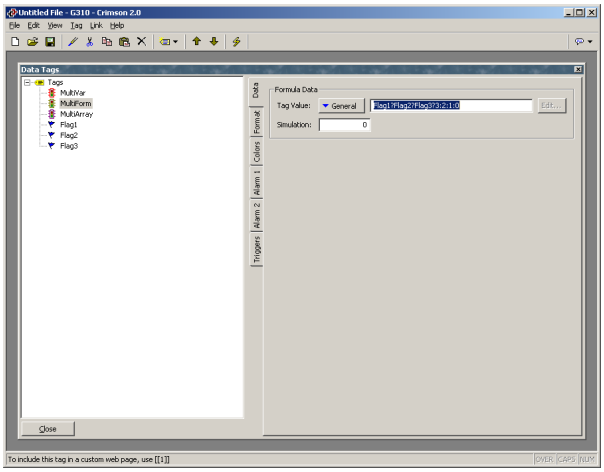


- La propriété *Mapping* permet de spécifier si la variable doit être mappée sur un registre dans un périphérique distant ou si elle existe uniquement dans le maître (CSMSTRSX ou GT ou Data Station Plus ou G3). Si vous appuyez sur le bouton fléché pour sélectionner un nom de périphérique dans le menu, une boîte de dialogue s'affiche, permettant se sélectionner le registre de l'API.

- Les propriétés restantes sont identiques à celles qui sont décrites dans l'onglet Données des étiquettes de type bit.

L'ONGLET DONNEES (FORMULES)

L'onglet Données d'une multi formule contient les propriétés suivantes...



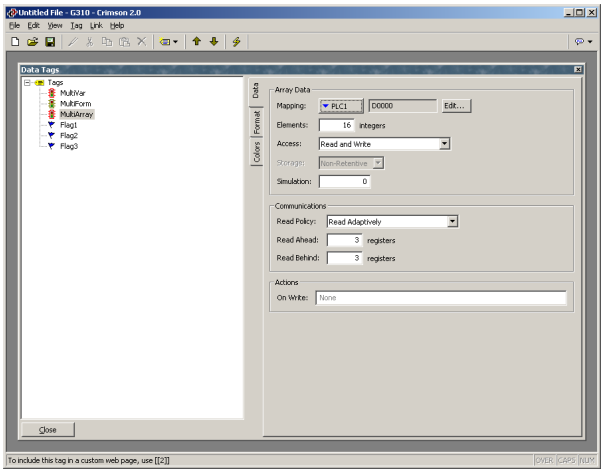
- La propriété *Valeur d'étiquette* permet de spécifier la valeur qui est représentée par cette étiquette. Elle est généralement définie sur une association de plusieurs autres étiquettes, liées les unes aux autres à l'aide d'opérateurs mathématiques. Dans l'exemple ci-dessus, l'étiquette est définie sur égale à une valeur de un, deux ou trois selon l'état de trois étiquettes différentes. Pour en savoir plus sur

l'opérateur **?:** utilisé dans cet exemple, reportez-vous à la section Ecriture d'expression.

- La propriété *Simulation* est identique à celle qui est décrite pour les étiquettes de type bit.

L'ONGLET DONNEES (TABLEAUX)

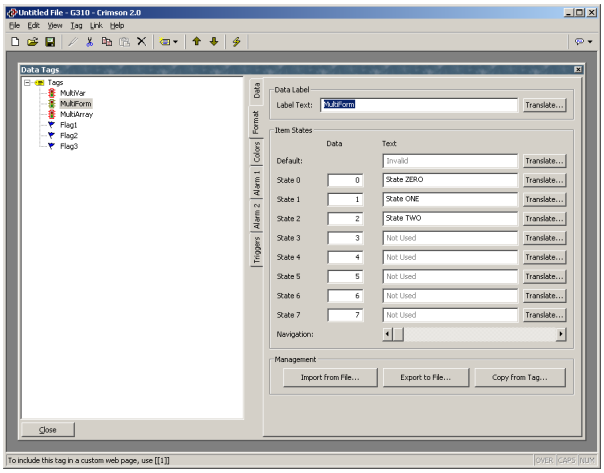
L'onglet Données d'un multi tableau contient les propriétés suivantes...



Toutes ces propriétés sont identiques à celles qui sont décrites pour les tableaux de bit.

L'ONGLET FORMAT

L'onglet Format d'une étiquette Multi contient les propriétés suivantes...



- La propriété *Texte d'étiquette* permet de spécifier le nom qui est affiché en regard de cette étiquette lors de l'ajout de l'étiquette sur une page d'affichage. Le nom diffère du nom d'étiquette car le premier peut être traduit pour les applications internationales alors que le second reste inchangé et ne s'affiche jamais à l'utilisateur du HMI virtuel.

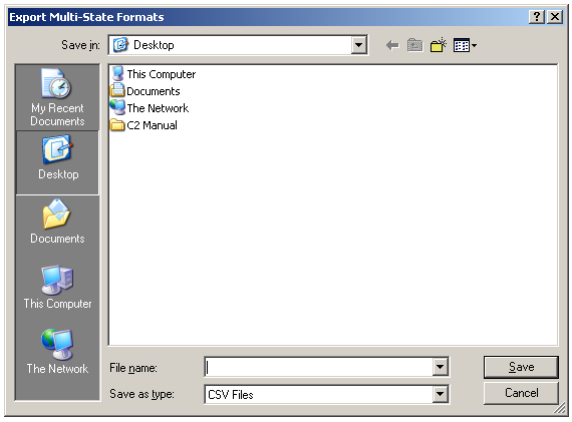
AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

- La propriété *Etats de l'Objet* permet de définir jusqu'à huit valeurs qui représentent différents états d'étiquette. Chaque état dispose d'une valeur entière qui lui est associée et une chaîne de texte pour indiquer ce qui pourrait être affiché lorsque l'étiquette possède cette valeur. Au moins deux états doivent être définis, mais l'équilibre peut être laissé sur sa condition par défaut s'ils ne sont pas nécessaires.

- La propriété *Par défaut* permet de définir le texte à afficher si l'étiquette possède une valeur différente de l'une de celles qui sont répertoriées dans les états de l'objet.

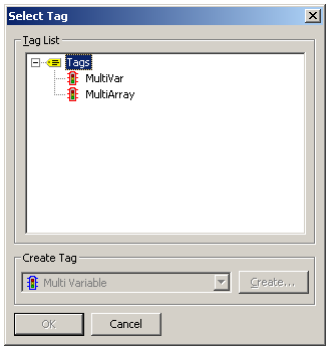
- La barre de défilement *Navigation* permet de consulter les 512 états qui peuvent être définis pour une étiquette spécifique. Si vous déplacez la barre de défilement vers la gauche ou la droite, le volet droit est mis à jour afin d'afficher les états sélectionnés.

Vous pouvez utiliser le bouton Exporter pour exporter les noms et les valeurs des états dans un fichier CSV…



Ce fichier CSV contient une ligne pour chaque état défini, indiquant le nom d'état, la valeur d'état et le texte qui lui est attribué. Si plusieurs langues sont en cours d'utilisation, une autre colonne est fournie pour chaque langue. Vous pouvez sélectionner la liste déroulante du type de fichier pour choisir un fichier au format Unicode si vous utilisez des langues qui ne peuvent pas être représentées au format ASCII standard. Vous pouvez ensuite réimporter le fichier à l'aide du bouton Importer.

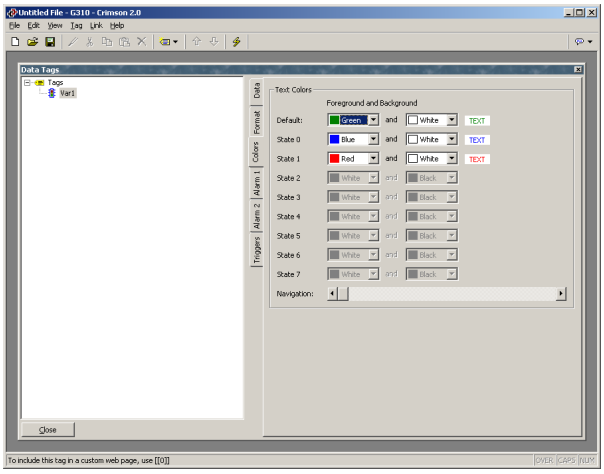
Le bouton Copier Depuis Etiquette affiche la boîte de dialogue suivante…



Vous pouvez utiliser cette boîte de dialogue pour sélectionner une autre étiquette Multi à partir de laquelle les informations de format sont copiées. Cette fonction permet de taper moins de choses si le même format est utilisé sur plusieurs étiquettes.

L'ONGLET COULEURS*

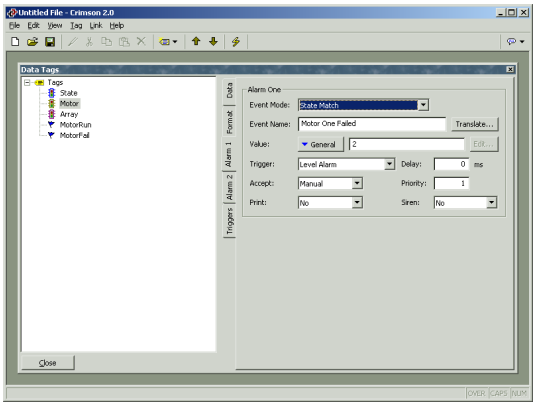
L'onglet Couleurs d'une étiquette Multi contient les propriétés suivantes...



- Les différentes paires de couleurs sont utilisées pour spécifier l'affichage de l'étiquette lorsqu'elle est dans chacun des états spécifiés sous l'onglet Format. Comme avec l'onglet Format, vous pouvez utiliser la barre de défilement *Navigation* vers le haut ou le bas dans la liste des paires de couleurs lorsque plus de huit états ont été définis.

LES ONGLETS ALARMES

Chaque onglet Alarme d'une multi variable ou d'une multi formule contient les propriétés suivantes...



- La propriété *Condition d'alarme* permet d'indiquer la logique qui est utilisée pour décider si l'alarme doit s'activer. Le tableau ci-dessous répertorie les modes disponibles.

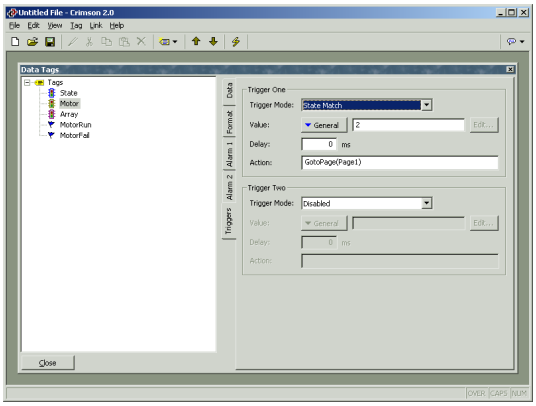
* Non présentes sur le HMI virtuel monochrome.

MODE	ACTIVATION DE L'ALARME LORSQUE...
Egal à	La valeur de l'étiquette est égale à la *valeur* de l'alarme.
Différent de	La valeur de l'étiquette n'est pas égale à la *valeur* de l'alarme.

- La propriété *Valeur* permet de définir les données de comparaison de l'alarme.
- Les propriétés restantes sont identiques à celles qui sont décrites dans l'onglet Alarmes des étiquettes de type bit.

L'ONGLET TRIGGERS

L'onglet Triggers d'une multi variable ou d'une multi formule contient les propriétés suivantes...



- La propriété *Condition d'alarme* est identique à celle de l'onglet Alarme.
- La propriété *Retard* est identique à celle de l'onglet Alarmes d'une étiquette de type bit.
- La propriété *Action* permet d'indiquer l'action qui doit être effectuée lors de l'activation du déclencheur. Reportez-vous à la section Ecriture d'actions pour obtenir une description de la syntaxe utilisée pour définir les différentes actions prises en charge par Crimson.

MODIFICATION D'ETIQUETTES DE REELS

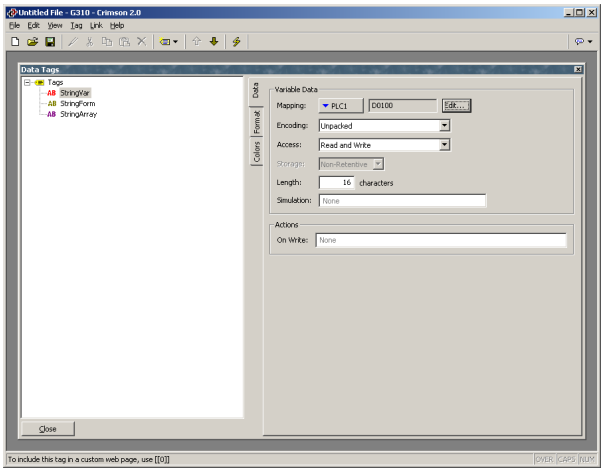
Rappelez-vous que les étiquettes de réels représentent une valeur en virgule flottante simple précision. Tous les onglets qui s'affichent pour les étiquettes de réels sont exactement identiques à ceux qui s'affichent pour les étiquettes d'entiers sauf que les données saisies pour les éléments comme les propriétés de la valeur et de l'hystérésis des alarmes et des déclencheurs peuvent contenir des décimales. Il convient donc de vous reporter aux sections sur les étiquettes d'entiers. Vous noterez que certaines sélections d'étiquettes d'entiers ne s'appliquent pas aux étiquettes de réels.

MODIFICATION D'ETIQUETTES DE CHAINES

Rappelez-vous que les étiquettes de chaînes représentent un élément de texte constitué de plusieurs caractères (ASCII). Les sections suivantes décrivent les différents onglets qui s'affichent dans la partie droite de la fenêtre Etiquettes de données lors de la modification de l'un des différents types d'étiquettes de chaînes.

L'ONGLET DONNEES (VARIABLES)

L'onglet Données d'une variable chaîne contient les propriétés suivantes...



- La propriété *Mapping* permet de spécifier si la variable doit être mappée sur un registre dans un périphérique distant ou si elle existe uniquement dans le maître (CSMSTRSX ou GT ou Data Station Plus ou G3). Si vous appuyez sur le bouton fléché pour sélectionner un nom de périphérique dans le menu, une boîte de dialogue s'affiche, permettant se sélectionner le registre de l'API.

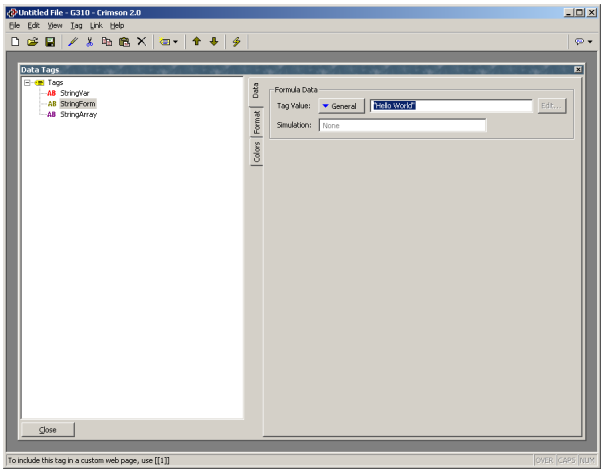
- La propriété *Encodage* permet de spécifier la façon dont le texte est compressé dans les registres mappés qui contiennent plus de 8 bits de données. Le mode Dégroupé permet d'enregistrer un caractère par registre quelle que soit la taille du registre, en laissant les bits de l'ordre supérieur vides. Le mode Groupé Bas vers Haut enregistre un caractère dans chacun des 8 bits du registre cible, en stockant le premier caractère dans les bits de l'ordre inférieur. Le mode Groupé Haut vers Bas enregistre un caractère dans chacun des 8 bits du registre cible, en stockant le premier caractère dans les bits de l'ordre supérieur.

- La propriété *Longueur* permet d'indiquer le nombre de caractères de la mémoire qui doivent être attribués pour cette chaîne. Vous devez saisir une valeur si vous avez configuré la variable pour une mémoire de rétention. Les chaînes qui sont conservées dans la RAM du maître et qui ne sont pas enregistrées en FLASH n'ont aucune limite pratique dans leur longueur.

- Les propriétés restantes sont identiques à celles qui sont décrites pour les variables bit.

L'ONGLET DONNEES (FORMULES)

L'onglet Données d'une formule chaîne contient les propriétés suivantes...

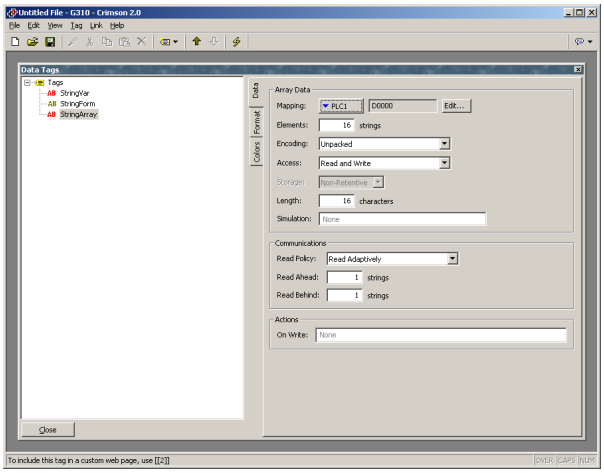


- La propriété *Valeur d'étiquette* permet de spécifier la valeur qui est représentée par cette étiquette. Elle est généralement définie sur une association de plusieurs autres étiquettes, liées les unes aux autres à l'aide d'opérateurs ou de fonctions mathématiques. Dans l'exemple ci-dessus, l'étiquette est définie sur égale à la combinaison de deux variables chaîne séparées par un espace. Pour en savoir plus que les opérateurs et les fonctions qui peuvent être réutilisées avec des chaînes, reportez-vous aux sections Ecriture d'expressions et Référence des fonctions à la fin du présent document.

- La propriété *Simulation* est identique à celle qui est décrite pour les variables bit.

L'ONGLET DONNEES (TABLEAUX)

L'onglet Données d'un tableau chaîne contient les propriétés suivantes...

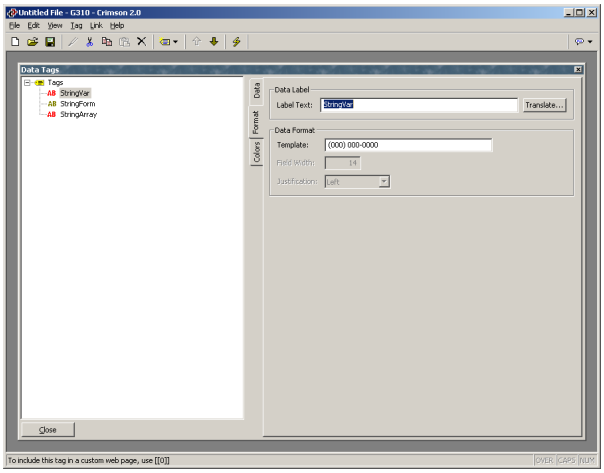


- Les propriétés *Longueur* et *Encodage* sont identiques à celles qui sont décrites pour les variables chaîne.

- Les propriétés restantes sont identiques à celles qui sont décrites pour les tableaux de bit.

L'ONGLET FORMAT

L'onglet Format d'une étiquette de chaîne contient les propriétés suivantes...



- La propriété *Texte d'étiquette* permet de spécifier le nom qui est affiché en regard de cette étiquette lors de l'ajout de l'étiquette sur une page d'affichage. Le nom diffère du nom d'étiquette car le premier peut être traduit pour les applications internationales alors que le second reste inchangé et ne s'affiche jamais à l'utilisateur du HMI virtuel.

- La propriété *Modèle* permet de fournir une « image » de la chaîne, indiquant de ce fait le type de caractères qui peut se produire dans chaque position. Si un modèle est spécifié, la saisie de données est limitée de telle façon que seul le type correct de caractères peut être sélectionné pour chaque caractère de la chaîne. Le tableau ci-dessous montre la signification des différents caractères spéciaux qui peuvent être ajoutés dans un modèle…

Caractère du modèle	Caractères autorisés				
	A-Z	a-z	0-9	Espace	Divers
A	Oui	-	-	-	-
a	Oui	Oui	-	-	-
S	Oui	-	-	Oui	-
s	Oui	Oui	-	Oui	-
N	Oui	-	Oui	-	-
n	Oui	Oui	Oui	-	-
M	Oui	-	Oui	Oui	-
m	Oui	Oui	Oui	Oui	-
0	-	-	Oui	-	-
X	Oui	Oui	Oui	Oui	Oui

AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

Les caractères supplémentaires auxquels la colonne « Divers » fait référence sont…

```
,.:;+-=!?%/$
```

Les caractères qui ne sont pas compris dans le tableau sont copiés textuellement à l'écran.

Par exemple, pour pouvoir saisir un numéro de téléphone aux Etats-Unis, utilisez le modèle…

```
(000) 000-0000
```

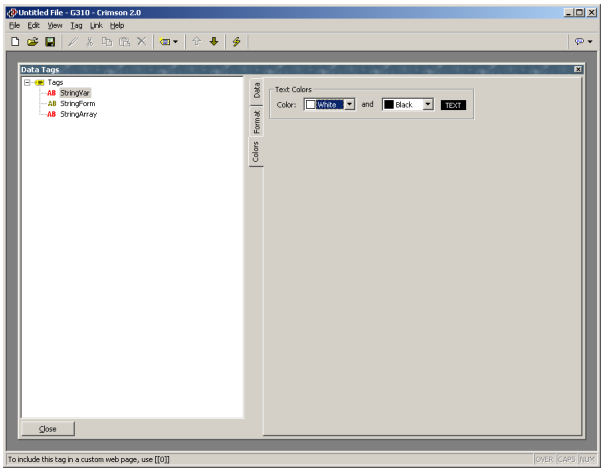
Les parenthèses, l'espace et le tiret sont tous ajoutés lorsque le champ s'affiche, mais seuls les 10 chiffres indiqués par les caractères « 0 » sont enregistrés dans la chaîne. De la même façon, si la saisie de données est activée pour un champ à l'aide de ce modèle, le curseur saute les différentes positions non numériques lorsqu'il se déplace vers la gauche ou la droite et n'autorise que la saisie des caractères numériques des positions qui peuvent être sélectionnées.

- La propriété *Longueur* est utilisée à la place du modèle pour indiquer le nombre de caractères qui doivent être réservés sur une page lors de l'affichage de cette chaîne. Si une variable chaîne est marquée Avec rétention, il est logique que cette propriété soit égale à la longueur saisie sous l'onglet *Données*, mais cela n'est pas obligatoire car vous souhaitez peut-être attribuer plus ou moins d'espace à l'écran à des fins de mise en page.

- La propriété *Justification* est utilisée lorsqu'un modèle n'est pas spécifié et elle indique le nombre de chaînes plus courtes que la propriété *Longueur* qui doivent être placées dans la mémoire attribuée à la chaîne. Elle se distingue de la propriété *Justification* du format d'affichage car elle influe sur les données qui sont vraiment enregistrées.

L'ONGLET COULEURS*

L'onglet Couleurs d'une étiquette de chaîne contient les propriétés suivantes...



* Non présentes sur le HMI virtuel monochrome.

Cet onglet permet de spécifier les couleurs par défaut qui sont utilisées pour afficher cette étiquette.

PLUS DE DEUX ALARMES

Si votre application nécessite plus de deux alarmes (ou Triggers) pour une étiquette, définissez une formule qui est égale en valeur à l'étiquette principale, puis définissez les alarmes supplémentaires sur l'alias. Ainsi, si vous disposez d'une variable appelée **`Level`** qui est mappée sur **`N7:100`** dans un API et que vous devez créer une troisième alarme pour cette étiquette, créez une variable appelée, par exemple, **`LevelAlias,`** puis définissez la propriété de sa valeur sur **`Level`**. Vous pouvez alors définir les alarmes supplémentaires sur cette étiquette d'alias.

VALIDATION D'ETIQUETTES

Lorsque vous sélectionnez l'icône Etiquettes dans le volet gauche de la fenêtre Etiquettes, vous pouvez alors accéder au bouton Valider toutes les étiquettes. Ce bouton vous permet de compiler à nouveau toutes les expressions de votre base de données en réparant toutes les références de communications cassées et en mettant à jour les compteurs de référence des étiquettes. Vous ne devez pas appuyer sur ce bouton sauf si vous avez supprimé, puis remplacé les étiquettes et que vous souhaitez réparer l'expression qui a été cassée lors de la suppression des étiquettes.

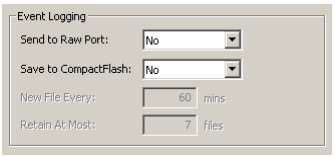
EXPORTATION DU MAPPING DES ETIQUETTES

Lorsque vous sélectionnez l'icône Etiquettes dans le volet gauche de la fenêtre Etiquettes, vous pouvez également accéder aux fonctions d'importation et d'exportation des étiquettes. Vous pouvez utiliser le bouton Exporter pour exporter les noms et mappings d'étiquettes dans un fichier CSV en vue d'une modification ultérieure dans Microsoft Excel ou tout autre outil approprié. Le bouton Importer peut alors être utilisé pour importer à nouveau le fichier en modifiant les mappings d'étiquettes conformément aux modifications apportées au fichier. Ces fonctions sont utiles lors du portage d'une application à partir d'un API vers une autre car elles permettent de modifier tous les mappings en une seule opération. Vous pouvez également utiliser la fonction d'importation pour créer des étiquettes pour qu'elles correspondent aux mappings de données qui ont été exportés dans un fichier CSV à partir d'un contrôleur modulaire Red Lion.

ENREGISTREMENT DE MESSAGES D'EVENEMENTS*

Lorsque vous sélectionnez l'icône Etiquettes dans le volet gauche, le volet droit de la fenêtre Etiquettes contient des options permettant de contrôler l'enregistrement des messages générés par les alarmes et les événements attachés à cette étiquette…

* Non présentes sur le HMI virtuel monochrome.



- La propriété *Envoyer vers Port* permet d'indiquer sur quel port de communication les alarmes doivent être imprimés. Le port en question doit être doté d'un pilote de port du type Raw Port tel que décrit dans le chapitre Utilisation du port Raw Port. Notez que vous pouvez utiliser un pilote série ou TCP/IP tel que l'application le requiert.

- La propriété *Sauvegarder sur CF* permet d'activer l'écriture d'événements dans des fichiers CSV sur la carte CompactFlash du maître. Les événements sont enregistrés à l'aide de techniques semblables à celles de l'enregistrement de données. Les propriétés *Nouveau Fichier tous les* et *Retenir au plus* contrôlent la façon dont les fichiers sont attribués. Reportez-vous au chapitre Configuration de l'enregistrement des données pour obtenir des informations sur l'écriture des données et l'attribution de noms aux fichiers.

REMARQUES POUR LES UTILISATEURS D'EDICT

Les utilisateurs du logiciel Edict-97 de Red Lion doivent noter ce qui suit…

- La fenêtre Etiquettes de données de Crimson permet d'effectuer toutes les fonctions différentes qui ont été déjà mises en œuvre à l'aide de la fenêtre Données/Etiquettes, le Scanner d'alarmes et la Table des Triggers.

- Crimson ne dispose pas du concept des constantes en tant que famille d'étiquettes distincte. Edict a utilisé des constantes lui permettant d'effectuer certaines optimisations que Crimson est dorénavant en mesure de réaliser automatiquement. Les constantes peuvent ainsi être mises en œuvre à l'aide de formules.

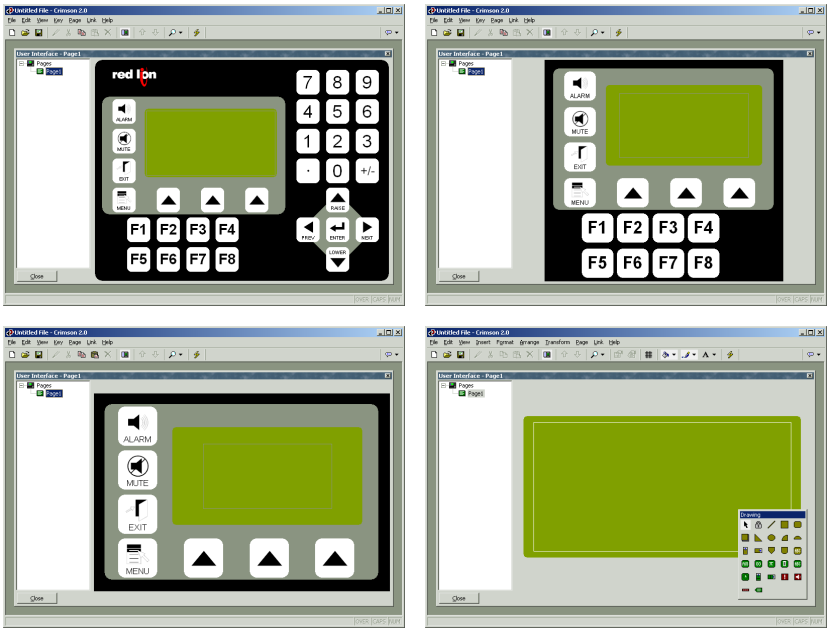
- Crimson associe les alarmes et les Triggers aux étiquettes, mais ne permet pas leur définition sur la base d'expressions arbitraires. Si une alarme ou un Trigger contrôle une expression générale, définissez une formule pour qu'elle soit égale à cette expression, puis définissez alors l'alarme et/ou le Trigger sur cette étiquette.

CONFIGURATION DE L'IHM VIRTUEL OU NON MONOCHROME

Maintenant que vous avez configuré vos options de communication et créé les étiquettes de données pour les différents éléments que vous souhaitez afficher, vous pouvez créer des pages d'affichage pour permettre à l'utilisateur de visualiser ou de modifier ces éléments de données. Vous pouvez manipuler ces pages en sélectionnant l'icône Interface utilisateur à partir de l'écran principal. Veuillez noter que ce chapitre se rapporte en particulier au HMI virtuel ou non monochrome. Si vous configurez le maître pour qu'il dispose d'un affichage couleur, reportez-vous au chapitre suivant pour obtenir des détails sur la configuration.

CONTROLE DE L'AFFICHAGE

Par défaut, la fenêtre Interface utilisateur affiche le panneau avant complet de l'HMI monochrome, notamment l'écran et toutes les touches disponibles. Si vous souhaitez qu'une plus grande partie de l'écran de votre ordinateur apparaisse sur l'écran du HMI, vous pouvez utiliser les quatre différents niveaux de zoom, comme décrit ci-dessous…



Comme vous pouvez le constater, à chaque niveau, de moins en moins de touches sont affichées et une plus grande partie de la fenêtre apparaît à l'écran. Vous pouvez contrôler le niveau du zoom à partir du menu Affichage à l'aide de l'icône de la loupe ou en appuyant sur la touche **`Alt`** simultanément aux chiffres **`1`** à **`4`**.

AUTRES OPTIONS D'AFFICHAGE

Outre le contrôle du zoom, le menu Affichage contient les options suivantes…

- La commande *Liste des pages* peut être utilisée pour afficher ou masquer le volet gauche de la fenêtre Interface utilisateur. Si la liste des pages est désactivée, davantage d'espace est disponible pour modifier l'affichage. La touche F4 bascule la liste des pages sur activée et désactivée.

- La commande *Conserver l'aspect* permet de contrôler si Crimson tente ou non de conserver les proportions de l'affichage. Si le fait de conserver l'aspect est activé, une figure qui pourrait apparaître (comme un cercle) sur le G303 s'affiche tel un cercle parfait sur votre PC. Si vous ne sélectionnez pas ce mode, Crimson peut agrandir la page d'affichage pour utiliser une plus grande partie de l'écran du PC, mais au détriment d'une certaine distorsion.

D'autres options sont disponibles lors de la modification des pages et sont décrites ci-dessous.

UTILISATION DE LA LISTE DES PAGES

Pour créer, renommer ou supprimer les pages d'affichage, cliquez sur le volet gauche de la fenêtre Interface utilisateur. Les différentes commandes du menu Page peuvent être utilisées pour apporter les modifications souhaitées. Une autre solution consiste à cliquer avec le bouton droit sur la page d'affichage requise, puis à effectuer votre choix à partir du menu.

Pour sélectionner une page, cliquez sur la page dans la liste des pages ou utilisez les flèches haut et bas de la barre d'outils. Une autre solution consiste à utiliser les associations de touches **`Alt+Gauche`** et **`Alt+Droite`** pour vous déplacer vers le haut ou le bas de la liste. Ces touches fonctionnent quel que soit le volet qui est sélectionné.

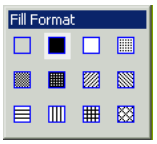
AFFICHAGE DES BOITES A OUTILS DE L'EDITEUR

Pour modifier le contenu d'une page d'affichage, sélectionnez d'abord la page de la façon décrite ci-dessus. Cliquez ensuite sur le rectangle vert qui représente l'affichage du maître. Un rectangle blanc s'affiche autour de l'écran, indiquant qu'il a été sélectionné. Plusieurs boîtes d'outils s'affichent également.

LA BOITE A OUTILS DE DESSIN

La boîte à outils de dessin permet d'ajouter différents éléments dans la page d'affichage, connus sous le nom de primitives. Les deux premières icônes contrôlent le mode d'insertion alors que le reste des icônes représente des primitives individuelles. Les primitives en jaune sont des éléments géométriques et d'animation de base alors que les primitives en vert sont des primitives riches qui utilisent la mise en forme et d'autres informations à partir d'une étiquette de données pour contrôler leur fonctionnement. Les primitives en rouge sont des éléments du système comme l'afficheur d'alarme actif. Vous pouvez également accéder à toutes les commandes de la boîte à outils via le menu Insertion.

LA BOITE A OUTILS DU FORMAT DE REMPLISSAGE



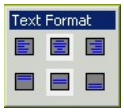
La boîte à outils du format de remplissage permet de contrôler le modèle qui est utilisé pour remplir une primitive d'affichage. Si vous sélectionnez une ou plusieurs primitives, le fait de cliquer sur un motif de remplissage modifie tous les éléments sélectionnés pour qu'ils utilisent ce modèle. Si vous ne

sélectionnez rien, le fait de cliquer sur un modèle définit le modèle par défaut des primitives qui viennent d'être créées. Vous pouvez également accéder aux différentes options via le menu Format ou l'icône du pot de peinture située dans la barre d'outils.

LA BOITE A OUTILS DU FORMAT DE LIGNE

La boîte à outils du format de ligne permet de contrôler la couleur qui est utilisée pour dessiner un contour autour de la primitive d'affichage. Si vous sélectionnez une ou plusieurs primitives, le fait de cliquer sur une couleur de ligne modifie tous les éléments sélectionnés pour qu'ils utilisent cette couleur. Si vous ne sélectionnez rien, le fait de cliquer sur une couleur définit la couleur du contour par défaut des primitives qui viennent d'être créées. Vous pouvez également accéder aux différentes options via le menu Format ou l'icône du pinceau située dans la barre d'outils.

LA BOITE A OUTILS DU FORMAT DE TEXTE



La boîte à outils du format de texte permet de contrôler l'alignement horizontal et vertical des primitives qui contiennent des éléments de texte. Si vous sélectionnez une ou plusieurs primitives, le fait de cliquer sur une icône modifie tous les éléments sélectionnés pour qu'ils utilisent la justification sélectionnée. Si vous ne sélectionnez rien, le fait de cliquer sur une icône définit le format par défaut des primitives qui viennent d'être créées. Vous pouvez également accéder aux différentes options via le menu Format.

LA BOITE A OUTILS DU PREMIER PLAN

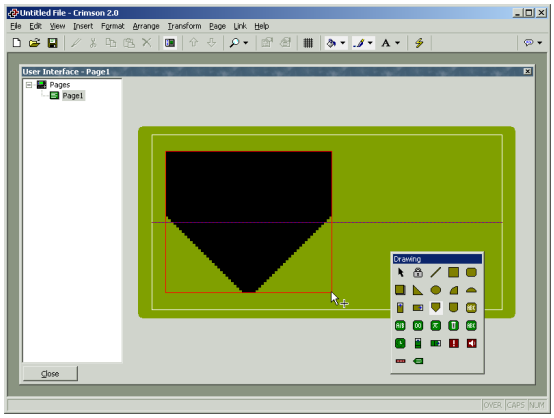
La boîte à outils de premier plan permet de contrôler la couleur du premier plan des primitives qui contiennent des éléments de texte. Si vous sélectionnez une ou plusieurs de ces primitives, le fait de cliquer sur une icône modifie tous les éléments sélectionnés pour qu'ils utilisent la couleur sélectionnée. Si vous ne sélectionnez rien, le fait de cliquer sur une icône définit la couleur par défaut des primitives qui viennent d'être créées. Vous pouvez également accéder aux différentes options via le menu Format.

LA BOITE A OUTILS DE L'ARRIERE-PLAN

La boîte à outils d'arrière-plan permet de contrôler la couleur de l'arrière-plan des primitives qui contiennent des éléments de texte. Si vous sélectionnez une ou plusieurs de ces primitives, le fait de cliquer sur une icône modifie tous les éléments sélectionnés pour qu'ils utilisent la couleur sélectionnée. Si vous ne sélectionnez rien, le fait de cliquer sur une icône définit la couleur par défaut des primitives qui viennent d'être créées. Vous pouvez également accéder aux différentes options via le menu Format.

AJOUT DE PRIMITIVES D'AFFICHAGE

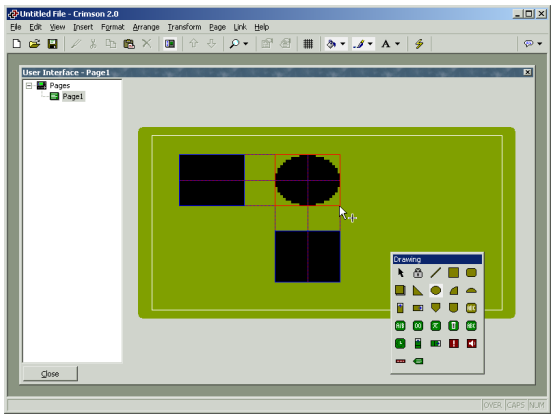
Pour ajouter une primitive d'affichage sur une page, cliquez sur l'icône requise dans la boîte à outils de dessin ou sélectionnez l'option requise à partir du menu Insertion. Le curseur de la souris est changé en une flèche avec une croix à sa base et vous pouvez alors faire glisser la position requise de la primitive dans la fenêtre d'affichage…



ALIGNEMENT INTELLIGENT

Si les fonctionnalités d'alignement intelligent du menu Affichage sont activées, Crimson vous fournit des instructions qui vous permettent d'aligner une nouvelle primitive sur des primitives existantes ou sur le centre de l'affichage. Dans l'exemple ci-dessus, la ligne horizontale en pointillés indique que le centre de la primitive du réservoir est aligné verticalement sur le centre de l'affichage. Avec un peu de pratique, cette fonctionnalité peut faciliter l'alignement des primitives lors de leur création, évitant une retouche de vos pages d'affichage afin d'aligner les différentes figures.

Dans l'exemple d'alignement intelligent ci-dessous, une ellipse qui vient d'être créée est alignée sur deux rectangles existants. Vous pouvez trouver des instructions sur les deux côtés et le centre des figures, indiquant qu'ils sont alignés. Le rectangle rouge met en valeur la primitive qui vient d'être créée alors que les rectangles bleus mettent en valeur les primitives d'où proviennent les instructions.



L'alignement intelligent est également activé lorsque les primitives sont déplacées ou redimensionnées.

OPTIONS DU CLAVIER

Lors de la création d'une primitive d'affichage, les options de clavier suivantes sont disponibles…

- Lorsque vous maintenez enfoncée la touche **Majuscule** tout en faisant glisser la primitive, la primitive est dessinée pour être centrée sur la position initiale de la souris et l'un de ses coins est défini par la position actuelle de la souris. (Si cela ne vous paraît pas logique, essayez. Cette opération est plus facile à faire qu'à expliquer !) Cette manipulation est utile pour dessiner des figures symétriques qui sont centrées sur un point initial.
- Lorsque vous maintenez enfoncée la touche **Ctrl** tout en faisant glisser la primitive, cette dernière conserve ses dimensions horizontales et verticales. Cette manipulation est utile lorsque vous souhaitez vous assurer que vous dessinez un cercle ou un carré exact à l'aide des primitives d'ellipse ou de rectangle.

Ces options sont également actives lorsque les primitives sont redimensionnées.

MODE D'INSERTION VERROUILLEE

Vous pouvez utiliser l'icône du cadenas présente dans la boîte à outils de dessin pour ajouter plusieurs primitives du même type de base sans avoir à cliquer sur l'icône de la boîte à outils pour chacun des éléments. Pour annuler le mode de verrouillage, cliquez une nouvelle fois sur l'icône du cadenas ou appuyez sur la touche **échap**. Vous pouvez effectuer la même opération à l'aide de la commande Mode Verrouiller du menu Insertion.

SELECTION DE PRIMITIVES

Pour sélectionner une primitive d'affichage, il vous suffit de déplacer le pointeur de votre souris sur la primitive en question, puis de cliquer avec le bouton gauche. Notez que lorsque le pointeur se trouve sur une primitive, un rectangle englobant bleu permet d'afficher la sélection. Lorsque vous effectuez une sélection réelle, le rectangle devient rouge et des poignées s'affichent pour vous aider à redimensionner la primitive. Si la primitive que vous souhaitez sélectionner est masquée derrière une autre primitive, appuyez sur le bouton **Alt** pour effectuer la sélection.

Pour sélectionner plusieurs primitives, faites glisser un rectangle de sélection autour des primitives que vous souhaitez sélectionner ou sélectionnez chaque primitive l'une après l'autre en maintenant enfoncée la touche **Majuscule** pour indiquer que vous souhaitez que chaque primitive soit ajoutée à la sélection. Si vous sélectionnez plusieurs primitives, le rectangle rouge entoure alors toutes les primitives et vous pouvez utiliser les poignées pour redimensionner les primitives en tant que groupe. La taille et la position relatives des primitives sont conservées tant que Crimson pourra le faire sans violer les conditions de taille minimales.

DEPLACEMENT ET REDIMENSIONNEMENT

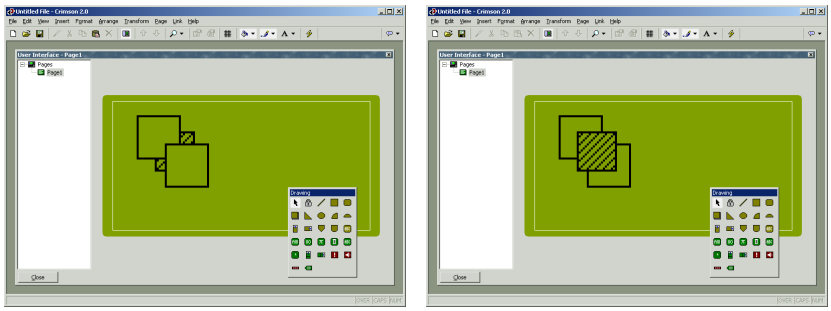
Vous pouvez déplacer les primitives en les sélectionnant, puis en les faisant glisser à la position souhaitée dans la page d'affichage. Si la fonctionnalité d'alignement intelligent est

activée, des instructions s'affichent pour vous aider à aligner les primitives sur les autres éléments de la page. Lorsque vous maintenez enfoncée la touche **Ctrl** tout en déplaçant une primitive, une copie de la primitive reste sur sa position originale, permettant donc de créer des doublons. Vous pouvez également utiliser les touches du curseur pour augmenter la sélection actuelle d'un seul pixel dans la direction requise. Lorsque vous maintenez enfoncée la touche **Ctrl** tout en augmentant la sélection, le mouvement des primitives est augmenté par un facteur de huit.

Vous pouvez redimensionner les primitives en les sélectionnant, puis en faisant glisser la poignée appropriée sur la position requise. Une fois encore, si la fonctionnalité d'alignement intelligent est activée, des instructions s'affichent pour vous aider à aligner les primitives sur les autres éléments de la page. Vous pouvez utiliser les touches **Majuscule** et **Ctrl** pour modifier le comportement du redimensionnement tel que décrit dans la section Ajout de primitives d'affichage. Notez que Crimson oblige toujours à effectuer des opérations de redimensionnement afin de garantir que les primitives restent à l'écran et que les éléments ne dépassent pas leur taille maximale autorisée ou qu'ils ne sont pas réduits au-dessous de la taille minimale appropriée à leur format.

ORGANISATION DES PRIMITIVES

Les primitives présentes sur une page d'affichage sont enregistrées dans ce qui est appelé un ordre Z. Il définit la séquence selon laquelle les primitives sont dessinées et si une primitive donnée s'affiche ou non devant ou derrière une autre primitive. Dans le premier exemple ci-dessous, le carré hachuré se trouve derrière les carrés pleins, c'est-à-dire en bas de l'ordre Z. Dans le deuxième exemple, il a été déplacé au premier plan de l'ordre et se trouve devant les autres figures.



Pour déplacer des éléments dans l'ordre Z, sélectionnez-les, puis utilisez les différentes commandes du menu Arranger. Les commandes Avancer et Reculer permettent de déplacer la sélection d'un niveau dans la direction indiquée alors que les commandes Placer au début et Placer à l'arrière permettent de la déplacer sur l'extrémité indiquée de l'ordre Z. Autre solution : si votre souris est dotée d'une roulette, vous pouvez utiliser la roulette pour déplacer la sélection. Lorsque vous faites défiler vers le haut, la sélection est déplacée vers la fin de l'ordre Z et lorsque vous faites défiler vers le bas, la sélection est déplacée vers le début.

MODIFICATION DES PRIMITIVES

Outre ce que nous venons de décrire, vous pouvez modifier les primitives de différentes façons…

- Vous pouvez utiliser les différentes commandes du Presse-papiers du menu Edition (par exemple, Couper, Copier et Coller) ou les icônes de la barre d'outils correspondantes pour dupliquer des éléments ou pour les déplacer dans une page ou entre plusieurs pages. Vous pouvez utiliser la commande Dupliquer pour effectuer une opération Copier, immédiatement suivie d'une opération Coller. Notez que lorsqu'une opération Copier est effectuée, Crimson décale l'élément qui vient d'être collé s'il superpose exactement un élément du même type.
- Vous pouvez modifier les différentes propriétés de mise en forme (par exemple, le motif de remplissage, la couleur du contour et la justification de texte…etc.) en sélectionnant une primitive, puis en cliquant sur les divers boutons dans les boîtes à outils appropriées ou en utilisant les commandes associées du menu Format. Si vous avez sélectionné plusieurs primitives, Crimson applique les modifications à toutes les primitives sélectionnées.
- Vous pouvez modifier les propriétés plus détaillées d'une primitive en double-cliquant sur la primitive ou en utilisant la commande Propriétés du menu Edition. Une boîte de dialogue s'affiche, permettant d'accéder à toutes les primitives. Les propriétés associées à chaque primitive sont décrites ci-dessous.

DESCRIPTIONS DE PRIMITIVES

Les sections ci-dessous décrivent chaque primitive qui se trouve dans la boîte à outils de dessin.

LA PRIMITIVE LIGNE

La primitive *Ligne* est une ligne dessinée entre deux points. Sa seule propriété est le style de ligne qui est utilisé. Outre les couleurs pleines qui se trouvent dans la boîte à outils de la ligne, vous pouvez également accéder à plusieurs styles en pointillés par le biais de la boîte de dialogue des propriétés.

LES PRIMITIVES GEOMETRIQUES SIMPLES

La primitive *Rectangle* est un rectangle avec un contour défini et un motif de remplissage. Vous pouvez définir le motif de remplissage sur Pas de remplissage pour dessiner uniquement le contour ou vous pouvez définir le contour sur Aucun pour dessiner une figure sans bordure.

La primitive *Rectangle rond* est semblable au rectangle, mais il possède des coins arrondis. Lorsque vous sélectionnez la primitive, une autre poignée s'affiche, permettant de modifier le rayon des coins en faisant glisser la poignée d'un côté vers l'autre.

AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

La primitive *Ombre* est semblable au rectangle, mais avec une ombre portée dans la partie inférieure droite de la figure. La primitive est souvent dessinée sans aucun motif de remplissage pour qu'elle agisse comme un cadre autour des primitives de texte.

La primitive *Coin* est un triangle à angle droit situé dans un quart de rectangle englobant. Outre les propriétés de contour et de remplissage, le coin possède une propriété qui indique quel quart il doit occuper.

La primitive *Ellipse* est une ellipse avec un contour défini et un motif de remplissage. Vous pouvez définir le motif de remplissage sur Pas de remplissage pour dessiner uniquement le contour ou vous pouvez définir le contour sur Aucun pour dessiner une figure sans bordure.

La primitive *Quart d'ellipse* est un des quarts d'une ellipse. Outre les propriétés de contour et de remplissage, le quart d'ellipse possède une propriété qui indique quel quart il doit occuper.

La primitive *Demi-ellipse* est une moitié d'une ellipse. Outre les propriétés de contour et de remplissage, la demi-ellipse possède une propriété qui indique laquelle des quatre moitiés possibles sera dessinée.

Les propriétés de ces primitives ne nécessitent aucune autre explication si ce n'est que le quart ou la moitié des primitives Coin, Quart d'ellipse ou Demi-ellipse peuvent également être modifiées via la commande qui se trouve dans le menu Transformation.

LES PRIMITIVES TYPE RESERVOIR

La primitive *Réservoir conique* est un réservoir conique avec un contour défini et un motif de remplissage. Lorsque vous sélectionnez la primitive, d'autres poignées s'affichent, permettant de modifier la forme exacte du réservoir en faisant glisser les poignées comme souhaité.

La primitive *Réservoir arrondi* est un réservoir avec un contour défini et un motif de remplissage. Lorsque vous sélectionnez la primitive, une autre poignée s'affiche, permettant de modifier la forme exacte du réservoir en faisant glisser la poignée comme requis.

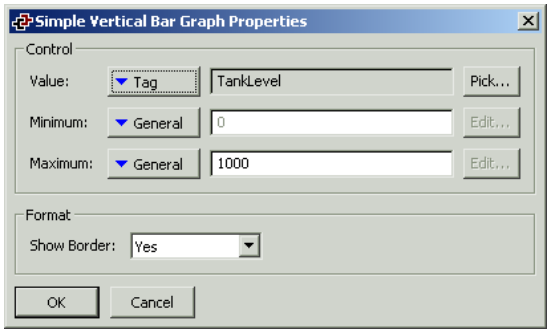
Les propriétés de ces primitives ne nécessitent aucune autre explication.

LES PRIMITIVES BARRES GRAPH SIMPLES

La primitive *Simple barre verticale* permet de dessiner un barre-graphe verticale entre les valeurs minimales et maximales spécifiées. Une autre propriété permet d'afficher ou de masquer la bordure de la primitive.

La primitive *Simple barre horizontale* permet de dessiner un barre-graphe horizontale entre les valeurs minimales et maximales spécifiées. Une autre propriété permet d'afficher ou de masquer la bordure de la primitive.

Vous pouvez accéder aux propriétés en double-cliquant sur la primitive…

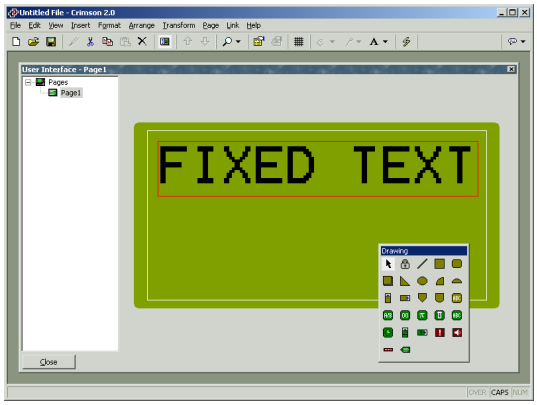


- La propriété *Valeur* permet de spécifier la valeur à afficher. Dans l'exemple donné ci-dessus, la primitive est configurée pour afficher le niveau du réservoir.
- Les propriétés *Minimum* et *Maximum* permettent de spécifier la gamme de valeurs à afficher. Dans l'exemple ci-dessus, une gamme de 0 à 1 000 est spécifiée.
- La propriété *Afficher bord* permet d'afficher ou de masquer la bordure de la primitive.

LA PRIMITIVE TEXTE FIXE

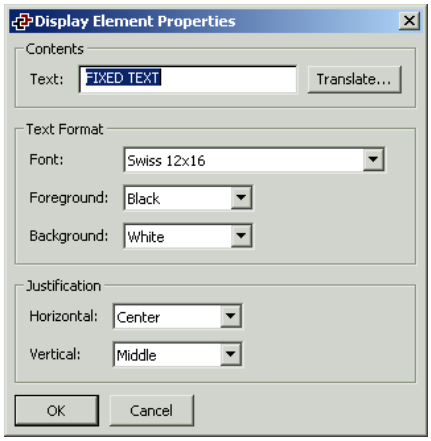
La primitive *Texte fixe* permet d'ajouter un texte fixe sur une page. Le texte est affiché dans une police, une couleur et une justification spécifiées. Vous pouvez également traduire le texte pour des applications internationales.

Lorsque le texte est créé, un curseur s'affiche, permettant de saisir du texte…



AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles / France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

Vous pouvez modifier les versions internationales du texte via la boîte de dialogue des propriétés à laquelle vous pouvez accéder en sélectionnant la primitive et en appuyant sur **`Alt+Entrée`** ou en sélectionnant la commande Propriétés à partir du menu Edition…



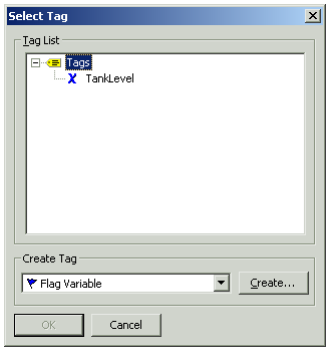
- La propriété *Texte* permet de spécifier le texte à afficher. Comme nous venons de l'expliquer, vous pouvez également modifier directement le texte sur la page d'affichage lors de la création de la primitive ou en cliquant sur une primitive existante.
- La propriété *Police* permet de spécifier la police à utiliser. Vous pouvez également modifier cette propriété à l'aide du bouton de la police situé dans la barre d'outils ou en utilisant le menu Format.
- Les propriétés *Avant-plan* et *Arrière-plan* permettent de spécifier les couleurs à utiliser pour dessiner le texte. Evidemment, si les deux paramètres ont la même couleur, le texte est illisible. Lorsque vous sélectionnez « Aucun pour » l'arrière-plan, un texte transparent est créé, permettant aux primitives sous-jacentes d'être considérées à travers les lettres.
- Les propriétés de justification *Horizontal* et *Vertical* permettent d'indiquer l'emplacement du texte dans le rectangle englobant de la primitive. Vous pouvez également modifier les propriétés par le biais de la boîte à outils associée ou du menu Format.

LA PRIMITIVE ETIQUETTE

La primitive *Etiquette* vous permet de sélectionner une étiquette (un mnémonique), puis de placer automatiquement la primitive appropriée sur l'écran. Par exemple, lorsque vous sélectionnez une étiquette d'entiers, l'insertion d'une primitive de texte d'entiers configurée de façon appropriée est effectuée.

C'est l'icône que vous utilisez le plus souvent pour ajouter des étiquettes sur une page. Elle affiche d'abord la boîte de dialogue ci-dessous pour permettre de sélectionner des étiquettes, puis elle crée l'une des cinq primitives Texte étiquette qui sont décrites dans la prochaine section. La nouvelle primitive est configurée pour afficher l'étiquette en question à l'aide de

son nom et de ses propriétés de mise en forme, tels que définis lors de la création de l'étiquette.



LES PRIMITIVES TEXTE ETIQUETTE

Les primitives Texte étiquette permettent d'afficher ou de modifier une expression dans une forme textuelle. Elles sont principalement utilisées pour afficher des étiquettes (des valeurs provenant de registres API et d'état de registres). Dans ce cas, le format par défaut provient de l'onglet Format associé à cette étiquette dans la fenêtre Etiquettes de données. Si une expression de non-étiquette est entrée (ou si vous souhaitez que la mise en forme diffère des valeurs par défaut d'une étiquette), vous pouvez remplacer les données de format tel que requis. Il existe un type de texte étiquette pour chaque famille d'étiquettes…

La primitive *Texte d'Etat* permet d'afficher un texte à l'état vrai et un texte à l'état faut.

La primitive *Texte entier* permet d'afficher la valeur d'un entier.

La primitive *Texte réel* permet d'afficher une valeur en virgule flottante.

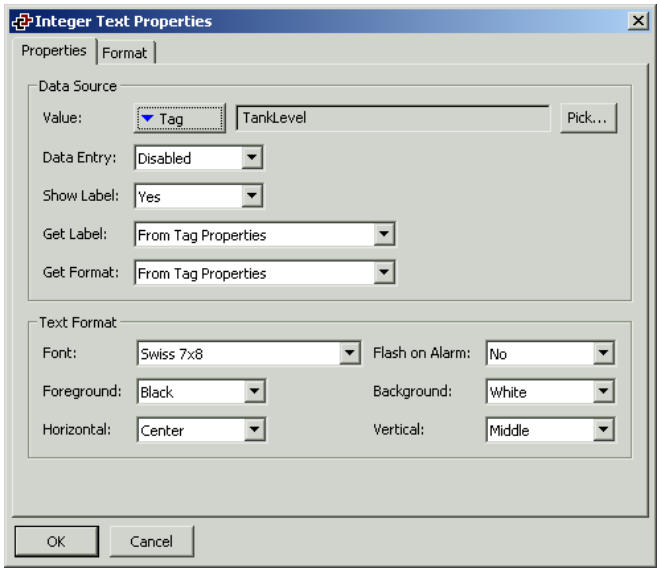
La primitive *Multi-texte* permet d'afficher une condition multi-état. Un texte en fonction d'une valeur dans un mot.

La primitive *Texte de chaîne* permet d'afficher une chaîne de caractère.

Les propriétés d'une primitive Texte étiquette sont affichées à l'aide de deux pages avec onglets.

AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

Le premier onglet est plus ou moins identique pour les cinq types de primitives…



- La propriété *Valeur* permet d'indiquer l'emplacement depuis lequel vous pouvez obtenir les données de cette primitive. Vous pouvez sélectionner une étiquette, un registre dans un périphérique de communication ou une expression qui associe plusieurs de ces éléments. Le type de données de l'élément doit être adapté à la primitive en question. Par exemple, la propriété Valeur d'une primitive Texte entier ne peut pas être définie pour une expression de chaîne.

- La propriété *Entrée de données* permet d'indiquer si vous souhaitez ou non que l'utilisateur de l'HMI puisse modifier la valeur via cette primitive. Pour que la saisie de données soit activée, sélectionner Activée. Par exemple, si une formule est saisie, la saisie de données n'est pas autorisée.

- La propriété *Afficher étiquette* permet d'indiquer si vous souhaitez ou non que la primitive comprenne un nom pour identifier les données affichées. Si cette propriété est définie sur *Oui*, le nom est justifié à gauche dans le rectangle englobant de la primitive alors que les données sont justifiées à droite. Si cette propriété est définie sur *Non*, la propriété Justification horizontale permet de rechercher les données dans le champ. Notez que cette propriété peut être modifiée via les commandes Etiquette de champ du menu Format. Lorsque aucune primitive n'est sélectionnée, vous pouvez également utiliser ces commandes pour définir la valeur par défaut des primitives qui viennent d'être créées.

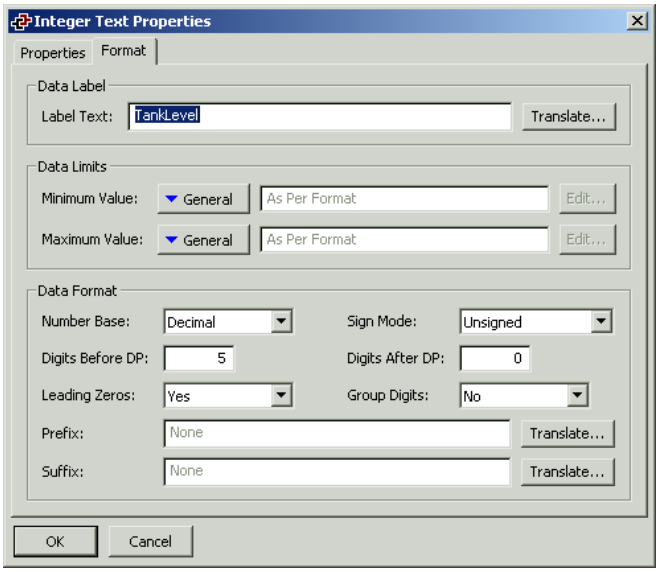
- La propriété *Texte d'étiquette* permet d'indiquer l'emplacement depuis lequel vous pouvez obtenir le texte de l'étiquette. Les options présentées dépendent de ce que vous avez saisi comme propriété Valeur. Si vous avez sélectionné une étiquette, vous pouvez utiliser le nom par défaut de l'étiquette ou entrer un nouveau nom sous l'onglet Format de la boîte de dialogue. Si vous avez sélectionné autre chose, vous ne pouvez appliquer que la seconde option.

- La propriété *Format d'étiquette* permet d'indiquer l'emplacement depuis lequel vous pouvez obtenir les informations de mise en forme de cette primitive. Les options présentées dépendent de ce que vous avez saisi comme propriété Valeur. Si vous avez sélectionné une étiquette du type de données correct, vous pouvez utiliser la mise en forme par défaut de l'étiquette ou entrer des informations modifiées sous l'onglet Format de la boîte de dialogue. Si vous avez sélectionné autre chose, vous ne pouvez appliquer que la seconde option.

- La propriété *Clignotement sur Alarme* permet d'indiquer si vous souhaitez ou non que le texte de l'écran clignote si l'étiquette entrée dans la propriété Valeur est actuellement en alarme. Cette propriété n'est pas disponible pour les primitives Texte de chaîne ou pour les primitives qui n'on pas d'étiquette définie et qui sont en accès registre API direct.

- Le reste des propriétés contrôle la police, les couleurs et la justification à utiliser lorsque vous dessinez la primitive. Ces propriétés ne nécessitent aucune explication supplémentaire.

Le deuxième onglet varie selon la primitive en question et affiche les mêmes informations que l'onglet Format du type d'étiquette associé. Différentes sections de la page sont activées en fonction des paramètres fournis pour les propriétés Texte d'étiquette et Format d'étiquette. L'exemple ci-dessous montre l'onglet Format d'une primitive Texte entier…



Comme vous pouvez le voir, les propriétés affichées sont identiques à celles qui sont affichées sous l'onglet Format d'une étiquette d'entiers. Comme nous venons de l'expliquer, les propriétés des autres types de primitives sont tout aussi identiques à celles de l'étiquette correspondante. Ainsi, vous pouvez vous reporter à la section précédente du manuel concernant les étiquettes de données afin d'obtenir plus d'informations sur chaque propriété.

MODIFICATION DE L'ETIQUETTE

Si vous souhaitez modifier les propriétés d'une primitive Texte étiquette, double-cliquez sur la primitive ou cliquez avec le bouton droit et sélectionnez la commande Propriétés dans le

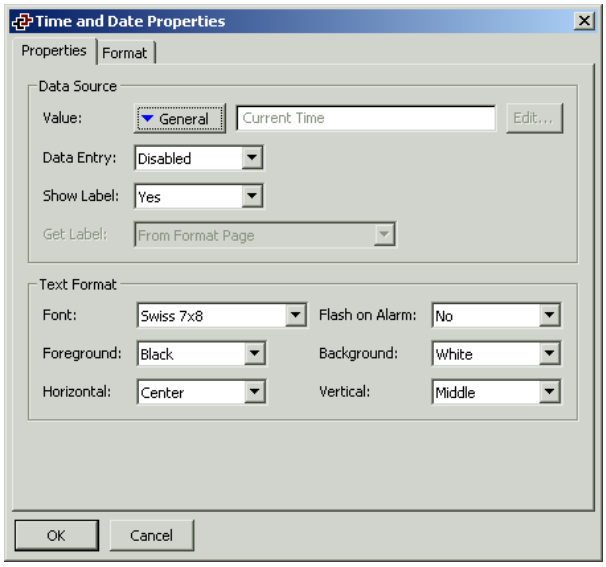
menu. Cependant, si vous souhaitez modifier les propriétés de l'étiquette qui est en cours d'utilisation pour contrôler la primitive, cliquez avec le bouton droit pour sélectionner la commande Détails d'étiquette. La boîte de dialogue affiche l'onglet Données et Format à partir de la fenêtre Etiquettes de données et vous permet de modifier les différentes propriétés. Notez qu'une modification apportée via ce mécanisme change toutes les primitives contrôlées par cette étiquette si ces primitives possèdent les propriétés Texte d'étiquette et Format d'étiquette définies sur Des propriétés de l'étiquette.

LA PRIMITIVE HEURE ET DATE

La primitive *Heure et date* permet d'afficher l'heure et la date actuelles ou le contenu d'une expression d'heure ou de date. Elle peut également être utilisée pour modifier une telle expression ou pour définir l'heure réelle du maître.

Les propriétés d'une primitive Heure et date sont affichées à l'aide de deux pages avec onglets.

Le premier onglet se présente comme suit…

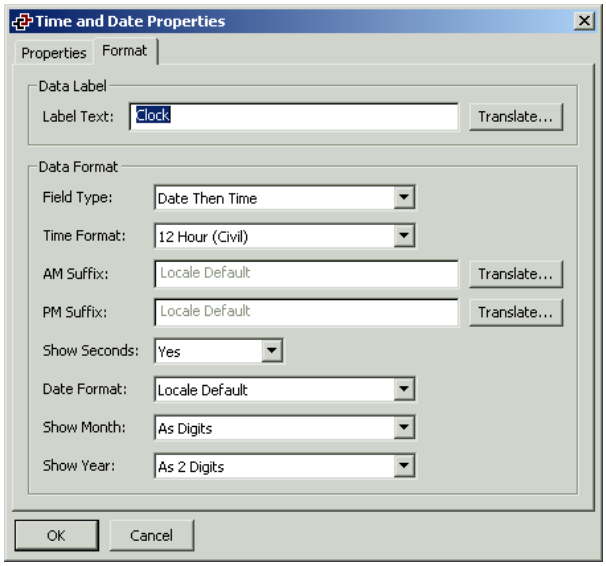


- La propriété *Valeur* permet d'indiquer la valeur du temps et la date à afficher. Si aucune valeur n'est saisie, l'heure et la date actuelles sont affichées. Si une expression est saisie, elle sert à représenter le nombre de secondes qui se sont écoulées depuis le 1er janvier 1997. Ces valeurs sont généralement obtenues à l'aide des différentes fonctions d'heure et de date décrites dans la section Référence des fonctions.

- La propriété *Entrée de données* permet d'indiquer si vous souhaitez ou non que l'utilisateur de l'HMI puisse modifier la valeur via cette primitive. Si aucune propriété de la valeur n'a été définie, cela équivaut à modifier l'heure et la date actuelles. Si une propriété de la valeur a été entrée, l'expression saisie pour la propriété de la valeur doit pouvoir être modifiée. Par exemple, si une formule est saisie, la saisie de données n'est pas autorisée.

- Le reste des propriétés est identique à celui qui est décrit pour les primitives Texte étiquette. (Même s'il peut sembler étrange d'avoir les propriétés Texte d'étiquette et Clignotement sur Alarme, souvenez-vous que la propriété de la valeur peut être une étiquette. Par conséquent, Crimson a accès au nom de l'étiquette et à l'état d'alarme de l'étiquette si vous décidez de les utiliser.)

Le second onglet se présente comme suit…



- La propriété *Texte d'étiquette* permet de définir un nom facultatif pour la primitive.
- La propriété *Type de champ* permet d'indiquer si le champ doit afficher l'heure, la date ou les deux. Dans le dernier cas, cette propriété indique également l'ordre dans lequel les deux éléments doivent être affichés.
- La propriété *Format de l'heure* permet d'indiquer si c'est le format horaire 12 heures (civil) ou 24 heures (militaire) qui doit être utilisé. Comme avec les autres propriétés, si vous laissez cette propriété définie sur Défaut local, Crimson peut choisir un format approprié en fonction de la langue sélectionnée dans Crimson.
- Les propriétés *Suffixe AM* et *Suffixe PM* sont utilisées avec le mode 12 heures pour indiquer le texte à ajouter au champ heure le matin et l'après-midi, le cas échéant. Si vous laissez la propriété non définie, Crimson utilise les paramètres par défaut.
- La propriété *Afficher secondes* permet d'indiquer si le champ heure doit comporter les secondes ou juste les heures et les minutes.
- La propriété *Format de la date* permet d'indiquer l'ordre dans lequel les différents éléments de la date (c'est-à-dire la date, le mois et l'année) doivent être affichés.
- La propriété *Afficher le mois* permet d'indiquer si le mois doit être affiché sous forme de chiffres (de 01 à 12) ou par son nom court (c'est-à-dire janv. A déc.).

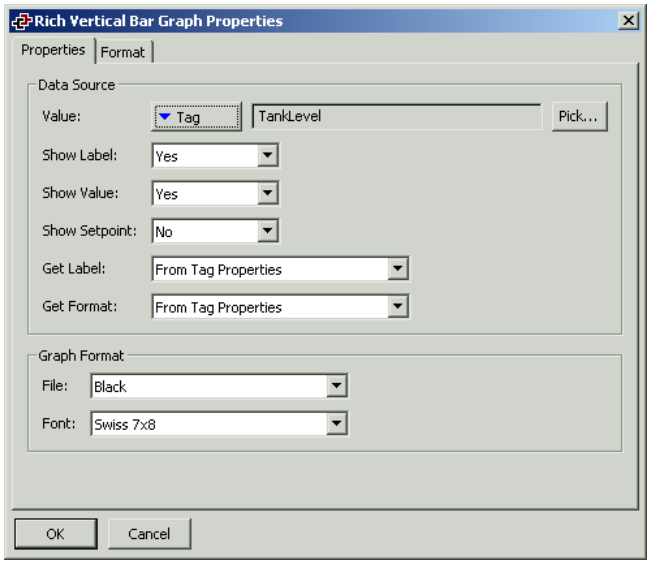
- La propriété *Afficher l'année* permet d'indiquer si le champ heure doit comprendre l'année et, le cas échéant, le nombre de chiffres qui doivent être affichés pour cet élément.

LES PRIMITIVES BARRES GRAPH RICHES

La primitive *Barre verticale riche* vous permet d'afficher un barre-graphe plus complexe qui comprend le nom de la variable, la valeur numérique de la donnée affichée et un point de consigne associé.

La primitive *Barre horizontale riche* vous permet d'afficher un barre-graphe plus complexe qui comprend le nom de la variable, la valeur numérique de la donnée affichée et un point de consigne associé.

Le fonctionnement de ces primitives riches est analogue à celui des différentes primitives Texte étiquette car elles peuvent trouver la majorité des informations de mise en forme requises à partir de l'étiquette utilisée comme leur valeur de contrôle. Tout comme avec les primitives Texte étiquette, deux pages avec onglets sont utilisées pour modifier les propriétés des primitives. Le premier de ces onglets se présente comme suit…

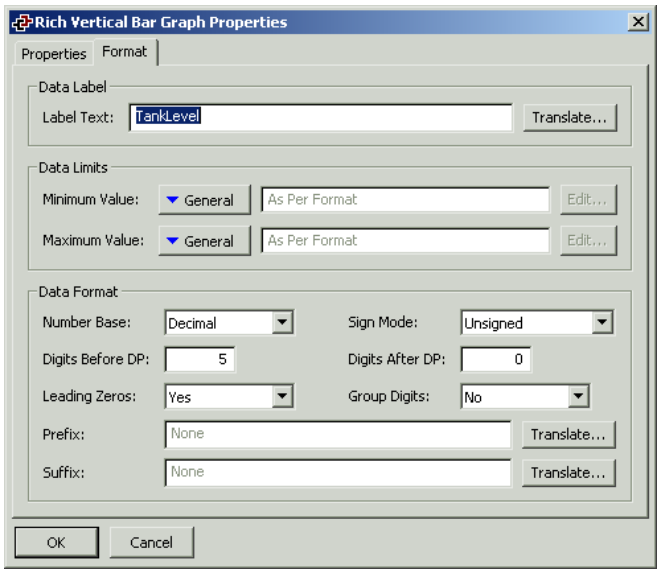


- La propriété *Valeur* permet de définir la valeur à afficher.
- La propriété *Afficher étiquette* permet d'indiquer si un nom (mnémonique) doit être ajouté ou non dans le barre-graphe. Pour les graphiques verticaux, le nom est ajouté en bas et pour les graphiques horizontaux, il est ajouté à gauche. Si vous utilisez une étiquette pour la propriété de la valeur, vous pouvez obtenir le nom à partir de l'étiquette. Sinon, il doit être entré sous l'onglet Format de la boîte de dialogue.
- La propriété *Afficher donnée* permet d'indiquer si la valeur de la donnée doit être affichée ou non dans le barre-graphe lui-même. Si vous utilisez une étiquette du type de donnée approprié pour la propriété de la valeur, vous pouvez obtenir le

format à partir de l'étiquette. Sinon, comme pour le nom, il doit être entré sous l'onglet Format.

- La propriété *Montrer Point de Consigne* permet d'afficher une barre représentant le point de consigne. Cette option est uniquement disponible si une étiquette a été saisie pour le champ de la valeur.
- Les propriétés *Texte d'étiquette* et *Format d'étiquette* sont telles que définies pour les différentes primitives Texte étiquette. Le format n'est pas requis si la propriété Afficher donnée est définie sur Non.
- La propriété *Remplissage* permet d'indiquer le modèle à utiliser pour la partie active du barre-graphe. S'il vous semble que votre barre-graphe ne fonctionne pas, assurez-vous que vous n'avez pas laissé cette propriété définie sur Aucun !
- La propriété *Police* permet d'indiquer la police à utiliser pour afficher la valeur incorporée dans le barre-graphe si cette valeur est activée via la propriété Afficher donnée.

Le second onglet contient les informations sur le nom et la mise en forme du champ...



Les propriétés affichées sont telles que celles qui sont décrites pour une étiquette d'entiers et vous devez donc vous reporter à la section précédente du manuel sur les étiquettes de données afin d'obtenir plus d'informations. Notez que l'existence de cette primitive explique la raison pour laquelle il faut saisir des valeurs minimales et maximales pour les formules car de telles étiquettes ne font jamais l'objet d'une entrée de données. Si ces limites n'ont pas été définies, comment Crimson pourrait-il savoir comment mettre le barre-graphe à l'échelle ?

AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

LES PRIMITIVES SYSTEME

La primitive *Afficheur d'alarme* permet de fournir à l'opérateur une méthode pour afficher et accepter les alarmes actives. Elle occupe toujours toute la largeur de l'écran, mais elle peut être limitée à moins que la hauteur complète, le cas échéant.

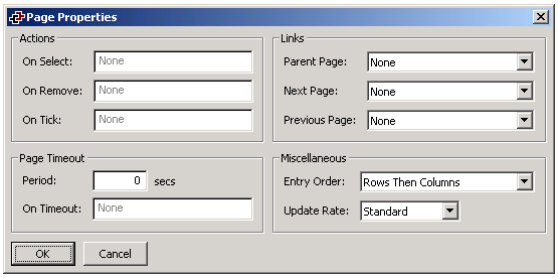
La primitive *Bandeau d'alarmes* permet de faire défiler les alarmes actives du système. Elle occupe une seule ligne et toute la largeur de l'écran. Elle ne permet pas à l'opérateur d'accepter les alarmes.

La primitive *Afficheur d'événements* permet de fournir à l'opérateur une méthode pour afficher les événements enregistrés dans le journal d'événements du système. Elle occupe toujours toute la largeur de l'écran, mais elle peut être limitée à moins que la hauteur complète, le cas échéant.

Si vous utilisez des alarmes acceptées manuellement dans votre système, vous devez fournir une page qui contient un afficheur d'alarme pour vous assurer que l'opérateur peut accepter ces alarmes. Vous voudrez peut-être ajouter le bandeau d'alarmes sur d'autres pages pour tenir l'opérateur au courant des alarmes lorsqu'il visualise d'autres pages. De la même façon, si vous utilisez des événements, vous devez fournir une page qui contient un afficheur d'événements pour permettre à l'opérateur d'afficher les événements qui se sont produits.

DEFINITION DES PROPRIETES DE LA PAGE

Chaque page dispose de plusieurs propriétés auxquelles vous pouvez accéder via le menu Page…



- Les propriétés *Sur apparition page* et *Sur changement page* permettent de définir les actions à faire lorsque la page est affichée sur l'écran ou lorsque vous changez de page. Reportez-vous aux sections Ecriture d'actions et Référence des fonctions pour obtenir une liste des actions prises en charge. Reportez-vous à la section Disponibilité des données de ce chapitre pour obtenir les détails sur un délai d'attente qui peut se produire lors de l'utilisation de ces propriétés.

- La propriété *A chaque Sec* permet de définir une action qui s'exécute chaque seconde lors de la période pendant laquelle cette page est affichée. Reportez-vous aux sections Ecriture d'actions et Référence des fonctions pour obtenir une liste des actions prises en charge. Si une absence de disponibilité des données entraîne l'impossibilité d'exécuter cette action, elle est sautée, puis retentée une seconde plus tard.

- La propriété *Au bout de* est le délai d'attente en secondes avant que l'exécution de l'action ne soit spécifiée.
- La propriété *Action sur Timeout* est l'action à effectuer lorsque le délai a expiré.
- La propriété *Page parente* permet d'indiquer la page à afficher lorsque vous appuyez sur la touche **EXIT** du HMI alors que la page est active. Vous pouvez remplacer la sélection de cette page à l'aide des techniques décrites ci-dessous.
- La propriété *Page suivante* permet d'indiquer la page à afficher lorsque vous appuyez sur la touche **NEXT** du HMI alors que cette page est active et lorsque le curseur se trouve sur la dernière entrée de données de la page Vous pouvez également remplacer cette sélection.
- La propriété *Page précédente* permet d'indiquer la page à afficher lorsque vous appuyez sur la touche **PREV** du HMI alors que cette page est active et lorsque le curseur se trouve sur la première entrée de données de la page Vous pouvez également remplacer cette sélection.

Si une seule page ne peut pas contenir tous les champs d'entrée de données, vous pouvez utiliser les propriétés Page suivante et Page précédente pour lier une série de pages afin que l'opérateur puisse modifier les champs dans la séquence. Crimson place automatiquement le curseur de façon appropriée. Ainsi, si vous appuyez sur la touche **PREC** dans le premier champ d'une page, la page précédente est activée avec le curseur sur le dernier champ de cette page.

- La propriété *Ordre saisie* permet de définir la façon dont le curseur se déplace entre les champs d'entrée de données sur le HMI. Les paramètres définissent si les champs organisés sous forme de grille sont saisis en ligne ou en colonne.
- La propriété *Rafraîchissement* permet de définir la fréquence de mise à jour des éléments à l'écran. Lorsque la fréquence des rafraîchissements augmente, les performances globales de l'HMI virtuel et G3, DSP ou Modular Controller peuvent diminuer. Vous pouvez laisser cette sélection sur son paramètre par défaut lorsque c'est possible.

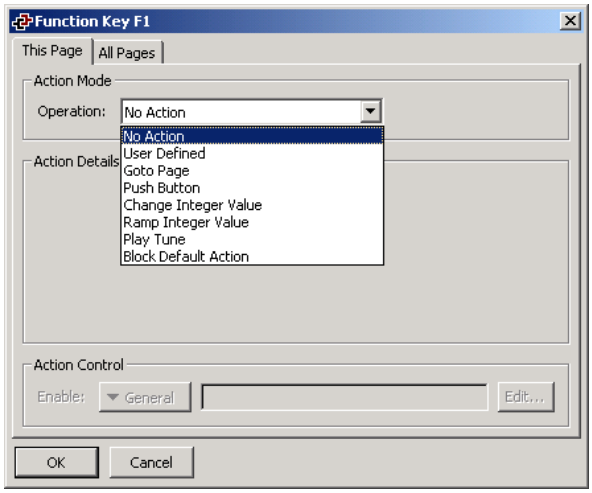
DEFINITION DES ACTIONS DU SYSTEME

Outre les différentes actions que vous pouvez définir via les propriétés de la page, Crimson vous permet de définir une action à exécuter lors du premier démarrage du système et une action à exécuter une fois par seconde, quelle que soit la page qui est affichée. Vous pouvez accéder à ces actions en sélectionnant l'icône *Pages* dans le volet gauche de la fenêtre Interface utilisateur.

DEFINITION DU COMPORTEMENT DES TOUCHES

Les sections précédentes ont fourni une description détaillée de l'utilisation de l'affichage du maître ou G3, DSP, MC pour obtenir des informations sur l'opérateur. Tout ce qui reste pour terminer la configuration de l'interface utilisateur consiste à définir l'utilisation par l'opérateur du clavier du maître ou G3, DSP, MC afin d'interagir avec le système.

Pour définir les actions que réalisent les touches, sélectionnez un niveau de zoom qui vous permet de voir les touches en question. Par exemple, si vous souhaitez configurer l'une des touches de fonction, sélectionnez le niveau de zoom un ou deux, tel que requis. Double-cliquez ensuite sur la touche pour afficher ce qui suit…



Notez que cette boîte de dialogue possède deux onglets. Le premier onglet permet de définir ce qui se produit lorsque vous appuyez sur la touche en question lorsque la page en cours est sélectionnée. Le deuxième onglet permet de définir ce qui se produit lorsque vous appuyez sur la touche quelle que soit la page sélectionnée. Le premier type d'action est appelé une action *locale* alors que le deuxième type est appelé une action *globale*. La couleur utilisée pour afficher la touche est modifiée selon les actions qui sont définies…

Si la touche est affichée en VIOLET, une action locale est définie pour cette PAGE.

Si la touche est affichée en VERT, une action GLOBALE est définie.

Si la touche est affichée en BLEU, une action globale et une action locale sont définies.

Une fois que vous avez sélectionné une action, vous pouvez cliquer avec le bouton droit sur la touche pour utiliser le menu afin de sélectionner Rendre global ou Rendre local pour modifier le type d'action. Ces options sont indisponibles si les deux types d'action ont déjà été définis.

CONDITION D'ACTIONS

Si vous souhaitez effectuer une action particulière qui dépend d'une certaine condition qui est vraie, entrez une expression pour cette condition dans le champ Conditions de l'action concernée. Cette expression peut faire directement référence à une étiquette de type bit ou utiliser tous les opérateurs de comparaison ou logiques définis dans la section Ecriture d'expressions. Si vous avez besoin d'une logique plus complexe de telle façon qu'une action est effectuée en fonction d'une prise de décision plus complexe, configurez la touche dans le

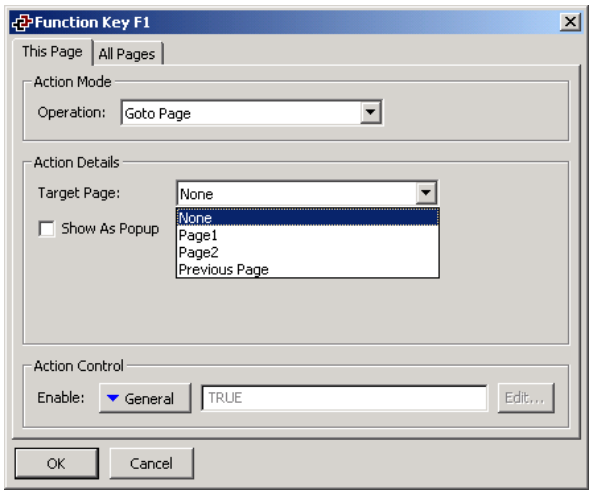
mode défini par l'utilisateur et utilisez-la pour appeler un programme qui implémente la logique requise.

DESCRIPTIONS D'ACTIONS

Les sections ci-dessous décrivent chaque type d'action disponible. Lorsque vous sélectionnez chaque type, la partie Détails des actions de la boîte de dialogue de l'action est modifiée pour afficher les options disponibles.

L'ACTION ALLER PAGE

Cette action permet de demander au G3 d'afficher une nouvelle page. Les options sont présentées ci-dessous…

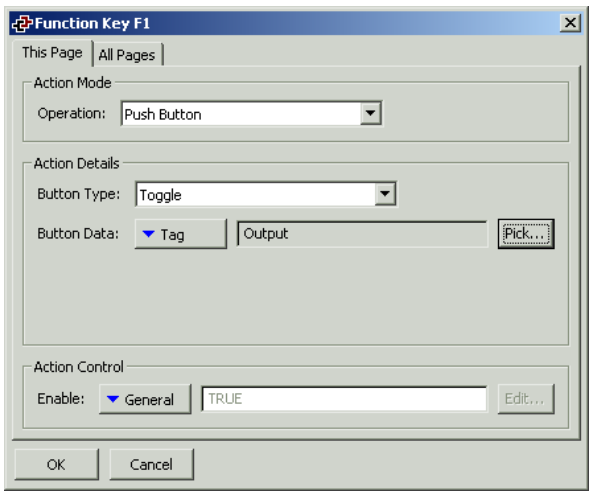


- La propriété *Page de destination* permet d'indiquer la page à afficher. Vous pouvez sélectionner une page spécifique à afficher ou la propriété Page Précédente pour revenir à ce qui a été affiché avant que la page actuelle ne soit appelée.

- La sélection *Afficher en popup* entraîne l'affichage de la page de destination sous forme de popup dans la partie supérieure de la page actuelle. Alors que ce popup est affichée, les touches supposent les définitions établies pour cette page, avec l'exception de la touche Quitter. La touche Quitter permet de supprimer le popup de l'écran.

L'ACTION BOUTON BISTABLE POUSSOIR ETC …

Cette action permet d'émuler un bouton poussoir. Les options sont présentées ci-dessous…



- La propriété *Type de bouton* permet de définir le comportement de la touche.

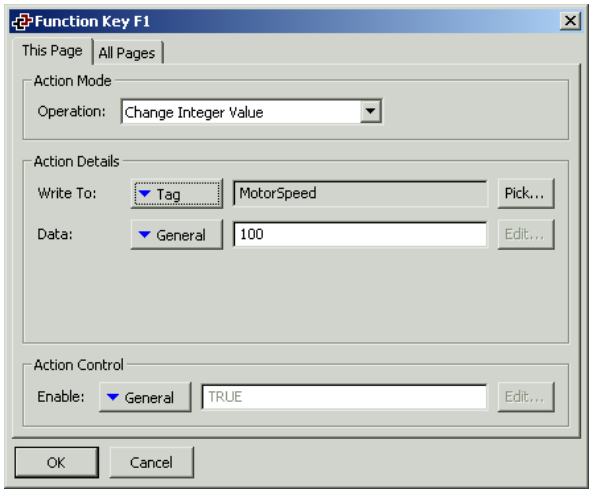
TYPE DE BOUTON	LE BOUTON…
Bistable	Modifie l'état des données en bistable ON/OFF lorsque vous appuyez sur la touche.
Momentané	Définit les données sur 1 lorsque vous appuyez sur la touche. Définit les données sur 0 lorsque vous relâchez la touche.
Actif ON	Définit les données sur 1 lorsque vous appuyez sur la touche.
Actif OFF	Définit les données sur 0 lorsque vous appuyez sur la touche.

- La propriété *Données* permet de définir les données que la touche va modifier.

Dans l'exemple ci-dessus, la touche bascule en bistable de ON à OFF et inversement la valeur de l'étiquette **`Output`**.

L'ACTION MODIFIER VALEUR D'UN ENTIER

Cette action permet d'écrire la valeur d'un entier dans un élément de données. Les options sont présentées ci-dessous…

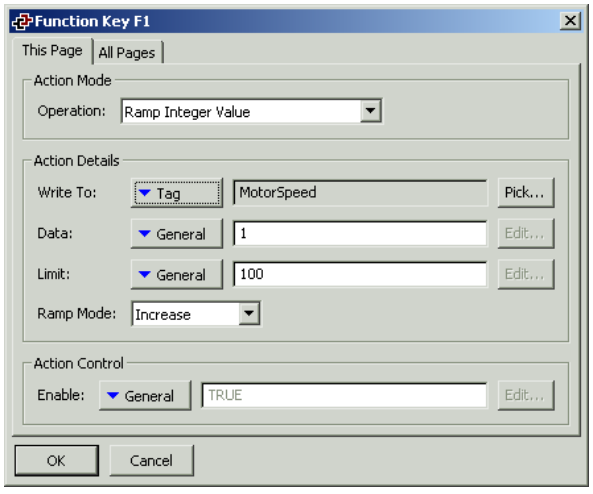


- La propriété *Ecrire dans* permet de définir l'élément de données (l'étiquette ou le registre API) à modifier.
- La propriété *Données* permet de définir la valeur à écrire dans l'étiquette.

Dans l'exemple ci-dessus, la touche écrit 100 dans l'étiquette **`MotorSpeed`**.

L'ACTION MODIFICATION VALEUR ENTIER SUIVANT RAMPE

Cette action permet d'augmenter ou de diminuer un élément de données (l'étiquette ou le registre API). Les options sont présentées ci-dessous…



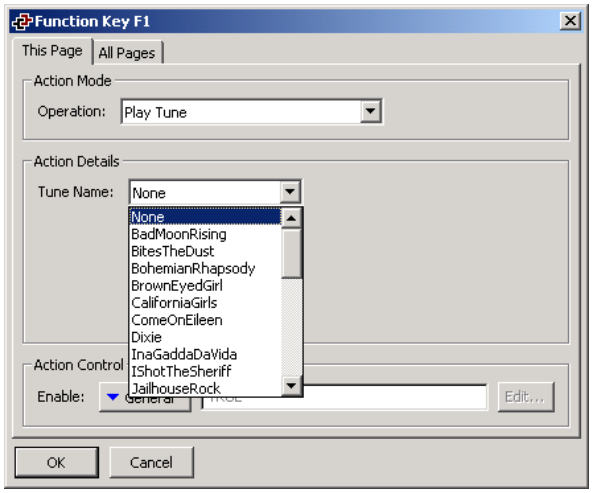
- La propriété *Ecrire dans* permet de définir l'élément de données (l'étiquette ou le registre API) à modifier.
- La propriété *Données* permet de définir le pas pour l'incrément ou le décrément de la valeur de l'étiquette.

- La propriété *Limite* permet de définir la valeur des données minimale ou maximale de l'étiquette.
- La propriété *Type de rampe* permet de définir si l'élément (l'étiquette) doit être augmenté ou réduit.

Dans l'exemple ci-dessus, le fait de maintenir la touche enfoncée permettra d'augmenter l'étiquette **`MotorSpeed`** de 1 jusqu'à ce qu'elle atteigne 100 et ce suivant une rampe logarithmique.

L'ACTION JOUER SONNERIE

Cette action joue une sonnerie (un air de musique) que vous avez sélectionnée à l'aide du beeper interne du maître.

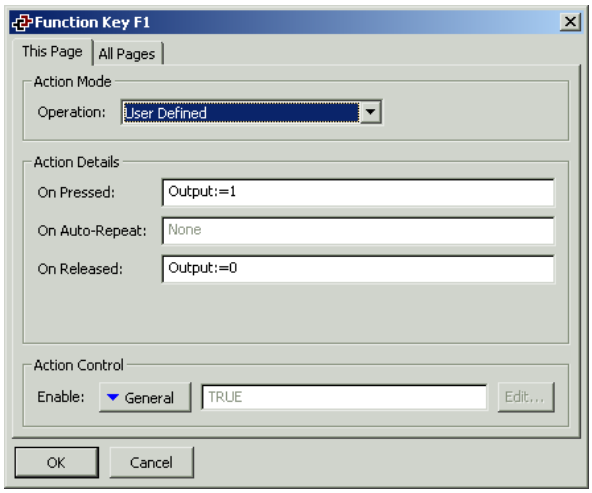


- La propriété *Nom de sonnerie* sélectionne la sonnerie à jouer.

Vous pouvez jouer des sonneries personnalisées à l'aide de la fonction **`PlayRTTTL()`**.

L'ACTION DEFINIE PAR L'UTILISATEUR

Cette action permet de faire tout le reste ! Les options sont présentées ci-dessous…



- La propriété *Sur Appui* permet de définir la ou les actions à effectuer lorsque vous appuyez sur la touche. Cette action peut appeler toutes les fonctions à partir de la section « Référence des fonctions » ou les « opérateurs » de modification des données décrits dans la section Ecriture d'actions ou elle peut encore exécuter un programme.

- La propriété *Sur Répétition Automatique* permet de définir la ou les actions à effectuer lorsque vous appuyez sur la touche et que vous la maintenez enfoncée. L'action se produit sur la pression initiale et sur les répétitions automatiques ultérieures. Par conséquent, il est inutile de définir cette propriété et Sur Appui en même temps. Cette action peut appeler toutes les fonctions à partir de la section « Référence des fonctions » ou les « opérateurs » de modification des données décrits dans la section Ecriture d'actions ou elle peut encore exécuter un programme.

- La propriété *Au Relâchement* permet de définir la ou les actions à effectuer lorsque vous relâchez sur la touche. Cette action peut appeler toutes les fonctions à partir de la section « Référence des fonctions » ou les « opérateurs » de la modification des données décrits dans la section Ecriture d'actions ou elle peut encore exécuter un programme.

Dans l'exemple ci-dessus, une action définie par l'utilisateur permet d'implémenter un bouton poussoir momentané.

BLOQUER ACTION PAR DEFAUT

Cette action ne fait rien à proprement parler, mais vous pouvez l'utiliser comme espace réservé pour empêcher d'autres traitements. Supposez par exemple que vous avez configuré la touche **`F1`** pour qu'elle effectue une action globale, mais vous souhaitez empêcher que cette action soit appelée sur une page spécifique. En configurant la touche **`F1`** sur cette page comme Bloquer action par défaut, l'action globale ne se produit pas.

MODIFICATION DE LA LANGUE

Pour configurer une touche pour qu'elle modifie la langue affichée par l' HMI virtuel, sélectionnez le mode Définie par l'utilisateur et entrez **`SetLanguage(n)`** comme propriété Sur Appui (où **`n`** est un chiffre entre 1 et 8) en fonction de la langue à afficher. La page d'affichage est recréée dans la langue sélectionnée et n'importe quel texte pour lequel des traductions ont été entrées (notamment les informations de texte fixe, de nom d'étiquette et de mise en forme d'étiquette) est ajusté comme requis par la langue sélectionnée. Les pages qui sont affichées sont également présentées dans la langue sélectionnée.

RUBRIQUES AVANCEES

Les sections suivantes concernent les questions plus avancées concernant les actions du clavier.

TRAITEMENT DES ACTIONS

Lorsque vous appuyez ou relâchez une touche, Crimson subit une séquence définie lors de la décision de l'action à prendre avec l'événement. Si une action est effectuée à n'importe quelle étape, la séquence est interrompue et les autres étapes ne peuvent pas traiter la touche.

La séquence se présente comme suit…

1. Si vous sélectionnez une primitive d'affichage pour l'interaction d'utilisateur, elle peut alors traiter la touche. Les champs d'entrée de données actifs utilisent les touches Raise, Lower, Exit et Enter, plus les autres touches appropriées à l'opération en cours d'exécution. Par exemple, les champs d'entrée des entiers utilisent également les touches numériques.

2. Si vous sélectionnez une primitive d'affichage pour l'interaction d'utilisateur et que vous appuyez sur les touches NEXT ou PREV, Crimson tente de rechercher la primitive d'affichage suivante ou précédente qui souhaite aussi l'interaction d'utilisateur. Si un tel champ existe, la touche est utilisée et cette primitive est activée.

3. Si vous définissez une action locale, l'action est effectuée et la touche utilisée.

4. Si vous définissez une action globale, l'action est effectuée et la touche utilisée.

5. Si vous n'utilisez pas la touche, les actions par défaut sont mises en œuvre…

EVENEMENT	ACTION
Touche **NEXT** appuyée	Affiche la page suivante de la page si aucune n'est définie.
Touche **PREV** appuyée	Affiche la page précédente de la page si aucune n'est définie.
Touche **EXIT** appuyée	Affiche la page parente de la page si aucune n'est définie.
Touche **MENU** appuyée	Affiche la première page dans la liste des pages.
Touche **MUTE** appuyée	Met le beeper de l'alarme interne en sourdine.

Comme nous venons de le décrire ci-dessus, la configuration d'une touche pour toute action globale ou locale (même une action qui n'effectue rien comme Bloquer action par défaut) empêche l'opération de cette séquence. Par conséquent, il est évident que cette action est utile même si elle ne semble avoir aucune utilité à première vue !

DISPONIBILITE DES DONNEES

L'infrastructure de communication de Crimson effectue une lecture des seuls éléments de données qui sont nécessaires et inclus dans la page en cours. C'est-à-dire que lorsque vous sélectionnez une page pour la première fois, certains éléments de données peuvent ne pas être disponibles. Ce n'est pas un problème pour une primitive d'affichage car dans ce cadre elle affiche uniquement un "état indéfini" (généralement un nombre de tirets) jusqu'à ce sur les données (les valeurs provenant des étiquette) soient disponibles. En revanche, les choses peuvent s'avérer plus complexes pour les actions.

Par exemple, supposez qu'une action locale augmente la vitesse d'un moteur de 50 tours/minute. Si la vitesse du moteur n'est pas enregistrée dans la page affichée précédemment, lorsque la page est affichée pour la première fois, Crimson ne connaît pas la vitesse actuelle et n'est donc pas en mesure d'écrire la nouvelle valeur. Pour la gérer, l'opérateur tente d'effectuer une action pour laquelle les données requises ne sont disponibles, le module maître MC, G3 ou DSP affiche « NOT READY » jusqu'à ce que la touche soit relâchée. L'opérateur doit alors attendre quelques minutes, puis retenter l'opération. En pratique, les mises à jour des communications se produisent normalement assez rapidement de sorte que même l'opérateur le plus doué aura du mal à obtenir que ce message d'erreur s'affiche. Mais comme il peut s'afficher, il est utile de l'expliquer.

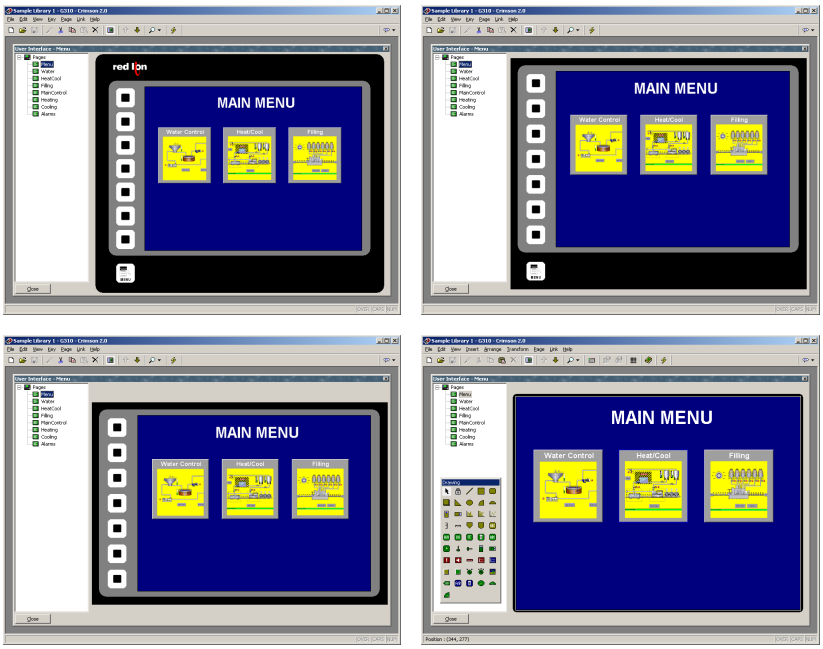
Un autre problème légèrement plus complexe se produit si l'action définie par une propriété Sur apparition page d'une page ne peut pas être poursuivie car elle trouve également que les données requises ne sont pas disponibles. à ce niveau, Crimson attend trente secondes avant que les données n'arrivent. Si elles n'arrivent pas, l'action n'est pas effectuée et un message « TIMEOUT » s'affiche pour l'opérateur. Ce mécanisme d'attente est requis pour éviter des problèmes si un lien de communication est coupé.

CONFIGURATION D'UN IHM VIRTUEL COULEUR

Maintenant que vous avez configuré vos options de communication et créé les étiquettes de données pour les différents éléments que vous souhaitez afficher, vous pouvez créer des pages d'affichage pour permettre à l'utilisateur de visualiser ou de modifier ces éléments de données. Vous pouvez manipuler ces pages en sélectionnant l'icône Interface utilisateur à partir de l'écran principal. Veuillez noter que ce chapitre se rapporte en particulier aux maîtres MC, G3 ou DSP configurés avec un HMI virtuel couleur. Si vous configurez le maître MC, G3 ou DSP pour qu'il dispose d'un HMI virtuel monochrome, reportez-vous au chapitre précédent pour obtenir des détails sur la configuration.

CONTROLE DE L'AFFICHAGE

Par défaut, la fenêtre Interface utilisateur tente d'afficher le panneau avant complet de l'IHM virtuel MC, G3 ou DSP, notamment l'écran et toutes les touches disponibles. Dans de nombreux cas, elle n'attribue pas assez d'espace écran pour modifier l'affichage. Par conséquent, vous utiliserez peut-être l'un des autres niveaux de zoom décrits ci-dessous…



Comme vous pouvez le constater, à chaque niveau, de moins en moins de touches sont affichées et une plus grande partie de la fenêtre apparaît à l'écran. Vous pouvez contrôler le niveau du zoom à partir du menu Affichage à l'aide de l'icône de la loupe ou en appuyant sur la touche **`Alt`** simultanément aux chiffres **`1`** à **`4`**.

AUTRES OPTIONS D'AFFICHAGE

Outre le contrôle du zoom, le menu Affichage contient les options suivantes…

- La commande *Liste des pages* peut être utilisée pour afficher ou masquer le volet gauche de la fenêtre Interface utilisateur. Si la liste des pages est désactivée,

davantage d'espace est disponible pour modifier l'affichage. La touche **F4** bascule la liste des pages sur activée et désactivée.

- La commande *Conserver l'aspect* permet de contrôler si Crimson tente ou non de conserver l'aspect de l'affichage. Si le fait de conserver l'aspect est activé, une figure qui pourrait apparaître (comme un cercle) sur le G303 s'affiche tel un cercle parfait sur votre PC. Si vous ne sélectionnez pas ce mode, Crimson peut agrandir la page d'affichage pour utiliser une plus grande partie de l'écran du PC, mais au détriment d'une certaine distorsion.

D'autres options sont disponibles lors de la modification des pages et sont décrites ci-dessous.

UTILISATION DE LA LISTE DES PAGES

Pour créer, renommer ou supprimer les pages d'affichage, cliquez sur le volet gauche de la fenêtre Interface utilisateur. Les différentes commandes du menu Page peuvent être utilisées pour apporter les modifications souhaitées. Une autre solution consiste à cliquer avec le bouton droit sur la page d'affichage requise, puis à effectuer votre choix à partir du menu.

Pour sélectionner une page, cliquez sur la page dans la liste des pages ou utilisez les flèches haut et bas de la barre d'outils. Une autre solution consiste à utiliser les associations de touches **Alt+Gauche** et **Alt+Droite** pour vous déplacer vers le haut ou le bas de la liste. Ces touches fonctionnent quel que soit le volet qui est sélectionné.

GRILLE

La commande Afficher grille du menu Affichage permet d'afficher ou de masquer une grille à huit pixels qui s'avère utile pour aligner des objets. Chacune des huit colonnes de la grille est affichée dans une couleur plus vive, comme chacune des six lignes. Vous pouvez configurer différentes opérations de dessin pour « attraper » les points de la grille si la grille est affichée ou non. Vous pouvez contrôler de façon individuelle les trois actions distinctes de création, déplacement et classement d'objets selon leur dimension ou utiliser les commandes Attraper pour Tout ou Attraper pour Aucun pour contrôler les trois actions en une seule fois.

LA BOITE A OUTILS DE DESSIN

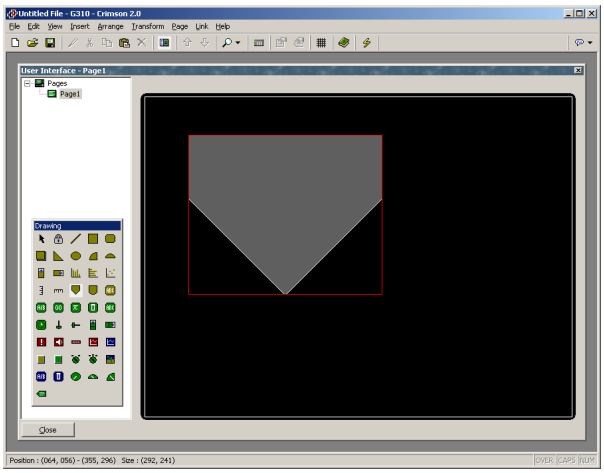
Pour modifier le contenu d'une page d'affichage, sélectionnez d'abord la page de la façon décrite ci-dessus. Cliquez ensuite sur le rectangle qui représente l'affichage du maître MC, G3 ou DSP. Un rectangle blanc s'affiche autour de l'écran, indiquant qu'il a été sélectionné. Plusieurs boîtes d'outils de dessin s'affichent.

Cette boîte à outils permet d'ajouter différents éléments dans la page d'affichage, connus sous le nom de primitives. Les deux premières icônes contrôlent le mode d'insertion mode alors que le reste des icônes représente des primitives individuelles. Les primitives en jaune sont des éléments géométriques et d'animation de base alors que les primitives en vert sont des primitives riches qui utilisent la mise en forme et d'autres informations à partir d'une étiquette de données pour contrôler leur fonctionnement. Les primitives en rouge sont des éléments du système comme l'afficheur d'alarme actif. Les primitives en bleu sont généralement des versions améliorées d'autres primitives qui ont été ajoutées plus récemment au logiciel.

Vous pouvez également accéder à toutes les commandes de la boîte à outils via le menu Insertion.

AJOUT DE PRIMITIVES D'AFFICHAGE

Pour implanter une primitive d'affichage sur une page, cliquez sur l'icône requise dans la boîte à outils de dessin ou sélectionnez l'option requise à partir du menu Insertion. Le curseur de la souris est changé en une flèche avec une croix à sa base et vous pouvez alors faire glisser la position requise de la primitive dans la fenêtre d'affichage…

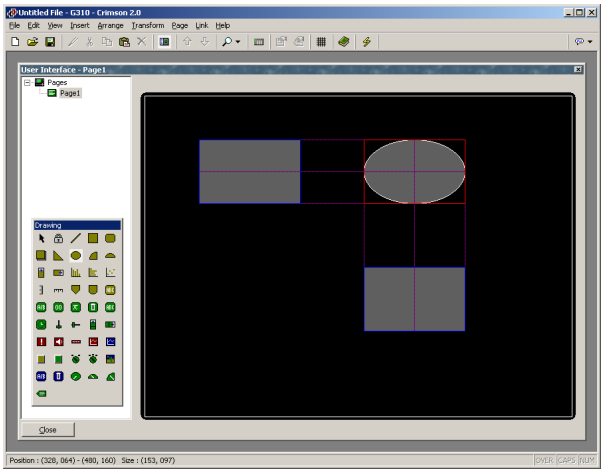


ALIGNEMENT INTELLIGENT

Si les fonctionnalités d'alignement intelligent du menu Affichage sont activées, Crimson vous fournit des instructions qui vous permettent d'aligner une nouvelle primitive sur des primitives existantes ou sur le centre de l'affichage. Dans l'exemple ci-dessus, la ligne

horizontale en pointillés indique que le centre de la primitive du réservoir est aligné verticalement sur le centre de l'affichage. Avec un peu de pratique, cette fonctionnalité peut faciliter l'alignement des primitives lors de leur création, sans avoir besoin de reprendre les pages de votre application pour « affiner » vos pages d'affichage afin d'aligner les différentes figures.

Dans l'exemple ci-dessous, une ellipse qui vient d'être créée est alignée sur deux rectangles...



Vous pouvez trouver des instructions sur les deux côtés et le centre des figures, indiquant qu'ils sont alignés. Le rectangle rouge met en valeur la primitive qui vient d'être créée alors que les rectangles bleus mettent en valeur les primitives d'où proviennent les instructions. L'alignement intelligent est également activé lorsque les primitives sont déplacées ou redimensionnées.

OPTIONS DU CLAVIER

Lors de la création d'une primitive d'affichage, les options de clavier suivantes sont disponibles…

- Lorsque vous maintenez enfoncée la touche **`Majuscule`** tout en faisant glisser la primitive, la primitive est dessinée pour être centrée sur la position initiale de la souris et l'un de ses coins est défini par la position actuelle de la souris. (Si cela ne vous paraît pas logique, essayez. Cette opération est plus facile à faire qu'à expliquer !) Cette manipulation est utile pour dessiner des figures symétriques qui sont centrées sur un point initial.

- Lorsque vous maintenez enfoncée la touche **`Ctrl`** tout en faisant glisser la primitive, cette dernière conserve ses dimensions horizontales et verticales. Cette manipulation est utile lorsque vous souhaitez vous assurer que vous dessinez un cercle ou un carré exact à l'aide des primitives d'ellipse ou de rectangle.

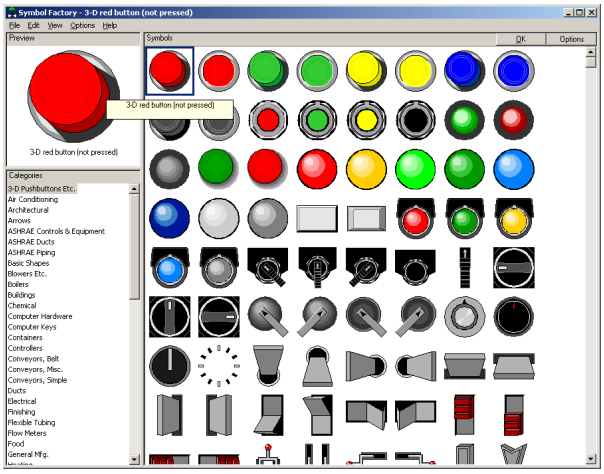
Ces options sont également actives lorsque les primitives sont redimensionnées.

MODE D'INSERTION VERROUILLEE

Vous pouvez utiliser l'icône du cadenas présente dans la boîte à outils de dessin pour ajouter plusieurs primitives du même type de base sans avoir à cliquer sur l'icône de la boîte à outils pour chacun des éléments, l'un après l'autre. Pour annuler le mode de verrouillage, cliquez une nouvelle fois sur l'icône du cadenas ou appuyez sur la touche **échap**. Vous pouvez effectuer la même opération à l'aide de la commande Mode Verrouiller du menu Insertion.

UTILISATION DE LA BIBLIOTHEQUE D'IMAGES

Pour ajouter une image à partir de la bibliothèque d'images étendue de Crimson, cliquez sur l'icône du livre dans la barre d'outils ou sélectionnez la commande Image dans le menu Insertion. La bibliothèque d'images s'ouvre sur la page à laquelle vous avez accédé en dernier lieu, ce qui permet de sélectionner une image…



Double-cliquez sur une image pour la sélectionner, puis faites glisser la taille requise de l'image comme vous le feriez lors de l'insertion de tout autre type de primitive. Le logiciel crée automatiquement une primitive *Image* contenant l'image sélectionnée. Reportez-vous aux sections ultérieures de ce chapitre pour obtenir des détails sur la façon dont cette primitive peut être gérée.

SELECTION DE PRIMITIVES

Pour sélectionner une primitive d'affichage, il vous suffit de placer le pointeur de votre souris sur la primitive en question, puis de cliquer dessus. Notez que lorsque le pointeur se trouve sur une primitive, un rectangle englobant bleu permet d'afficher la sélection. Lorsque vous effectuez une sélection réelle, le rectangle devient rouge et des poignées s'affichent pour vous aider à redimensionner la primitive. Si la primitive que vous souhaitez sélectionner est masquée derrière une autre primitive, appuyez sur le bouton **Alt** pour effectuer la sélection.

Pour sélectionner plusieurs primitives, faites glisser un rectangle de sélection autour des primitives que vous souhaitez sélectionner ou sélectionnez chaque primitive l'une après l'autre en maintenant enfoncée la touche **Majuscule** pour indiquer que vous souhaitez que chaque primitive soit ajoutée à la sélection. Si vous sélectionnez plusieurs primitives, le rectangle rouge entoure alors toutes les primitives et vous pouvez utiliser les poignées pour

redimensionner les primitives en tant que groupe. La taille et la position relatives des primitives sont conservées tant que Crimson pourra le faire sans violer les conditions de taille minimales.

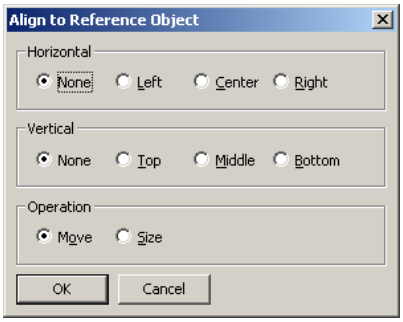
DEPLACEMENT ET REDIMENSIONNEMENT

Vous pouvez déplacer les primitives en les sélectionnant, puis en les faisant glisser sur la position requise dans la page d'affichage. Si la fonctionnalité d'alignement intelligent est activée, des instructions s'affichent pour vous aider à aligner les primitives sur les autres éléments de la page. Lorsque vous maintenez enfoncée la touche **Ctrl** tout en déplaçant une primitive, une copie de la primitive reste sur sa position originale, permettant donc de créer des doublons. Vous pouvez également utiliser les touches du curseur pour apporter à la sélection actuelle un seul pixel dans la direction requise. Lorsque vous maintenez enfoncée la touche **Ctrl** tout en augmentant la sélection, le mouvement des primitives est augmenté par un facteur de huit.

Vous pouvez redimensionner les primitives en les sélectionnant, puis en faisant glisser la poignée appropriée sur la position requise. Une fois encore, si la fonctionnalité d'alignement intelligent est activée, des instructions s'affichent pour vous aider à aligner les primitives sur les autres éléments de la page. Vous pouvez utiliser les touches **Majuscule** et **Ctrl** pour modifier le comportement du redimensionnement tel que décrit dans la section Ajout de primitives d'affichage. Notez que Crimson oblige toujours à effectuer des opérations de redimensionnement afin de garantir que les primitives restent à l'écran et que les éléments ne dépassent pas leur taille maximale autorisée ou qu'ils ne sont pas réduits au-dessous de la taille minimale appropriée à leur format.

ALIGNEMENT DES PRIMITIVES

Alors que les options d'alignement intelligent décrites ci-dessus permettent d'effectuer manuellement de nombreuses opérations d'alignement, vous voudrez peut-être parfois que le logiciel réalise l'alignement à votre place. Cette opération peut être effectuée en sélectionnant plusieurs primitives, en commençant par celle que vous souhaitez utiliser comme point de référence de l'alignement. Notez que la primitive de référence est toujours affichée avec un carré double en son centre. Une fois que vous avez effectué votre sélection, utilisez la commande Aligner du menu Arranger pour afficher la boîte de dialogue suivante…



Vous pouvez utiliser les paramètres Horizontal et Vertical pour indiquer le type d'alignement que vous allez effectuer alors que le paramètre Opération indique si les primitives doivent être redimensionnées ou déplacées pour que le résultat souhaité soit atteint.

Comme exemple, en mode Déplacer, si vous sélectionnez Gauche pour Horizontal, le côté gauche de toutes les primitives est aligné sur le côté droit de la primitive de référence. De la même façon, si vous sélectionnez Milieu pour Vertical, les primitives sont alignées de façon à ce que la ligne horizontale qui passe par le centre de chacune des primitives est alignée sur la même ligne qui passe par la primitive de référence.

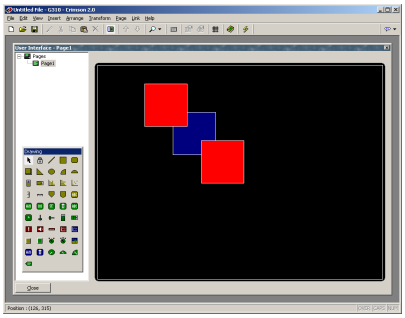
En mode Taille, les opérations d'alignement des côtés fonctionnent en augmentant les primitives de non-référence pour obtenir les résultats souhaités alors que les opérations d'alignement du centre fonctionnent en modifiant la hauteur ou la largeur des primitives afin qu'elles correspondent à la primitive de référence. Vous pouvez expérimenter le mode Taille pour avoir une meilleure idée de son fonctionnement.

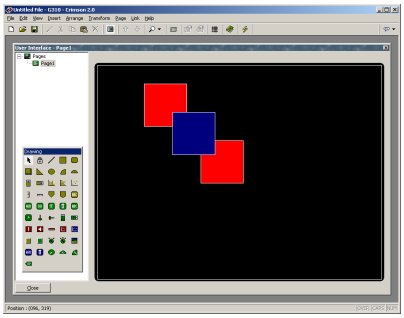
PRIMITIVES D'ESPACEMENT

Si vous souhaitez espacer de façon égale plusieurs primitives sur la page, vous pouvez utiliser les commandes Espacer Verticalement ou Espacer Horizontalement dans le menu Arranger. Ces commandes fonctionnent sur les primitives actuellement sélectionnées et tente d'attribuer à nouveau l'espace disponible entre les éléments afin d'obtenir un espacement égal. Les deux primitives extérieures sont laissées sur leur position actuelle. Notez que la commande peut échouer si un groupe de primitives inapproprié est sélectionné et elle peut ne pas atteindre un espacement parfait si l'espace disponible est trop limité.

REORGANISATION DE PRIMITIVES

Les primitives présentes sur une page d'affichage sont enregistrées dans ce qui est appelé un ordre Z. Il définit la séquence selon laquelle les primitives sont dessinées et si une primitive donnée s'affiche ou non devant ou derrière une autre primitive. Dans le premier exemple ci-dessous, le carré bleu se trouve derrière les carrés rouges, c'est-à-dire en bas de l'ordre Z. Dans le deuxième exemple, il a été déplacé au premier plan de l'ordre et se trouve devant les autres figures.





Pour déplacer des éléments dans l'ordre Z, sélectionnez-les, puis utilisez les différentes commandes du menu Arranger. Les commandes Avancer et Reculer permettent de déplacer la sélection d'un niveau dans la direction indiquée alors que les commandes Placer au début et Placer à l'arrière permettent de la déplacer sur l'extrémité indiquée de l'ordre Z. Autre solution : si votre souris est dotée d'une roulette, vous pouvez utiliser la roulette pour déplacer la sélection. Lorsque vous faites défiler vers le haut, la sélection est déplacée vers la fin de l'ordre Z et lorsque vous faites défiler vers le bas, la sélection est déplacée vers le début.

GROUPEMENT DE PRIMITIVES

Si vous souhaitez gérer plusieurs primitives comme un objet unique, vous pouvez les sélectionner tel que décrit ci-dessus, puis utiliser la commande Grouper du menu Arranger. Vous effectuerez la même opération en appuyant sur la combinaison de touches **Ctrl+G**. Une fois que vous avez créé un groupe, vous pouvez le déplacer, le classer selon sa taille et le copier comme vous le feriez avec un objet unique. Vous pouvez fragmenter un groupe en primitives de composants en le sélectionnant et en utilisant la commande Dégrouper ou la combinaison de touches **Ctrl+U**. Notez que les groupes peuvent comprendre des primitives et d'autres groupes et que ces groupes peuvent être indéfiniment imbriqués. Toutefois, vous devez en règle générale éviter les niveaux de groupement excessifs car cela peut rendre difficile la modification des primitives les plus profondément imbriquées.

MODIFICATION DE PRIMITIVES

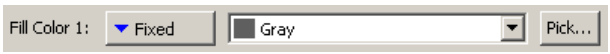
Outre ce que nous venons de décrire, vous pouvez modifier les primitives de différentes façons…

- Vous pouvez utiliser les différentes commandes du Presse-papiers du menu Edition (par exemple, Couper, Copier et Coller) ou les icônes de la barre d'outils correspondantes pour dupliquer des éléments ou pour les déplacer dans une page ou entre plusieurs pages. Vous pouvez utiliser la commande Dupliquer pour effectuer une opération Copier, immédiatement suivie d'une opération Coller. Notez que lorsqu'une opération Copier est effectuée, Crimson décale l'élément qui vient d'être collé s'il superpose exactement un élément du même type.

- Vous pouvez modifier les propriétés plus détaillées d'une primitive en double-cliquant sur la primitive ou en utilisant la commande Propriétés du menu Edition. Une boîte de dialogue s'affiche, permettant d'accéder à toutes les primitives. Les propriétés associées à chaque primitive sont décrites ci-dessous.

DEFINITION DE COULEURS

De nombreuses propriétés des primitives concernent les couleurs du dessin de la primitive. L'exemple ci-dessous montre l'une des couleurs de remplissage à partir d'une primitive Rectangle…



Notez que la propriété de la couleur est présentée par un bouton de menu déroulant, une liste déroulante et le bouton Choisir. Le menu déroulant permet de sélectionner le mode de la couleur, c'est-à-dire l'un des modes suivants…

- En mode *Fixe*, la couleur n'est pas modifiée. Vous la sélectionnez dans la liste déroulante ou en appelant la boîte de dialogue de sélection des couleurs en appuyant sur le bouton Choisir.

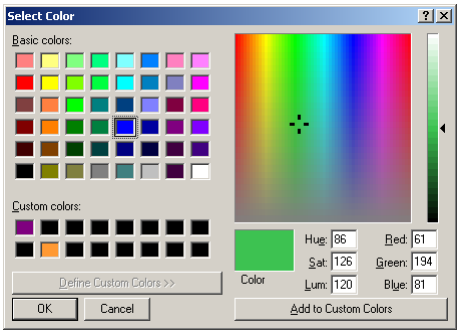
- En mode *Texte Etiquette*, la couleur est animée pour correspondre à la couleur du premier plan définie par une étiquette particulière. Vous pouvez sélectionner l'étiquette spécifique en appuyant sur le bouton Choisir.

- En mode *Ar Plan Etiquette*, la couleur est animée pour correspondre à la couleur de l'arrière-plan définie par une étiquette particulière. Vous pouvez sélectionner l'étiquette spécifique en appuyant sur le bouton Choisir.

La liste déroulante contient les couleurs fixes suivantes…

- Les seize couleurs VGA standard.
- Les seize couleurs personnalisées, définies par l'utilisateur.
- Quatorze tons de gris qui varient entre le noir et blanc.

Vous trouverez ci-dessous la boîte de dialogue de sélection des couleurs référencées ci-dessus…



Grâce à cette boîte de dialogue, vous pouvez définir une couleur de plusieurs façons. Vous pouvez les choisir dans la palette, dans la fenêtre de « l'arc en ciel » ou entrer les paramètres TSL ou RVB explicites. La boîte de dialogue vous permet également d'ajouter des couleurs personnalisées à la palette. Elles s'affichent dès que la boîte de dialogue est appelée ainsi que dans la liste déroulante décrite ci-dessus. Notez que les couleurs qui s'affichent dans « l'arc-en-ciel » ne peuvent pas toutes être rendues sur l'écran 256 couleurs du HMI. Crimson choisit la couleur la plus proche dans les capacités du périphérique.

DEFINITION DEMOTIFS DE REMPLISSAGE

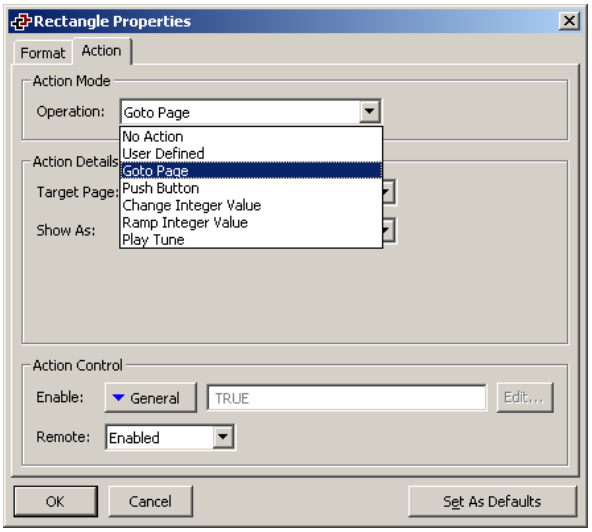
Un motif de remplissage est défini comme dans l'exemple ci-dessous…

La propriété *Type de remplissage* permet de sélectionner le motif de hachure ou de pointillé à utiliser alors que les deux propriétés de couleurs permettent de définir les couleurs qui seront utilisées pour créer le motif. Chaque couleur est définie comme décrit ci-dessus. La deuxième couleur n'est pas nécessaire lorsque vous sélectionnez un remplissage plein.

DEFINITION DES ACTIONS

Vous pouvez rendre de nombreuses primitives tactiles pour que certaines actions se produisent lorsque vous appuyez dessus, que vous les maintenez enfoncées ou que vous les

relâchez. Pour définir les actions qu'une primitive effectue, affichez les propriétés de cette primitive et sélectionnez l'onglet Action…



La liste déroulante permet de sélectionner l'un des modes d'action, pour lesquels vous trouverez les descriptions ci-dessous.

CONDITION DES ACTIONS

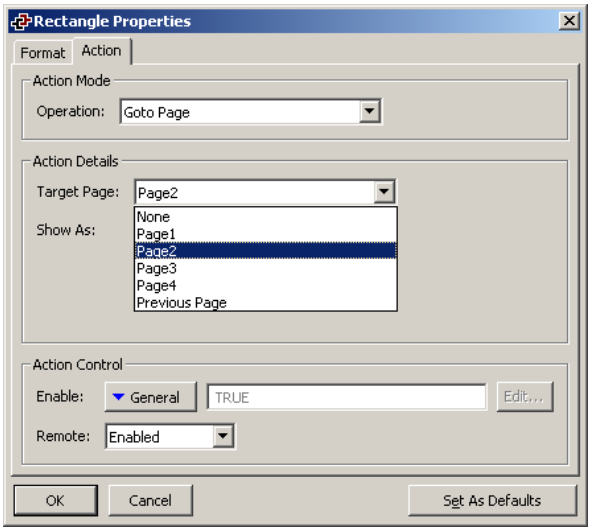
Si vous souhaitez effectuer une action particulière qui dépend d'une certaine condition qui est vraie, entrez une expression pour cette condition dans le champ Conditions de l'action concernée. Cette expression peut faire directement référence à une étiquette de type bit ou utiliser tous les opérateurs de comparaison ou logiques définis dans la section Ecriture d'expressions. Si vous avez besoin d'une logique plus complexe de telle façon qu'une action est effectuée en fonction d'une prise de décision plus complexe, configurez la primitive dans le mode défini par l'utilisateur et utilisez-la pour appeler un programme qui implémente la logique requise. De plus, vous pouvez utiliser la propriété Contrôle à Distance pour activer ou désactiver cette action via la fonction de I'IHM virtuel du serveur Web : pour autoriser l'accès à distance, l'expression Activer doit être sélectionnée si vous souhaitez juste un contrôle local sélectionnez Désactivé.

DESCRIPTIONS DES ACTIONS

Les sections ci-dessous décrivent chaque type d'action disponible. Lorsque vous les sélectionnez, la partie Détails des actions de la boîte de dialogue de l'action est modifiée pour afficher les options disponibles.

L'ACTION ALLER PAGE

Cette action permet de demander au G3 d'afficher une nouvelle page. Les options sont présentées ci-dessous…

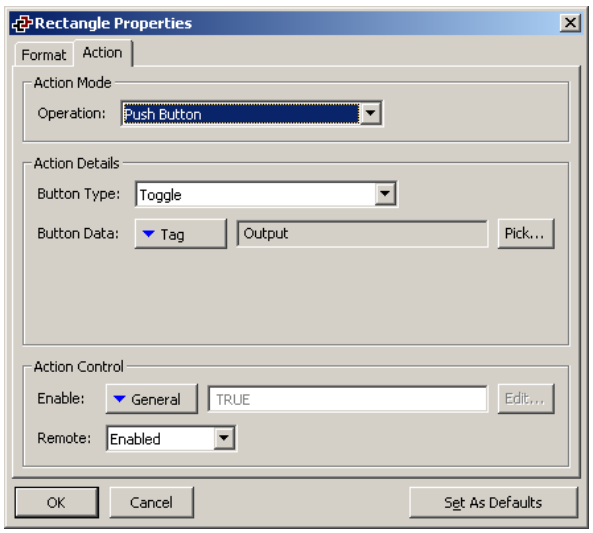


- La propriété *Page de destination* permet d'indiquer la page à afficher. Vous pouvez sélectionner une page spécifique à afficher ou la propriété Page Précédente pour revenir à ce qui a été affiché avant que la page actuelle ne soit appelée.

- La propriété *Affiché Comme* permet de définir la façon d'afficher la page. Outre l'affichage sous forme de page normale, vous pouvez l'afficher sous forme de page ou de menu contextuel(le). Les deux types de popup apparaissent dans la partie supérieure de la page existante et lorsqu'elles sont affichées, les touches et l'écran tactile de l'HMI assurent les fonctions de la nouvelle page. Les menus contextuels sont affichés avec un alignement à gauche pour qu'ils correspondent aux touches d'affichage alors que les pages contextuelles s'affichent à l'emplacement indiqué par les propriétés de la page. Notez que vous devez attribuer l'action HidePopup() à une primitive ou une touche de la nouvelle page pour supprimer le popup.

L'ACTION BOUTON BISTABLE POUSSOIR

Cette action permet d'émuler un bouton poussoir. Les options sont présentées ci-dessous…



- La propriété *Type de bouton* permet de définir le comportement de la primitive.

TYPE DE BOUTON	LE BOUTON…
Bistable	Modifie l'état des données en bistable ON/OFF lorsque vous appuyez sur la primitive.
Momentané	Ecrit les données à 1 lorsque vous appuyez sur la primitive. Ecrit les données à 0 lorsque vous relâchez sur la primitive.
Actif ON	Ecrit les données à 1 lorsque vous appuyez sur la primitive.
Actif OFF	Ecrit les données à 0 lorsque vous appuyez sur la primitive.

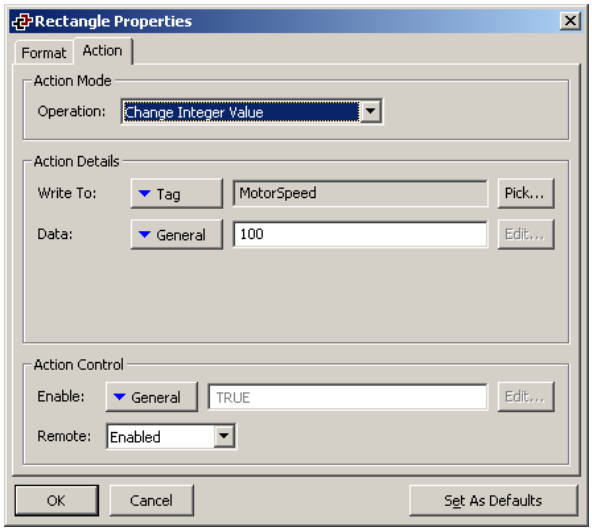
- La propriété *Données/Etiquettes* permet de définir les données à modifier.

Dans l'exemple ci-dessus, la primitive bascule de ON à OFF et inversement la valeur de l'étiquette **`Output`**.

AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

L'ACTION MODIFIER VALEUR D'UN ENTIER

Cette action permet d'écrire la valeur d'un entier dans un élément de données. Les options sont présentées ci-dessous…

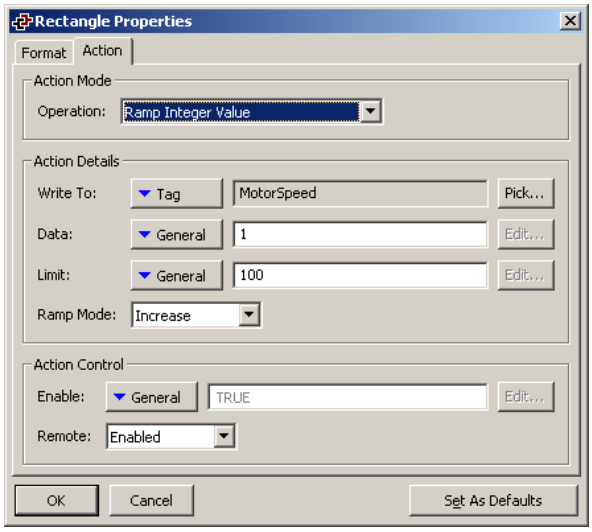


- La propriété *Ecrire dans* permet de définir l'élément de données (l'étiquette) à modifier.
- La propriété *Données* permet de spécifier la valeur à écrire dans l'étiquette.

Dans l'exemple ci-dessus, la touche écrit 100 dans l'étiquette **`MotorSpeed`**.

L'ACTION MODIFICATION VALEUR ENTIER SUIVANT RAMPE

Cette action permet d'augmenter ou de diminuer un élément de données. Les options sont présentées ci-dessous…



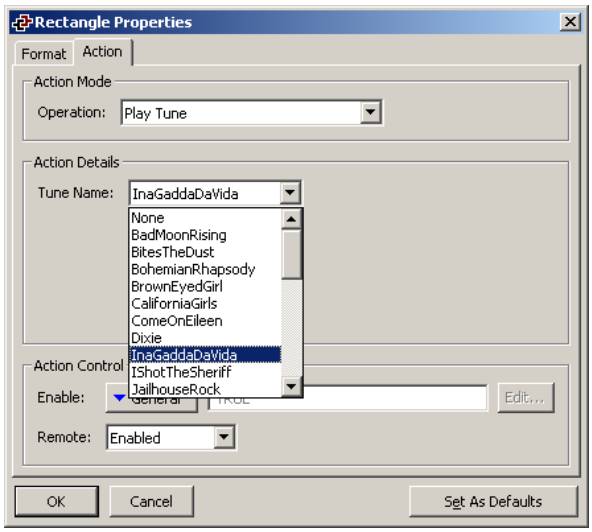
- La propriété *Ecrire dans* permet de définir l'élément de données (l'étiquette) à modifier.

- La propriété *Données* permet de définir le pas pour incrémenté ou décrémenté la valeur de l'élément.
- La propriété *Limite* permet de définir la valeur des données minimale ou maximale.
- La propriété *Type de rampe* permet de définir si l'élément doit être incrémenté ou décrémenté.

Dans l'exemple ci-dessus, le fait de maintenir la primitive enfoncée permettra d'incrémenter l'étiquette **`MotorSpeed`** de 1 jusqu'à ce qu'elle atteigne 100.

L'ACTION JOUER SONNERIE

Cette action joue une sonnerie que vous avez sélectionnée à l'aide du beeper interne du maître.



- La propriété *Nom de sonnerie* sélectionne la sonnerie à jouer.

Vous pouvez jouer des sonneries personnalisées à l'aide de la fonction **`PlayRTTTL()`**.

L'ACTION DEFINIE PAR L'UTILISATEUR

Cette action permet de faire tout le reste ! Les options sont présentées ci-dessous…

- La propriété *Sur Appui* permet de définir l'action à effectuer lorsque vous appuyez sur la primitive. Cette action peut appeler toutes les fonctions à partir de la section « Référence des fonctions » ou les « opérateurs » de la modification des données décrits dans la section Ecriture d'actions ou elle peut encore exécuter un programme.
- La propriété *Sur Répétition Automatique* permet de définir l'action à effectuer lorsque vous appuyez sur la primitive et que vous la maintenez enfoncée. L'action se produit sur la pression initiale et sur les répétitions automatiques ultérieures. Par conséquent, il est inutile de définir cette propriété et Sur Appui. Cette action peut appeler toutes les fonctions à partir de la section « Référence

des fonctions » ou les « opérateurs » de la modification des données décrits dans la section Ecriture d'actions ou elle peut encore exécuter un programme.

- La propriété *Au Relâchement* permet de définir l'action à effectuer lorsque vous relâchez sur la primitive. Cette action peut appeler toutes les fonctions à partir de la section « Référence des fonctions » ou les « opérateurs » de la modification des données décrits dans la section Ecriture d'actions ou elle peut encore exécuter un programme.

Dans l'exemple ci-dessus, une action définie par l'utilisateur permet d'implémenter un bouton poussoir momentané.

UTILISATION DE PARAMETRES PAR DEFAUT

La boîte de dialogue utilisée pour modifier les propriétés de chaque type de primitive dispose d'un bouton situé dans le coin inférieur droit et nommé Mettre Par défaut. Vous pouvez utiliser ce bouton pour enregistrer les paramètres actuels de certaines propriétés de la primitive car les paramètres par défaut sont utilisés lors de la création d'une nouvelle primitive. Vous pouvez effectuer cette même fonction en sélectionnant la commande Enregistrer par Défaut dans le menu Edition ou en appuyant sur la combinaison de touches **`Ctrl+E`**. Notez que les propriétés ne sont pas toutes comprises dans les paramètres par défaut. Seules celles qui concernent la mise en forme (par opposition aux données sous-jacentes présentées par la primitive) sont enregistrées. Vous pouvez appliquer les paramètres par défaut à la ou aux primitives actuellement sélectionnée(s) à l'aide de la commande Appliquer par Défaut du menu Edition ou en appuyant sur la combinaison de touches **`Ctrl+L`**.

DESCRIPTIONS DE PRIMITIVES

Les sections ci-dessous décrivent chaque primitive qui se trouve dans la boîte à outils de dessin.

LA PRIMITIVE LIGNE

La primitive *Ligne* est une ligne dessinée entre deux points. Ses seules propriétés sont le style de ligne à utiliser. Outre les couleurs pleines qui se trouvent dans la boîte à outils de la ligne, vous pouvez également accéder à plusieurs styles en pointillés par le biais de la boîte de dialogue des propriétés.

LES PRIMITIVES GEOMETRIQUE SIMPLES

La primitive *Rectangle* est un rectangle avec un contour défini et un motif de remplissage. Vous pouvez définir le motif de remplissage sur Pas de remplissage pour dessiner uniquement le contour ou vous pouvez définir le contour sur Aucun pour dessiner une figure sans bordure.

La primitive *Rectangle rond* est semblable au rectangle, mais il possède des coins arrondis. Lorsque vous sélectionnez la primitive, une autre poignée s'affiche, permettant de modifier le rayon des coins en faisant glisser la poignée d'un côté vers l'autre.

La primitive *Ombre* est semblable à celle du rectangle, mais avec une ombre portée ou un effet 3D ombré. La primitive est souvent dessinée de façon à ce qu'elle agisse comme un cadre autour des primitives de texte ou des autres groupes d'éléments.

La primitive *Coin* est un triangle à angle droit situé dans un quart de rectangle englobant. Outre les propriétés de contour et de remplissage, le coin possède une propriété qui indique quel quart il doit occuper.

La primitive *Ellipse* est une ellipse avec un contour défini et un motif de remplissage. Vous pouvez définir le motif de remplissage sur Pas de remplissage pour dessiner uniquement le contour ou vous pouvez définir le contour sur Aucun pour dessiner une figure sans bordure.

La primitive *Quart d'ellipse* est un des quarts d'une ellipse. Outre les propriétés de contour et de remplissage, le quart d'ellipse possède une propriété qui indique quel quart il doit occuper.

La primitive *Demi-ellipse* est une moitié d'une ellipse. Outre les propriétés de contour et de remplissage, la demi-ellipse possède une propriété qui indique laquelle des quatre moitiés possibles sera dessinée.

Les propriétés de ces primitives ne nécessitent aucune autre explication si ce n'est que le quart ou la moitié des primitives Coin, Quart d'ellipse ou Demi-ellipse peuvent également être modifiées via la commande qui se trouve dans le menu Transformation.

LES PRIMITIVES DE RESERVOIR

La primitive *Réservoir conique* est un réservoir conique avec un contour défini et un motif de remplissage. Lorsque vous sélectionnez la primitive, d'autres poignées s'affichent, permettant de modifier la forme exacte du réservoir en faisant glisser les poignées comme requis.

La primitive *Réservoir arrondi* est un réservoir avec un contour défini et un motif de remplissage. Lorsque vous sélectionnez la primitive, une autre poignée s'affiche, permettant de modifier la forme exacte du réservoir en faisant glisser la poignée comme requis.

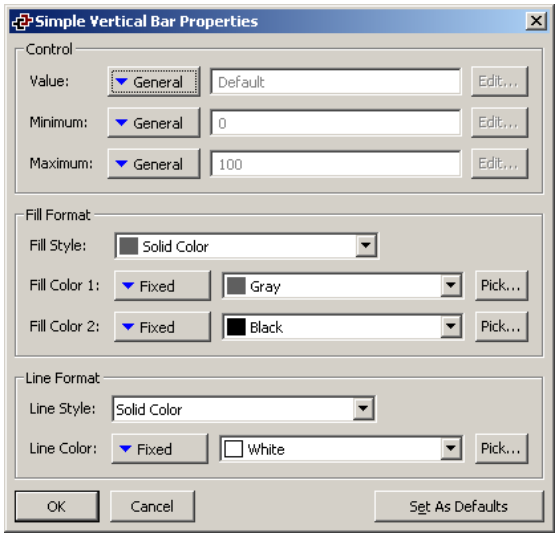
Les propriétés de ces primitives ne nécessitent aucune autre explication.

LES PRIMITIVES BARRES GRAPH SIMPLES

La primitive *Simple barre verticale* permet de dessiner un barre-graphe verticales entre des limites spécifiées. Les autres propriétés permettent de définir la couleur de remplissage et le style de bordure de la primitive.

La primitive *Simple barre horizontale* permet de dessiner un barre-graphe horizontales entre des limites spécifiées. Les autres propriétés permettent de définir la couleur de remplissage et le style de bordure de la primitive.

Vous pouvez accéder aux propriétés en double-cliquant sur la primitive…



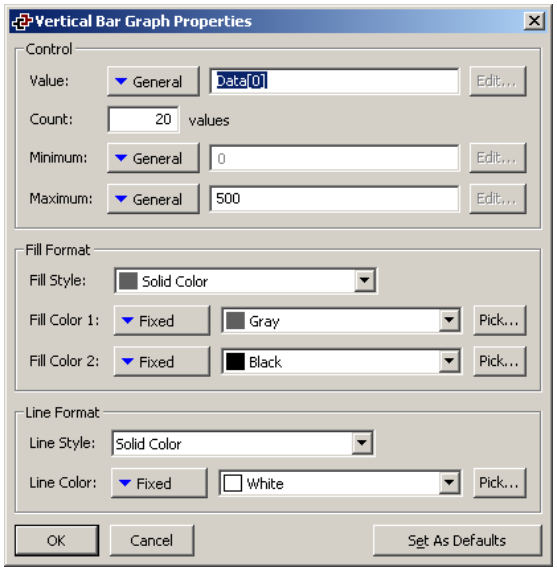
- La propriété *Valeur* permet de spécifier la valeur ou l'étiquette à afficher.
- Les propriétés *Minimum* et *Maximum* permettent de spécifier la gamme de valeurs à afficher.
- Les propriétés *Format de Remplissage* permettent de définir la couleur de remplissage de la primitive. La zone de remplissage de la barre est dessinée dans le motif et les couleurs indiqués alors que la zone non remplie est dessinée avec la *Couleur* 2 pleine.
- Les propriétés *Format Ligne* permettent de définir la bordure de la primitive.

LES PRIMITIVES GRAPHIQUE A BARRES

La primitive *Graph Barres Verticales* affiche un groupe de valeurs d'étiquettes à partir d'un tableau, sous forme de plusieurs barres verticales. Chaque valeur est mise à l'échelle selon les mêmes valeurs minimales ou maximales. De 2 à 400 valeurs (étiquette) peuvent être affichées.

La primitive *Graph Barres Horizontales* affiche un groupe de valeurs (d'étiquettes) à partir d'un tableau, sous forme de plusieurs barres horizontales. Chaque valeur est mise à l'échelle selon les mêmes valeurs minimales ou maximales. De 2 à 400 valeurs (étiquette) peuvent être affichées.

Vous pouvez accéder aux propriétés en double-cliquant sur la primitive…



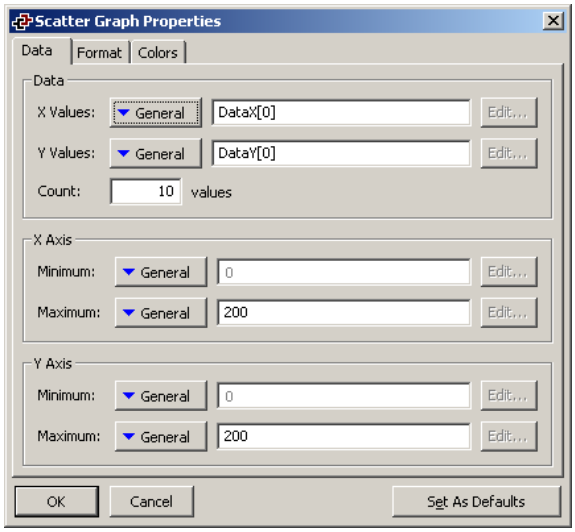
- La propriété *Valeur* permet de spécifier le premier élément (étiquette) du tableau à afficher.
- La propriété *Nombre de valeurs* permet de spécifier le nombre de valeurs (d'étiquette) à afficher.
- Les propriétés *Minimum* et *Maximum* permettent d'indiquer la mise à l'échelle.
- Les propriétés *Format de Remplissage* permettent de définir la couleur de remplissage de la primitive. Les zones de remplissage des barres sont dessinées dans le motif et les couleurs indiqués alors que les zones non remplies sont dessinées avec la *Couleur 2* pleine.
- Les propriétés *Format Ligne* permettent de définir la bordure de la primitive.

LA PRIMITIVE GRAPHIQUE A POINTS

La primitive *Graph à points* affiche un groupe de valeurs de données qui seront reliées entre elles par des points ou des segments de droite. Elle fait appel à un tableau de données.

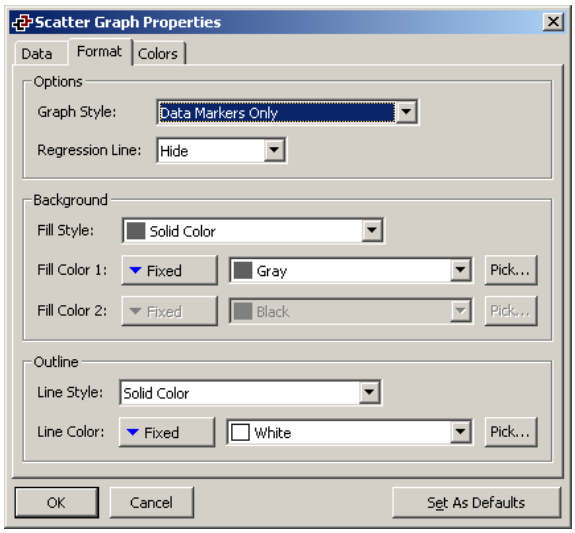
AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

Les propriétés sont divisées sur trois onglets et vous pouvez y accéder en double-cliquant sur la primitive. L'onglet Données définit les sources et la mise à l'échelle des données et vous pouvez le trouver ci-dessous…



- La propriété *Valeurs X* permet de définir l'élément du tableau qui contient la première coordonnée *x* à combiner alors que la propriété *Valeurs Y* définit l'élément du tableau qui contient la coordonnée *y* associée.
- La propriété *Nombre de valeurs* permet de spécifier le nombre de valeurs (d'étiquette du tableau) à afficher.
- Les propriétés *Axe X minimum* et *Axe X maximum* permettent de spécifier la mise à l'échelle de l'axe horizontal alors que les propriétés *Axe Y minimum* et *Axe Y maximum* permettent de spécifier la mise à l'échelle de l'axe vertical.

L'onglet Format définit les différentes options de mise en forme…



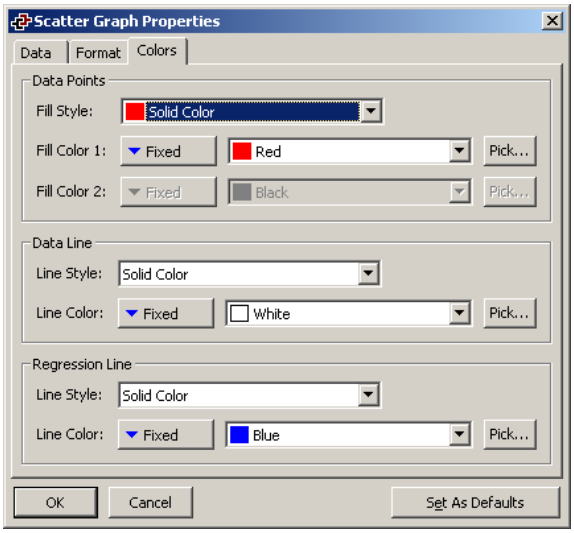
- La propriété *Style du Graphique* vous permet de choisir entre un graphique à segment de droite, un graphique à ligne avec des indicateurs de données ou un

ensemble d'indicateurs de données sans aucune ligne. Les autres options de mise en forme de la page Couleur sont activées ou désactivées, le cas échéant.

- La propriété *Ligne de Régression* permet d'afficher ou masquer la ligne de régression des jeux de données. Si la ligne est activée, le logiciel calcule la ligne qui correspond le mieux en se basant sur la méthode des moindres carrés et la dessine en haut des jeux de données.

- Les propriétés restantes permettent de définir la couleur de l'arrière-plan de la primitive et si une bordure doit être dessinée ou non autour de son bord extérieur.

L'onglet Couleurs définit les couleurs des différents éléments du tableau…



- Les propriétés *Points des Données* permettent de définir le motif et les couleurs utilisés pour créer les indicateurs des points de données. Ces propriétés sont uniquement accessibles si ces indicateurs des points de données du tableau sont activés.

- Les propriétés *Ligne des Données* permettent de définir le format et la couleur de la ligne dessinée entre les points des données. Ces propriétés sont uniquement accessibles si cette ligne du tableau est activée.

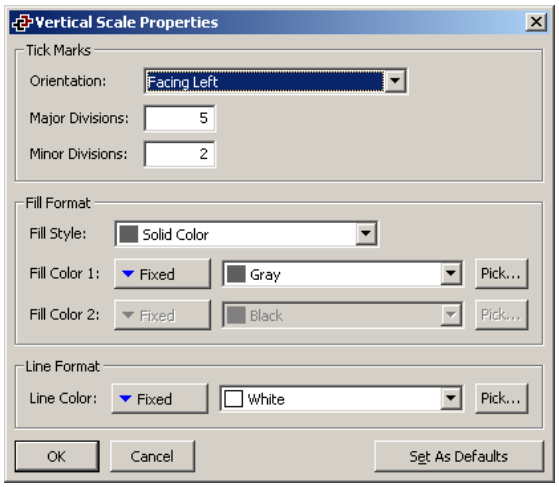
- Les propriétés *Ligne de Régression* permettent de définir le format et la couleur de la ligne dessinée pour une correspondance parfaite avec les points des données. Ces propriétés sont uniquement accessibles si cette ligne du tableau est activée.

LES PRIMITIVES ECHELLE

La primitive *Echelle Horizontale* affiche une échelle avec un nombre spécifié de divisions mineures et majeures. Elle est souvent utilisée pour donner une échelle à d'autres primitives comme les graphiques à barres.

La primitive *Echelle Verticale* affiche une échelle avec un nombre spécifié de divisions mineures et majeures. Elle est souvent utilisée pour donner une échelle à d'autres primitives comme les graphiques à barres.

Vous pouvez accéder aux propriétés en double-cliquant sur la primitive…

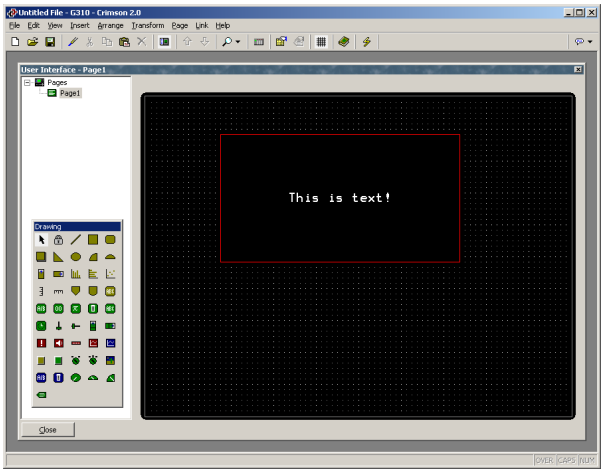


- La propriété *Orientation* permet d'indiquer la direction vers laquelle les l'échelle doit être. Les échelles verticales prennent en charge les sélections de la gauche à la droite alors que les échelles horizontales prennent en charge les sélections du haut vers le bas.
- La propriété *Divisions Principales* permet d'indiquer le nombre de divisions principales dans lequel l'échelle doit être divisée. Pour chaque division, de grands échelons sont dessinés. Le nombre minimal de divisions principales est un, auquel cas de grands échelons sont dessinés aux extrémités de l'échelle et non pas sur sa longueur.
- La propriété *Divisions Secondaires* permet d'indiquer le nombre de divisions secondaires dans lequel chaque division principale doit être divisée. Pour chaque division, des échelons plus petits sont dessinés. Lorsque vous sélectionnez une valeur de 1 pour cette propriété, les divisions secondaires sont désactivées.
- Les propriétés restantes définissent le motif de remplissage et le style de ligne de l'échelle.

LA PRIMITIVE TEXTE FIXE

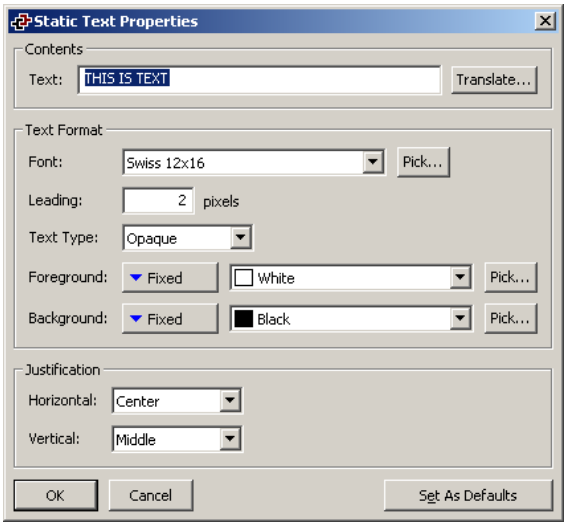
La primitive *Texte fixe* permet d'ajouter du texte fixe sur une page. Le texte est affiché dans une police, une couleur et une justification spécifiées. Vous pouvez également traduire le texte pour des applications internationales.

Lorsque le texte est créé, un curseur s'affiche, permettant de saisir du texte…



L'éditeur de texte prend en charge les opérations de couper et coller ainsi que toutes les autres options qui se trouvent normalement dans un éditeur Windows. L'éditeur configure également le clavier pour qu'il utilise l'éditeur de méthode d'entrée approprié pour la langue par défaut actuellement sélectionnée.

Notez que seul le texte de la langue par défaut peut être directement modifié. Vous devez modifier les autres versions du texte via la boîte de dialogue des propriétés à laquelle vous pouvez accéder en sélectionnant la primitive et en appuyant sur **`Alt+Entrée`** ou en sélectionnant la commande Propriétés à partir du menu Edition…



- La propriété *Texte* permet de spécifier le texte à afficher. Comme nous venons de l'expliquer, vous pouvez également modifier directement la version du texte de la langue par défaut sur la page d'affichage lors de la création de la primitive ou en cliquant sur une primitive existante.

- La propriété *Police* permet de spécifier la police à utiliser. La liste des polices comprend les huit polices résidantes qui se trouvent dans tous les maîtres MC, DSP et G3, plus toutes les polices personnalisées qui sont déjà créées dans la

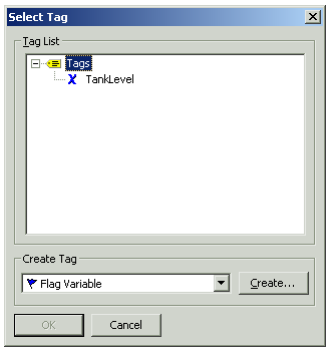
base de données. Vous pouvez utiliser le bouton Choisir pour appeler la boîte de dialogue de sélection des polices. Elle permet à toutes les polices qui sont installées sur votre système d'être rendues dans une forme que le périphérique de destination peut utiliser. Notez qu'il est de votre ressort de vous assurer que votre licence relative aux polices autorise ce type d'utilisation.

- La propriété *Type de Texte* permet d'indiquer si le texte doit être dessiné avec un arrière-plan plein ou transparent. Vous pouvez utiliser du texte transparent pour superposer plusieurs primitives tout en autorisant l'affichage de ces primitives.
- Les propriétés *Avant-plan* et *Arrière-plan* permettent de spécifier les couleurs à utiliser pour dessiner le texte. A l'évidence, si les deux paramètres possèdent la même couleur, le texte sera invisible, un fait qui peut être exploité pour afficher ou masquer le texte, le cas échéant.
- Les propriétés de justification *Horizontal* et *Vertical* permettent d'indiquer l'emplacement du texte dans le rectangle englobant de la primitive.

LA PRIMITIVE ETIQUETTE

La primitive *Etiquette* vous permet de sélectionner une étiquette, puis de placer automatiquement la primitive de texte appropriée sur l'écran. Par exemple, lorsque vous sélectionnez une étiquette d'entiers, l'insertion d'une primitive de texte d'entiers configurée de façon appropriée est effectuée.

C'est l'icône que vous utilisez le plus souvent pour ajouter des étiquettes sur une page. Elle affiche d'abord la boîte de dialogue ci-dessous pour permettre de sélectionner des étiquettes, puis elle crée l'une des cinq primitives Texte étiquette qui sont décrites dans la prochaine section. La nouvelle primitive est configurée pour afficher l'étiquette en question à l'aide de son nom et de ses propriétés de mise en forme, tels que définis lors de la création de l'étiquette.



LES PRIMITIVES TEXTE ETIQUETTE

Les primitives Texte étiquette permettent d'afficher ou de modifier une expression dans une forme textuelle. Elles sont principalement utilisées pour afficher des étiquettes. Dans ce cas, le format par défaut provient de l'onglet Format associé à cette étiquette dans la fenêtre Etiquettes de données. Si une expression de non-étiquette est entrée (ou si vous souhaitez que la mise en forme diffère des valeurs par défaut d'une étiquette), vous pouvez remplacer les

données de format tel que requis. Il existe un type de texte étiquette pour chaque famille d'étiquettes…

La primitive *Texte d'etat* permet d'afficher une condition vraie ou fausse.

La primitive *Texte entier* permet d'afficher une expression d'entiers.

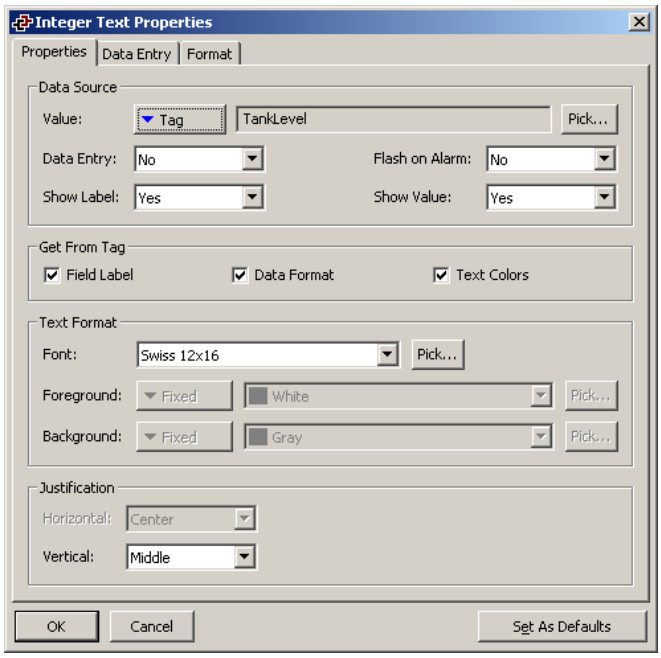
La primitive *Texte réel* permet d'afficher une expression en virgule flottante.

La primitive *Multi-texte* permet d'afficher une condition multi-état.

La primitive *Texte de chaîne* permet d'afficher une expression de chaîne de caractères.

Les propriétés d'une primitive Texte étiquette sont affichées à l'aide de trois pages avec onglets.

Le premier onglet est plus ou moins identique pour les cinq types de primitives…



- La propriété *Valeur* permet d'indiquer l'emplacement à partir duquel vous pouvez obtenir les données de cette primitive. Vous pouvez sélectionner une étiquette, un registre dans un périphérique de communication ou une expression qui associe plusieurs de ces éléments. Le type de données de l'élément doit être

adapté à la primitive en question. Par exemple, la propriété Valeur d'une primitive Texte entier ne peut pas être définie sur égale à une expression de chaîne.

- La propriété *Entrée de données* permet d'indiquer si vous souhaitez ou non que l'utilisateur du HMI virtuel puisse modifier la valeur sous-jacente via cette primitive. Si vous sélectionnez Local, la saisie de données est activée, mais l'accès n'est pas autorisé via la fonction du HMI virtuel du serveur Web. Pour que la saisie de données soit activée, l'expression saisie pour la propriété de la valeur doit pouvoir être modifiée. Par exemple, si une formule est saisie, la saisie de données n'est pas autorisée.

- La propriété *Afficher étiquette* permet d'indiquer si vous souhaitez ou non que la primitive comprenne un nom pour identifier les données affichées. Si cette propriété est définie sur Oui, le nom est justifié à gauche dans le rectangle englobant de la primitive alors que les données sont justifiées à droite. Si cette propriété est définie sur Non, la propriété Justification horizontale permet de rechercher les données dans le champ.

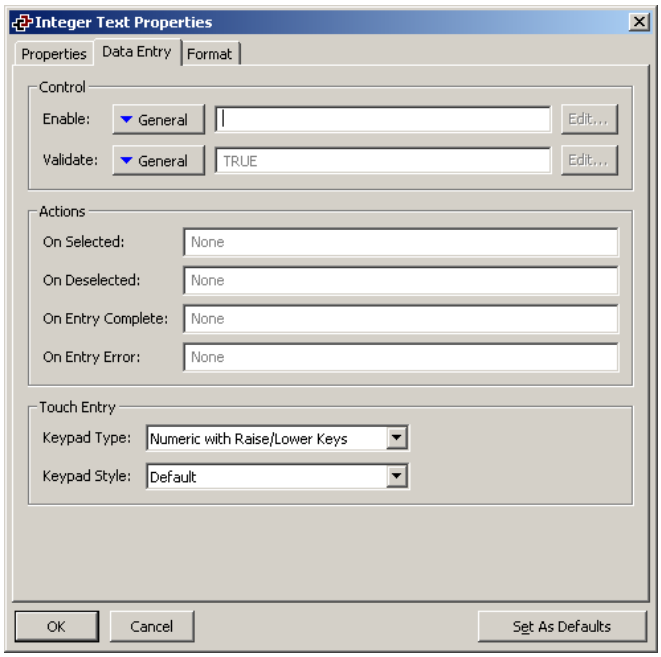
- La propriété *Afficher les Données* permet d'indiquer si la primitive doit inclure ou non la valeur des données ou si elle doit simplement afficher le nom. Comme la primitive peut refléter l'état de l'élément de données sous-jacent par le biais de la couleur uniquement, la valeur réelle peut parfois être omise.

- Les propriétés *Obtenir depuis étiquette* permettent d'indiquer l'emplacement à partir duquel vous pouvez obtenir le texte de l'étiquette, le format du champ et les couleurs du texte. Les options présentées dépendent de ce que vous avez saisi comme propriété Valeur. Dans chacun des cas, vous pouvez saisir manuellement les données dans les propriétés appropriées ou, en supposant qu'une expression adaptée a été définie, vous pouvez demander à la primitive d'obtenir les informations requises à partir de l'étiquette de données sous-jacente.

- La propriété *Clignotement sur Alarme* permet d'indiquer si vous souhaitez ou non que le texte de l'écran du maître clignote si l'étiquette entrée dans la propriété Valeur est actuellement sur l'état d'alarme. Cette propriété n'est pas disponible pour les primitives Texte de chaîne ou pour les primitives qui ont une valeur de non-étiquette définie pour la propriété Valeur.

- L'équilibre des propriétés contrôle la police, les couleurs et la justification à utiliser lorsque vous dessinez la primitive. Ces propriétés ne nécessitent aucune explication supplémentaire.

Le deuxième onglet est uniquement utilisé pour les champs qui sont sélectionnés pour la saisie de données…



- La propriété *Activer* permet de définir une expression qui doit être vraie pour que la saisie de données soit autorisée. Ainsi, vous pouvez utiliser cette propriété pour implémenter un système de sécurité ou pour limiter l'entrée sur certains états de machine.

- La propriété *Condition Validation* permet de définir une expression qui est utilisée pour valider toutes les valeurs d'entrée. Par exemple : **`DATA %25 == 0`** permet uniquement de saisir des multiples de **`25`** dans la variable **`DATA`**. La variable système particulière Data conserve la valeur qui vient d'être entrée, mais seulement lors de l'exécution de cette expression. Le code doit faire référence à une valeur autre que zéro pour permettre la saisie ou zéro pour la bloquer.

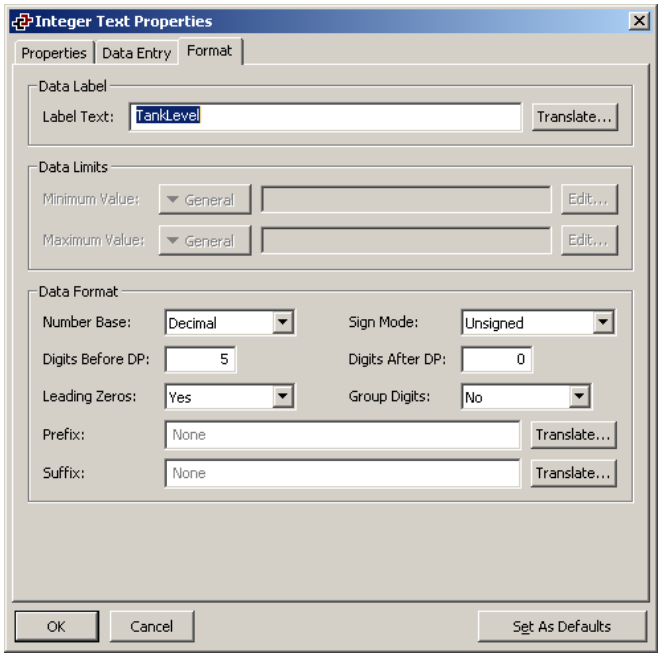
- Les propriétés *Sur Sélection Champ* et *Sur Désélection Champ* sont respectivement utilisées pour définir les actions à exécuter lorsque l'utilisateur sélectionne le champ d'entrée ou qu'il désélectionne le champ pour en sélectionner un autre. Les actions peuvent appeler toutes les différentes fonctions à partir de la section Référence des fonctions ou les opérateurs de la modification des données décrits dans la section Ecriture d'actions ou elles peuvent encore exécuter un programme.

- Les propriétés *Sur Saisie Valide* et *Sur Erreurs* sont respectivement utilisées pour définir les actions à exécuter lorsque la saisie de données s'est correctement terminée ou qu'une valeur invalide a été saisie. Les actions peuvent appeler toutes les fonctions à partir de la section Référence des fonctions ou les opérateurs de la modification des données décrits dans la section Ecriture d'actions ou elles peuvent encore exécuter un programme.

- La propriété *Type de Pavé Numérique* permet de sélectionner le type de pavé numérique à afficher lors de la modification de la valeur. Par défaut, un pavé numérique complet avec les touches Raise et Lower s'affiche. Les options disponibles varient en fonction du type de primitive.

- La propriété *Style de Pavé Numérique* permet de remplacer le jeu de couleurs par défaut qui est associé au pavé numérique contextuel par une version à contraste important. Elle est utilisée dans les applications à faible visibilité comme la lumière directe du soleil.

Le troisième onglet varie selon la primitive en question et affiche les mêmes informations que l'onglet Format du type d'étiquette associé. Différentes sections de la page sont activées en fonction des paramètres fournis pour les propriétés Texte d'étiquette et Format d'étiquette. L'exemple ci-dessous montre l'onglet Format d'une primitive Texte entier…



Comme vous pouvez le voir, les propriétés affichées sont identiques à celles qui sont affichées sous l'onglet Format d'une étiquette d'entiers. Comme nous venons de l'expliquer, les propriétés des autres types de primitives sont tout aussi identiques à celles de l'étiquette correspondante. Ainsi, vous pouvez vous reporter à la section précédente du manuel concernant les étiquettes de données afin d'obtenir plus d'informations sur chaque propriété.

MODIFICATION DE L'ETIQUETTE ASSOCIEE

Si vous souhaitez modifier les propriétés d'une primitive Texte étiquette, double-cliquez sur la primitive ou cliquez avec le bouton droit et sélectionnez la commande Propriétés dans le menu. Cependant, si vous souhaitez modifier les propriétés de l'étiquette qui est en cours d'utilisation pour contrôler la primitive, cliquez avec le bouton droit pour sélectionner la commande Détails d'étiquette. La boîte de dialogue affiche les onglets Données, Format et Couleurs à partir de la fenêtre Etiquettes et vous permet de modifier les différentes propriétés. Notez qu'une modification apportée via ce mécanisme change toutes les primitives contrôlées

par cette étiquette si ces primitives sont configurées pour obtenir leur configuration à partir de cette source.

LES PRIMITIVES TEXTE MULTI-LIGNE

La primitive *Multi-Ligne Texte de status* permet d'afficher une valeur on-off, mais qui est séparée sur plusieurs lignes. Ainsi, de plus grandes quantités de texte sont affichées, peut-être pour fournir des invites ou des informations d'aide à l'opérateur.

La primitive *Multi-Ligne Multi Texte* permet d'afficher l'une des séries des valeurs de texte, mais qui est séparée sur plusieurs lignes. Ainsi, de plus grandes quantités de texte sont affichées, peut-être pour fournir des invites ou des informations d'aide à l'opérateur.

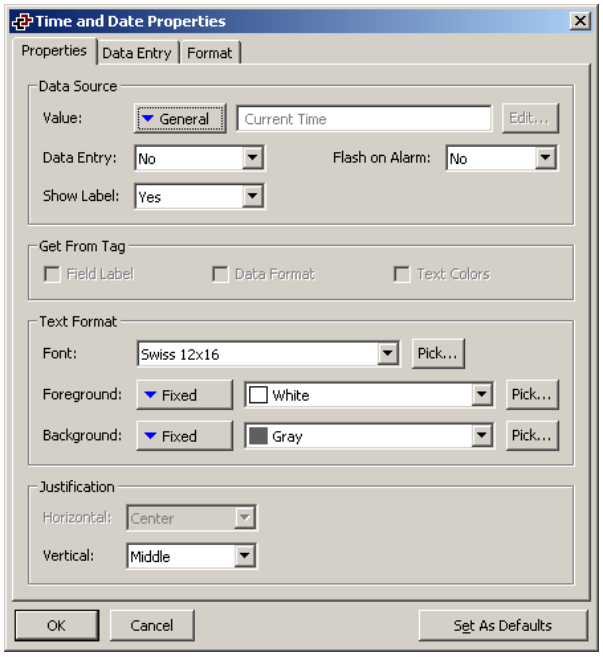
Chacune de ces primitives est comme la primitive de ligne unique associée sauf qu'elles ne prennent pas en charge la saisie de données. La chaîne de texte à afficher est divisée en lignes par l'ajout de caractères de barre verticale (« | ») là où un saut de ligne est nécessaire.

LA PRIMITIVE HEURE ET DATE

La primitive *Heure et date* permet d'afficher l'heure et la date actuelles ou le contenu d'une expression d'heure ou de date. Elle peut également être utilisée pour modifier une telle expression ou pour définir l'heure réelle du maître.

Les propriétés d'une primitive Heure et date sont affichées à l'aide de trois pages avec onglets.

Le premier onglet se présente comme suit…



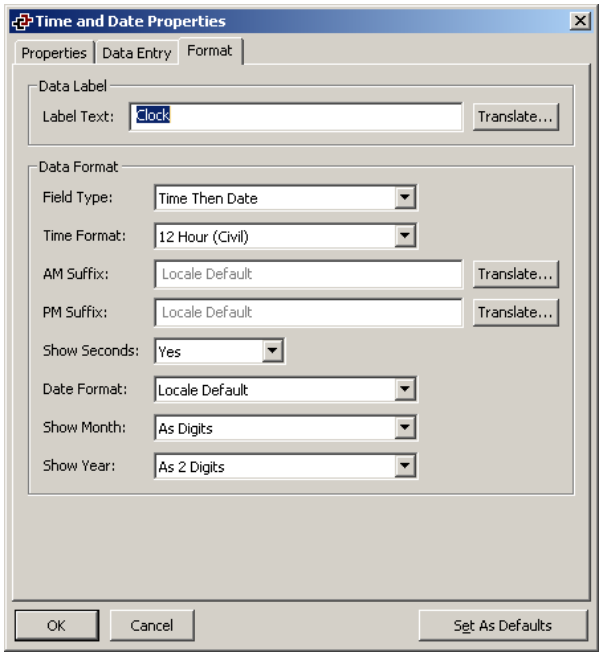
- La propriété *Valeur* permet d'indiquer la valeur du temps et la date à afficher. Si aucune valeur n'est saisie, l'heure et la date actuelles sont affichées. Si une expression est saisie, elle sert à représenter le nombre de secondes qui se sont écoulées depuis le 1er janvier 1997. Ces valeurs sont généralement obtenues à l'aide des différentes fonctions d'heure et de date décrites dans la section Référence des fonctions.

- La propriété *Entrée de données* permet d'indiquer si vous souhaitez ou non que l'utilisateur du HMI virtuel puisse modifier la valeur de l'étiquette associé via cette primitive. Si vous sélectionnez Local, la saisie de données est activée, mais l'accès n'est pas autorisé via la fonction du HMI virtuel du serveur Web. Si aucune propriété de la valeur n'a été définie, cela équivaut à modifier l'heure et la date actuelles. Si une propriété de la valeur a été entrée, l'expression saisie pour la propriété de la valeur doit pouvoir être modifiée. Par exemple, si une formule est saisie, la saisie de données n'est pas autorisée.

- L'équilibre des propriétés est identique à celui qui est décrit pour les primitives Texte étiquette. (Même s'il peut sembler étrange d'avoir les propriétés Obtenir depuis étiquette et Clignotement sur Alarme, souvenez-vous que la propriété de la valeur peut être une étiquette. Par conséquent, Crimson a accès au nom de l'étiquette et à l'état d'alarme de l'étiquette si vous décidez de les utiliser.)

Le deuxième onglet contient les propriétés de la saisie de données. Elles sont identiques à celles des primitives Texte Etiquette.

Le troisième onglet se présente comme suit…



- La propriété *Texte d'étiquette* permet de définir un nom facultatif pour la primitive.

- La propriété *Type de champ* permet d'indiquer si le champ doit afficher l'heure, la date ou les deux. Dans le dernier cas, cette propriété indique également l'ordre d'affichage de ces deux éléments.
- La propriété *Format de l'heure* permet d'indiquer le format horaire à utiliser : 12 heures (civil) ou 24 heures (militaire). Comme avec les autres propriétés, si vous laissez cette propriété définie sur Défaut local, Crimson peut choisir un format approprié en fonction de la langue sélectionnée dans le HMI.
- Les propriétés *Suffixe AM* et *Suffixe PM* sont utilisées avec le mode 12 heures pour indiquer le texte à ajouter au champ heure le matin et l'après-midi, le cas échéant. Si vous laissez ces champs à blanc, Crimson utilisera les paramètres par défaut.
- La propriété *Afficher secondes* permet d'indiquer si le champ heure doit mentionner les secondes ou juste les heures et les minutes.
- La propriété *Format de la date* permet d'indiquer l'ordre d'affichage des différents éléments de la date (c'est-à-dire la date, le mois et l'année).
- La propriété *Afficher le mois* permet d'indiquer si le mois doit être affiché sous forme de chiffres (de 01 à 12) ou par son abréviation (c'est-à-dire janv. A déc.).
- La propriété *Afficher l'année* permet de choisir si le champ heure doit indiquer l'année et, le cas échéant, le nombre de chiffres à afficher pour cet élément.

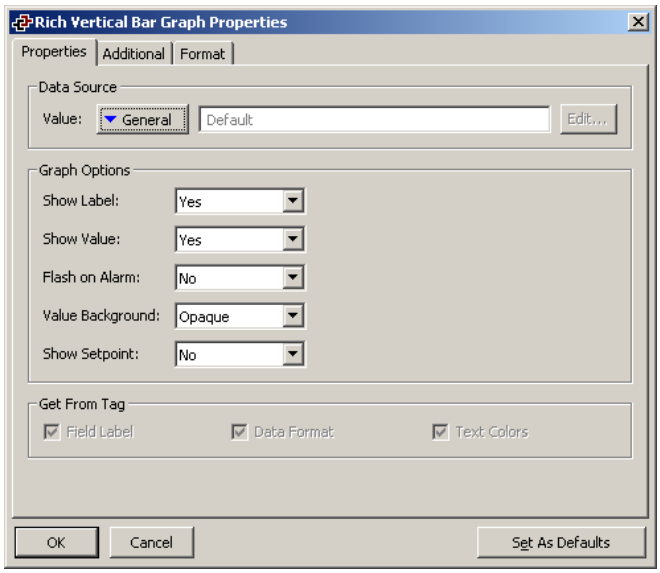
LES PRIMITIVES BARRES GRAPH RICHES

La primitive *Barre verticale légendable* vous permet d'afficher un graphique à barres plus complexe qui comprend un nom, une version numérique des données affichées et des indicateurs Tick permettant d'indiquer un point de consigne éventuellement associé.

La primitive *Barre horizontale riche* vous permet d'afficher un graphique à barres plus complexe qui comprend un nom, une version numérique des données affichées et des indicateurs Tick permettant d'indiquer un point de consigne éventuellement associé.

AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

Le fonctionnement de ces primitives riches est analogue à celui des différentes primitives Texte étiquette car elles peuvent trouver la majorité des informations de mise en forme requises dans l'étiquette utilisée comme valeur de contrôle. Tout comme avec les primitives Texte étiquette, plusieurs pages avec onglets sont utilisées pour modifier les propriétés des primitives. Le premier de ces onglets se présente comme suit…

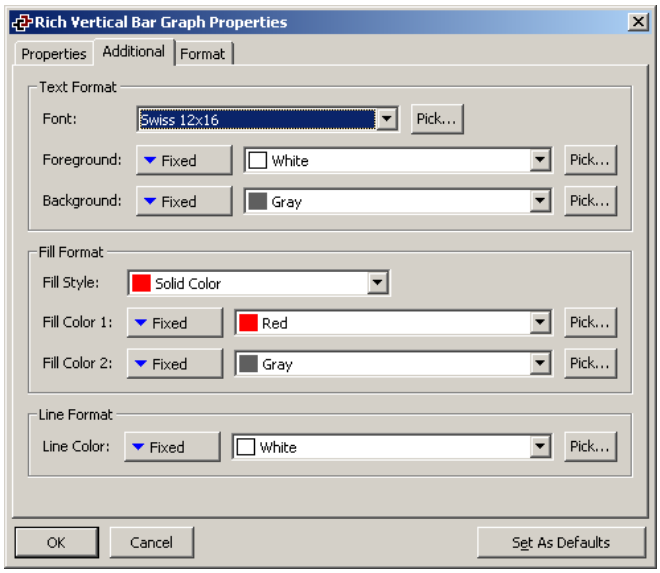


- La propriété *Valeur* permet de définir la valeur à afficher.
- La propriété *Afficher étiquette* permet de choisir si un nom doit être ajouté ou non dans le graphique à barres. Pour les graphiques verticaux, le nom est ajouté en bas et pour les graphiques horizontaux, il est ajouté à gauche. Si vous utilisez une étiquette pour la propriété de la valeur, vous pouvez obtenir le nom à partir de l'étiquette. Sinon, il doit être entré sous l'onglet Format de la boîte de dialogue.
- La propriété *Afficher donnée* permet de choisir si la valeur de la donnée doit être affichée ou non dans le graphique lui-même. Si vous utilisez une étiquette du type de donnée approprié pour la propriété de la valeur, vous pouvez obtenir le format à partir de l'étiquette. Sinon, comme pour le nom, il doit être entré sous l'onglet Format.
- La propriété *Montrer Point de Consigne* permet de définir les indicateurs Tick qui doivent être ajoutés d'un côté ou l'autre de la barre pour spécifier le point de consigne de la valeur de contrôle. Cette option n'est disponible que si une étiquette a été saisie pour la valeur.
- La propriété *Clignotement sur Alarme* permet d'indiquer si vous souhaitez ou non que le texte de l'écran du maître clignote si l'étiquette entrée dans la propriété Valeur est actuellement sur l'état d'alarme. Cette option n'est pas disponible pour les primitives qui ont une valeur de non-étiquette définie pour la propriété Valeur.

- La propriété *Arrière-Plan Donnée* permet d'indiquer si la valeur doit être dessinée avec un arrière-plan opaque ou transparent. Le choix du format dépend en général de la visibilité de la valeur par rapport à la barre.

- Les propriétés *Obtenir depuis étiquette* permettent d'indiquer l'emplacement à partir duquel vous pouvez obtenir le texte de l'étiquette, le format du champ et les couleurs du texte. Les options présentées dépendent de ce que vous avez saisi comme propriété Valeur. Dans chacun des cas, vous pouvez saisir manuellement les données dans les propriétés appropriées ou, en supposant qu'une expression adaptée a été définie, vous pouvez demander à la primitive d'obtenir les informations requises à partir de l'étiquette de données liée.

Le second onglet contient des informations supplémentaires de mise en forme du champ...



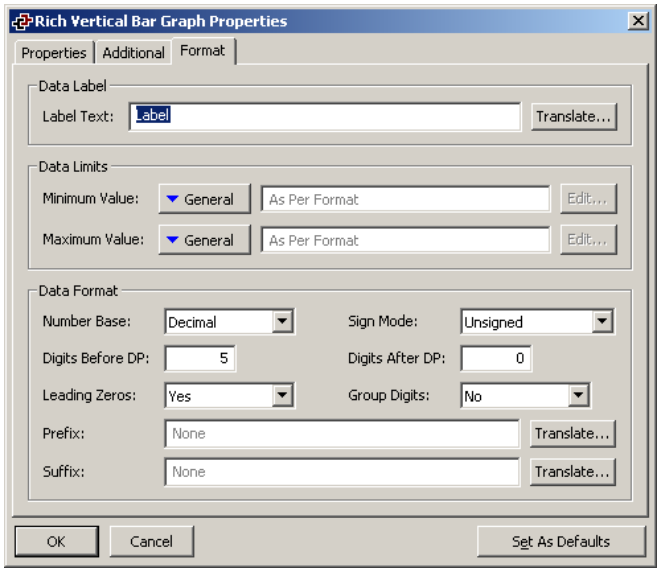
- Les propriétés *Format Texte* permettent de définir la police et les couleurs utilisées pour afficher la valeur et le nom du champ, en supposant que ces éléments sont activés.

- Les propriétés *Format de Remplissage* permettent de définir la couleur de remplissage de la primitive. Les zones de remplissage des barres sont affichées dans le motif et les couleurs indiqués alors que les zones non remplies sont dessinées avec la *Couleur* 2 pleine.

- Les propriétés *Format Ligne* permettent de définir le style de la bordure de la primitive.

AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

Le troisième onglet contient les informations concernant le nom et la mise en forme du champ...



Les propriétés affichées sont les mêmes que celles qui sont décrites pour une étiquette d'entiers et vous devez donc vous reporter à la section précédente du manuel sur les étiquettes de données pour en savoir plus. Notez que l'existence de cette primitive explique la raison pour laquelle il faut saisir des valeurs minimales et maximales pour les formules car de telles étiquettes ne font jamais l'objet d'une entrée de données. Si ces limites n'ont pas été définies, comment Crimson pourrait-il savoir comment mettre la barre à l'échelle ?

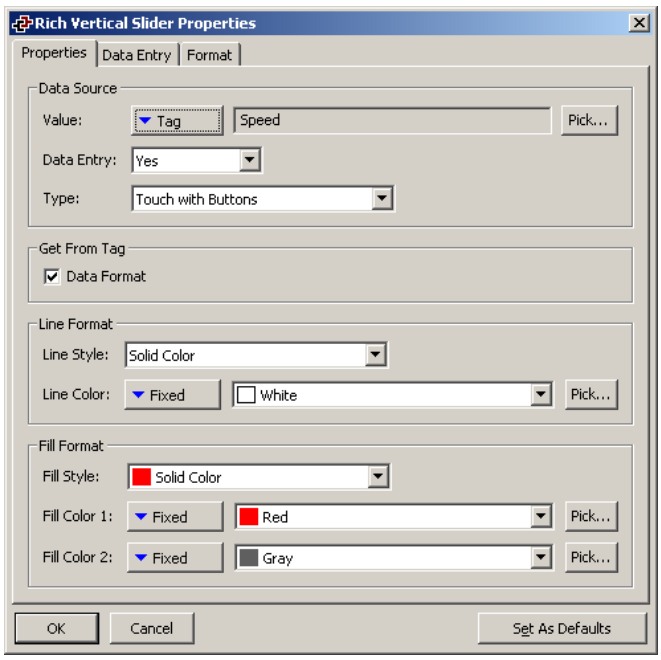
LES PRIMITIVES GLISSEUR RICHES

La primitive *Glisseur Vertical* permet d'afficher une valeur, en général à partir d'une étiquette d'entiers car un glisseur peut afficher une valeur ou permettre de la manipuler en touchant un emplacement spécifique sur la primitive ou en appuyant sur des boutons à l'une des extrémités.

La primitive *Glisseur Horizontal* permet d'afficher une valeur, en général à partir d'une étiquette d'entiers car un glisseur peut afficher une valeur ou permettre de la manipuler en touchant un emplacement spécifique sur la primitive ou en appuyant sur des boutons à l'une des extrémités.

Tout comme avec les autres primitives riches, les primitives Glisseur peuvent obtenir la plupart des informations de mise en forme requises à partir de l'étiquette utilisée comme leur valeur de contrôle. Tout comme avec les primitives Texte étiquette, plusieurs pages avec onglets sont utilisées pour modifier les propriétés des primitives.

Le premier de ces onglets se présente comme suit…



- La propriété *Valeur* permet d'indiquer l'emplacement à partir duquel vous pouvez obtenir les données de cette primitive. Vous pouvez sélectionner une étiquette, un registre dans un périphérique de communication ou une expression qui associe plusieurs de ces éléments.

- La propriété *Entrée de données* permet d'indiquer si vous souhaitez ou non que l'utilisateur du HMI virtuel puisse modifier la valeur de l'étiquette via cette primitive. Si vous sélectionnez Local, la saisie de données est activée, mais l'accès n'est pas autorisé via la fonction du HMI virtuel du serveur Web. Pour que la saisie de données soit activée, l'expression saisie dans le champ Donnée doit pouvoir être modifiée. Par exemple, si une formule est saisie, la saisie de données n'est pas autorisée.

- La propriété *Type* permet d'indiquer le type de glisseur à afficher. Les trois types de glisseurs permettent d'entrer des données via des boutons, une manipulation directe ou ces deux méthodes. N.B : la manipulation directe peut être assez risquée car les contacts accidentels peuvent entraîner d'importantes modifications dans le traitement des valeurs.

- La propriété *Obtenir depuis étiquette* permet d'indiquer si le format des données doit être obtenu à partir de la valeur de contrôle ou de la page Format. Notez que la majorité des valeurs du format ne sont pas utilisées sauf les valeurs minimum et maximum.

- Les propriétés *Format Ligne* permettent de définir le style de la bordure de la primitive.

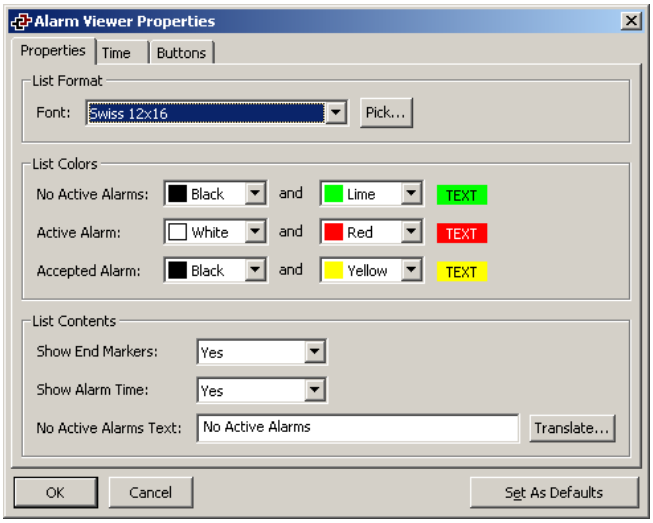
- Les propriétés *Format de Remplissage* permettent de définir la couleur et le style de l'arrière-plan de la primitive Glisseur. L'arrière-plan est dessiné dans *Couleur* 2 alors que le glisseur est dessiné dans *Couleur* 1 ou dans l'association des deux couleurs qui est spécifiée par Type de remplissage.

Le deuxième onglet permet de contrôler la saisie de données et les fonctions sont identiques à celles des primitives Texte étiquette. Pour en savoir plus, il convient donc de vous reporter à la section précédente. Le troisième onglet permet de définir le nom facultatif ainsi que les valeurs minimum et maximum, peut-être au moyen d'un format de données complet. Cet onglet fonctionne comme nous venons de le décrire pour les étiquettes de données d'entiers. Reportez-vous donc à cette section pour en savoir plus.

LA PRIMITIVE AFFICHEUR D'ALARME

La primitive *Afficheur d'alarme* permet de fournir à l'opérateur une méthode pour afficher et accepter les alarmes actives. Plusieurs paires de couleurs sont utilisées pour afficher les différents états d'alarme. Vous pouvez afficher d'autres données sur les alarmes, le cas échéant.

Si vous utilisez des alarmes acceptées manuellement par votre système, vous devez fournir une page qui contient un afficheur d'alarme pour vous assurer que l'opérateur peut accepter ces alarmes. Vous pouvez peut-être envisager de créer une page contextuelle qui vous permettra d'afficher l'afficheur d'alarme même si les restrictions de taille sur les popups peuvent vous pousser à ne pas le faire. Les propriétés de l'afficheur d'alarme sont affichées sur les trois pages avec onglets, et le premier d'entre eux est décrit ci-dessous…

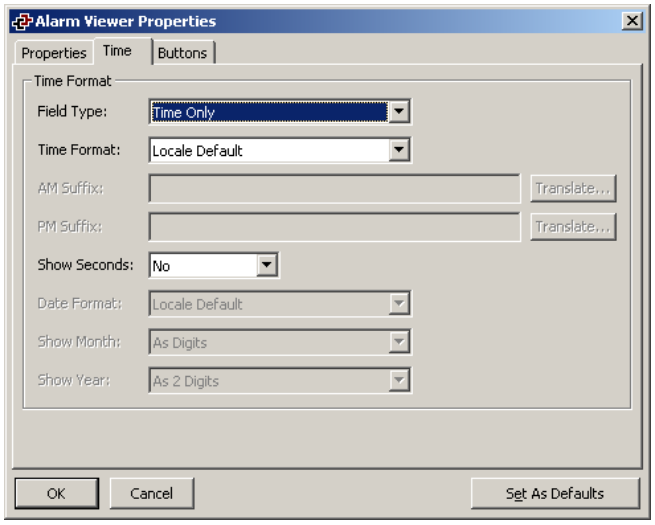


- La propriété *Police* permet de sélectionner la police à utiliser pour dessiner la primitive. L'idéal est d'utiliser une police non proportionnelle pour garantir que les différents champs de données restent dans l'alignement correct.

- Les propriétés *Liste des couleurs* permettent de définir les couleurs du premier plan et de l'arrière-plan qui sont utilisées pour afficher l'état de l'alarme. Les valeurs par défaut doivent être adaptées à la plupart des applications.

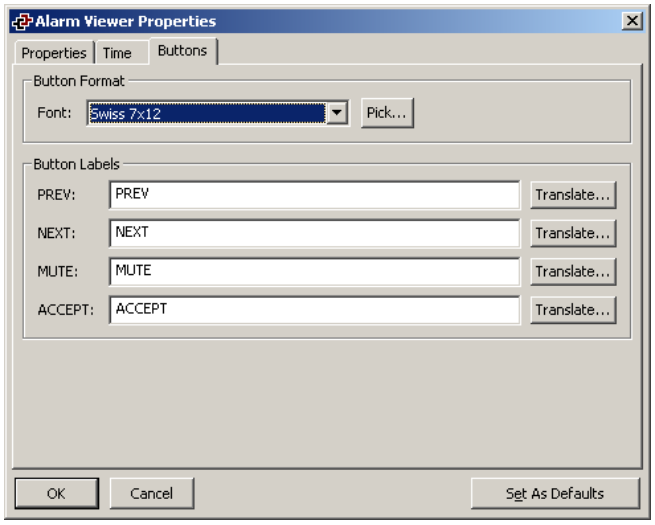
- La propriété *Afficher Indicateurs d'Arrêt* permet d'indiquer s'il est nécessaire d'afficher une colonne contenant des indicateurs qui montrent le début et la fin de la liste des alarmes. Si la colonne est omise, la primitive prend moins d'espace, mais l'opérateur a plus de mal à déterminer les limites de la liste.
- La propriété *Afficher Heure de l'Alarme* permet d'indiquer si l'heure de l'alarme doit être ajoutée dans la primitive. Si l'heure est affichée, le deuxième onglet permet de définir le format à utiliser.
- La propriété *Texte Pas d'Alarmes Actives* permet de remplacer le texte par défaut qui est affiché en l'absence d'alarmes ou d'entrer les versions localisées de ce texte sur les systèmes qui prennent plusieurs langues en charge.

Le deuxième onglet des propriétés permet de définir le format de l'heure de l'alarme…



Les propriétés sont identiques à celles de la primitive Heure et date.

Le dernier onglet des propriétés permet de contrôler la façon dont les boutons de la primitive sont nommés…



AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

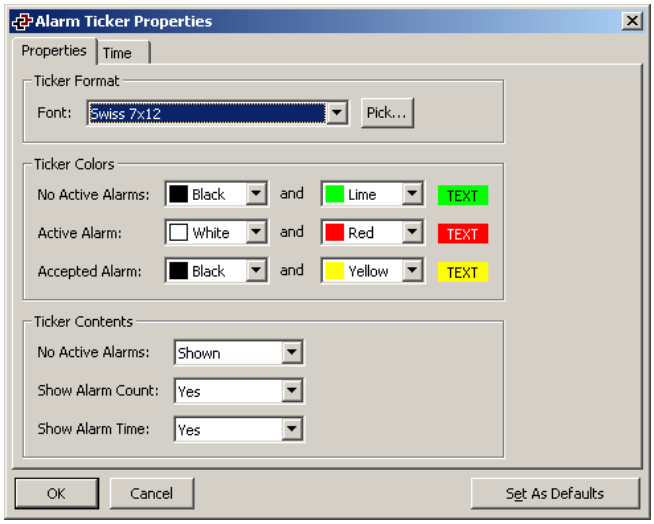
- La propriété *Police* permet de sélectionner la police requise.
- Les propriétés *Textes des Boutons* permettent de remplacer le nom par défaut qui s'affiche sur chacun des quatre autres boutons ou d'entrer les versions localisées de ce texte sur les systèmes qui prennent plusieurs langues en charge.

LA PRIMITIVE BANDEAU D'ALARMES

La primitive *Bandeau d'alarmes* permet de faire défiler les alarmes actives du système. Elle occupe une seule ligne et elle est généralement placée dans la partie inférieure de l'écran, peut-être sur chaque page. Elle ne permet pas à l'opérateur d'accepter les alarmes.

Les propriétés du bandeau d'alarmes sont affichées sur deux pages avec onglets.

Le premier de ces onglets se présente comme suit…



- La propriété *Police* permet de sélectionner la police à utiliser pour dessiner la primitive. L'idéal est d'utiliser une police non proportionnelle pour garantir que les différents champs de données restent dans l'alignement correct.
- Les propriétés *Couleur du Bandeau* permettent de définir les couleurs du premier plan et de l'arrière-plan qui sont utilisées pour afficher chaque état de l'alarme. Les valeurs par défaut doivent être adaptées à la plupart des applications.
- La propriété *Pas d'Alarmes Actives* permet d'indiquer si un message doit être affiché en l'absence des alarmes ou si la barre doit être laissée vierge dans de telles circonstances.
- La propriété *Afficher Compteur d'alarmes* permet d'indiquer si le nombre d'alarmes actuellement actives doit être affiché dans la primitive. Sauf si l'espace d'affichage est restreint, l'affichage de ce champ améliore généralement la lisibilité de l'opérateur.

- La propriété *Afficher Heure de l'Alarme* permet d'indiquer si l'heure de l'alarme doit être ajoutée dans la primitive. Si l'heure est affichée, le deuxième onglet permet de définir le format à utiliser.

Le deuxième onglet est le format de l'heure et il est décrit pour la primitive Heure et date.

LA PRIMITIVE AFFICHEUR D'EVENEMENTS

La primitive *Afficheur d'événements* permet de fournir à l'opérateur une méthode pour afficher les événements enregistrés dans le journal d'événements du système. Comme pour l'afficheur d'alarme, il est parfois placé dans une page contextuelle.

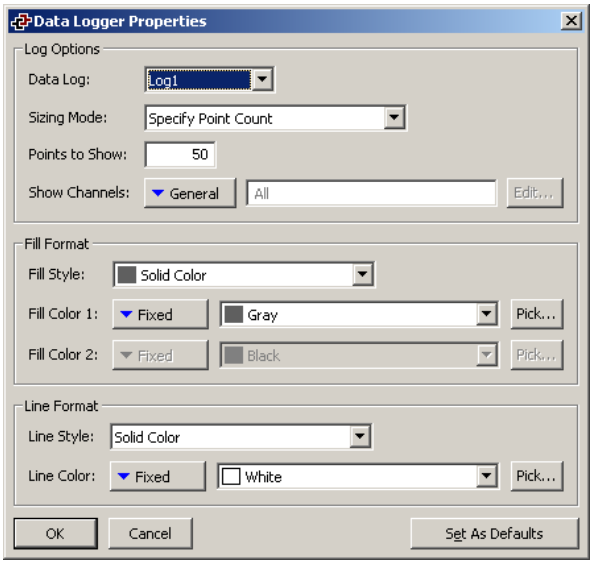
Les propriétés de cette primitive sont identiques à celles de l'afficheur d'alarme. Pour en savoir plus, veuillez vous reporter à la section précédente. La seule propriété supplémentaire est *Afficher le type d'évènements,* qui est utilisée pour indiquer si chaque ligne doit être ou non nommée avec le type d'événement qui a provoqué l'entrée de journal. Les types d'événements possibles sont les activations, acquittements et désactivations d'alarmes ainsi que les activations d'événements.

LES PRIMITIVES HISTORIQUE/COURBES DE DONNEES

La primitive *Historique de Données* donne une vue fixe des données contenues dans un historique de données. Vous pouvez définir le nombre de points de données à afficher et afficher ou masquer les canaux à l'aide d'un masque de bits.

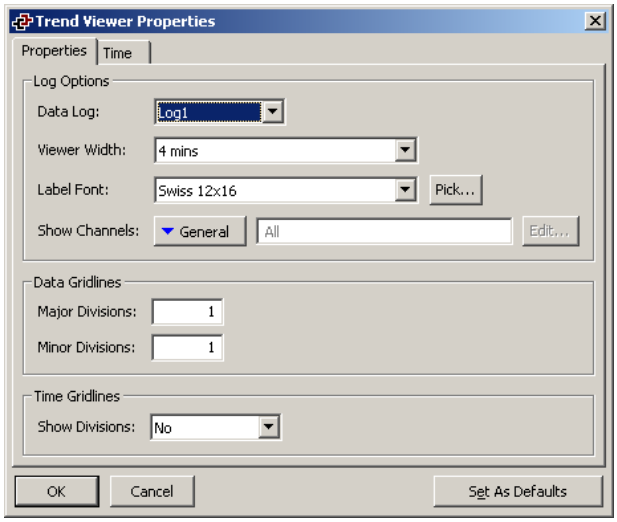
La primitive *Historique de Courbes* donne une vue interactive plus avancée d'un historique de données, ce qui permet à l'opérateur de faire un zoom avant, arrière ou de faire défiler vers l'arrière ou l'avant dans les données d'historique qui sont enregistrées dans la mémoire tampon de l'historique.

La primitive Historique de Données est configurée via une seule page de propriété présentée ci-dessous…



- La propriété *Historique de Données* permet de sélectionner l'historique de données à afficher.
- La propriété *Dimensionnement* permet d'indiquer si vous souhaitez ou non spécifier le nombre de points de données à afficher ou si vous souhaitez que le logiciel affiche un point de données pour chaque pixel horizontal de la primitive. La propriété *Nombres de points* permet de définir le nombre de points à afficher lorsque le Dimensionnement est configuré de cette façon.
- La propriété *Voir Canaux* permet de définir une valeur entière facultative qui contrôle les canaux affichés. Si une expression est entrée, le canal *n* s'affiche si et uniquement si le $n^{ème}$ bit de la valeur est défini. Le $n^{ème}$ bit est défini comme le bit ayant un poids de 2^n, de telle façon que le bit de l'ordre le plus bas est le bit 0.
- Les propriétés *Format de Remplissage* permettent de définir l'arrière-plan de la primitive.
- Les propriétés *Format Ligne* permettent de spécifier le format de la bordure facultative entourant la primitive. Notez que ces propriétés ne modifient pas les stylos utilisés pour dessiner les données actuelles du canal : les couleurs de ces lignes sont définies par le système.

La primitive Historique de Courbes est semblable, mais elle possède un deuxième onglet qui est utilisé pour spécifier le format de l'heure et de la date qui sert à indiquer les extrémités des données d'affichage. Le premier onglet des propriétés se présente comme suit…



- La propriété *Historique de Données* permet de sélectionner l'historique de données à afficher. Si vous souhaitez que l'opérateur puisse faire défiler vers l'arrière dans les données d'historique, assurez-vous que la mémoire tampon de l'historique du log est activée. Pour en savoir plus, reportez-vous au chapitre Enregistrement des données.
- La propriété *Taille Visu* permet de définir la quantité de données par défaut à afficher lorsque la primitive est affichée pour la première fois. Notez que

l'opérateur peut faire un zoom avant ou arrière s'il le souhaite et peut donc choisir d'afficher plus ou moins de données.

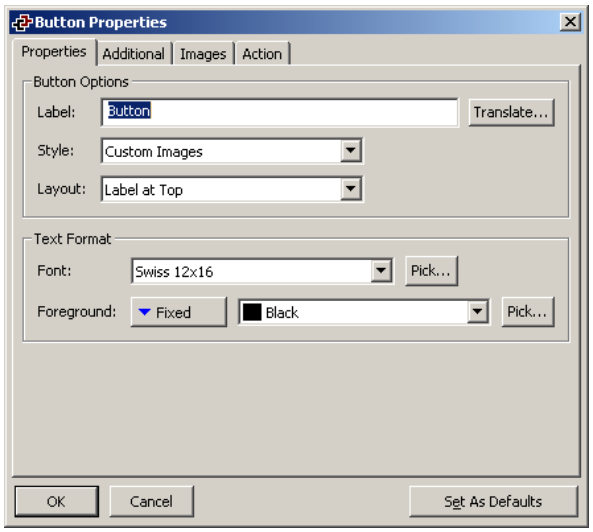
- La propriété *Police du Label* permet de définir la police utilisée pour dessiner les différents noms de la primitive. La police par défaut est en général trop grande pour les applications où la primitive ne prend pas tout l'écran.
- La propriété *Voir Canaux* est identique à celle qui est définie pour la primitive Historique de Données.
- Les propriétés *Quadrillages pour Données* permettent de définir les divisions principales et secondaires à utiliser pour nommer l'axe vertical de l'afficheur. Notez que chaque étiquette qui s'affiche est mise à l'échelle selon les propriétés de son format et que différentes étiquettes peuvent donc avoir différentes mises à l'échelle. Dans l'idéal, vous devez définir des quadrillages logiques pour toutes les étiquettes qui s'affichent et vous assurer que vous nommez la page d'affichage pour que l'opérateur connaisse la mise à l'échelle que vous avez sélectionnée.
- La propriété *Voir Grille* permet d'indiquer si des quadrillages doivent être dessinés pour l'axe du temps. Vous devez choisir les divisions principales et secondaires que le système utilise en fonction du niveau actuel du zoom.

LA PRIMITIVE BOUTON GENERAL

La primitive *Bouton Général* affiche un bouton animé qui peut répondre à l'entrée de l'utilisateur. Plusieurs styles de boutons différents sont fournis, notamment celui qui utilise des images personnalisées à partir de la bibliothèque des images du logiciel.

Les propriétés du bouton général sont définies à l'aide de quatre onglets.

Le premier de ces onglets se présente comme suit…



- La propriété *Label* permet de définir le texte à afficher sur le bouton.

AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

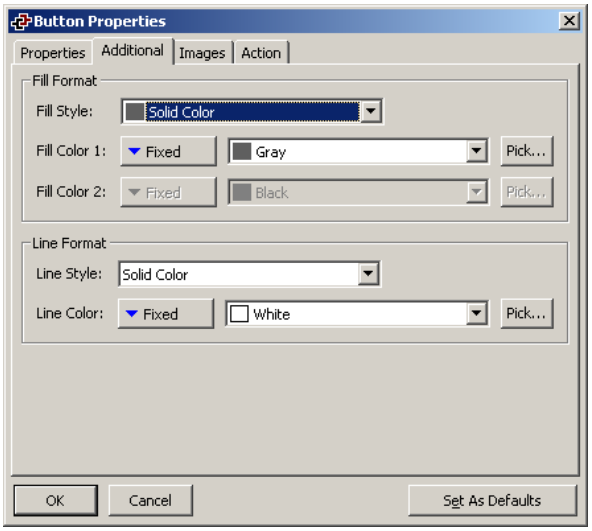
- La propriété *Style* permet de définir la valeur du bouton à afficher…

STYLE	DESCRIPTION
Rond	Un bouton arrondi comprenant deux cercles concentriques.
Rectangle Plat	Un bouton rectangulaire comprenant deux rectangles imbriqués.
3D Rectangle	Un bouton rectangulaire dessiné à l'aide des effets de couleur en 3D.
3D Rectangle avec Bords	Un bouton rectangulaire avec des effets en 3D plus prononcés.
Images	Un bouton basé sur deux images personnalisées.

- La propriété *Disposition* permet d'indiquer l'endroit où, le cas échéant, le nom doit être placé lors de l'utilisation d'images personnalisées pour définir l'aspect du bouton.

- Les propriétés *Format Texte* permettent de définir la police et la couleur du nom.

Le deuxième onglet permet de définir d'autres informations sur la mise en forme…



- Les propriétés *Format de Remplissage* permettent de définir la couleur et le motif à utiliser pour remplir l'intérieur du bouton. Pour les boutons en 3D, cette couleur permet de dessiner le bouton alors que le système utilise différentes teintes de gris pour dessiner la bordure afin d'obtenir l'effet 3D. Pour les boutons basés sur des images personnalisées, le format de remplissage définit l'arrière-plan qui est dessiné sous les images.

- Les propriétés *Format Ligne* permettent de définir le style et la couleur des lignes qui constituent les bordures des styles Rond et Rectangle Plat. Les propriétés ne sont pas utilisées lorsque vous sélectionnez d'autres styles.

Le troisième onglet est utilisé pour les boutons du style Images et permet de définir les images à afficher lorsque vous relâchez ou appuyez sur le bouton. La sélection des images est

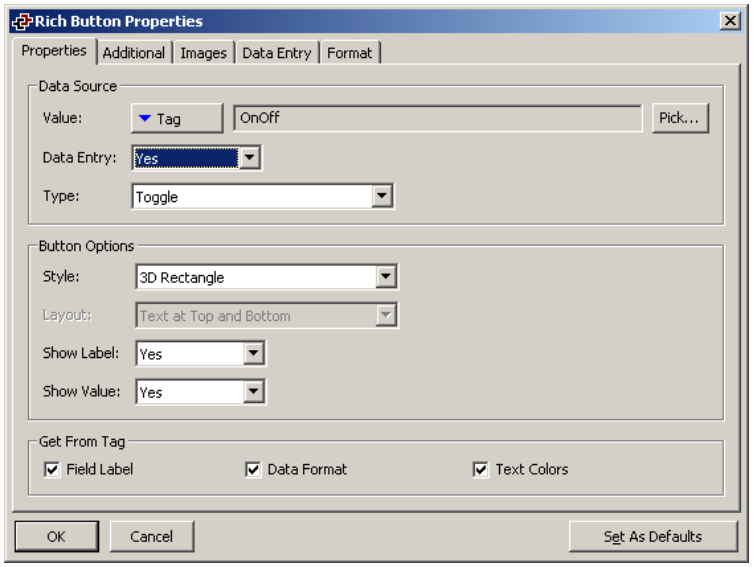
décrite en détail sous la primitive de l'image et vous pouvez donc vous reporter à ces sections. Le quatrième onglet permet de définir l'action que le bouton effectue. Pour en savoir plus, reportez-vous à la section précédente de ce chapitre.

LA PRIMITIVE BOUTON A BIT

La primitive *Bouton à bit* affiche un bouton animé qui permet de contrôler l'état d'une étiquette de type bit. Alors que vous pouvez obtenir la même fonctionnalité à l'aide d'un bouton général, la version riche obtient automatiquement les données à partir de l'étiquette qui en dépend.

Les propriétés du bouton à bit sont définies à l'aide de cinq onglets.

Le premier de ces onglets se présente comme suit…



- La propriété *Valeur* permet d'indiquer l'emplacement depuis lequel vous pouvez obtenir les données de cette primitive. Vous pouvez sélectionner une étiquette, un registre dans un périphérique de communication ou une expression qui associe plusieurs de ces éléments. Le type de données de l'élément doit être un entier ou un bit.

- La propriété *Entrée de données* permet d'indiquer si vous souhaitez ou non que l'utilisateur du HMI virtuel puisse modifier la valeur de l'étiquette associée via cette primitive. Si vous sélectionnez Local, la saisie de données est activée, mais l'accès n'est pas autorisé via la fonction du HMI virtuel du serveur Web. Pour que la saisie de données soit activée, l'expression saisie pour la propriété de la valeur doit pouvoir être modifiée. Par exemple, si une formule est saisie, la saisie de données n'est pas autorisée. La saisie de données de la plupart des boutons est activée.

- La propriété *Type* définit la mesure à prendre lorsque vous appuyez et relâchez le bouton...

TYPE DE BOUTON	LE BOUTON…
Bistable	Modifie l'état des données lorsque vous appuyez sur la primitive.
Momentané	Définit les données sur 1 lorsque vous appuyez sur la primitive. Définit les données sur 0 lorsque vous appuyez sur la primitive.
Actif ON	Définit les données sur 1 lorsque vous appuyez sur la primitive.
Actif OFF	Définit les données sur 0 lorsque vous appuyez sur la primitive.

- La propriété *Style* est identique à celle des boutons de type général.
- La propriété *Disposition* permet de définir l'endroit où le nom et les valeurs de données en option doivent être positionnés en fonction du bouton, lors de l'utilisation d'images personnalisées. Les champs de texte sont toujours placés dans le bouton pour les autres styles de bouton.
- La propriété *Afficher étiquette* permet d'indiquer si vous souhaitez ou non que la primitive comprenne un nom pour identifier les données affichées et contrôlées.
- La propriété *Afficher les Données* permet d'indiquer si la primitive doit inclure ou non la valeur des données associées. Pour les boutons configurés avec un type Bistable ou Momentané, la valeur des données affichée représente la valeur de l'étiquette liée. Pour les autres boutons, la valeur affichée représente la valeur sur laquelle l'étiquette est définie.
- Les propriétés *Obtenir depuis étiquette* permettent d'indiquer l'emplacement à partir duquel vous pouvez obtenir le texte de l'étiquette, le format du champ et les couleurs du texte. Les options présentées dépendent de ce que vous avez saisi comme propriété Valeur. Dans chacun des cas, vous pouvez saisir manuellement les données dans les propriétés appropriées ou, en supposant qu'une expression adaptée a été définie, vous pouvez demander à la primitive d'obtenir les informations requises à partir de l'étiquette de données qui en dépend.

Le deuxième onglet définit les autres informations de mise en page et il est identique à celui de la primitive Bouton général. Le troisième onglet permet de définir les images personnalisées à utiliser pour refléter les états du bouton et il est identique à celui du bouton général sauf que différentes images peuvent être spécifiées en fonction de l'activation ou de la désactivation de l'étiquette qui leurs est liée. Encore une fois, reportez-vous à la primitive de l'image pour obtenir des informations sur la sélection des images. Le quatrième onglet définit plusieurs propriétés qui sont spécifiques à la saisie de données. Elles sont identiques à celles de la primitive Texte étiquette de type bit et vous vous reporterez à cette section pour en savoir plus. Le cinquième et dernier onglet définit le nom et le format à utiliser pour la primitive ; il est identique à celui des étiquettes de type bit.

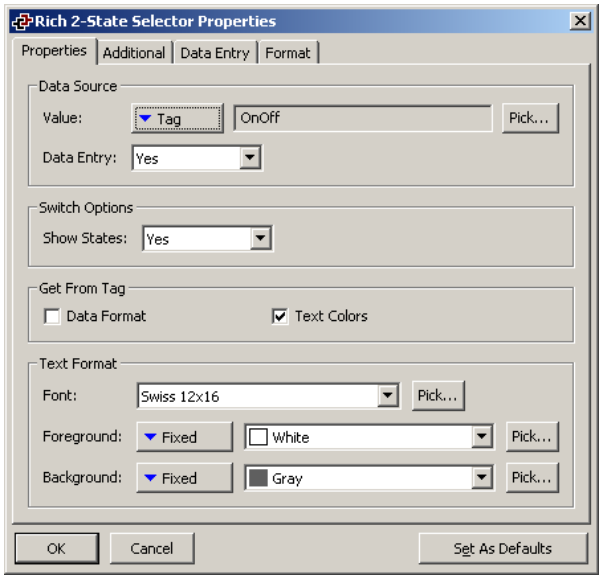
LES PRIMITIVES SELECTEUR

La primitive *Sélecteur 2 états* affiche un interrupteur de type rotatif qui peut être utilisé pour activer ou désactiver une étiquette de type bit. Comme avec toutes les primitives riches, vous pouvez obtenir la majorité des données de configuration à partir de l'étiquette liée.

La primitive *Sélecteur Multi-états* affiche un interrupteur de type rotatif qui peut être utilisé pour activer ou désactiver une étiquette Multi. Comme avec toutes les primitives riches, vous pouvez obtenir la majorité des données de configuration à partir de l'étiquette.

Chacune de ces primitives affiche un interrupteur de sélecteur circulaire dans la zone utilisée pour définir la primitive. Si la primitive est assez grande pour que l'interrupteur circulaire dispose de suffisamment d'espace au-dessus de lui, vous pouvez ajouter des noms à la primitive afin de pouvoir identifier les différents états.

Les deux primitives sont configurées à l'aide de quatre pages avec onglets et le premier d'entre eux se présente comme suit...



- La propriété *Valeur* permet d'indiquer l'emplacement depuis lequel vous pouvez obtenir les données de cette primitive. Vous pouvez sélectionner une étiquette, un registre dans un périphérique de communication ou une expression qui associe plusieurs de ces éléments. Le type de données de l'élément doit être adapté à la primitive en question. Par exemple, la propriété Valeur d'une primitive Sélecteur Multi-états ne peut pas être définie sur égale à une expression de chaîne.

- La propriété *Entrée de données* permet d'indiquer si vous souhaitez ou non que l'utilisateur du HMI virtuel puisse modifier la valeur dépendante de cette primitive. Si vous sélectionnez Local, la saisie de données est activée, mais l'accès n'est pas autorisé via la fonction du HMI virtuel du serveur Web. Pour que la saisie de données soit activée, l'expression saisie pour la propriété de la

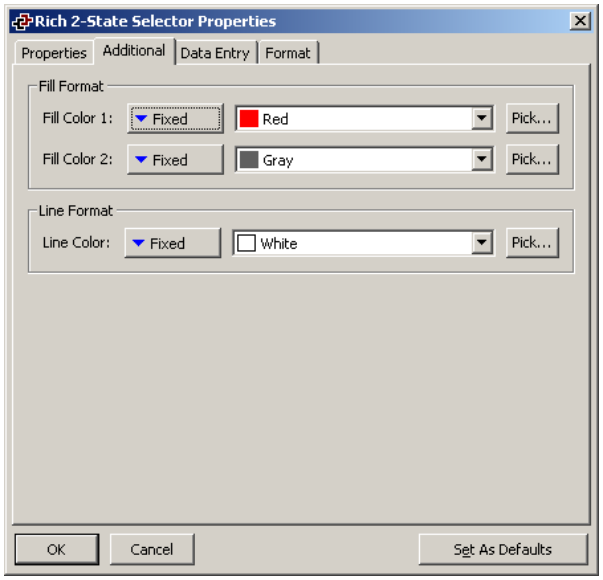
valeur doit pouvoir être modifiée. Par exemple, si une formule est saisie, la saisie de données n'est pas autorisée.

- La propriété *Montrer Etat* permet d'indiquer si vous souhaitez ou non que la primitive tente de nommer chacun des états possibles de l'étiquette. Les états sont uniquement affichés si un espace suffisant existe en haut de la primitive. Il est également important de sélectionner une police assez petite pour éviter que du texte ne se chevauche.

- Les propriétés *Obtenir depuis étiquette* permettent d'indiquer l'emplacement à partir duquel vous pouvez obtenir le format des données et couleurs du texte associées. Les options présentées dépendent de ce que vous avez saisi comme propriété Valeur. Dans chacun des cas, vous pouvez saisir manuellement les données dans les propriétés appropriées ou, en supposant qu'une expression adaptée a été définie, vous pouvez demander à la primitive d'obtenir les informations requises à partir de l'étiquette de données liée.

- Les propriétés *Format Texte* permettent de sélectionner la police et les couleurs de texte utilisées pour nommer les états. Si la primitive est configurée de façon à obtenir ses couleurs de texte à partir de l'étiquette associée, les champs des couleurs sont désactivés.

Le deuxième onglet contient d'autres informations de mise en forme…



- La propriété *Couleur 1* permet de définir la couleur de la partie rectangulaire du sélecteur qui se déplace afin d'indiquer l'état de l'étiquette. La propriété *Couleur 2* permet de définir la couleur du reste de la primitive.

- La propriété *Format Ligne* permet de définir la couleur des différentes lignes utilisées pour dessiner la primitive. Elles comprennent le cercle autour du sélecteur et les autres lignes qui définissent le rectangle dans la primitive.

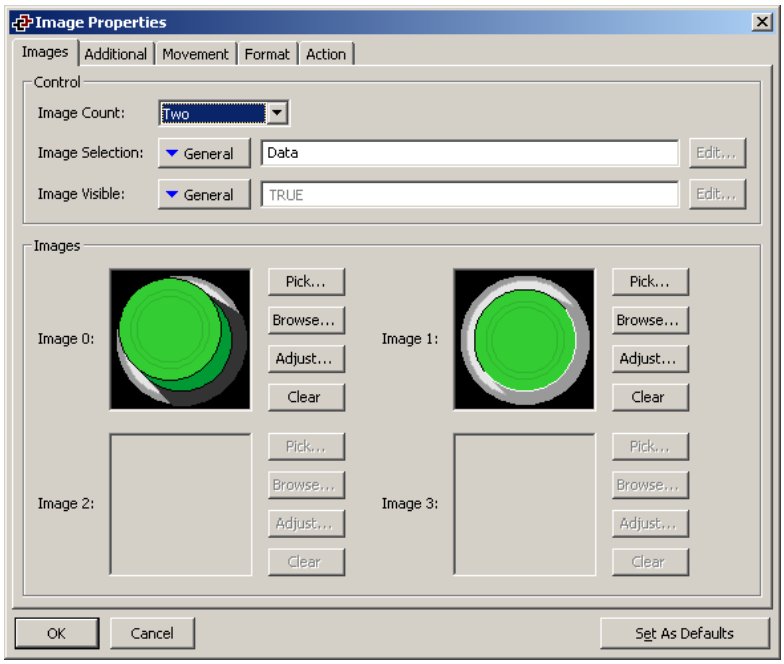
Le quatrième onglet définit plusieurs propriétés qui sont spécifiques à la saisie de données. Elles sont identiques à celles de la primitive Texte étiquette et vous vous reporterez à cette section pour en savoir plus. Le quatrième onglet définit le nom et le format à utiliser pour la primitive et il est identique à celui des étiquettes de type bit ou Multi-états, selon le type de sélecteur qui est configuré. Une fois encore, reportez-vous au chapitre sur les étiquettes pour obtenir des informations sur les différentes options de mise en forme.

LA PRIMITIVE IMAGE

La primitive *Image* permet d'afficher une image parmi plusieurs, en fonction d'une valeur de données facultative. Vous pouvez manipuler les images de différentes façons et les déplacer dans la primitive selon les valeurs de données internes ou externes.

La primitive fournit des fonctions exhaustives d'affichage des bitmaps, images JPEG ou métafichiers à partir de la bibliothèque d'images étendue de Crimson ou de fournisseurs d'images tiers. Cinq pages distinctes avec onglets contrôlent les différentes options.

Le premier onglet permet de sélectionner les images à afficher…



- La propriété *Nombre d'Images* permet d'indiquer le nombre d'images différentes qui doivent être affichées par cette primitive. Vous pouvez afficher jusqu'à dix images différentes.

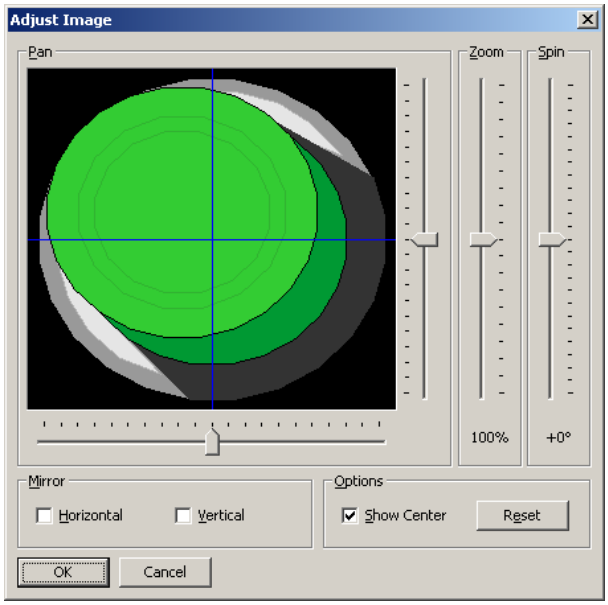
- La propriété *Sélection de l'Image* est une valeur numérique utilisée pour choisir l'une des différentes images si un nombre d'images supérieur à un a été configuré. Une valeur de zéro affiche Image 0, et ainsi de suite.

- La propriété *Image Visible* est une valeur vraie ou fausse qui permet de masquer ou d'afficher l'image sélectionnée. Si vous souhaitez que l'image soit masquée,

vous ne devez pas sélectionner Pas de Remplissage pour l'arrière-plan lorsque vous définissez le format de l'arrière-plan.

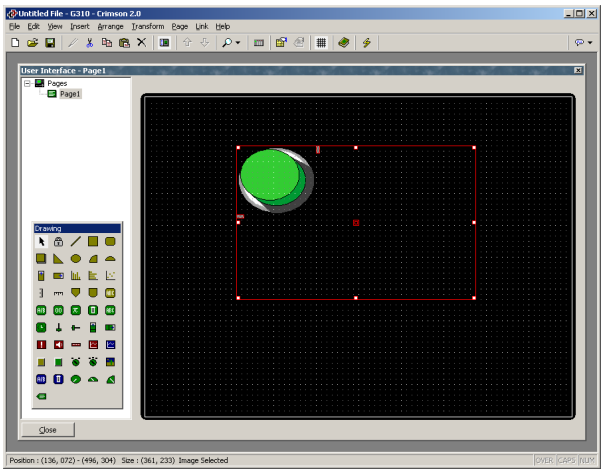
- Les propriétés *Image* définissent chaque image particulière. Vous pouvez utiliser le bouton Choisir situé à coté de chaque image pour sélectionner un symbole depuis la bibliotèque. Le bouton Parcourir vous permet d'ouvrir un fichier contenant une bitmap, un fichier JPEG ou un métafichier Windows. Le bouton Effacer permet de supprimer l'image et le bouton Ajuster de la modifier, comme cette opération est décrite ci-dessous.

Si vous utilisez l'un des boutons Ajuster pour manipuler une image, vous serez d'abord averti des problèmes que vous rencontrerez si vous essayez de télécharger une base de données contenant des images manipulées à l'aide de versions antérieures des systèmes d'exploitation Windows. En supposant que vous écartiez l'utilisation de ces versions antérieures, la boîte de dialogue suivante s'affiche…

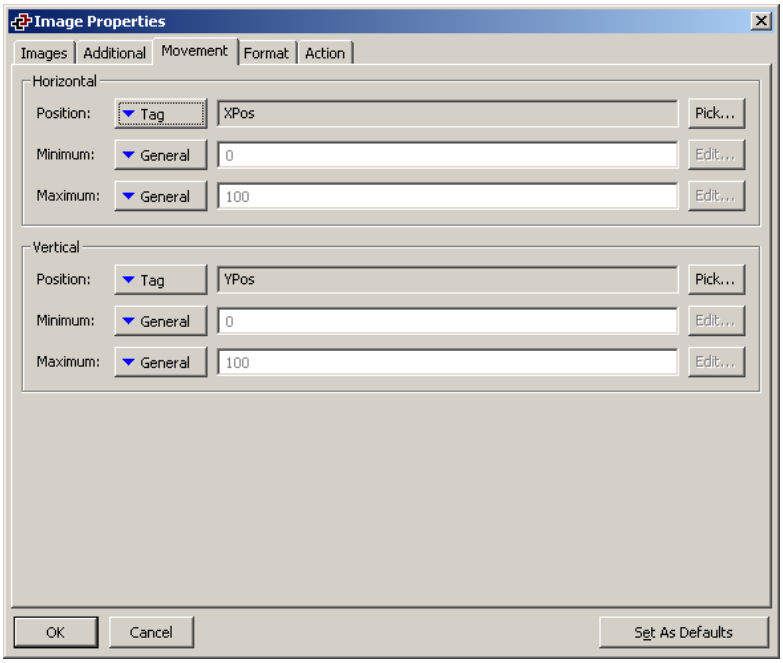


Vous pouvez utiliser les différents glisseurs pour faire un panoramique, zoomer et faire tourner l'image ; les cases à cocher vous permettront de la refléter horizontalement ou verticalement. La case à cocher Options affiche ou masque les lignes bleues qui marquent le centre de l'image alors que le bouton Réinitialiser permet de restaurer l'image dans son état d'origine. Les différentes options de manipulation sont généralement utilisées pour modifier une image afin de créer différents états qui sont utilisés dans une animation.

Le deuxième onglet des propriétés de la primitive Image contient toutes les autres images qui n'ont pas pu être affichées dans le premier onglet. Il est uniquement nécessaire lorsque la propriété Nombre d'Images est définie sur une valeur supérieure à quatre. Le troisième onglet contrôle le mouvement de l'image dans la primitive. Pour activer cette fonction, faites glisser l'une des deux poignées ou les deux poignées rectangulaires ombrées dans le coin supérieur gauche de la primitive pour définir une sous-région dans laquelle l'image doit résider…

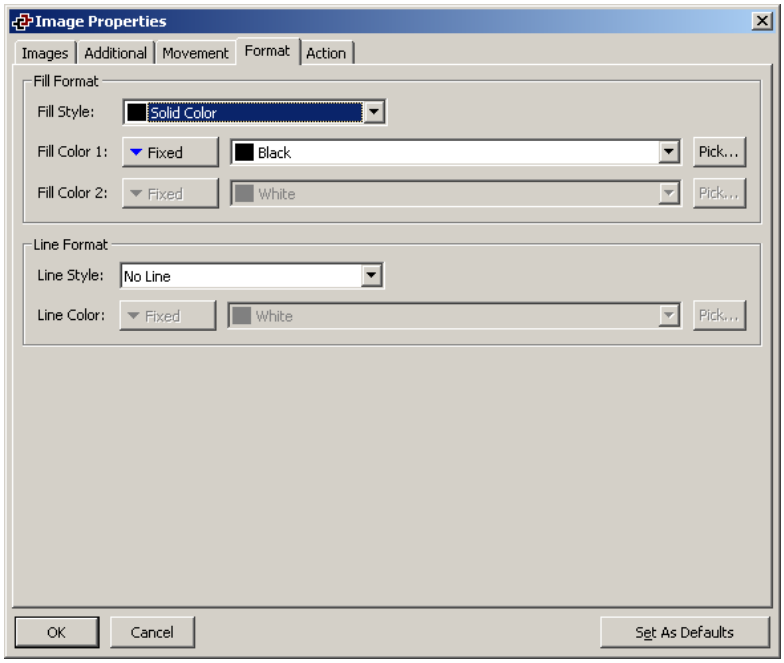


Vous pouvez déplacer l'image maintenant réduite dans toute la primitive en définissant des valeurs pour contrôler sa position horizontale et verticale. Ces valeurs sont définies avec des limites minimales et maximales qui spécifient les valeurs correspondant aux extrêmes du mouvement de l'image dans le rectangle englobant de la primitive…



Dans cet exemple, la définition de **`XPos`** sur 0 place l'image à gauche de la primitive alors que la définition de **`XPos`** sur 100 la place à droite. Vous pouvez obtenir un comportement semblable concernant les limites supérieures et inférieures de la primitive via la valeur enregistrée dans **`YPos`**.

Le quatrième onglet des propriétés de la primitive permet de contrôler la mise en forme de base…



- Les propriétés *Format de Remplissage* permettent de définir le motif l'arrière-plan de la primitive. Notez que si vous souhaitez animer la primitive de quelque façon que ce soit, vous devez spécifier un type de couleur d'arrière-plan pour que le système puisse effacer les anciennes images.
- Les propriétés *Format Ligne* permettent de définir le style de la bordure de la primitive.

Le dernier onglet des propriétés de la primitive permet de définir une action facultative qui peut être déclenchée lorsque l'opérateur touche l'image. Pour en savoir plus sur la configuration de cette fonctionnalité, reportez-vous à la section précédente sur l'attribution des actions aux primitives.

LES PRIMITIVES CADRAN

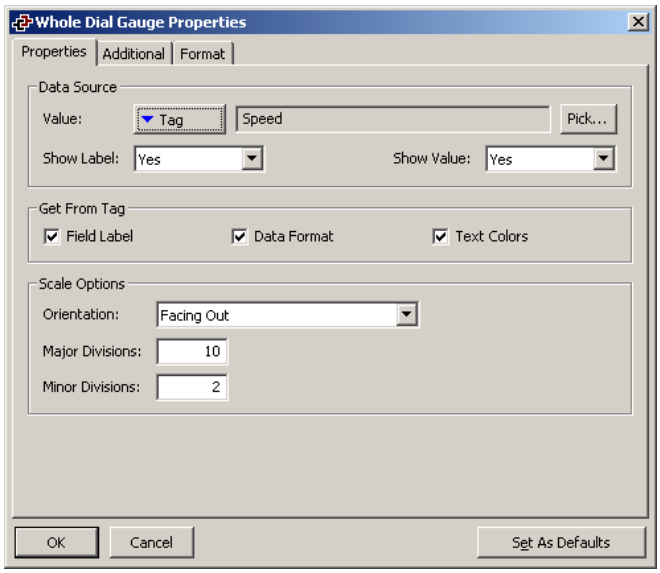
La primitive *Cadran Complet* affiche une valeur entière telle un pointeur avec un arc à 270º dans un cercle complet. La primitive peut afficher facultativement la valeur des données et l'étiquette de données associée. Vous pouvez également définir le nombre de divisions de l'échelle.

La primitive *Demi Cadran* fonctionne comme le cadran complet, mais elle affiche un pointeur avec un arc à 180ºdans un demi-cercle. Toutes les autres options de mise en forme et d'affichage sont identiques.

La primitive *Quart de Cadran* fonctionne comme le cadran complet, mais elle affiche un pointeur avec un arc à 90ºdans un quart de cercle. Toutes les autres options de mise en forme et d'affichage sont identiques.

Tout comme avec les autres primitives riches, les primitives Cadran peuvent obtenir la plupart des informations de mise en forme requises à partir de l'étiquette utilisée comme leur valeur de contrôle. Tout comme avec les primitives Texte étiquette, plusieurs pages avec onglets sont utilisées pour modifier les propriétés des primitives.

Le premier de ces onglets se présente comme suit…

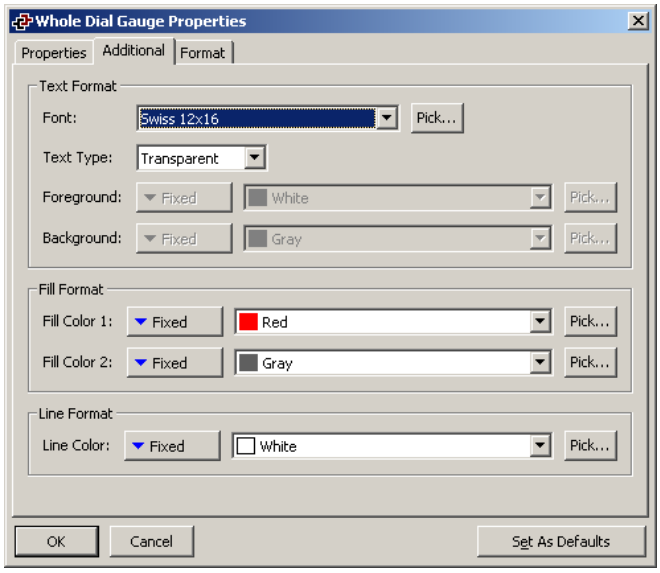


- La propriété *Valeur* permet de définir la valeur à afficher.
- La propriété *Afficher étiquette* permet d'indiquer si un nom doit être ajouté ou non dans le cadran. Ce nom s'affiche au centre de la primitive, au-dessus de la valeur facultative. Si vous utilisez une étiquette pour la propriété de la valeur, vous pouvez obtenir le nom à partir de l'étiquette. Sinon, il doit être entré sous l'onglet Format de la boîte de dialogue.
- La propriété *Afficher donnée* permet d'indiquer si la valeur de la donnée doit être affichée ou non dans le cadran. Si vous utilisez une étiquette du type de donnée approprié pour la propriété de la valeur, vous pouvez obtenir le format à partir de l'étiquette. Sinon, comme pour le nom, il doit être entré sous l'onglet Format.
- Les propriétés *Obtenir depuis étiquette* permettent d'indiquer l'emplacement à partir duquel vous pouvez obtenir le texte de l'étiquette, le format du champ et les couleurs du texte. Les options présentées dépendent de ce que vous avez saisi comme propriété Valeur. Dans chacun des cas, vous pouvez saisir manuellement les données dans les propriétés appropriées ou, en supposant qu'une expression adaptée a été définie, vous pouvez demander à la primitive d'obtenir les informations requises à partir de l'étiquette de données liée.
- La propriété *Orientation* permet d'indiquer la direction vers laquelle les repères secondaires de l'échelle doivent pointer. Les repères principaux pointent toujours vers l'intérieur.

- La propriété *Divisions Principales* permet d'indiquer le nombre de divisions principales dans lequel l'échelle doit être divisée. Pour chaque division, de grands repères sont dessinés. Le nombre minimal de divisions principales est un, auquel cas de grands repères sont dessinés aux extrémités de l'échelle et non pas le long de son arc.

- La propriété *Divisions Secondaires* permet d'indiquer le nombre de divisions secondaires dans lequel chaque division principale doit être divisée. Pour chaque division, des repères plus petits sont dessinés. Lorsque vous sélectionnez une valeur de 1 pour cette propriété, les divisions secondaires sont désactivées.

Le deuxième onglet définit les différentes options de mise en forme…



- Les propriétés *Format Texte* permettent de définir la couleur utilisée pour afficher la valeur et l'étiquette de données facultative ainsi que les couleurs qui sont utilisées pour le texte. De plus, une propriété est fournie pour définir si la police soit être opaque ou si le pointeur doit être visible dans le texte.

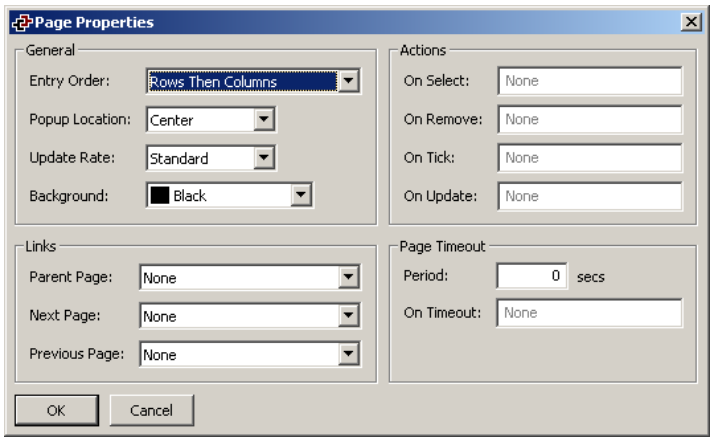
- Les propriétés *Format de Remplissage* permettent de définir la couleur de l'arrière-plan du cadran et la couleur à utiliser pour dessiner l'intérieur du pointeur et les repères de l'échelle. Couleur 1 définit l'arrière-plan et Couleur 2 les autres éléments.

- La propriété *Format Ligne* permet de définir la couleur des lignes utilisées pour démarquer le pointeur, les repères de l'échelle et la bordure du cadran lui-même.

Le troisième onglet permet de spécifier le nom facultatif, les valeurs minimales et maximales et le format des données des valeurs de données facultatives. Cet onglet fonctionne comme nous l'avons déjà décrit pour les étiquettes de données d'entiers. Reportez-vous donc à cette section pour en savoir plus.

DEFINITION DES PROPRIETES DE LA PAGE

Chaque page dispose de plusieurs propriétés auxquelles vous pouvez accéder via le menu Page…



- La propriété *Ordre saisie* permet de définir la façon dont le curseur se déplace entre les champs d'entrée de données sur les pages du HMI. Le paramètre définit si les champs organisés sous forme de grille sont saisis en ligne ou en colonne.

- La propriété *Emplacement du Popup* permet de définir l'emplacement de la fenêtre contextuelle ou du pavé numérique contextuel lorsque la page d'affichage est visible. Vous voudrez peut-être ajuster cette propriété pour que les popups ne cache pas des éléments de données importants.

- La propriété *Rafraîchissement* permet de définir la fréquence de mise à jour des éléments à l'écran. Lorsque la fréquence des rafraîchissements augmente, les performances de communications globales du HMI virtuel peuvent diminuer. Vous pouvez laisser cette sélection sur son paramètre par défaut lorsque c'est possible.

- La propriété *Arrière-plan* permet de définir la couleur d'arrière-plan de la page d'affichage. Notez que l'arrière-plan ne peut pas être animé car un changement de couleur vous forcerait à redessiner toute la page, ce qui réduirait les performances.

- Les propriétés *Sur apparition page* et *Sur changement page* permettent de définir les mesures à prendre lorsque la page est d'abord sélectionnée pour l'affichage ou lorsque la page est supprimée de l'écran. Reportez-vous aux sections Ecriture d'actions et Référence des fonctions pour obtenir la liste des actions prises en charge. Reportez-vous à la section Disponibilité des données de ce chapitre pour obtenir les détails sur un délai d'attente qui peut se produire lors de l'utilisation de ces propriétés.

- La propriété *A chaque Sec* permet de définir une action qui s'exécute chaque seconde lorsque la page est affichée. Reportez-vous aux sections Ecriture d'actions et Référence des fonctions pour obtenir la liste des actions prises en

charge. Si une absence de disponibilité des données entraîne l'impossibilité d'exécuter cette action, elle est sautée, puis retentée une seconde plus tard.

- La propriété *Sur Rafraîchissement* permet de définir une action qui s'exécute chaque fois qu'une page est redessinée. Reportez-vous aux sections Ecriture d'actions et Référence des fonctions pour obtenir la liste des actions prises en charge. Si une absence de disponibilité des données entraîne l'impossibilité d'exécuter cette action, elle est sautée, puis retentée lors de la mise à jour suivante. Vous devez noter que vous pouvez sérieusement réduire les performances système en effectuant des actions complexes sur chaque mise à jour de l'affichage. Notez également que souvent, les actions qui, selon vous, doivent être exécutées sur chaque mise à jour peuvent être effectuées à l'aide de déclencheurs ou de blocs de mapping.

- La propriété *Page parente* permet d'indiquer la page à afficher lorsque vous appuyez sur la touche **`Quitter`** du HMI alors que la page est active. Vous pouvez remplacer la sélection de cette page à l'aide des techniques décrites ci-dessous.

- La propriété *Page suivante* permet d'indiquer la page à afficher lorsque vous appuyez sur la touche **`SUIV`** du HMI alors que cette page est active et lorsque le curseur se trouve sur la dernière entrée de données de la page. Vous pouvez également remplacer cette sélection.

- La propriété *Page précédente* permet d'indiquer la page à afficher lorsque vous appuyez sur la touche **`PREC`** du HMI alors que cette page est active et lorsque le curseur se trouve sur la première entrée de données de la page. Vous pouvez également remplacer cette sélection.

- La propriété *Au bout de* permet de définir le temps d'attente en secondes sans qu'aucune interaction d'utilisateur ne se produise avant d'effectuer l'action spécifiée dans la propriété *Action sur Timeout*. Ces propriétés sont en général utilisées pour supprimer un popup ou pour revenir à un certain type d'écran de menu après plusieurs secondes d'inactivité.

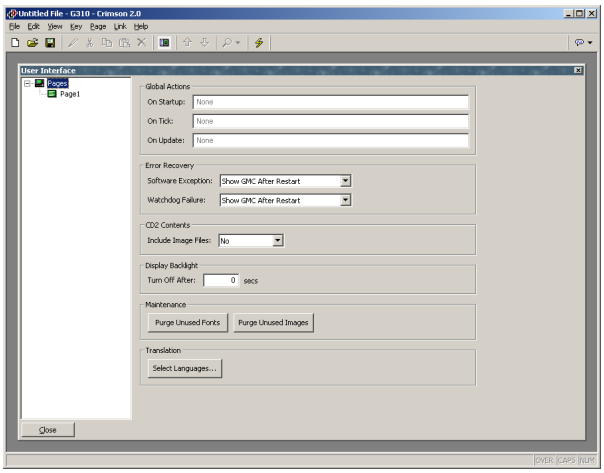
Si une seule page ne peut pas contenir tous les champs d'entrée de données, vous pouvez utiliser les propriétés Page suivante et Page précédente pour lier une série de pages afin que l'opérateur puisse modifier les champs dans la séquence. Crimson place automatiquement le curseur de façon appropriée. Ainsi, si vous appuyez sur la touche **`PREC`** dans le premier champ d'une page, la page précédente est activée avec le curseur sur le dernier champ de cette page.

DEFINITION DES ACTIONS DU SYSTEME

Outre les différentes actions que vous pouvez définir via les propriétés de la page, Crimson vous permet de définir une action à exécuter lors du premier démarrage du système et une action à exécuter une fois par seconde ou lors des mises à jour de page, quelle que soit la page qui est affichée. Vous pouvez accéder à ces actions en sélectionnant l'icône *Pages* dans le volet gauche de la fenêtre Interface utilisateur. Reportez-vous à l'avertissement ci-dessus concernant l'utilisation de la propriété *Sur Rafraîchissement*.

AUTRES PROPRIETES DU SYSTEME

Outre les actions du système décrites ci-dessus, la page des propriétés, à laquelle vous pouvez accéder en sélectionnant l'icône Pages, vous permet d'accéder à plusieurs autres paramètres de tout le système…



- Les propriétés *Démarrage Après Erreurs* permettent de définir le comportement du système lorsqu'il rencontre un problème logiciel ou lorsque la mise à jour de l'affichage échoue sur une longue période, suite à une erreur de codage. Par défaut, le système réinitialise et affiche ce qu'on appelle un code « Guru Meditation Code» qui permet aux ingénieurs de développement de localiser le problème. Vous pouvez désactiver l'affichage de ce code pour permettre au système de récupérer plus rapidement et sans aucune intervention de l'utilisateur.

- La propriété *Inclure les Images* permet d'indiquer que Crimson doit enregistrer dans le fichier de la base de données une copie de toutes les images basées sur le disque qui ont été importées dans le projet. Par défaut, le nom de fichier seul de l'image est enregistré, ce qui nécessite que les images soient disponibles sur le disque dès que vous travaillez sur le projet. Si vous souhaitez créer un seul fichier contenant toutes les données requises pour le projet, activez cette option. Notez que les bases de données qui contiennent les fichiers image sont généralement très volumineuses et peuvent empêcher le fonctionnement de la prise en charge de l'extraction.

- La propriété *Désactiver le Rétro-éclairage* permet de configurer le système pour qu'il désactive le rétro-éclairage de l'affichage une fois que le nombre de

secondes d'inactivité a été spécifié. Cette fonction permet d'étendre la durée de vie des composants du rétro-éclairage. L'opérateur peut réactiver le rétro-éclairage en appuyant sur une touche ou en touchant une zone active de l'écran tactile. Si vous appuyez sur une touche ou si vous touchez l'écran tactile lorsque le rétro-éclairage est allumé, cette action est reportée.

- Les boutons *Maintenance* permettent de supprimer les polices ou les images inutilisées dans la base de données, ce qui réduit la taille des fichiers et l'utilisation de la mémoire. Vous devez en règle générale utiliser ces options avant que la base de données soit utilisée car elles suppriment la plupart des détritus qui s'accumulent lors du développement.
- Le bouton *Sélectionner une Langue* est décrit ci-dessous.

SELECTION DE LANGUES

Pour sélectionner les différentes langues qui sont prises en charge dans votre base de données, sélectionnez l'icône Pages dans la fenêtre Interface utilisateur, puis appuyez sur le bouton Sélectionner Langue afin d'afficher la boîte de dialogue suivante…

Vous pouvez définir jusqu'à huit langues différentes que vous pouvez choisir à partir d'une liste déroulante. Crimson configure à nouveau Windows pour qu'il utilise l'éditeur de méthode d'entrée lorsqu'une langue complexe (par exemple, une langue basée sur l'Unicode) est modifiée. Pour utiliser cette fonction, vous devez vous assurer que la prise en charge linguistique requise est installée en vous reportant à la documentation Windows appropriée. Si vous souhaitez utiliser une langue basée sur l'Unicode qui n'est pas comprise dans la liste déroulante, sélectionnez le mode Unicode générique. Vous pouvez alors saisir tous les caractères Unicode même si vous devez manuellement sélectionner le mode clavier approprié ou l'éditeur de méthode d'entrée.

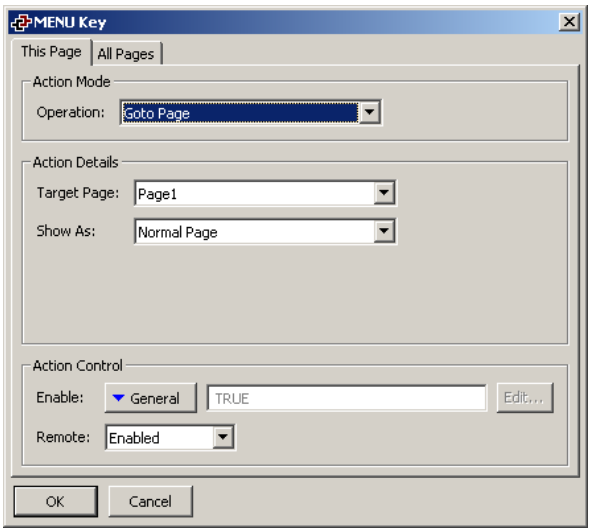
MODIFICATION DE LA LANGUE

Pour configurer une touche ou une primitive pour qu'elle modifie la langue affichée par le HMI virtuel, sélectionnez dans l'onglet Option le mode Définie par l'utilisateur et entrez **`SetLanguage(n)`** comme propriété Sur Appui (où **`n`** est un chiffre entre 1 et 8) en fonction de la langue à afficher. La page d'affichage est recréée dans la langue sélectionnée et n'importe

quel texte pour lequel des traductions ont été entrées (notamment les informations de texte fixe, de nom d'étiquette et de mise en forme d'étiquette) est ajusté comme requis. Les pages qui sont affichées sont également présentées dans la langue sélectionnée.

DEFINITION DU COMPORTEMENT DE LA TOUCHE

Outre la définition des actions qui se produisent lorsque les primitives sont touchées, vous pouvez définir des actions qui sont exécutées lorsque vous appuyez sur les touchesdu coté de l'écran. Pour cela, modifiez d'abord le niveau de zoom pour pouvoir afficher la touche requise. Double-cliquez ensuite sur la touche pour afficher la boîte de dialogue suivante…



Notez que cette boîte de dialogue est semblable à celle que nous avons précédemment vue sur les primitives, mais qu'elle possède deux pages avec onglets. Le premier onglet permet de définir ce qui se produit lorsque vous appuyez sur la touche en question lorsque la page en cours est sélectionnée. Le deuxième onglet permet de définir ce qui se produit lorsque vous appuyez sur la touche lorsque n'importe quelle page est sélectionnée. Le premier type d'action est appelé une action *locale* alors que le deuxième type est appelé une action *globale*.

La couleur utilisée pour afficher la touche est modifiée selon les actions qui sont définies…

Si la touche est affichée en VIOLET, une action LOCAL est définie pour cette PAGE.

Si la touche est affichée en VERT, une action GLOBALE est définie.

Si la touche est affichée en BLEU, l'action globale et l'action locale sont définies.

Une fois que vous avez sélectionné une action, vous pouvez cliquer avec le bouton droit sur la touche pour utiliser le menu afin de sélectionner Rendre global ou Rendre local pour modifier le type d'action. Ces options sont indisponibles si les deux types d'action ont déjà été définis.

BLOCAGE DES ACTIONS PAR DEFAUT

Lorsque vous définissez des actions de touches, vous pouvez utiliser la sélection Bloquer action par défaut comme espace réservé pour empêcher d'autres traitements. Supposez par exemple que vous avez configuré la touche **F1** pour qu'elle effectue une action globale, mais vous souhaitez empêcher que cette action soit appelée sur une page spécifique. En configurant la touche **F1** sur cette page comme Bloquer action par défaut, l'action globale ne se produit pas.

DISPONIBILITE DES DONNEES

L'infrastructure de communication de Crimson effectue une lecture seule des éléments de données qui sont nécessaires à la page en cours. C'est-à-dire que lorsque vous sélectionnez une page pour la première fois, certains éléments de données peuvent ne pas être disponibles. Ce n'est pas un problème pour une primitive d'affichage car cette dernière affiche uniquement un état indéfini (généralement des tirets) jusqu'à ce sur les données soient disponibles. En revanche, les choses peuvent s'avérer plus complexes pour les actions.

Par exemple, supposez qu'une action locale augmente la vitesse d'un moteur de 50 tours/minute. Si la vitesse du moteur n'est pas enregistrée dans la page affichée précédemment, lorsque la page est affichée pour la première fois, Crimson ne connaît pas la vitesse actuelle et n'est donc pas en mesure d'écrire la nouvelle valeur. Pour la gérer, l'opérateur tente d'effectuer une action pour laquelle les données requises ne sont disponibles, le module maître n'affiche le message « NOT READY » jusqu'à ce que la touche soit relâchée. L'opérateur doit alors attendre quelques minutes, puis retenter l'opération. En pratique, les mises à jour des communications se produisent normalement assez rapidement de sorte que même l'opérateur le plus agile avec ses doigts aura du mal à obtenir que le message s'affiche. Mais comme il peut s'afficher, il est utile de l'expliquer.

Un autre problème légèrement plus complexe se produit si l'action définie par une propriété Sur apparition page d'une page ne peut pas être poursuivie car elle trouve également que les données requises ne sont pas disponibles. A ce niveau, Crimson patiente trente secondes avant que les données n'arrivent. Si elles n'arrivent pas, l'action n'est pas effectuée et un message « TIMEOUT » s'affiche pour l'opérateur. Ce mécanisme d'attente est requis pour éviter des problèmes si un lien de communication est coupé.

REMARQUES POUR LES UTILISATEURS D'EDICT

Les utilisateurs du logiciel Edict-97 de Red Lion doivent noter ce qui suit…

- Les pages ne contiennent plus de couches graphiques ou de texte car toutes les primitives sont graphiques par nature. Cela signifie que le concept d'un format de page est tout aussi redondant.
- Les catégories de pages ont été remplacées par des primitives système. Par exemple, lorsque Edict utilise une page complète pour son afficheur d'alarme, la primitive système correspondante peut être utilisée pour attribuer un espace aussi petit ou grand que nécessaire.

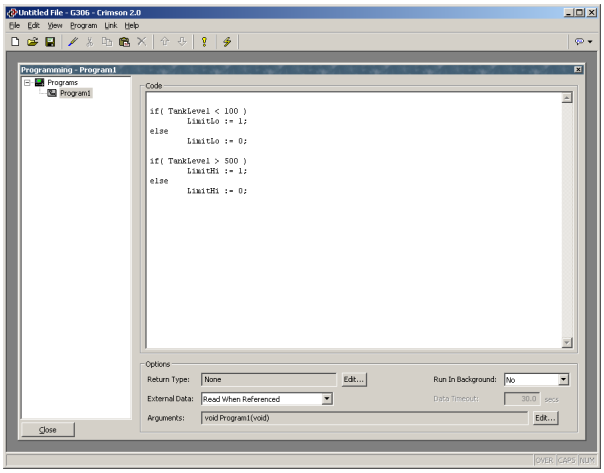
- Les actions définies en double-cliquant sur une touche remplacent les tables d'événements globales et locales. Si votre application a utilisé plus d'une ligne par événement, vous devrez sans doute utiliser un programme pour implémenter la logique requise.

- Les événements comme la mise à jour des communications terminée et le repère d'une seconde ont été supprimés car la majorité des actions effectuées par de tels événements peuvent désormais être gérées via d'autres mécanismes. Par exemple, comms update complet a souvent été utilisée pour déplacer les données entre les périphériques. Elle peut maintenant être effectuée à l'aide de la fonctionnalité de conversion du protocole de la fenêtre Communications. De plus, ces événements étaient souvent utilisés de manière incorrecte et entraînent la création de trop de bases de données complexes.

- Alors qu'Edict pourrait généralement gérer entre deux et cinq mises à jour d'affichage par seconde, Crimson est conçu pour redessiner l'affichage toutes les 100 msec, fournissant ainsi, par exemple, un retour plus régulier à l'opérateur lors de la saisie de données.

CONFIGURATION DE PROGRAMMES

Les sections précédentes de ce manuel décrivent l'utilisation des actions pour effectuer toutes sortes d'opérations une fois que vous avez appuyé sur des touches ou modifié des étiquettes de données. Si vous devez effectuer une action qui est trop complexe pour s'adapter sur une ligne unique ou qui exige une logique de prise de décision plus complexe, vous pouvez utiliser l'icône Programmation de l'écran principal pour créer et gérer des programmes. Notez que de nombreuses applications ne nécessitent aucun programme. Si vous le souhaitez, vous pouvez donc choisir de passer ce chapitre.



UTILISATION DE LA LISTE DES PROGRAMMES

Pour créer, renommer ou supprimer des programmes, cliquez sur le volet gauche de la fenêtre. Les différentes commandes du menu Programme peuvent être utilisées pour apporter les modifications souhaitées. Une autre solution consiste à cliquer avec le bouton droit sur le programme requis, puis à effectuer votre choix à partir du menu.

Pour sélectionner un programme, cliquez sur le nom dans la liste ou utilisez les flèches haut et bas de la barre d'outils. Une autre solution consiste à utiliser les associations de touches **`Alt+Gauche`** et **`Alt+Droite`** pour vous déplacer vers le haut ou le bas de la liste. Ces touches fonctionnent quel que soit le volet qui est sélectionné.

MODIFICATION DE PROGRAMMES

Pour modifier un programme, il vous suffit de modifier le texte du programme à l'aide de la grande zone située dans le volet droit de la fenêtre Programmation. Une fois terminé, appuyez sur l'association de touches **`Ctrl+T`** ou sélectionnez la commande Compiler dans le menu Programme. Cette opération lit le programme et vérifie la présence d'éventuelles erreurs. Si une erreur est détectée, une boîte de dialogue s'affiche et le curseur est déplacé sur l'emplacement approximatif de l'erreur. Si aucune erreur n'est détectée, une boîte de dialogue s'affiche pour confirmer ce fait et le programme est compilé dans le format interne de Crimson en vue d'une exécution ultérieure par le maître.

PROPRIETES DU PROGRAMME

Les différents champs situés dans la partie inférieure droite du volet permettent de modifier les propriétés du programme…

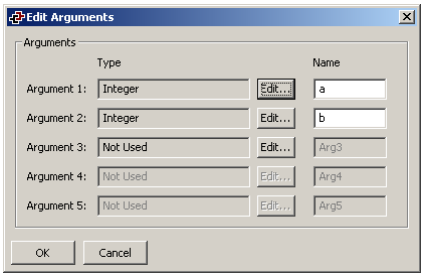
- La propriété *Données Retournées* permet d'indiquer si ce programme doit uniquement effectuer une série d'actions ou s'il doit effectuer un calcul, puis renvoyer la valeur de ce calcul à l'utilisateur. Les programmes qui renvoient des valeurs sont décrits plus en détail ci-dessous.

- La propriété *En Arrière-Plan* permet d'indiquer si Crimson doit attendre que le programme se termine avant de continuer le traitement de n'importe quelle tâche appelée qui a appelé le programme. Par exemple, si cette propriété est définie sur Non, si vous exécutez un programme après avoir appuyé sur une touche, une pause se produit dans les mises à jour de l'affichage jusqu'à ce que le programme se termine. (Comme la majorité des programmes mettent très peu de temps à s'exécuter, vous ne le noterez peut-être même pas.) Si cette propriété est définie sur Oui, les mises à jour de l'affichage continuent immédiatement et le programme s'exécute en arrière-plan, avec une priorité inférieure. Seul un programme en arrière-plan s'exécute à la fois. Par conséquent, les requêtes suivantes sont mises en file d'attente en vue d'une exécution ultérieure. Notez également que les programmes qui renvoient des valeurs ne peuvent pas être exécutés en arrière-plan car leur valeur de retour ne serait alors pas disponible pour que l'appelant puisse l'utiliser !

- Les propriétés *Données externes* et *Timeout* permettent de contrôler l'interaction du programme avec l'infrastructure de communication de Crimson en ce qui concerne les éléments de données externes auxquels le programme fait référence. Rappelez-vous que Crimson lit uniquement les données lorsqu'elles sont utilisées. Cette propriété permet de contrôler l'interprétation exacte de cette règle en ce qui concerne les programmes…

MODE	COMPORTEMENT
Les lire lorsqu'on y fait référence	Les données externes utilisées par le programme sont ajoutées à l'analyse des communications dès que le programme est référencé. Si le programme est référencé par une page d'affichage, les données sont lues lorsque cette page s'affiche. Si le programme est référencé par une action globale ou un déclencheur, les données sont toujours lues. C'est là le mode par défaut, qui est acceptable pour tous les programmes sauf ceux qui utilisent de grandes quantités de données externes.
Toujours les lire	Les données externes utilisées par le programme sont toujours lues, que le programme soit référencé ou non. Ainsi, le programme est toujours prêt à s'exécuter et l'opérateur ne voit pas le message « NOT READY » qui peut s'afficher lorsque le programme est référencé pour la première fois. L'inconvénient de ce mode réside dans le fait que les performances de communications peuvent être réduites si de grandes quantités de données sont référencées par le programme.
Lire avant exécution programme	Les données externes utilisées par le programme sont uniquement lues lorsque le programme est appelé. Le programme attend que la période définie dans la propriété Timeout de ces données soit disponible. Si les données ne peuvent être lues (peut-être parce qu'un périphérique est hors ligne), le programme ne s'exécute pas. Ce mode est généralement utilisé avec les programmes référencés de façon globale, qui consomment de grande quantités de données qui ralentiraient par ailleurs l'analyse des communications.
Lire Avant et Exec Prog tout de même	Les données externes sont gérées comme pour le mode Toujours les lire, mais le programme s'exécute si les données ont été correctement lues ou non. L'opérateur ne voit donc jamais le message « NOT READY », mais si un périphérique est hors ligne, il n'existe aucune garantie que les éléments de données du programme contiennent des données valides.

- La propriété *Arguments* permet de spécifier jusqu'à cinq arguments qui peuvent être passé dans le programme. Chaque argument possède un nom et un type de données, comme l'indique la boîte de dialogue qui s'affiche lorsque vous appuyez sur le bouton Editer...



- Le passage des arguments aux programmes est décrit plus en détail ci-dessous.

AJOUT DE COMMENTAIRES

Vous pouvez ajouter des commentaires de deux façons à vos programmes. Tout d'abord, vous pouvez utiliser la séquence **//** pour présenter un commentaire qui se poursuit dans le reste de la ligne en cours. Ensuite, vous pouvez utiliser la séquence **/*** pour présenter un commentaire d'une ou plusieurs lignes. Ce commentaire se poursuit jusqu'à ce que la séquence ***/** apparaisse. L'exemple ci-dessous montre les deux styles de commentaires…

```
// Ceci est un commentaire sur une ligne

/* Ceci est la ligne 1 du commentaire
   Ceci est la ligne 2 du commentaire
   Ceci est la ligne 3 du commentaire */
```

Vous pouvez également placer un commentaire sur une seule ligne à la fin d'une ligne contenant du code.

RETOUR DE VALEURS

Comme nous venons de l'expliquer ci-dessus, les, programmes peuvent retourner des valeurs. Ce type de programmes peuvent être appelés par d'autres programmes ou par des expressions, quel que soit l'endroit dans la base de données. Par exemple, si vous souhaitez effectuer un décodage complexe particulier sur plusieurs conditions relatives à un moteur, puis retourner une valeur afin d'indiquer l'état actuel, vous devez créer un programme qui renvoie un entier tel que celui-ci…

```
if( MotorRunning )
       return 1;
else {
       if( MotorTooHot )
              return 2;
       if( MotorTooCold )
              return 3;
       return 0;
       }
```

Vous devez ensuite configurer une formule multi-états pour appeler ce programme, puis utiliser l'onglet Format de cette étiquette pour définir les noms des différents états. L'appel est effectué en définissant la propriété Valeur de l'étiquette sur **Name()**, où **Name** représente le nom du programme en question. Les parenthèses permettent d'indiquer un appel de fonction et ne doivent pas être omises.

ATTENTION !

Notez que vous devez être très attentif lorsque vous utilisez des programmes pour renvoyer des valeurs. En particulier, vous devez éviter de diffuser en boucle sur de trop longues périodes ou d'effectuer des actions illogiques dans le contexte dans lequel la fonction est appelée. Par exemple, si le fragment de code ci-dessus a appelé la fonction **GotoPage** pour modifier la page, l'affichage est modifié chaque fois que le programme est appelé. Imaginez

ce qui pourrait vous arriver si vous avez par exemple essayé d'enregistrer des données à partir de l'étiquette associée et que vous vous êtes rendu compte que cela n'est pas une bonne idée ! Voilà pourquoi vous devez conserver les programmes qui renvoient des valeurs simples et toujours prendre en compte le contexte dans lequel ils sont exécutés. En cas de doute, évitez de faire autre chose que des mathématiques simples et des instructions **`if`**.

PASSAGE D'ARGUMENTS

Comme nous l'avons également expliqué ci-dessus, le programme peut accepter les arguments. Par exemple, supposez que vous souhaitiez écrire un programme appelé **`FindMean`** pour prendre la moyenne des deux valeurs. Vous pouvez configurer le programme pour qu'il accepte deux arguments d'entiers, **`a`** et **`b`**, comme l'exemple fourni le montre lors de la définition de l'objectif de la propriété *Arguments*. Vous pouvez également configurer le programme pour qu'il renvoie une valeur entière. Le code du programme serait alors défini comme…

```
return (a+b)/2;
```

Une fois que vous avez créé et compilé ce programme, vous pouvez saisir une expression telle que **`FindMean(Tag1, Tag2)`** pour l'appeler avec les arguments appropriés. Dans ce cas, l'expression est égale à la moyenne de **`Tag1`** et **`Tag2`**.

CONSEILS DE PROGRAMMATION

Les sections ci-dessous fournissent une vue d'ensemble des constructions de programmation que Crimson prend en charge. La syntaxe de base utilisée est celle du langage de programmation C. Notez que l'objectif consiste à essayer de vous apprendre à devenir un programmeur ou à maîtriser les subtilités du langage C. Ces sujets dépassent le périmètre du présent manuel. Nous tenterons plutôt de vous fournir une rapide vue d'ensemble des fonctions disponibles afin que l'utilisateur intéressé puisse approfondir son expérience.

PLUSIEURS ACTIONS

Le type de programme le plus simple comprend une liste d'actions et chaque action occupe une ligne et est suivie d'un point-virgule. Toutes les différentes actions définies dans la section Ecriture d'actions sont disponibles pour être utilisées. Ces programmes simples sont en règle générale utilisés lorsque l'association des actions en une seule définition d'actions s'avère illisible.

AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

L'exemple ci-dessous définit plusieurs variables, puis modifie la page d'affichage…

```
Motor1 := 0;
Motor2 := 1;
Motor3 := 0;

GotoPage(Page1);
```

Les actions sont exécutées dans l'ordre et le programme revient vers l'appelant.

INSTRUCTIONS IF

Ce type d'instruction est utilisé dans un programme afin de prendre une décision. La construction se compose d'une instruction **`si`** avec une condition entre parenthèses, suivie d'une action (ou de plusieurs actions) qui est exécutée si la condition est vraie. Si plusieurs actions sont spécifiées, vous devez placer chacune d'entre elles sur une ligne distincte et utiliser des accolades pour regrouper les instructions. Vous pouvez utiliser une clause **`else`** facultative pour exécuter du code si la condition est fausse.

L'exemple ci-dessous montre une instruction **`if`** avec une action unique…

```
if( TankFull )
       StartPump := 1;
```

L'exemple ci-dessous montre une instruction **`if`** avec deux actions…

```
if( TankEmpty ) {
       StartPump := 0;
       OpenValue := 1;
       }
```

L'exemple ci-dessous montre une instruction **`if`** avec une clause **`else`**…

```
if( MotorHot )
       StartFan := 1;
else
       StartFan := 0;
```

Notez qu'il est très important de ne pas oublier de placer les accolades autour des groupes d'actions à exécuter dans la partie **`if`** ou **`else`** de l'instruction. Si vous omettez les accolades, Crimson interprétera sans doute mal les actions exactes dont vous souhaitez dépendre lors de la condition **`if`**. Même si nous vous conseillons d'ajouter des sauts de ligne entre les actions, ils ne servent pas à comprendre ce qui est inclus ou pas dans l'instruction conditionnelle.

INSTRUCTIONS SWITCH

Une instruction **`switch`** permet de comparer une valeur entière à un nombre de constantes possibles et d'effectuer une action qui repose sur la valeur correspondante. La syntaxe exacte

prend en charge plusieurs options au-delà de celles qui sont présentées dans l'exemple ci-dessous, mais pour la majorité des applications, cette forme simple est acceptable.

Cet exemple démarre un moteur sélectionné par la valeur dans l'étiquette **`MotorIndex`**...

```
switch( MotorIndex ) {

       case 1:
              MotorA := 1;
              break;
       case 2:
       case 3:
              MotorB := 1;
              break;
       case 4:
              MotorC := 1;
              break;
       default:
              MotorD := 1;
              break;
       }
```

Une valeur de 1 démarre le moteur A, une valeur de 2 ou 3 démarre le moteur B et une valeur de 4 démarre le moteur C. Toute valeur qui n'est pas explicitement répertoriée démarre le moteur D. Concernant la syntaxe, vous devez noter les choses suivantes : l'utilisation des accolades autour des instructions **`case`**, l'utilisation de **`break`** pour conclure chaque bloc conditionnel, l'utilisation de deux instructions séquentielles **`case`** pour correspondre à plusieurs valeurs et l'utilisation de l'instruction facultative **`default`** pour indiquer une action à effectuer si la valeur ne correspond à aucune des valeurs spécifiées dans l'expression de contrôle. (Si cette syntaxe semble trop intimidante, vous pouvez utiliser une série d'instructions **`if`** à sa place pour obtenir les mêmes résultats, mais avec des performances et une lisibilité relativement inférieures.)

VARIABLES LOCALES

Certains programmes utilisent des variables pour stocker les résultats intermédiaires ou pour contrôler l'une des différentes constructions de boucles décrites ci-dessous. Plutôt que de définir une étiquette pour qu'elle contienne ces valeurs, vous pouvez déclarer ce qu'on appelle des variables locales à l'aide de la syntaxe décrite ci-dessous…

```
int     a;          // Declare local integer 'a'
float   b;          // Declare local real    'b'
cstring c;          // Declare local string  'c'
```

Vous pouvez initialiser les variables locales lorsqu'elles sont déclarées en faisant suivre le nom de la variable de **`:=`** et de la valeur à attribuer. Les variables qui ne sont pas initialisées de cette façon sont définies sur zéro ou sur une chaîne vide, le cas échéant.

Notez que les variables locales sont réellement locales dans leur portée et leur durée de vie. C'est-à-dire qu'elles ne peuvent pas être référencées en dehors du programme et qu'elles ne conservent pas leur valeur entre les appels de fonctions. Si une fonction est appelée de façon récursive, chaque appel possède ses propres variables.

CONSTRUCTIONS DE BOUCLES

Vous pouvez utiliser les trois différentes constructions de boucles pour réaliser une section de code donnée alors qu'une certaine condition est vraie. La boucle **`while`** teste sa condition avant l'exécution du code alors que la boucle **`do`** teste la condition après. La boucle **`for`** est un moyen plus rapide de définir une boucle **`while`** en vous permettant de combiner trois éléments communs en une instruction.

Notez que vous devez être attentive lorsque vous utilisez des boucles dans vos programmes car vous pouvez faire des erreurs de programmation et une boucle ne se terminerait ainsi jamais. Selon la situation dans laquelle le programme est appelé, cela peut sérieusement perturber l'activité de l'interface utilisateur du maître ou de ses communications. Les boucles qui se répètent de trop nombreuses fois peuvent également provoquer des problèmes de performances au sous-système qui les a appelés.

LA BOUCLE WHILE

Ce type de boucle répète l'action qui la suit alors que la condition de l'instruction **`while`** reste vraie. Si la condition n'est jamais vraie, l'action n'est jamais exécutée et la boucle n'effectue aucune opération au-delà de l'évaluation de la condition de contrôle. Si vous souhaitez que la boucle comprenne plusieurs actions, assurez-vous que vous avez entouré les instructions d'accolades, comme avec l'instruction **`if`**. L'exemple ci-dessous initialise une paire de variables locales, puis utilise la première variable pour diffuser en boule dans le contenu d'un tableau, totalisant les dix premiers éléments et renvoyant la valeur totale à l'appelant…

```
int i:=0, t:=0;

while( i < 10 ) {
       t := t + Data[i];
       i := i + 1;
       }

return t;
```

L'exemple ci-dessous montre le même programme, mais réécrit dans une forme compressée. Comme l'instruction de la boucle contrôle maintenant une seule action, les accolades ont été omises…

```
int i:=0, t:=0;

while( i < 10 )
       t += Data[i++];

return t;
```

LA BOUCLE FOR

Notez que la boucle **`while`** affichée ci-dessus dispose de quatre éléments…

1. L'initialisation de la variable de contrôle de la boucle.
2. L'évaluation d'un test pour vérifier si la boucle doit continuer.
3. L'exécution de l'action que la boucle doit effectuer.
4. La modification apportée à la variable du contrôle.

La boucle **`for`** permet d'associer les éléments 1, 2 et 4 dans une seule instruction de telle façon que l'action qui suit l'instruction doit seulement implémenter l'élément 3. Le résultat de cette syntaxe est assez semblable à celui de la boucle FOR-NEXT qui se trouve dans BASIC et d'autres langages similaires. En utilisant cette instruction, vous pouvez réécrire l'exemple ci-dessous sous la forme…

```
int i, t;

for( i:=t:=0; i<10; i++ )
       t += Data[i];

return t;
```

Notez que l'instruction **`for`** contient trois éléments distincts, chacun étant séparé par un point-virgule. Le premier élément est l'étape d'initialisation qui est effectuée une fois que la boucle a commencé. Le deuxième élément est la condition qui est testée au début de chaque itération de boucle pour vérifier si la boucle doit continuer. Le dernier élément est l'étape d'induction qui permet d'apporter des modifications à la variable de contrôle afin de déplacer la boucle vers sa prochaine itération. Une nouvelle fois, n'oubliez pas que si vous souhaitez inclure plusieurs actions dans la boucle, ajoutez-les entre des accolades !

LA BOUCLE DO

Ce type de boucle est semblable à la boucle **`while`** sauf que la condition est testée à la fin de la boucle. C'est-à-dire que la boucle s'exécute toujours au moins une fois. L'exemple ci-dessous montre l'exemple précédent, mais réécrit pour utiliser une boucle **`do`**…

```
int i:=0, t:=0;

do {
       t += Data[i];
       } while( ++i < 10 );

return t;
```

CONTROLE DES BOUCLES

Vous pouvez utiliser deux autres instructions dans les boucles. L'instruction **`break`** permet de terminer la boucle par anticipation alors que l'instruction **`continue`** permet de reprendre au la boucle à son début sans atteindre la fin de celle-ci. Pour être logiques, ces instructions doivent être utilisées avec des instructions **`if`** pour que leur exécution soit conditionnelle. L'exemple ci-dessous montre une boucle qui s'est terminée par anticipation lorsqu'un autre programme était vrai…

```
for( i:=0; i<10; i++ ) {
       if( LoopAbort() )
              break;
       LoopBody();
       }
```

REMARQUES POUR LES UTILISATEURS D'EDICT

Les utilisateurs du logiciel Edict-97 de Red Lion doivent noter ce qui suit…

- Crimson prend en charge les variables locales à l'aide des déclarations de style C dans le corps du programme plutôt que par le biais de la table des variables locales. A la différence des variables locales d'Edict, les variables de Crimson sont conservées dans la pile et peuvent donc être utilisées si un programme est appelé de façon récursive. Cela signifie également que les variables locales de Crimson ne conservent pas leurs valeurs entre les appels de programmes.

  - Crimson prend en charge le passage des arguments en fonctions. Par conséquent, il est inutile d'utiliser des variables globales pour improviser une telle fonctionnalité. Comme avec les variables locales, les arguments sont stockés sur la pile et peuvent donc être utilisés de façon récursive.

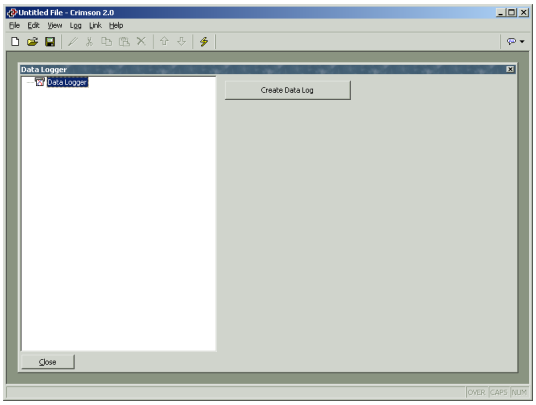
  - Crimson ne prend pas en charge la fonction **`Dispatch`**. La décision d'exécuter un programme au premier plan ou en arrière-plan repose sur les propriétés du programme et non pas sur la méthode utilisée pour son appel.

  - Crimson appelle des programmes à l'aide de la syntaxe de style C et (alors que l'ancienne syntaxe est toujours prise en charge) la fonction **`Run`** n'a pas besoin d'être utilisée. Les programmes qui renvoient les valeurs doivent être appelés à l'aide de la syntaxe plus récente car les fonctions de la famille Edict comme **`RunInteger`** ne sont pas fournies.

  - Les programmes s'exécutent beaucoup plus rapidement dans Crimson qu'ils ne le faisaient dans Edict !

CONFIGURATION DE L'ENREGISTREMENT DE DONNEES

Maintenant que vous avez configuré le cœur de votre application, vous pouvez décider d'utiliser l'historique de données de Crimson pour enregistrer certaines valeurs d'étiquette sur la carte CompactFlash. Les données ainsi enregistrées sont stockées dans des fichiers CSV normalisés et vous pouvez facilement les importer dans des applications comme Excel à l'aide de différentes méthodes. Pour configurer l'enregistrement des données, sélectionnez l'icône Historique de Données dans l'écran principal…



CREATION D'UN HISTORIQUES DE DONNEES

Vous pouvez utiliser le bouton Créer un historique de données pour créer autant d'historiques de données que vous le souhaitez. Comme chaque historique peut enregistrer un nombre illimité d'étiquettes de données, la plupart des applications utilisent uniquement un seul historique. Cependant, comme chaque historique possède un groupe fixe de propriétés en termes de taux d'échantillonnage, vous pouvez décider d'utiliser plusieurs historiques si vous voulez échantillonner différentes données à des taux différents.

UTILISATION DE LA LISTE DES HISTORIQUES

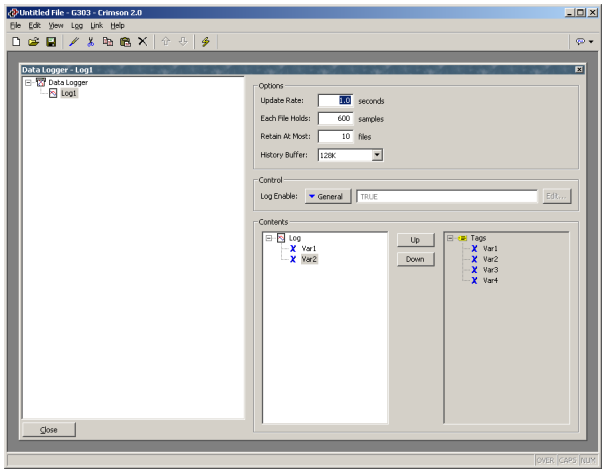
Pour renommer ou supprimer des historiques de données, cliquez sur le volet gauche de la fenêtre Historique de Données. Les commandes du menu Historique peuvent être alors utilisées pour apporter les modifications souhaitées. Une autre solution consiste à cliquer avec le bouton droit sur l'historique de données requis, puis à effectuer votre choix à partir du menu.

(Notez que le nom d'un historique de données doit comporter huit caractères ou moins car le nom est utilisé pour définir le répertoire sous lequel les fichiers journaux sont enregistrés et le module maître n'est pas en mesure de gérer des noms qui ne sont pas conformes à l'attribution de nom 8.3 de style FAT.)

Pour sélectionner un historique de données, cliquez sur le nom dans la liste ou utilisez les flèches haut et bas de la barre d'outils. Une autre solution consiste à utiliser les associations de touches **`Alt+Gauche`** et **`Alt+Droite`** pour vous déplacer vers le haut ou le bas de la liste. Ces touches fonctionnent quel que soit le volet qui est sélectionné.

PROPRIETES DE L'HISTORIQUE DE DONNEES

Chaque historique de données comporte les propriétés suivantes…



- La propriété *Rafraîchissement* permet d'indiquer la fréquence à laquelle Crimson prend un échantillon d'éléments de données pour l'enregistrer. Le taux d'échantillonnage le plus rapide est d'une seconde, mais notez que l'utilisation d'un tel taux élevé produit de grandes quantités de données ! Toutes les étiquettes de l'historique sont échantillonnées au même taux.

- La propriété *Chaque fichier* contient permet d'indiquer le nombre d'échantillons qui sont compris dans chaque fichier journal. Lorsque tous ces échantillons ont été enregistrés, un nouveau fichier journal est créé à l'aide d'un autre nom. En règle générale, cette valeur est définie de telle façon que chaque fichier journal contient une quantité pertinente de données. Par exemple, l'historique affiché ci-dessus est configuré pour utiliser un nouveau fichier journal chaque jour.

  - La propriété *Nombre max*. de fichiers permet d'indiquer le nombre de fichiers journaux qui sont conservés sur la carte CompactFlash avant que le plus ancien fichier ne soit supprimé. Cette propriété doit être définie pour permettre à tout élément qui consomme des informations enregistrées d'extraire les données à partir du module maître avant que les informations ne soient supprimées. L'historique affiché ci-dessus est configuré pour conserver les données pendant une semaine.

  - La propriété *Condition* permet d'autoriser ou d'empêcher l'enregistrement. Si l'expression saisie est vraie, l'enregistrement est activé. Si l'expression est fausse, l'enregistrement est désactivé. Si aucune expression n'est saisie, l'enregistrement est active par défaut.

  - La propriété *Mémoire pour Historique* permet d'indiquer la quantité de RAM qui doit être attribuée à la mémoire tampon pour cet historique de données. La mémoire tampon de l'historique permet de prendre en charge la Primitive Courbes d'historique de données et permet à l'utilisateur de faire défiler vers l'arrière pour afficher d'anciennes données qui sont disponibles. Un maximum

de 256 Ko doit être attribué à tous les historiques de données. Cette propriété est ignorée lorsque le maître est configuré pour afficher un HMI virtuel monochrome.

- La propriété *Contenu* permet d'indiquer l'étiquette qui doit être enregistrée. La première liste montre les étiquettes sélectionnées alors que la seconde montre celles qui sont disponibles dans la base de données. Vous pouvez ajouter des étiquettes à l'historique en double-cliquant dessus dans la liste située à droite. Vous pouvez les supprimer en double-cliquant dessus dans la liste située à gauche ou en appuyant sur la touche Suppr de votre clavier lorsque l'étiquette est sélectionnée. Vous pouvez utiliser les boutons Haut et Bas pour déplacer les étiquettes dans la liste. Peut-être qu'un jour quelqu'un inventera une solution de glisser-déplacer pour permettre de manipuler plus facilement cette liste !

STOCKAGE DES FICHIERS JOURNAUX

Comme nous l'avons vu précédemment, un historique de données enregistre ses données dans une série de fichiers sur la carte CompactFlash du maître. Ces fichiers sont placés dans un sous-répertoire nommé d'après le nom de l'historique de données et ce répertoire est stocké sous une entrée de répertoire racine appelée LOGS. Les fichiers sont nommés d'après l'heure et la date auxquelles la planification de l'historique doit commencer. Si chaque fichier contient une heure ou plus d'informations, les fichiers sont nommés **`AAMMJJhh.CSV`**, où **`AA`** représente l'année du fichier, **`MM`** le mois, **`JJ`** la date et **`hh`** l'heure. Si chaque fichier contient moins d'une heure d'informations, les fichiers sont alors nommés **`MMJJhhmm.CSV`**, où les six premiers caractères sont décrits ci-dessus et où **`mm`** représente la minute à laquelle l'historique a débuté. Ces règles garantissent que chaque historique dispose d'un nom unique.

LE PROCESSUS D'ENREGISTREMENT

L'historique de données de Crimson fonctionne à l'aide de deux processus distincts. Le premier échantillonne chaque point d'entrée au taux spécifié dans les propriétés et place les données enregistrées dans une mémoire tampon de la RAM du module maître. Le second processus s'exécute toutes les deux minutes et écrit les données à partir de la RAM vers la carte CompactFlash. Cette structure présente plusieurs avantages…

- Les écritures sur la carte CompactFlash sont garanties pour commencer uniquement sur un intervalle de deux minutes, c'est-à-dire à exactement 2, 4 ou 6 minutes après l'heure, et ainsi de suite. Cela signifie que si votre module maître prend en charge l'échange à chaud des cartes CF, vous pouvez attendre la prochaine apparition d'écritures pour commencer et, lorsque la LED d'activité de la carte CompactFlash située à l'avant du maître cesse de clignoter, vous avez jusqu'au démarrage de l'intervalle de deux minutes suivant avant que d'autres tentatives d'écritures ne se produisent. Ainsi, vous pouvez retirer la carte sans craindre l'endommagement des données. Si vous insérez une nouvelle carte avant la fin des quatre minutes, aucune donnée n'est perdue.

- Les écritures sur la carte CompactFlash atteignent un niveau de performances supérieur en évitant de devoir en permanence mettre à jour la structure des données

du système de fichiers de la carte pour chaque échantillon. Pour les historiques qui sont configurés pour échantillonner à des taux de données très élevés, la bande passante d'une carte classique CompactFlash ne permet pas d'écrire de façon fiable les données en l'absence d'un tel processus de mise en mémoire tampon.

Notez que comme les données ne sont pas validées sur la carte CompactFlash jusqu'à deux minutes, une telle quantité de données peut être perdue lorsque le maître est éteint. De plus, si le maître est éteint lorsqu'une écriture est en cours, la carte CompactFlash peut être endommagée. Pour vous assurer qu'un tel endommagement n'est pas permanent, le module maître utilise un système de journalisation qui met en cache les écritures sur une autre mémoire non volatile dans le maître. Si le maître détecte qu'une écriture a été interrompue lors de la mise hors tension, l'écriture est répétée lors de la mise sous tension, ce qui annule tout endommagement et répare la carte CompactFlash.

Ainsi, si vous souhaitez retirer une carte CompactFlash d'un maître effectuant un enregistrement de données, vous devez observer la procédure décrite ci-dessus, relative à l'activité de la LED, puis le mettre hors tension lorsque l'activité est terminée. Si vous n'êtes pas sûr que le maître a été éteint correctement, remettez-le sous tension, autorisez une séquence d'écritures complète sur la carte CompactFlash, puis mettez-le hors tension en suivant la procédure correcte. Vous pouvez retirer la carte en toute sécurité.

Comme les procédures nécessaires au retrait d'une carte CompactFlash sont relativement complexes, Crimson fournit deux autres mécanismes permettant d'accéder aux fichiers journaux, éliminant de ce fait le besoin de retirer la carte. Ces méthodes sont décrites ci-dessous.

ACCES AUX FICHIERS JOURNAUX

Il existe deux autres méthodes d'accès aux fichiers journaux…

- La méthode la moins recommandée consiste à monter la carte comme lecteur sur un PC en appliquant le processus décrit au début de ce manuel pour que les historiques puissent être copiés à l'aide de Windows Explorer. Notez que Windows 2000 ou version ultérieure est conseillée si vous utilisez cette méthode car les versions antérieures de Windows peuvent verrouiller la carte CompactFlash et empêcher l'enregistrement de données.

- La méthode préférée consiste à utiliser le serveur Web tel que décrit dans le chapitre suivant. Une fois le serveur Web activé, vous pouvez accéder aux fichiers journaux sur le port Ethernet du maître en utilisant un navigateur Web comme Microsoft Internet Explorer ou le processus automatisé mis en œuvre par l'utilitaire WebSync, qui est fourni avec le logiciel de configuration de Crimson.

UTILISATION DE WEBSYNC

Vous pouvez exécuter l'utilitaire WebSync (qui est stocké dans le répertoire spécifié lors de l'installation du logiciel) pour synchroniser un répertoire sur un PC avec le contenu des historiques de données du maître. Vous pouvez décider de configurer une application comme Windows Scheduler (ou peut-être un démon **`cron`**), pour qu'elle utilise cet utilitaire de façon

régulière ou vous pouvez utiliser un commutateur de ligne de commande pour demander à WebSync d'effectuer automatiquement l'interrogation. Vous pouvez également décider d'héberger WebSync sur un serveur central pour que les fichiers journaux soient disponibles pour les utilisateurs sélectionnés sur votre réseau d'entreprise.

SYNTAXE DE WEBSYNC

WebSync est appelé à partir de la ligne de commande à l'aide de la syntaxe suivante…

```
websync {switches} <nom_hôte>
```

…où **<nom_hôte>** est remplacé par l'adresse IP du maître à interroger.

SWITCHES FACULTATIFS

Le champ **switches** peut contenir une ou plusieurs des options suivantes…

- Vous pouvez utiliser **–terse** pour supprimer les informations de progression.
- Vous pouvez utiliser **–poll <n>** pour interroger le maître toutes les **n** minutes.
- Vous pouvez utiliser **–path <dir>** pour spécifier dir comme le répertoire dans lequel se trouvent les fichiers journaux.
- Vous pouvez utiliser **-ras <nom>** pour appeler une connexion d'appel sortant afin d'accéder à l'unité.
- Vous pouvez utiliser **-user <nom>** pour spécifier le nom d'utilisateur de la connexion.
- Vous pouvez utiliser **-pass <mot_de_passe>** pour spécifier le mot de passe de la connexion.
- Vous pouvez utiliser **-num <num>** pour remplacer le numéro de téléphone de la connexion.

EXEMPLE

Vous trouverez ci-dessous un exemple de ligne de commande…

```
websync –poll 10 –path C:\Logs 192.9.200.52
```

…lit les fichiers journaux de tous les historiques de données sur le maître avec l'adresse IP 192.9.200.52, puis enregistre ces journaux dans des sous-répertoires du répertoire **C:\Logs**. WebSync continue à s'exécuter et répète le processus d'interrogation toutes les dix minutes. Il est conseillé d'avoir un intervalle de téléchargement inférieur au rafraîchissement multiplié par le nombre d'échantillons par fichier multiplié par le nombre de fichiers. Si cette contrainte est remplie, le répertoire du PC accumule des copies de tous les fichiers journaux à partir du maître.

REMARQUES POUR LES UTILISATEURS D'EDICT

Les utilisateurs du logiciel Edict-97 de Red Lion doivent noter ce qui suit…

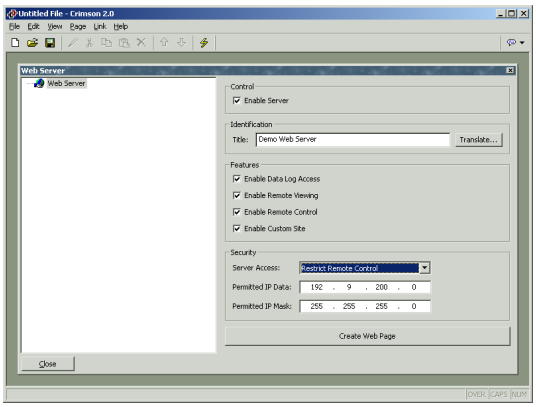
- Les historiques de Crimson enregistrent l'heure et la date de chaque échantillon et n'ont pas besoin d'enregistrer les valeurs « vides » utilisées par Edict pour marquer les périodes de désactivation. Lorsqu'un module maître est mis sous tension, il s'affiche simplement un vide dans les fichiers journaux.

CONFIGURATION DU SERVEUR WEB

Le serveur Web de Crimson permet d'exposer différentes données via le port Ethernet du module maître, permettant un accès à distance pour diagnostiquer les informations ou aux valeurs enregistrées par l'historique de données. Vous pouvez configurer le serveur Web en sélectionnant l'icône Serveur Web dans l'écran principal.

PROPRIETES DU SERVEUR WEB

Le serveur Web est doté des propriétés suivantes…



- La propriété *Activer le Serveur* permet d'activer ou de désactiver le serveur Web. Si le serveur est activé, le maître contrôle les demandes entrantes du port 80 et répond aux demandes, le cas échéant. Si le serveur est désactivé, les connexions à ce port sont refusées. Gardez à l'esprit que pour que le serveur fonctionne, le port Ethernet du maître doit avoir été activé depuis la fenêtre Communications.

- La propriété *Titre* permet de fournir le titre qui sera affiché dans le menu du serveur Web. Vous pouvez utiliser ce titre pour différencier plusieurs maîtres sur le réseau, garantissant ainsi que vous accédez au maître correct.

- La propriété *Activer l'accès aux données* permet d'activer ou de désactiver l'accès Web aux fichiers créés par l'historique de données. A l'évidence, cette fonction doit être activée si vous devez utiliser l'utilitaire WebSync pour copier des fichiers journaux sur un PC.

- La propriété *Activer visualisation à distance* permet d'activer ou de désactiver la fonction grâce à laquelle vous pouvez utiliser un navigateur Web pour afficher le HMI virtuel du maître. Cette fonction est très utile lorsque vous diagnostiquez à distance des problèmes qu'un opérateur peut rencontrer avec une machine.

- La propriété *Activer Contrôle à distance* permet d'activer ou de désactiver une option grâce à laquelle la fonctionnalité d'affichage à distance est étendue pour permettre d'utiliser un navigateur Web afin de simuler l'utilisation des touches sur le HMI virtuel, ce qui permet donc de contrôler à distance le maître ou la machine qu'il commande. Même si cette fonctionnalité est très utile, vous devez être attentif lors de l'utilisation des différents paramètres de sécurité afin d'éviter une manipulation non

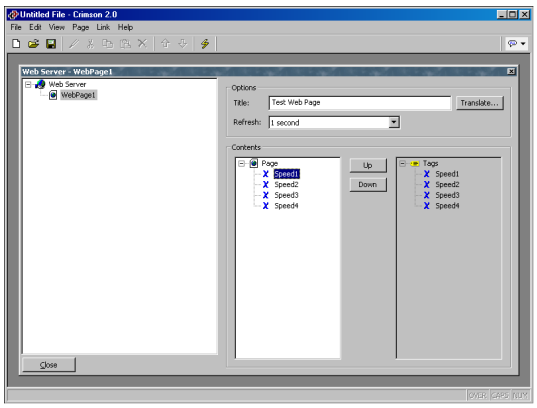
autorisée de la machine. En outre, nous vous recommandons fortement d'utiliser un pare-feu externe si le maître peut être atteint à partir d'Internet.

- La propriété *Activer site Web personnalisé* permet d'activer ou de désactiver une fonction grâce à laquelle les fichiers enregistrés dans le répertoire WEB de la carte CompactFlash sont exposés via le serveur Web. Vous trouverez une description plus détaillée de cette fonction ci-dessous.

- Les propriétés *Sécurité* permettent de restreindre l'accès au serveur Web aux hôtes dont l'adresse IP correspond au masque et aux données spécifiés. Tous les accès peuvent être restreints ou le filtre peut être utilisé pour limiter uniquement les tentatives d'utilisation de la fonction de contrôle à distance. Il est de votre ressort d'utiliser un pare-feu externe pour empêcher tout accès non autorisé si la fonction de contrôle à distance est activée car le filtre IP peut être vaincu par certaines techniques avancées de piratage et qu'il n'est pas garanti par Red Lion Controls.

AJOUT DE PAGES WEB

Outre les fonctions décrites ci-dessus, le serveur Web prend en charge l'affichage des pages Web génériques, chacune contenant une liste prédéfinie de valeurs d'étiquettes. Ces pages sont créées en appuyant sur le bouton Créer une page Web d'accès de données situé sous les propriétés du serveur Web et elles sont enregistrées dans une liste semblable à celle qui est utilisée pour les pages d'affichage, les historiques de données, etc.



Chaque page Web comporte les propriétés suivantes…

- La propriété *Titre* permet d'identifier la page Web dans le menu présenté à l'utilisateur via le navigateur Web. Même si le titre peut être traduit, les versions actuelles de Crimson utilisent uniquement la version américaine du texte.

- La propriété *Actualiser* permet d'indiquer si l'on doit demander ou non au navigateur Web de rafraîchir automatiquement le contenu de la page. Des rafraîchissements entre 1 et 8 secondes sont pris en charge. Notez que le rafraîchissements représenté sur le navigateur Web varie en fonction de la base de donnée exact et des performances de la machine utilisés. La mise à jour n'est pas conçue pour être sans scintillement.

- La propriété *Contenu* permet d'indiquer l'étiquette qui doit être ajoutée à la page. La première liste montre les étiquettes sélectionnées alors que la seconde montre celles qui sont disponibles dans la base de données. Vous pouvez ajouter des étiquettes à la page en double-cliquant dessus dans la liste située à droite. Vous pouvez les supprimer en double-cliquant dessus dans la liste située à gauche ou en appuyant sur la touche **Suppr** de votre clavier lorsque l'étiquette est sélectionnée. Vous pouvez utiliser les boutons Haut et Bas pour déplacer les étiquettes dans la liste. Peut-être qu'un jour quelqu'un inventera une solution de glisser-déplacer pour permettre de manipuler plus facilement cette liste !

UTILISATION D'UN SITE WEB PERSONNALISE

Alors que les pages Web standard fournissent un accès rapide et simple aux données dans le maître, vous risquez de trouver que votre incapacité à modifier leur mise en forme précise contrarie assez vos capacités artistiques. Ainsi, vous pouvez utiliser la fonction de site personnalisé du maître pour créer un site Web entièrement personnalisé à l'aide de votre éditeur HTML tiers favori et, en insérant certaines séquences spéciales et en enregistrant les fichiers obtenus sur la carte CompactFlash du maître, exposer ce site à l'aide du serveur Web de l'appareil.

CREATION DU SITE

Le site Web peut utiliser n'importe quelle fonction HTML prise en charge par votre navigateur, mais il ne doit pas utiliser ASP, CGI ou d'autres astuces côté serveur. Les noms de fichiers utilisés pour les fichiers HTML et les graphiques associés doivent également être conformes à l'ancienne convention d'attribution de nom 8.3. Ainsi, les extensions de fichier sont, par exemple, **HTM** au lieu de **HTML** et **JPG** au lieu de **JPEG**. Cela signifie également que le corps du nom de fichier doit comporter huit caractères maximum et que vous ne devez pas vous fier à la différence entre les majuscules et les minuscules pour différencier les pages. Vous pouvez utiliser n'importe quelle structure de répertoire si vous vous assurez que vos répertoires observent la convention d'attribution de nom 8.3 et si vous ne vous fiez pas aux différences de casse.

INCORPORATION DE DONNEES

Pour incorporer des données d'étiquettes dans une page Web, insérez la séquence **[[N]]** en remplaçant **N** par le numéro d'index de l'étiquette en question. Ce numéro d'index s'affiche dans la barre d'état lorsqu'une étiquette est sélectionnée dans la fenêtre Etiquettes de données et correspond plus ou moins à l'ordre de création des étiquettes. Lorsque la page Web qui contient cette séquence est utilisée, la séquence est remplacée par la valeur actuelle de l'étiquette, mise en forme selon les propriétés de l'étiquette.

DEPLOIEMENT DU SITE

Pour déployer votre site Web personnalisé, copiez-le dans le répertoire **\WEB** de la carte CompactFlash qui doit être installée sur le maître. Pour copier les fichiers, montez la carte comme lecteur sur votre PC, tel que décrit au début du présent manuel, ou utilisez un programme d'écriture de carte approprié connecté à votre PC. Assurez-vous que la propriété

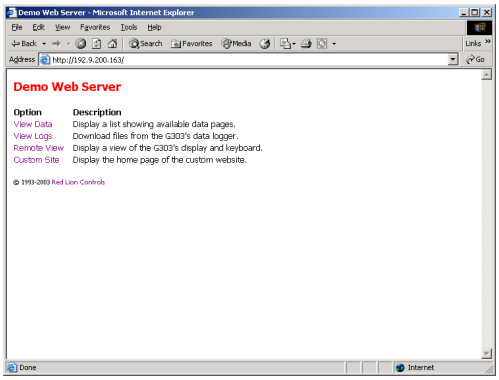
Activer site Web personnalisé est définie et le site personnalisé s'affichera alors dans le menu du serveur Web. Une fois que le site est sélectionné, un fichier appelé **DEFAULT.HTM** du répertoire **\WEB** s'affiche. Au-delà de ce point, la navigation s'effectue en fonction des liens présents dans le site.

ACCES A LA CARTE COMPACTFLASH

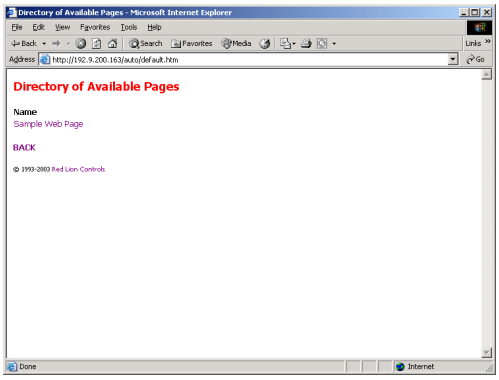
Notez que pour utiliser des pages Web personnalisées (ou pour fournir un accès à l'historique de données du maître), le serveur Web doit pouvoir accéder à la carte CompactFlash de l'unité. Si vous avez monté la carte comme lecteur sur votre PC et effectué des opérations d'écriture, vous devrez peut-être patienter une minute environ que le PC déverrouille la carte et permette au maître d'obtenir l'accès. Si vous utilisez un système d'exploitation antérieur à Windows 2000 pour effectuer une telle opération, votre PC verrouille peut-être la carte lorsque le lecteur est monté pour la première fois, qu'une écriture soit effectuée ou non. Une nouvelle fois, ce verrou est libéré en une minute environ.

EXEMPLES DU SERVEUR WEB

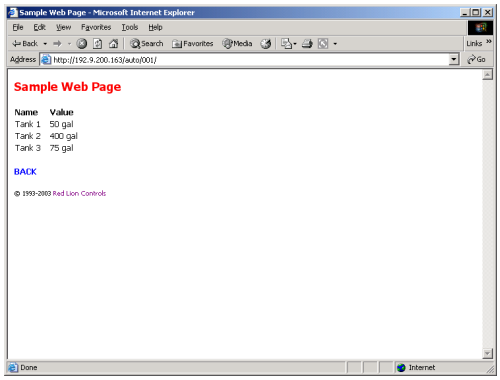
La capture d'écran ci-dessous montre le menu principal que le serveur Web affiche…



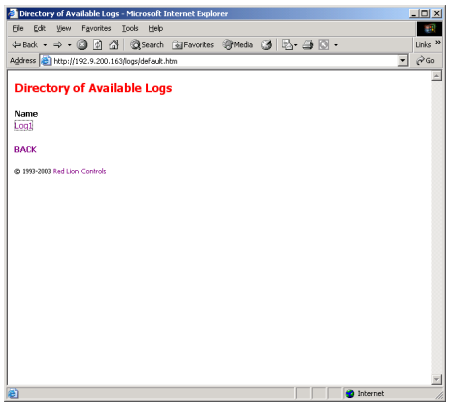
La capture d'écran ci-dessous montre une liste de pages Web standard…



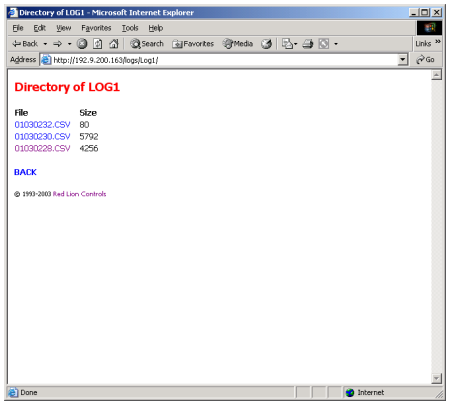
La capture d'écran ci-dessous montre une page Web standard contenant trois étiquettes…



La capture d'écran ci-dessous montre le menu de l'historique de données…

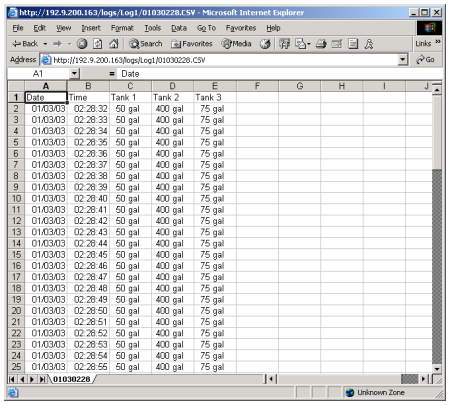


La capture d'écran ci-dessous montre le contenu d'un historique de données…

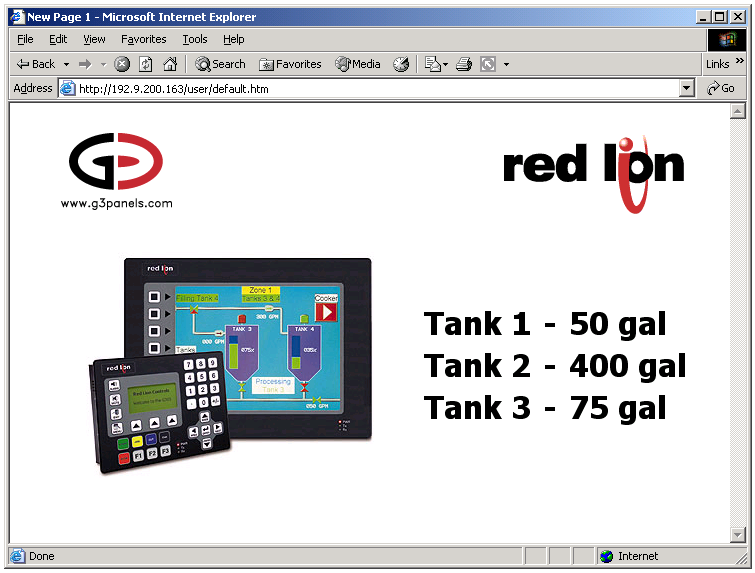


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

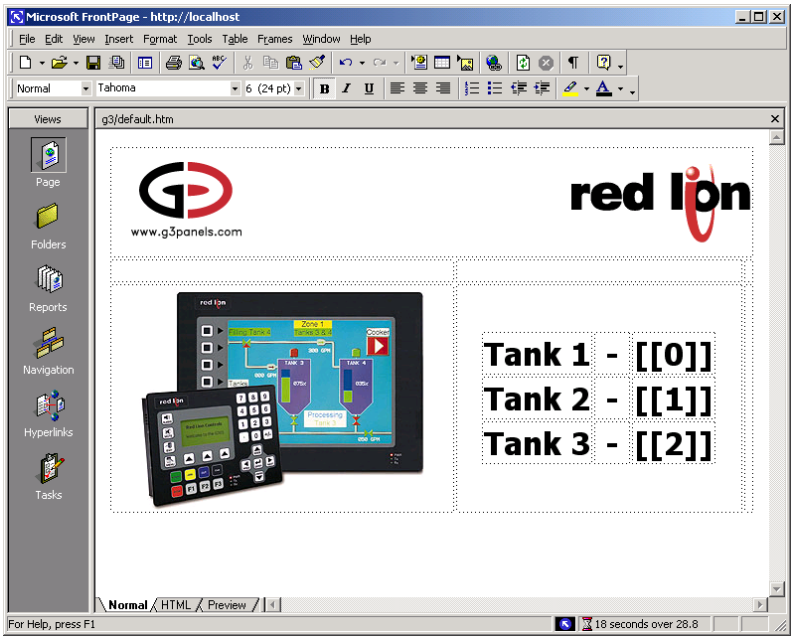
La capture d'écran ci-dessous montre le contenu d'un fichier journal…



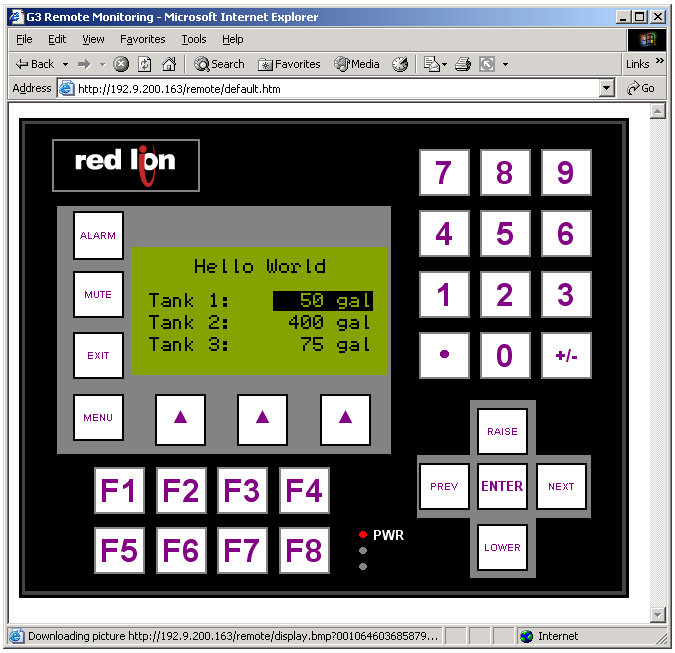
La capture d'écran ci-dessous montre une page personnalisée contenant trois étiquettes…



La capture d'écran ci-dessous montre la création d'une page personnalisée dans FrontPage…



La capture d'écran ci-dessous montre l'affichage à distance et/ou l'affichage de contrôle du G303…



UTILISATION DU SYSTEME DE SECURITE

Crimson contient des fonctionnalités qui vous permettent de définir les opérateurs qui ont accès à telle ou telle page d'affichage et de limiter ces opérateurs qui peuvent apporter des modifications aux données sensibles. Le logiciel contient également une fonctionnalité de journalisation de la sécurité que vous pouvez utiliser pour enregistrer les modifications apportées aux valeurs de données, indiquant la date de la modification et l'utilisateur à l'origine de cette modification.

CONCEPTS DE BASE DE LA SECURITE

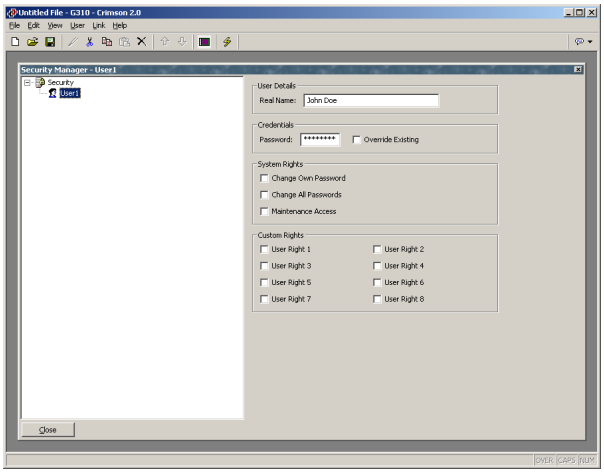
Les sections suivantes détaillent certains des concepts de base que le système de sécurité utilise.

SECURITE BASEE SUR L'OBJET

Le système de sécurité de Crimson est basé sur l'objet. Ainsi, les caractéristiques de sécurité sont appliquées à une page d'affichage ou une étiquette, et pas à l'élément de l'interface utilisateur qui accède à la page ou apporte une modification à l'étiquette. L'autre approche basée sur le sujet implique généralement que vous devez être attentif à appliquer les paramètres de sécurité à chaque élément de l'interface utilisateur qui peut modifier les données sécurisée. L'approche de Crimson évite cette duplication et garantit qu'une fois que vous avez décidé de protéger une étiquette, elle reste protégée dans toute votre base de données.

UTILISATEURS NOMMES

Crimson prend en charge la fonctionnalité de création d'utilisateurs (quel que soit leur nombre). Chacun d'entre eux aura alors un nom d'utilisateur, un nom complet et un mot de passe. Le nom d'utilisateur est une chaîne respectant la casse sans aucun espace incorporé, qui permet d'identifier l'utilisateur lors de la connexion alors que le nom réel est en général une chaîne plus longue utilisée dans les fichiers de connexion pour enregistrer l'identité lisible de l'utilisateur qui apporte la modification. Notez que vous êtes libre d'utiliser ces champs d'autres façons s'ils sont adaptés à votre application. Vous pouvez par exemple créer des utilisateurs qui représentent des groupes d'individus ou peut-être des rôles comme Opérateurs, Superviseurs et Managers. Vous pouvez également décider d'utiliser le nom complet pour conserver un élément comme un numéro d'horloge afin de lier les identités des utilisateurs à votre système MRP.



DROITS D'UTILISATEUR

Chaque utilisateur dispose d'aucun ou de plusieurs droits d'accès. Un utilisateur sans aucun droit peut accéder aux objets qui nécessitent simplement l'enregistrement de l'identité de l'utilisateur alors que les utilisateurs avec plusieurs droits peuvent accéder aux objets qui exigent que ces droits soient présents. Les droits sont divisés en Droits système et en Droits utilisateur, les premiers contrôlant l'accès aux fonctions du logiciel Crimson et les seconds étant disponibles pour une utilisation générale. Par exemple, le Droit utilisateur 1 peut être utilisé dans votre base de données pour contrôler l'accès aux cibles de production. Seuls les utilisateurs qui, selon vous, devraient pouvoir modifier ces éléments disposent de ce droit.

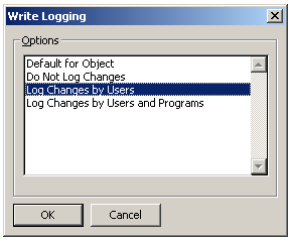
CONTROLE DE L'ACCES

Les objets qui sont soumis à la sécurité possèdent un paramètre de contrôle d'accès associé…

Ce paramètre vous permet de spécifier si n'importe quel utilisateur peut accéder à l'élément, si n'importe quel opérateur authentifié peut y accéder ou si les utilisateurs avec des droits d'utilisateur spécifiques peuvent y accéder. Le paramètre du contrôle d'accès vous permet également de spécifier si une étiquette peut être modifiée par un programme qui s'exécute suite à quelque chose d'autre que l'action d'utilisateur. Cette fonctionnalité vous permet de garantir qu'aucune modification en arrière-plan des données sensible ne s'est produite même si une erreur de programmation tente d'apporter une telle modification.

JOURNALISATION D'ECRITURES

Les étiquettes disposent également d'une propriété de journalisation des écritures...



Elle indique si les modifications apportées à une étiquette par des utilisateurs ou des programmes doivent être enregistrées. Cette fonctionnalité vous permet de créer un suivi d'audit des modifications apportées à votre système, simplifiant de ce fait la détection d'erreurs et fournissant des informations de contrôle qualité permettant de traiter la configuration. Notez que vous devez être attentif lorsque vous enregistrez les modifications apportées par les programmes car certaines bases de données peuvent enregistrer des quantités incontrôlables de données dans de telles circonstances.

ACCES PAR DEFAUT

Pour accélérer le processus de configuration, Crimson permet également de spécifier un accès par défaut et des paramètres de journalisation d'écritures pour les étiquettes mappées et internes ainsi que pour les pages d'affichage. La différenciation entre les étiquettes mappées et non mappées est importante dans les systèmes où toutes les modifications apportées aux données externes doivent être enregistrées, mais où les données internes de Crimson peuvent être manipulées sans avoir à recourir à un tel suivi d'audit.

CONNEXION A LA DEMANDE

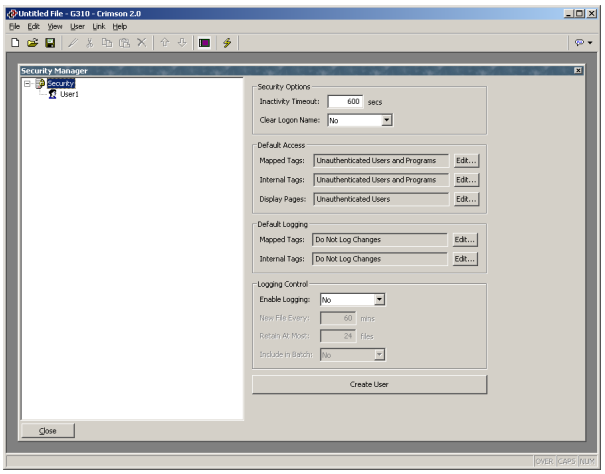
Le système de sécurité de Crimson prend en charge la connexion classique et la connexion à la demande. Une connexion classique peut se produire lorsque vous utilisez un élément de l'interface utilisateur comme un bouton-poussoir pour activer l'action Authentification Utilisateur ou pour appeler la fonction **`UserLogOn()`**. Une connexion à la demande se produit si l'opérateur tente une action sans disposer des droits d'accès suffisants et si cette tentative de connexion à abouti. Par exemple, un utilisateur peut appuyer sur un bouton qui exécute un programme pour réinitialiser plusieurs valeurs. Dès que le programme tente de modifier une valeur qui nécessite un accès sécurisé, le système invite à entrer les informations d'identification d'ouverture de session. Cette méthode réduit l'interaction de l'opérateur et le système est plus réactif.

ACCES A LA MAINTENANCE

Le système fournit également une fonctionnalité appelée Mode Maintenance qui permet de supprimer le délai d'inactivité de l'utilisateur lors de la mise en service du système. Ce mode est activé si une page d'affichage est marquée comme étant accessible avec le droit Accès Maintenance et si l'utilisateur actuel a obtenu l'accès à la page en conséquence de ce droit. L'utilisation de ce mode vous évite de devoir vous connecter de façon répétée lorsque vous testez le système.

PARAMETRES DE SECURITE

Vous pouvez accéder aux paramètres du système de sécurité via l'icône Gestionnaire de sécurité…



Les propriétés disponibles se présentent comme suit…

- La propriété *Délai d'inactivité* permet d'indiquer la durée écoulée dans aucune entrée d'utilisateur avant que l'utilisateur actuel soit automatiquement déconnecté. Si la valeur du paramètre est trop élevée, le système n'est pas sécurisé. Si la valeur du paramètre est trop basse, les opérateurs peuvent rencontrer des difficultés manipulations avec le système.

- La propriété *Effacer le Nom d'Utilisateur* permet d'indiquer si le nom d'utilisateur doit être effacé ou non avant de demander à l'opérateur de se connecter. Si ce paramètre est désactivé, le nom d'utilisateur précédent est affiché et seul le mot de passe doit être ressaisi. Si cette fonctionnalité est activée, une plus grande sécurité est garantie. Elle peut également être requise pour une conformité avec les normes de sécurité de certains secteurs

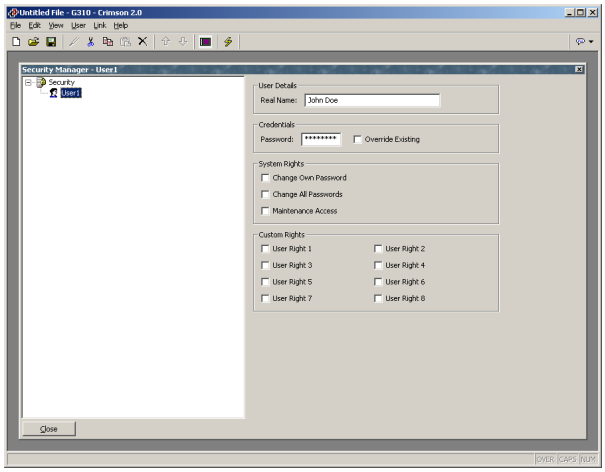
- La propriété *Accès par défaut* permet d'indiquer l'accès à fournir aux différents objets au cas où aucun accès spécifique ne serait défini pour cet élément. Les paramètres sont identiques à ceux qui sont décrits dans la section Contrôle de l'accès ci-dessus.

- Les propriétés *Enregistrement par Défaut* permettent d'indiquer si les modifications apportées aux étiquettes mappées et non mappées doivent être enregistrées si aucun critère de journalisation spécifique n'est défini pour une étiquette. Il est impossible d'enregistrer un accès par programmation par défaut car vous devez envisager une telle journalisation avec soin pour éviter une activité d'historique excessive.

- Les propriétés *Contrôle de l'Enregistrement* permettent de définir si et comment les journaux de sécurité doivent être créés. Reportez-vous au chapitre Configuration de l'enregistrement des données pour obtenir des informations sur l'écriture des données et l'attribution de noms aux fichiers.

CREATION DES UTILISATEURS

Vous pouvez utiliser le bouton Créer un utilisateur pour créer autant d'utilisateurs que vous le souhaitez. Vous pouvez renommer ou supprimer les utilisateurs à l'aide du volet gauche. Pour sélectionner un utilisateur, cliquez sur le nom dans la liste ou utilisez les flèches haut et bas de la barre d'outils. Une autre solution consiste à utiliser les associations de touches **`Alt+Gauche`** et **`Alt+Droite`** pour vous déplacer vers le haut ou le bas de la liste. Ces touches fonctionnent quel que soit le volet qui est sélectionné.



Chaque utilisateur comporte les propriétés suivantes…

- La propriété *Nom Réel* permet d'enregistrer l'identité d'un utilisateur dans les journaux de sécurité et dans la primitive Gestionnaire de sécurité qui est utilisée pour modifier les mots de passe à partir du terminal de l'opérateur. Si la sécurité maximale est requise, le nom de l'utilisateur ne doit pas être facilement dérivé du nom complet.

- La propriété *Mot de passe* permet de spécifier un mot de passe initial pour cet utilisateur. Le mot de passe respecte la casse et se compose de caractères alphanumériques. Notez que si la case *Réécriture existant* est cochée, toutes les modifications apportées à ce mot de passe à partir du panneau de l'opérateur sont remplacées lorsque cette base de données est téléchargée dans le panneau.

- Les propriétés *Droits système* permettent d'attribuer à un utilisateur la capacité à effectuer certaines actions du système. Les propriétés relatives aux modifications du mot de passe sont évidentes et l'utilisateur du Mode de maintenance est décrit ci-dessus.

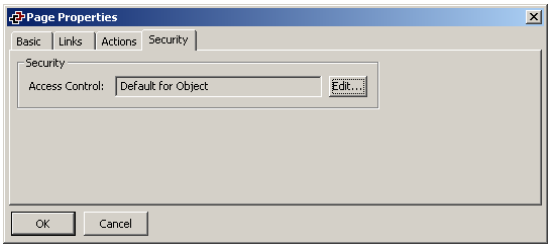
- Les propriétés *Droits personnalisés* permettent d'attribuer à un utilisateur certains droits qu'il peut alors utiliser dans la base de données pour autoriser l'accès aux groupes d'étiquettes ou de pages d'affichage. L'utilisation exacte de ces droits dépend du concepteur du système.

SPECIFICATION DE LA SECURITE DES ETIQUETTES

Chaque étiquette accessible en écriture dispose d'un onglet appelé Sécurité qui permet de définir les paramètres du contrôle d'accès et de la journalisation d'écritures de cette étiquette. Si vous ne définissez pas les paramètres spécifiques, le système utilise les paramètres par défaut appropriés, selon qu'il est mappé ou non aux données externes.

SPECIFICATION DE LA SECURITE DES PAGES

Les paramètres du contrôle d'accès d'une page d'affichage sont définis via la boîte de dialogue Propriétés…



Une nouvelle fois, si aucun paramètre n'est défini, les paramètres par défaut sont utilisés.

LA PRIMITIVE GESTIONNAIRE DE SECURITE

La primitive *Gestionnaire de sécurité* permet d'afficher les noms des utilisateurs présents sur le système. Vous pouvez l'utiliser pour modifier un mot de passe d'utilisateur en fonction des droits attribués à l'utilisateur en cours.

Les seules propriétés modifiables de cette primitive définissent les polices à utiliser et si la primitive doit être affichée ou non. Reportez-vous aux autres primitives pour obtenir des descriptions de ces paramètres.

FONCTIONS RELATIVES A LA SECURITE

Reportez-vous à l'Annexe A de ce manuel pour obtenir des détails sur les fonctions **`UserLogOn()`**, **`UserLogOff()`** et **`TestAccess()`**. Cette troisième fonction est utile lorsque vous modifiez plusieurs valeurs dans un programme car elle vous permet de forcer une vérification anticipée de l'accès dans le code pour éviter d'apporter des modifications uniquement pour que les opérations ultérieures échouent à cause de droits d'utilisateur insuffisants.

ECRITURE D'EXPRESSIONS

Rappelez-vous que nous avons vu dans les sections précédentes de ce manuel que de nombreux champs de Crimson sont configurés sous forme de propriétés Expression. Rappelez-vous également que ces champs sont configurés à l'aide d'un élément de l'interface utilisateur qui est semblable à celui qui se présente comme suit…

Dans de nombreuses situations, vous configurerez ces propriétés pour qu'elles soient égales à la valeur d'une étiquette ou au contenu du registre dans un périphérique de communication distant, auquel cas vous effectuerez votre sélection en cliquant sur l'option appropriée dans le menu déroulant, puis en sélectionnant l'élément requis dans la boîte de dialogue qui s'affiche.

Toutefois, vous pourrez vouloir rendre une propriété dépendante d'une association plus complexe d'éléments de données, peut-être en utilisant les mathématiques pour combiner ou comparer leurs valeurs. Vous pouvez gérer de telles éventualités par le biais de ce qu'on appelle des expressions, que vous pouvez saisir dans la zone d'édition de la propriété lorsque vous sélectionnez le mode Général via la liste déroulante.

VALEURS DE DONNEES

Toutes les expressions contiennent au moins une valeur de données. Les expressions les plus simples sont ainsi des références à des constantes uniques, des étiquettes uniques ou des registres d'API uniques. Si vous choisissez l'une des deux dernières options, Crimson simplifie le processus d'édition en modifiant automatiquement le mode de la propriété, le cas échéant. Par exemple, si vous saisissez un nom d'étiquette dans le mode Général, Crimson bascule en mode Etiquette et affiche le nom de l'étiquette dans le champ de sélection.

CONSTANTES

Les constantes représentent des nombres ou des chaînes de constantes.

CONSTANTES ENTIERES

Les constantes entières représentent des nombres signés à 32 bits. Vous pouvez les saisir au format décimal, binaire, octal ou hexadécimal. Les exemples ci-dessous présentent le même nombre entré dans les quatre bases différentes…

BASE	EXEMPLE
Décimal	`123`
Binaire	`0b1111011`
Octal	`0173`
Hexadécimal	`0x7B`

Les suffixes « U » et « L » pris en charge par les versions antérieures du logiciel ne sont pas utilisés.

CONSTANTES DE CARACTERES

Les constantes de caractères représentent un caractère ASCII unique, codé dans les 8 bits inférieurs d'un nombre signé à 32 bits. Une constante de caractères comprend un caractère unique compris entre des guillemets simples. Ainsi, vous pouvez utiliser **`'A'`** pour représenter une valeur de 65. Vous pouvez coder certains caractères impossibles à imprimer ou à représenter à l'aide de ce qu'on appelle les séquences d'échappement et chacune d'entre elles est présentée avec une barre oblique inverse unique…

SEQUENCE	VALEUR	ASCII
`\a`	Hex 0x07, Décimal 7	BEL
`\t`	Hex 0x09, Décimal 9	TAB
`\n`	Hex 0x0A, Décimal 10	LF
`\f`	Hex 0x0C, Décimal 12	FF
`\r`	Hex 0x0D, Décimal 13	CR
`\e`	Hex 0x1B, Décimal 27	ESC
`\x`***nnn***	La valeur hexadécimale représentée par ***nnn***.	-
`\`***nnn***	La valeur octale représentée par ***nnn***.	-
`\\`	Un caractère de barre oblique inverse unique.	-
`\'`	Un caractère de guillemet simple unique.	-
`\"`	Un caractère de guillemet double unique.	-

CONSTANTES LOGIQUES

Les constantes logiques représentent une valeur de 1 ou 0 qui est utilisée pour indiquer la vérité ou une expression oui ou non. Un exemple d'élément qui peut être fixé comme étant égal à une constante logique est une étiquette qui représente une sortie numérique dans une API. Vous pouvez saisir des constantes logiques en **`1`** ou **`0`**,ou à l'aide des mots-clés **`true`** ou **`false`**.

CONSTANTES EN VIRGULE FLOTTANTE

Les constantes en virgule flottante représentent une valeur en virgule flottante simple précision à 32 bits. Elles sont représentées (comme vous pouvez vous y attendre) par la partie entière, suivie d'un seule virgule décimale, puis de la partie décimale. La notation exponentielle n'est pas prise en charge.

CONSTANTES DE CHAINES

Les constantes de chaînes représentent des séquences de caractères. Elles comprennent les caractères à représenter, inclus entre des guillemets doubles. Par exemple, la chaîne `"ABCD"` représente une chaîne à quatre caractères, comprenant les valeurs 65, 66, 67 et 68. (En fait, cinq octets sont utilisés pour enregistrer la chaîne, avec une valeur nulle qui est ajoutée pour indiquer la fin de la chaîne.) Vous pouvez également utiliser dans les chaînes les différentes séquences d'échappement décrites ci-dessus.

VALEURS D'ETIQUETTES

La valeur d'une étiquette est représentée dans une expression par le nom d'étiquette. Les caractères majuscules et minuscules sont considérés comme étant équivalents lorsque vous recherchez l'étiquette requise. De plus, une fois que vous avez entré une expression, toutes les modifications apportées au nom de l'étiquette modifient toutes les expressions qui lui font référence. Ainsi, vous n'avez pas besoin de rééditer les expressions pour « réparer » le nom.

REFERENCES DES COMMUNICATIONS

Vous pouvez saisir les références aux registres des périphériques de communication maîtres dans une expression à l'aide d'une syntaxe comprenant un crochet ouvrant, le nom du registre et un crochet fermant. Vous pouvez préfixer le nom d'un périphérique facultatif au nom du registre, puis le séparer par un point. Il est inutile de spécifier le nom du périphérique pour les registres du premier (ou unique) périphérique de la base de données. Vous trouverez ci-dessous des exemples de cette syntaxe…

EXEMPLE	SIGNIFICATION
`[D100]`	Registre `D100` dans le premier périphérique.
`[AB.N7:0]`	Registre `N7:0` dans le périphérique AB.
`[FX.D100]`	Registre `D100` dans le périphérique FX.

MATHEMATIQUES SIMPLES

Comme nous venons de l'expliquer, les expressions contiennent souvent plusieurs valeurs de données et leurs valeurs sont combinées mathématiquement. La plus simple de ces expressions peut ajouter une paire de valeurs alors qu'une expression plus complexe peut obtenir la moyenne de trois valeurs. Ces opérations sont effectuées à l'aide de la syntaxe habituelle que vous pouvez rencontrée dans des applications comme Excel. Les exemples ci-dessous montrent les opérations de base que vous pouvez effectuer…

OPERATEUR	PRIORITE	EXEMPLE
Addition	Groupe 4	`Etiquette1 + Etiquette2`
Soustraction	Groupe 4	`Etiquette1 - Etiquette2`
Multiplication	Groupe 3	`Etiquette1 * Etiquette2`

OPERATEUR	PRIORITE	EXEMPLE
Division	Groupe 3	`Etiquette1 / Etiquette2`
Reste	Groupe 3	`Etiquette1 % Etiquette2`

Même si ces exemples affichent des espaces autour des opérateurs, ils ne sont pas nécessaires.

PRIORITE DES OPERATEURS

Vous avez remarqué la colonne Priorité dans le tableau ci-dessus. Vous vous souvenez sans aucun doute de vos cours d'algèbre : lorsque plusieurs opérateurs sont utilisés en même temps, ils sont évalués selon un ordre défini. Par exemple, la multiplication est toujours évaluée avant l'addition. Crimson implémente cet ordre à l'aide de ce qui est appelé les priorités des opérateurs : chaque opérateur est placé dans un groupe, puis les opérateurs sont appliqués par ordre, du groupe le plus bas au groupe le plus haut. (Sauf lorsque le texte le précise, les opérateurs d'un groupe sont évalués de la gauche vers la droite.) L'ordre par défaut de l'évaluation peut être remplacé à l'aide de parenthèses.

CONVERSION DE TYPE

Normalement, Crimson décide automatiquement lorsqu'il bascule de l'évaluation d'une expression en mathématiques d'entiers à son évaluation à l'aide de la virgule flottante. Par exemple, si vous divisez une valeur entière par une valeur en virgule flottante, l'entier est converti en virgule flottante avant que la division ne soit effectuée. Toutefois, vous pouvez peut-être vouloir forcer une conversion.

Supposez par exemple que vous ajoutez trois entiers qui représentent les niveaux de trois réservoirs, puis que vous divisez le total par le nombre de réservoirs pour obtenir le niveau moyen. Si vous utilisez une expression comme **`(Tank1+Tank2+Tank3)/3`**, votre résultat peut ne pas être aussi précis que vous le souhaitez car la division se produit à l'aide des mathématiques d'entiers et la moyenne ne contient aucune décimale. Pour forcer Crimson à évaluer le résultat à l'aide des mathématiques la virgule flottante, la technique la plus simple consiste à modifier le `3` en `3.0`, ce qui force Crimson à convertir la somme en virgule flottante avant que la division ne se produise. Une technique légèrement plus complexe consiste à utiliser une syntaxe comme **`float(Tank1+Tank2+Tank3)/3`**. Elle appelle une « conversion de type » sur le terme entre parenthèses, le convertissant manuellement en une virgule flottante.

Vous pouvez également utiliser les conversions de type pour convertir une valeur entière, en abandonnant peut-être délibérément de la précision d'une valeur intermédiaire avant de l'enregistrer dans un registre de l'API. Par exemple, l'expression **`int(cos(Theta)*100)`** calcule le cosinus d'un angle, multiplie cette valeur par 100 à l'aide des mathématiques de la virgule flottante, puis la convertit en un entier, en abandonnant tous les chiffres derrière la virgule.

COMPARAISON DES VALEURS

Vous voudrez souvent comparer la valeur d'une donnée à une autre, puis prendre votre décision en fonction du résultat. Par exemple, vous pouvez vouloir définir une formule bit

pour signaler lorsqu'un réservoir dépasse une valeur particulière ou vous pouvez vouloir utiliser une instruction **`if`** dans un programme afin d'exécuter du code lorsqu'un moteur atteint sa vitesse souhaitée. Les opérateurs de comparaison suivants sont fournis…

OPERATEUR	PRIORITE	EXEMPLE
Egal à	Groupe 7	`Données == 100`
Différent de	Groupe 7	`Données != 100`
Supérieur à	Groupe 6	`Données >  100`
Supérieur ou égal à	Groupe 6	`Données >= 100`
Inférieur à	Groupe 6	`Données <  100`
Inférieur ou égal à	Groupe 6	`Données <= 100`

Chaque opérateur produit une valeur de 0 ou 1, en fonction de la condition qu'il teste. Vous pouvez utiliser les opérateurs sur des entiers, des valeurs en virgule flottante ou des chaînes de texte. Si vous comparez des chaînes, cette comparaison ne respecte pas la case, c'est-à-dire que « abc » est considérée comme étant égale à « ABC ».

TEST DES BITS

Crimson vous permet de tester la valeur d'un bit dans une valeur de données grâce à l'opérateur de sélection des bits, représenté par un point unique. La partie gauche de l'opérateur doit représenter la valeur dans laquelle vous devez tester le bit et la partie droite une expression indiquant le numéro du bits à tester. La valeur droite doit se situer entre 0 et 31. Le résultat de l'opérateur est égal à 0 ou 1 selon la valeur du bit en question.

OPERATEUR	PRIORITE	EXEMPLE
Sélection de bits	Groupe 1	`Entrée.2`

L'exemple présenté teste le bit 2 (c'est-à-dire le bit avec une valeur de 4) dans l'étiquette spécifiée.

Si vous souhaitez tester un bit égal à zéro, vous pouvez utiliser l'opérateur NON logique …

OPERATEUR	PRIORITE	EXEMPLE
NON logique	Groupe 2	`!Entrée.2`

Cet exemple est égal à 1 si le bit 2 de l'étiquette spécifiée est égal à 0, et vice-versa.

CONDITIONS MULTIPLES

Si vous souhaitez définir une expression qui est vraie si plusieurs conditions sont *toutes* vraies, vous pouvez utiliser l'opérateur ET logique. De la même façon, si vous souhaitez définir une expression qui est vraie si *n'importe laquelle* des conditions est vraie, vous pouvez utiliser l'opérateur OU logique. Les exemples ci-dessous montrent chaque opérateur en cours d'utilisation…

OPERATEUR	PRIORITE	EXEMPLE
ET logique	Groupe 11	`A>10 && B>10`
OU logique	Groupe 12	`A>10 || B>10`

L'opérateur ET logique donne une valeur de 1 si et uniquement si les expressions situées à gauche et à droite sont vraies alors que l'opérateur OU logique donne une valeur de 1 si n'importe laquelle des expressions est vraie. Notez qu'à la différence des opérateurs de manipulation de bits auxquels il est fait référence dans la présente section, les opérateurs logiques cessent d'effectuer l'évaluation une fois qu'ils connaissent la réponse. Ainsi, dans l'exemple ci-dessus de ET logique, le côté droit de l'opérateur est évalué uniquement si A est supérieur à 10 car, si cela n'est pas vrai, le résultat de l'opérateur ET doit déjà être zéro. Même si cette propriété ne fait qu'une petite différence dans les exemples fournis ci-dessus, si les expressions à gauche et à droite appellent un programme ou apportent une modification à la valeur des données, ce comportement doit être pris en compte.

CHOIX DES VALEURS

Parfois, vous voulez faire votre choix parmi deux valeurs (que ce soit des entiers, des valeurs en virgule flottante ou des chaînes) en fonction de la valeur d'une certaine condition. Par exemple, vous voudrez peut-être définir la vitesse d'un moteur pour qu'il soit égal à 500 tours/minute ou 2 000 tours/minute en fonction d'une étiquette de type bit. Vous pouvez effectuer cette opération à l'aide de l'opérateur **`?:`** qui est unique dans le sens où il prend trois arguments, comme l'exemple ci-dessous le décrit…

OPERATEUR	PRIORITE	EXEMPLE
Sélection	Groupe 13	`Rapide ? 2000 : 500`

Cet exemple évalue jusqu'à 2 000 si **`Rapide`** est vraie, sinon jusqu'à 500. L'opérateur peut être considéré comme étant équivalent à la fonction **`IF`** dans des applications comme Microsoft Excel.

MANIPULATION DE BITS

Crimson fournit également des opérateurs qui permettent d'effectuer des opérations qui ne gèrent pas les entiers comme des valeurs numériques, mais plutôt comme des séquences de bits. Ces opérateurs sont connus sous le nom d'opérateurs de manipulation de bits.

ET (AND), OU (OR), OU EXCLUSIF (XOR)

Ces trois opérateurs de manipulation de bits fournissent tous un résultat dans lequel chaque bit est défini pour être égal aux bits correspondants dans les valeurs situées à gauche et à droite de l'opérateur, associées à l'aide d'une table de vérité spécifique…

OPERATEUR	PRIORITE	EXEMPLE
ET de manipulation de bits	Groupe 8	`Données & Masque`
OU de manipulation de bits	Groupe 9	`Données | Masque`
OU EXCLUSIF de manipulation de bits	Groupe 10	`Données ^ Masque`

Le tableau ci-dessous présente les tables de vérité associées…

A	B	A & B	A | B	A ^ B
0	0	0	0	0
0	1	0	1	1
1	0	0	1	1
1	1	1	1	0

OPERATEURS DE DEPLACEMENT

Crimson fournit également des opérateurs permettant de déplacer un entier ***n*** bits vers la gauche ou la droite…

OPERATEUR	PRIORITE	EXEMPLE
Déplacement à gauche	Groupe 5	`Données << 2`
Déplacement à droite	Groupe 5	`Données >> 2`

Chaque exemple décale **`Données`** de deux bits vers la direction spécifiée.

NON DE MANIPULATION DE BITS

Pour finir, Crimson fournit un opérateur NON de manipulation de bits pour inverser le sens des bits dans une valeur…

OPERATEUR	PRIORITE	EXEMPLE
NON de manipulation de bits	Groupe 2	`~Masque`

Cet exemple fournit une valeur dans laquelle chaque bit est égal à l'opposé de sa valeur dans **`Masque`**.

INDEXATION DE TABLEAUX

Vous pouvez sélectionner les éléments d'une étiquette de tableau en faisant suivre le nom du tableau par des crochets qui contiennent une expression d'indexation. Cette expression doit varier de 0 à 1 de moins que le nombre d'éléments du tableau. Par exemple, si vous créez un tableau à 10 éléments, le premier élément est **`Name[0]`** et le dernier **`Name[9]`**.

INDEXATION DE CHAINES

Vous pouvez utiliser les crochets pour sélectionner des caractères dans une chaîne. Par exemple, si vous disposez d'une étiquette appelée Texte, qui contient la chaîne "ABCD", l'expression **`Text[0]`** renvoie une valeur de 65, égale à la valeur ASCII du premier caractère. Les valeurs d'indexation au-delà de l'extrémité de la chaîne renvoient zéro.

AJOUT DE CHAINES

Outre l'ajout de nombre, vous pouvez utiliser l'opérateur d'addition pour concaténer les chaînes. Ainsi, l'expression `"AB"+"CD"` évalue "ABCD". Vous pouvez également utiliser l'opérateur d'addition pour ajouter un entier à une chaîne. De cette façon, un seul caractère égal au code ASCII représenté par l'entier est ajouté aux données de la chaîne.

APPEL DE PROGRAMMES

Vous pouvez appeler les programmes qui renvoient des valeurs dans des expressions en faisant suivre le nom du programme de deux parenthèses. Par exemple, **`Program1()*10`** appelle le programme associé et multiplie la valeur affichée par 10. A l'évidence, vous devez définir le type de retour de **`Program1`** sur un entier ou une virgule flottante pour qu'il soit logique.

UTILISATION DE FONCTIONS

Crimson fournit plusieurs fonctions prédéfinies que vous pouvez utiliser pour accéder aux informations du système ou pour effectuer des opérations mathématiques courantes.

Ces fonctions sont définies en détail dans la section Référence des fonctions. Elles sont appelées à l'aide d'une syntaxe qui est semblable à celle des programmes, avec plusieurs arguments de la fonction qui sont compris entre parenthèses. Par exemple, cos(0) appelle la fonction de cosinus avec un argument de 0, en renvoyant une valeur de +1,0.

SYNTHESE DES PRIORITES

Le tableau ci-dessous présente la priorité de tous les opérateurs définis dans cette section…

GROUPE	OPERATEURS
Groupe 1	`.`
Groupe 2	`! ~`
Groupe 3	`* / %`
Groupe 4	`+ -`

AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

GROUPE	OPERATEURS
Groupe 5	<< >>
Groupe 6	< > <= >=
Groupe 7	== !=
Groupe 8	&
Groupe 9	|
Groupe 10	^
Groupe 11	&&
Groupe 12	||
Groupe 13	?:

Les opérateurs situés dans les groupes aux nombres inférieurs sont appliqués les premiers.

REMARQUES POUR LES UTILISATEURS D'EDICT

Les utilisateurs du logiciel Edict-97 de Red Lion doivent noter ce qui suit…

- Les opérateurs **&&** et **||** arrêtent l'évaluation une fois que le résultat est connu.
- Les seuls types de données disponibles sont la chaîne, l'entier et la virgule flottante.
- L'opérateur ternaire **?**: est désormais pris en charge.

ECRITURES D'ACTIONS

Alors que les expressions permettent de définir des valeurs, les actions permettent de définir ce que vous souhaitez qu'il se produise lorsqu'un déclencheur ou un autre événement a lieu. Comme la plupart des actions d'une base de données concernent les touches sur lesquelles vous appuyez et comme Crimson fournit une méthode simple de définition des actions couramment utilisées à l'aide de la boîte de dialogue décrite dans la section Interface utilisateur, vous pouvez souvent éviter d'écrire des actions « à la main ». Toutefois, les actions sont nécessaires si vous souhaitez utiliser des déclencheurs, écrire des programmes ou utiliser une touche dans le mode Définie par l'utilisateur.

MODIFICATION DE LA PAGE

Pour créer une action qui modifie la page affichée dans le HMI virtuel du maître, utilisez la syntaxe **`GotoPage(Nom)`**, où **`Nom`** est le nom de la page d'affichage en question. La page en cours est alors supprimée et la nouvelle page s'affiche à sa place.

MODIFICATION DE VALEURS NUMERIQUES

Crimson fournit plusieurs façons de modifier des valeurs de données.

ATTRIBUTION SIMPLE

Pour créer une action qui attribue une nouvelle valeur à une étiquette ou à un registre dans un périphérique de communication, utilisez la syntaxe **`Données:=Valeur`**, où **`Données`** représente l'élément de données à modifier et **`Valeur`** la valeur à attribuer. Notez que **`Valeur`** ne doit pas être une valeur de constante, mais qu'elle peut être toute expression valide de même type. Reportez-vous à la section précédente pour obtenir des détails sur la façon d'écrire des expressions. Par exemple, vous pouvez utiliser le code tel que **`[N7:0]:=Tank1+Tank2`** pour ajouter deux niveaux de réservoir et enregistrer la quantité totale directement dans un registre d'API.

ATTRIBUTION COMPOSEE

Pour créer une action qui définit une valeur de données égale à sa valeur actuelle combinée à une autre valeur par le biais de tous les opérateurs définis dans la section précédente, utilisez la syntaxe **`Données`***`op`***`=Valeur`**, où **`Données`** représente l'étiquette à modifier, **`Valeur`** la valeur que l'opérateur utilise et *op* n'importe lequel des opérateurs disponibles. Par exemple, le code **`Tag+=10`** augmente **`Etiquette`** d'une valeur de 10 alors que le code **`Tag*=10`** multiplie la valeur actuelle par 10.

INCREMENTATION ET DECREMENTATION

Pour créer une action qui augmente une valeur de données de 1, utilisez la syntaxe **`Données++`**. Pour créer une action qui diminue une étiquette de 1, utilisez la syntaxe **`Données--`**. Notez que vous pouvez placer les opérateurs **`++`** ou **`--`** avant ou après la valeur de données en question. Dans le premier cas, la valeur de l'expression représentée par **`++Données`** est égale à la valeur de **`Données`** *après* son incrémentation. Dans le second cas, l'expression est égale à la valeur *avant* sa modification.

MODIFICATION DES VALEURS DE BITS

Pour modifier un bit dans une étiquette, utilisez la syntaxe **`Données.Bit:=1`** ou **`Données.Bit:=0`** pour définir ou effacer le bit le cas échéant, où **`Données`** représente l'étiquette en question et **`Bit`** le nombre de bits basés sur zéro. Notez une nouvelle fois que la valeur située à droite de l'opérateur **`:=`** peut être une expression de telle façon qu'un exemple comme **`Données.1:=(Niveau>10)`** peut être utilisé pour définir ou effacer un bit si le niveau d'un réservoir dépasse ou non une valeur prédéfinie.

EXECUTION DE PROGRAMMES

Vous pouvez appeler des programmes dans des actions en faisant suivre le nom du programme de deux parenthèses. Par exemple, **`Program1()`** appelle le programme associé. Le programme s'exécute au premier plan ou en arrière-plan, tel que défini par les propriétés du programme.

UTILISATION DE FONCTIONS

Crimson fournit plusieurs fonctions prédéfinies que vous pouvez utiliser pour effectuer différentes opérations. Ces fonctions sont définies en détail dans la section Référence des fonctions. Elles sont appelées à l'aide d'une syntaxe qui est semblable à celle des programmes, avec plusieurs arguments de la fonction qui sont compris entre parenthèses. Par exemple, **`SetLanguage(1)`** définit la langue du maître sur 1.

PRIORITE DES OPERATEURS

Tous les opérateurs d'attribution sont répartis dans le Groupe 14. En d'autres termes, ils sont évalués après tous les autres opérateurs. Ils sont également uniques dans le sens où ils sont groupés de la droite vers la gauche. Ainsi, vous pouvez utiliser du code semblable à **`Tag1:=Tag2:=Tag3:=0`** pour effacer les trois étiquettes en une seule fois.

REMARQUES POUR LES UTILISATEURS D'EDICT

Les utilisateurs du logiciel Edict-97 de Red Lion doivent noter ce qui suit…

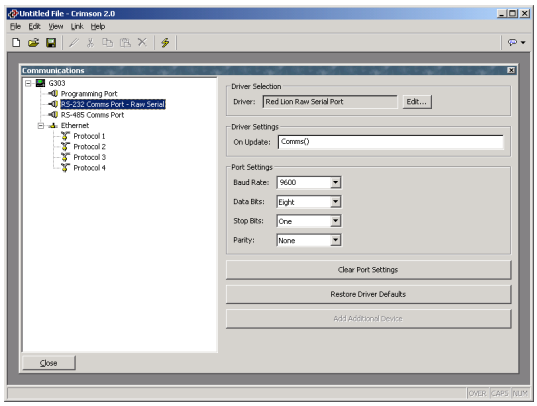
- Vous pouvez désormais utiliser l'opérateur **`=`** au lieu de l'opérateur **`:=`**.

UTILISATION DES PORTS BRUTS

Pour permettre aux clients d'implémenter des protocoles ASCII simples sans demander à Red Lion de développer des pilotes personnalisés, Crimson fournit une nouvelle fonctionnalité grâce à laquelle le langage de programmation du logiciel peut être utilisé pour contrôler directement les ports série ou les sockets réseau TCP/IP. Cette fonctionnalité (connue sous le nom Raw port Access) remplace le pilote et les fonctions utilisées pour prendre en charge la fonction Roll Your Own Protocol d'Edict 97. Elle remplace également le protocole General ASCII Frame en fournissant une fonction qui permet d'effectuer les opérations d'analyse que le pilote a précédemment mises en œuvre. Notez que si vous n'utilisez pas les protocoles ASCII personnalisés, mais plutôt les pilotes standard fournis par Crimson, vous pouvez sauter cette section.

CONFIGURATION D'UN PORT SERIE

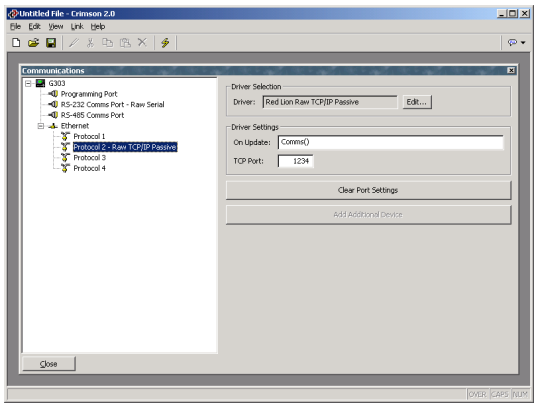
Pour utiliser un port série en mode brut, sélectionnez le pilote Raw Serial Port comme expliqué ci-dessous…



Le débit en bauds du port et les autres paramètres du format de type octet doivent être configurés pour indiquer les paramètres de communication requis et la propriété Sur Rafraîchissement doit être définie pour spécifier le programme qui effectue la communication. Ce programme est appelé en permanence par la tâche de communication du port.

CONFIGURATION D'UN SOCKET TCP/IP

Pour utiliser un socket TCP/IP en mode brut, sélectionnez Pilote TCP/IP Passif tel que décrit ci-dessous…



La propriété Sur Rafraîchissement est configurée comme nous venons de le décrire alors que la propriété Port doit être configurée pour indiquer le port TCP sur lequel vous souhaitez que le pilote effectue ses contrôles. Le pilote accepte des connexions sur ce port, puis appelle le programme Sur Rafraîchissement pour gérer les communications.

LECTURE DE CARACTERES

Pour lire des données à partir d'un port brut, un caractère à la fois, utilisez la fonction **`PortRead`** telle que décrite dans la section Référence des fonctions de ce manuel. Comme avec toutes les fonctions du port brut, l'argument ***`port`*** de cette fonction est calculé en descendant la liste des ports situés dans le volet gauche de la fenêtre Communications, avec le port de programmation qui est le port 1.

L'exemple ci-dessous affiche l'utilisation de **`PortRead`** pour accepter les caractères…

```
int Data;

for(;;) {

       if( (Data := PortRead(2, 100)) >= 0 ) {

              /* Add code to process data */
              }
       }
```

Notez qu'en passant une valeur autre que zéro pour l'argument ***`period`***, le besoin d'appeler la fonction **`Sleep`** est supprimé. Si vous utilisez une valeur zéro pour cet argument, vous devez vous assurer que vous suspendez la tâche des communications à un certain point, sinon vous pouvez empêcher le fonctionnement du système.

LECTURE DE TRAMES COMPLETES

Pour lire une trame complète à partir d'un port brut, utilisez la fonction **PortInput** telle que décrite dans la section Référence des fonctions de ce manuel. Cette fonction vous permet de spécifier les délimiteurs de la trame, la longueur de trame requise et un délai de trame, supprimant de ce fait le besoin d'écrire votre propre machine de réception. Voici un exemple de programme…

```
cstring input;
int     value;

for(;;) {

       input := PortInput(5, 42, 13, 0, 0);

       if( value := TextToInt(input, 10) ) {

              Speed := value;

              PortPrint(5, "Value is ");
              PortPrint(5, IntToText(value,10,5));
              PortPrint(5, "\r\n");
              }
       }
```

Cet exemple écoute sur un socket TCP/IP une trame qui démarre avec un astérisque et se termine par un retour chariot. Il convertit ensuite la trame en une valeur décimale, l'enregistre dans une étiquette d'entiers, puis renvoie la valeur au client.

ENVOI DE DONNEES

Pour envoyer des données sur un port brut, utilisez les fonctions **PortWrite** ou **PortPrint**, telles que décrites dans la section Référence des fonctions de ce manuel. La première fonction envoie un seul octet alors que la seconde envoie une chaîne complète. Pour envoyer des valeurs numériques, utilisez la fonction **IntToText** pour les convertir en chaînes.

REMARQUES POUR LES UTILISATEURS D'EDICT

Les utilisateurs du logiciel Edict-97 de Red Lion doivent noter ce qui suit…

- Le pilote du périphérique Raw Serial Port contrôle les lignes d'établissement de liaison du port. Par conséquent, il est inutile d'appeler **SetRTS**, **HoldTx** ou n'importe quelle autre fonction de gestion des ports. C'est la raison pour laquelle, Crimson ne fournit pas ces fonctions.

- Lors de l'envoi de données, Crimson gère automatiquement les événements de débordement de la mémoire tampon et garantit qu'aucun donnée n'est perdue. Les fonctions **PortWrite** et **PortPrint** ne fournissent donc jamais une valeur affichée et une telle valeur n'est pas requise.

- Pour émuler directement GAF, utilisez un programme qui est semblable à l'exemple ci-dessus, mais enregistrez la chaîne reçue dans une étiquette de chaîne, puis incrémentez une étiquette d'entiers. Cette émulation directe est déconseillée car vous aurez presque toujours un déclencheur pour répondre à la modification dans le numéro de séquence, auquel cas vous pouvez aussi bien gérer cette logique dans le programme de communication.
- La prise en charge améliorée de la programmation par Crimson permet d'atteindre des niveaux de performance supérieurs lors de l'utilisation des pilotes de ports bruts. En règle générale, la performance obtenue est supérieure à celle que permet la configuration Edict équivalente d'un ordre de magnitude ou plus.

AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

REFERENCE DES VARIABLES SYSTEME

Les pages suivantes décrivent les différentes variables système qui existent dans Crimson. Vous pouvez appeler ces variables système dans des actions ou des expressions, telles que décrites dans les deux chapitres précédents.

UTILISATION DES VARIABLES SYSTEME

Les variables système permettent de refléter l'état du système ou de modifier le comportement du système d'une certaine façon. Le premier type de variable est en lecture seule alors que le second peut avoir une valeur qui lui est attribuée.

ACTIVEALARMS

DESCRIPTION

Renvoie le nombre d'alarmes actuellement actives.

TYPE DE VARIABLE

Entier.

TYPE D'ACCES

Lecture seule.

AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

COMMSERROR

DESCRIPTION

Renvoie un masque de bits qui indique si chaque périphérique de communication est hors ligne ou non. Une valeur de 1 dans une position de bits donnée indique que le périphérique correspondant rencontre des erreurs de communication. Le bit 0 (c'est-à-dire le bit avec une valeur de 1) correspond au premier périphérique de communication.

TYPE DE VARIABLE

`Entier`.

TYPE D'ACCES

Lecture seule.

DISPBRIGHTNESS

DESCRIPTION

Renvoie un nombre qui indique la luminosité de l'affichage, de 0 à 100, zéro signifiant désactivé.

TYPE DE VARIABLE

`Entier`.

TYPE D'ACCES

Ecriture/Lecture.

AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

DISPCONTRAST

DESCRIPTION

Renvoie un nombre qui indique la quantité de contraste d'affichage, de 0 à 100.

TYPE DE VARIABLE

`Entier`.

TYPE D'ACCES

Ecriture/Lecture.

DISPCOUNT

DESCRIPTION

Renvoie un nombre qui indique le nombre de mises à jour de l'affichage depuis la dernière réinitialisation.

TYPE DE VARIABLE

Entier.

TYPE D'ACCES

Lecture seule.

AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

DISPUPDATES

DESCRIPTION

Renvoie un nombre qui indique la rapidité de la mise à jour de l'affichage.

TYPE DE VARIABLE

`Entier`.

TYPE D'ACCES

Lecture seule.

PI

DESCRIPTION

Renvoie *pi* comme nombre en virgule flottante.

TYPE DE VARIABLE

`Virgule flottante`.

TYPE D'ACCES

Lecture seule.

AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

REFERENCE DES FONCTIONS

Les pages suivantes décrivent les différentes fonctions standard qui existent dans Crimson. Vous pouvez appeler ces fonctions dans des programmes, tel que décrit dans les chapitres précédents. Les fonctions qui sont marquées comme étant *actives* ne doivent pas être utilisées dans les expressions qui ne sont pas autorisées à modifier les valeurs. Exemple : dans l'expression de contrôle d'une primitive d'affichage. Les fonctions qui sont marquées comme étant *passives* peuvent être utilisées dans n'importe quel contexte.

REMARQUES POUR LES UTILISATEURS D'EDICT

Les utilisateurs du logiciel Edict-97 de Red Lion doivent noter ce qui suit…

- Les différentes fonctions **`Port`** remplacent les fonctions RYOP **`Serial`**.

ABS(*VALEUR*)

ARGUMENT	TYPE	DESCRIPTION
valeur	entier/virgule flottante	La valeur à traiter.

DESCRIPTION

Renvoie la valeur absolue de l'argument. En d'autres termes, si ***valeur*** est une valeur positive, elle est retournée. Si ***valeur*** est une valeur négative, une valeur de la même magnitude, mais avec le signe opposé, est retournée.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

entier ou **virgule flottante**, selon le type de l'argument ***valeur***.

EXEMPLE

```
Erreur := abs(PV – SP)
```


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

ACOS(*VALEUR*)

ARGUMENT	TYPE	DESCRIPTION
valeur	virgule flottante	La valeur à traiter.

DESCRIPTION

Renvoie l'angle ***thêta*** dans les radians de telle façon que **cos(*thêta*)** est égal à ***valeur***.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

virgule flottante.

EXEMPLE

```
thêta := acos(1.0)
```

ASIN(*VALEUR*)

ARGUMENT	TYPE	DESCRIPTION
valeur	virgule flottante	La valeur à traiter.

DESCRIPTION

Renvoie l'angle ***thêta*** dans les radians de telle façon que **sin(*thêta*)** est égal à ***valeur***.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

virgule flottante.

EXEMPLE

```
thêta := asin(1.0)
```


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

ATAN(*VALEUR*)

ARGUMENT	TYPE	DESCRIPTION
valeur	virgule flottante	La valeur à traiter.

DESCRIPTION

Renvoie l'angle ***thêta*** dans les radians de telle façon que **tan(*thêta*)** est égal à ***valeur***.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

virgule flottante.

EXEMPLE

```
thêta := atan(1.0)
```

ATAN2(*A, B*)

ARGUMENT	TYPE	DESCRIPTION
a	virgule flottante	La valeur du côté opposé de l'angle thêta.
b	virgule flottante	La valeur du côté adjacent à l'angle thêta.

DESCRIPTION

Cette fonction est équivalente à **atan(*a*/*b*)**, sauf qu'elle prend également en compte le signe de ***a*** et ***b*** et elle garantit que la valeur affichée est dans le quadrant approprié. Elle peut aussi gérer une valeur nulle pour ***b***, évitant donc l'infini qui en résulterait si la forme de l'argument unique de **tan** était utilisée à la place.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

virgule flottante.

EXEMPLE

```
thêta := atan2(1,1)
```


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

BEEP(*FREQUENCE, PERIODE*)

ARGUMENT	TYPE	DESCRIPTION
fréquence	entier	La fréquence requise dans les demi-tons.
période	entier	La période requise en millisecondes.

DESCRIPTION

Fait retentir le signal sonore du maître sur la période spécifiée et dans le ton spécifié. Une valeur de zéro pour ***période*** désactive le signal sonore. Les demandes de signal sonores ne sont pas mises en file d'attente. Par conséquent, l'appel de la fonction annule immédiatement les appels précédents. Pour les utilisateurs avec un penchant musical, l'argument ***fréquence*** est calibré en demi-tons. Sur une « note » plus sérieuse, la fonction du signal sonore peut s'avérer être un outil de déboguage précieux car elle fournit une méthode asynchrone de signalement de la gestion d'un événement ou d'exécution de l'étape d'un programme.

TYPE DE FONCTION

Cette fonction est active.

TYPE DE RETOUR

Cette fonction ne renvoie aucune valeur.

EXEMPLE

```
Beep(60, 100)
```

CLEAREVENTS()

ARGUMENT	TYPE	DESCRIPTION
aucun		

DESCRIPTION

Efface la liste des événements qui sont affichés dans le journal d'événements.

TYPE DE FONCTION

Cette fonction est active.

TYPE DE RETOUR

Cette fonction ne renvoie aucune valeur.

EXEMPLE

```
ClearEvents()
```

CLOSEFILE(*FICHIER*)

ARGUMENT	TYPE	DESCRIPTION
fichier	entier	Descripteur de fichier tel que renvoyé par OpenFile.

DESCRIPTION

Ferme un fichier qui a déjà été ouvert dans un appel à **FileOpen().**

TYPE DE FONCTION

Cette fonction est active.

TYPE DE RETOUR

Cette fonction ne renvoie aucune valeur.

EXEMPLE

```
CloseFile(Fichierh)
```

COMPACTFLASHEJECT()

ARGUMENT	TYPE	DESCRIPTION
aucun		

DESCRIPTION

Interrompt tout accès de la carte CompactFlash, ce qui permet de la retirer en tout sécurité.

TYPE DE FONCTION

Cette fonction est active.

TYPE DE RETOUR

Cette fonction ne renvoie aucune valeur.

EXEMPLE

```
CompactFlashEject()
```


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

COMPACTFLASHSTATUS()

ARGUMENT	TYPE	DESCRIPTION
aucun		

DESCRIPTION

Renvoie l'état actuel de l'emplacement de la carte CompactFlash sous forme d'entier.

VALEUR	ETAT	DESCRIPTION
0	Vide	Aucune carte n'est installée ou la carte a été éjectée via un appel à la fonction **CompactFlashEject**.
1	Invalide	La carte est endommagée, formatée de façon incorrecte ou pas formatée du tout. Gardez à l'esprit que seul FAT16 est pris en charge.
2	Examen	Le G3 vérifie l'état de la carte. Cet état de produit lorsqu'une carte est insérée pour la première fois dans le G3.
3	Formatage	Le G3 formate la carte. Cet état de produit lorsqu'une opération de formatage est demandée par le PC de programmation.
4	Verrouillée	L'interface d'opérateur écrit sur la carte ou la carte est montée et Windows y accède.
5	Montée	Une carte valide est installée, mais n'est pas verrouillée par l'interface d'opérateur ou Windows.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

entier.

EXEMPLE

```
d := CompactFlashStatus()
```

CONTROLDEVICE(*PERIPHERIQUE,ACTIVER*)

ARGUMENT	TYPE	DESCRIPTION
périphérique	entier	Périphérique à activer ou désactiver.
activer	entier	Détermine si le périphérique est activé ou désactivé.

DESCRIPTION

Permet à la base de données de désactiver ou d'activer un périphérique de communication donné. Le numéro à insérer dans l'argument ***périphérique*** pour identifier le périphérique s'affiche dans la barre d'état de la catégorie Communications lorsque le nom du périphérique est mis en surbrillance.

TYPE DE FONCTION

Cette fonction est active.

TYPE DE RETOUR

Cette fonction ne renvoie aucune valeur.

EXEMPLE

```
ControlDevice(1, vrai)
```

COPY(*DESTINATION*, *SOURCE*, *NOMBRE*)

ARGUMENT	TYPE	DESCRIPTION
destination	entier/virgule flottante	Le premier élément du tableau vers lequel la copie est effectuée.
source	entier/virgule flottante	Le premier élément du tableau à partir duquel la copie est effectuée.
nombre	entier	Le nombre d'éléments à traiter.

DESCRIPTION

Copie ***nombre*** d'éléments du tableau depuis ***source*** vers ***destination***.

TYPE DE FONCTION

Cette fonction est active.

TYPE DE RETOUR

Cette fonction ne renvoie aucune valeur.

EXEMPLE

```
Copy(Enregistrer[0], Travail[0], 100)
```

COS(*THÊTA*)

ARGUMENT	TYPE	DESCRIPTION
thêta	virgule flottante	L'angle, en radians, à traiter.

DESCRIPTION

Renvoie le cosinus de l'angle ***thêta***.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

virgule flottante.

EXEMPLE

```
xp := rayon*cos(thêta)
```


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

CREATEDIRECTORY(*NOM*)

ARGUMENT	TYPE	DESCRIPTION
nom	cstring	Le répertoire à créer.

DESCRIPTION

Crée un nouveau répertoire sur la carte CompactFlash. Notez que le système de remplissage utilisé sur la carte ne prend pas en charge les longs noms de fichiers et que si des barres obliques inverses (backslash) sont ajoutées dans le chemin d'accès pour séparer les éléments du chemin, elles doivent être doubles conformément aux règles de Crimson sur les constantes de chaînes, telles que décrites dans le chapitre Ecriture d'expressions. Pour éviter cette complication, des barres obliques (slash) peuvent être utilisées à la place des barres obliques inverses (backslash). Par conséquent, il est inutile de les doubler. La fonction renvoie la valeur 1 si elle a réussi et la valeur 0 si elle a échoué.

TYPE DE FONCTION

Cette fonction est active.

TYPE DE RETOUR

entier.

EXEMPLE

```
Résultat := CreateDirectory("/LOGS/LOG1")
```

CREATEFILE(*NOM*)

ARGUMENT	TYPE	DESCRIPTION
nom	cstring	Le fichier à créer.

DESCRIPTION

Crée un fichier vide sur la carte CompactFlash. Notez que le système de remplissage utilisé sur la carte ne prend pas en charge les longs noms de fichiers et que si des barres obliques inverses (backslash) sont ajoutées dans le chemin d'accès pour séparer les éléments du chemin, elles doivent être doubles conformément aux règles de Crimson sur les constantes de chaînes, telles que décrites dans le chapitre Ecriture d'expressions. Pour éviter cette complication, des barres obliques (slash) peuvent être utilisées à la place des barres obliques inverses (backslash). Par conséquent, il est inutile de les doubler. La fonction renvoie une valeur de 1 si elle a réussi et une valeur de 0 si elle a échoué. Notez que le fichier n'est pas ouvert après sa création. Un appel ultérieur doit être effectué sur **`OpenFile()`** pour lire ou écrire des données.

TYPE DE FONCTION

Cette fonction est active.

TYPE DE RETOUR

`entier`.

EXEMPLE

```
Succès := CreateFile("/logs/custom/myfile.txt")
```


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

DATATOTEXT(*DONNEES, LIMITE*)

ARGUMENT	TYPE	DESCRIPTION
données	entier	Le premier élément d'un tableau.
limite	entier	Le nombre d'éléments à traiter.

DESCRIPTION

Forme une chaîne à partir du tableau en prenant chaque élément du tableau comme un caractère ASCII unique.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

cstring.

EXEMPLE

```
chaîne := DataToText(Données[0], 8)
```

DATE(*A, M, J*)

ARGUMENT	TYPE	DESCRIPTION
a	entier	L'année à coder, sous la forme de quatre chiffres.
m	entier	Le mois à coder, de 1 à 12.
j	entier	La date à coder, à partir de 1.

DESCRIPTION

Renvoie une valeur qui représente la date indiquée comme le nombre de secondes écoulées depuis le point de données du 1er janvier 1997. Cette valeur peut également être utilisée avec d'autres fonctions de temps/date.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

entier.

EXEMPLE

```
t := Date(2000,12,31)
```


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

DECTOTEXT(*DONNEES, SIGNE, AVANT, APRES, NON SIGNIFICATIF, GROUPE*)

ARGUMENT	TYPE	DESCRIPTION
données	entier/virgule flottante	Données numériques à mettre en forme.
signé	entier	0 – sans signe, 1 – signe facultatif, 2 – signe forcé.
avant	entier	Nombre de chiffres à gauche de la virgule décimale.
après	entier	Nombre de chiffres à droite de la virgule décimale.
non significatif	entier	0 – zéros non significatifs, 1 – zéros significatifs.
groupe	entier	0 – aucun groupement, 1– chiffres de groupe par trois.

DESCRIPTION

Met en forme la valeur dans ***données*** sous forme de valeur décimale selon les autres paramètres. La fonction est généralement utilisée pour générer une option de mise en forme avancée via des programmes ou pour préparer des chaînes à envoyer via un pilote Raw Serial Port.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

cstring.

EXEMPLE

```
Texte := DecToText(var1, 2, 5, 2, 1, 1)
```

DEG2RAD(*THÊTA*)

ARGUMENT	TYPE	DESCRIPTION
thêta	virgule flottante	L'angle à traiter.

DESCRIPTION

Renvoie ***thêta*** converti de degrés en radians.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

virgule flottante.

EXEMPLE

```
Charge := Poids * cos(Deg2Rad(Angle))
```


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

DELETEDIRECTORY(*NOM*)

ARGUMENT	TYPE	DESCRIPTION
nom	cstring	Le répertoire à supprimer.

DESCRIPTION

Supprime un répertoire, ses sous-répertoires et son contenu à partir de la carte CompactFlash. Notez que le système de remplissage utilisé sur la carte ne prend pas en charge les longs noms de fichiers et que si des barres obliques inverses (backslash) sont ajoutées dans le chemin d'accès pour séparer les éléments du chemin, elles doivent être doubles conformément aux règles de Crimson sur les constantes de chaînes, telles que décrites dans le chapitre Ecriture d'expressions. Pour éviter cette complication, des barres obliques (slash) peuvent être utilisées à la place des barres obliques inverses (backslash). Par conséquent, il est inutile de les doubler. La fonction renvoie une valeur de 1 si elle a réussi et une valeur de 0 si elle a échoué.

TYPE DE FONCTION

Cette fonction est active.

TYPE DE RETOUR

entier.

EXEMPLE

```
Succès := DeleteDirectory("/logs/custom")
```

DELETEFILE(*FICHIER*)

ARGUMENT	TYPE	DESCRIPTION
fichier	entier	Descripteur de fichier tel que renvoyé par OpenFile.

DESCRIPTION

Ferme, puis supprime un fichier qui se trouve sur la carte CompactFlash card.

TYPE DE FONCTION

Cette fonction est active.

TYPE DE RETOUR

entier.

EXEMPLE

```
Résultat := DeleteFile(Fichierh)
```


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

DEVCTRL(*PÉRIPHÉRIQUE, FONCTION, DONNÉES*)

ARGUMENT	TYPE	DESCRIPTION
périphérique	entier	L'index du périphérique à contrôler.
fonction	entier	La fonction requise à exécuter.
données	cstring	N'importe quel paramètre de la fonction.

DESCRIPTION

Cette fonction permet d'effectuer une opération spéciale sur un périphérique de communication. Le numéro à insérer dans l'argument ***périphérique*** pour identifier le périphérique s'affiche dans la barre d'état de la catégorie Communications lorsque le nom du périphérique est mis en surbrillance. L'action spécifique à effectuer est indiquée par le paramètre ***fonction*** dont les valeurs dépendent du type de périphérique qui est adressé. Le paramètre ***données*** peut être utilisé pour passer d'autres informations au pilote. La plupart des lecteurs ne prennent pas en charge cette fonction. Lorsqu'elles sont prises en charge, les opérations sont spécifiques au pilote et décrites de façon séparées.

TYPE DE FONCTION

Cette fonction est active.

TYPE DE RETOUR

entier.

EXEMPLE

Reportez-vous aux remarques relatives aux applications du pilote de communication pour consulter des exemples spécifiques.

DISABLEDEVICE(*PÆRIPHÆRIQUE*)

ARGUMENT	TYPE	DESCRIPTION
périphérique	entier	Le périphérique à désactiver.

DESCRIPTION

Désactive les communications du périphérique spécifié. Le numéro à insérer dans l'argument ***périphérique*** pour identifier le périphérique s'affiche dans la barre d'état de la catégorie Communications lorsque le nom du périphérique est mis en surbrillance.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

Cette fonction ne renvoie aucune valeur.

EXEMPLE

```
DisableDevice(1)
```


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

DISPOFF()

ARGUMENT	TYPE	DESCRIPTION
aucun	virgule flottante	Désactive le rétro-éclairage de l'écran.

DESCRIPTION

Désactive le rétro-éclairage de l'écran.

TYPE DE FONCTION

Cette fonction est active.

TYPE DE RETOUR

Cette fonction ne renvoie aucune valeur.

EXEMPLE

```
DispOff()
```

DISPON()

ARGUMENT	TYPE	DESCRIPTION
aucun		Active le rétro-éclairage de l'écran.

DESCRIPTION

Active le rétro-éclairage de l'écran.

TYPE DE FONCTION

Cette fonction est active.

TYPE DE RETOUR

Cette fonction ne renvoie aucune valeur.

EXEMPLE

```
DispOn()
```


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

DRVCTRL(*PORT, FONCTION, DONNEES*)

ARGUMENT	TYPE	DESCRIPTION
port	entier	L'index du pilote à contrôler.
fonction	entier	La fonction requise à exécuter.
données	cstring	N'importe quel paramètre de la fonction.

DESCRIPTION

Cette fonction permet d'effectuer une opération spéciale sur un pilote de communication. Le numéro à ajouter à l'argument ***port*** pour identifier le pilote est le numéro de port auquel le pilote est lié. L'action spécifique à effectuer est indiquée par le paramètre ***fonction*** dont les valeurs dépendent du pilote lui-même. Le paramètre ***données*** peut être utilisé pour passer d'autres informations au pilote. La plupart des pilotes ne prennent pas en charge cette fonction. Lorsqu'elles sont prises en charge, les opérations sont propres au pilote et elles décrites séparément..

TYPE DE FONCTION

Cette fonction est active.

TYPE DE RETOUR

entier.

EXEMPLE

Reportez-vous aux remarques relatives aux applications du pilote de communication pour consulter des exemples spécifiques.

ENABLEDEVICE(*PERIPHERIQUE*)

ARGUMENT	TYPE	DESCRIPTION
périphérique	entier	Le périphérique à activer.

DESCRIPTION

Active les communications du périphérique spécifié. Le numéro à insérer dans l'argument ***périphérique*** pour identifier le périphérique s'affiche dans la barre d'état de la catégorie Communications lorsque le nom du périphérique est mis en surbrillance.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

Cette fonction ne renvoie aucune valeur.

EXEMPLE

```
EnableDevice(1)
```


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

EXP(*VALEUR*)

ARGUMENT	TYPE	DESCRIPTION
valeur	virgule flottante	La valeur à traiter.

DESCRIPTION

Renvoie *e* (2,7183) majoré de la puissance de ***valeur***.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

virgule flottante.

EXEMPLE

```
Variable2 := exp(1.609)
```

EXP10(*VALEUR*)

ARGUMENT	TYPE	DESCRIPTION
valeur	virgule flottante	La valeur à traiter.

DESCRIPTION

Renvoie 10 majoré de la puissance de ***valeur***.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

virgule flottante.

EXEMPLE

```
Variable4 := exp10(0.699)
```


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

FILL(*ELEMENT, DONNEES, NOMBRE*)

ARGUMENT	TYPE	DESCRIPTION
élément	entier/virgule flottante	Le premier élément du tableau à traiter.
données	entier/virgule flottante	La valeur des données à écrire.
nombre	entier	Le nombre d'éléments à traiter.

DESCRIPTION

Définit ***nombre*** d'éléments du tableau à partir d'***élément*** pour être égal à ***données***.

TYPE DE FONCTION

Cette fonction est active.

TYPE DE RETOUR

Cette fonction ne renvoie aucune valeur.

EXEMPLE

```
Fill(Liste[0], 0, 100)
```

FIND(*CHAINE, CARACTERE, SAUTER*)

ARGUMENT	TYPE	DESCRIPTION
chaîne	cstring	La chaîne à traiter.
caractère	entier	Le caractère à rechercher.
sauter	entier	Le nombre de fois que le caractère est sauté.

DESCRIPTION

Renvoie la position de ***caractère*** dans ***chaîne***, en prenant en compte le nombre d'occurrences ***sauter*** spécifiées. La première position comptée est 0. Renvoie -1 si ***caractère*** est introuvable. Dans l'exemple ci-dessous, la position de « : », sautant la première occurrence, est 7.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

entier

EXEMPLE

```
Position := Find("one:two:three",':',1)
```


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

FINDFILEFIRST(*REPERTOIRE*)

ARGUMENT	TYPE	DESCRIPTION
répertoire	cstring	Le répertoire utilisé lors des recherches.

DESCRIPTION

Renvoie le nom de fichier du nom du premier fichier ou répertoire qui se trouve dans le répertoire ***répertoire*** de la carte CompactFlash. Renvoie une chaîne vide si aucun fichier ou aucune carte n'est présent(e). Cette fonction peut être utilisée avec les fonctions **FindFileNext** pour analyser tous les fichiers d'un répertoire donné.

TYPE DE FONCTION

Cette fonction est active.

TYPE DE RETOUR

cstring.

EXEMPLE

```
Nom := FindFileFirst("/LOGS/LOG1")
```

FINDFILENEXT()

ARGUMENT	TYPE	DESCRIPTION
aucun		

DESCRIPTION

Renvoie le nom de fichier du fichier ou du répertoire suivant dans le répertoire spécifié dans un appel précédent à la fonction **FindFileFirst**. Renvoie une chaîne vide si plus aucun fichier n'est présent. Cette fonction peut être utilisée avec les fonctions **FindFileFirst** pour analyser tous les fichiers d'un répertoire donné.

TYPE DE FONCTION

Cette fonction est active.

TYPE DE RETOUR

cstring.

EXEMPLE

```
Nom := FindFileNext()
```


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

FORMATCOMPACTFLASH()

ARGUMENT	TYPE	DESCRIPTION
aucun		

DESCRIPTION

Formate la carte CompactFlash du maître et supprime toutes les données de la carte. Assurez-vous par conséquent que l'utilisateur a été correctement averti avant d'appeler cette fonction.

TYPE DE FONCTION

Cette fonction est active.

TYPE DE RETOUR

Cette fonction ne renvoie aucune valeur.

EXEMPLE

```
FormatCompactFlash()
```

GETDATE (*TEMPS*) ET FAMILLE

ARGUMENT	TYPE	DESCRIPTION
heure	entier	La valeur du temps à décoder.

DESCRIPTION

Chaque membre de cette famille de fonctions renvoie un composant donné d'une valeur de temps/de date, déjà créée par **GetNow**, **Time** ou **Date**. Les fonctions disponibles se présentent comme suit…

FONCTION	DESCRIPTION
GetDate	Renvoie la partie jour du mois de ***temps***.
GetDay	Renvoie la partie jour de la semaine de ***temps***.
GetDays	Renvoie le nombre de jours dans ***temps***.
GetHour	Renvoie la partie heures de ***temps***.
GetMin	Renvoie la partie minutes de ***temps***.
GetMonth	Renvoie la partie mois de ***temps***.
GetSec	Renvoie la partie secondes de ***temps***.
GetWeek	Renvoie la partie semaine de l'année de ***temps***.
GetWeeks	Renvoie le nombre de semaines dans ***temps***.
GetWeekYear	Renvoie la semaine de l'année lors de l'utilisation des numéros de semaine.
GetYear	Renvoie la partie année de ***temps***.

Notez que les fonctions **GetDays** et **GetWeeks** sont généralement utilisées avec la différence entre deux valeurs de temps pour calculer la durée qui s'est écoulée en termes de jours ou de semaines. Notez également que l'année retournée par la fonction ***GetWeekYear*** n'est pas toujours identique à celle qui est retournée par la fonction ***GetYear*** car la première peut renvoyer une valeur inférieure si la dernière semaine d'une année s'étend au-delà de la fin de l'année.

TYPE DE FONCTION

Ces fonctions sont passives.

TYPE DE RETOUR

entier.

EXEMPLE

```
d := GetDate(GetNow() – 12*60*60)
```


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

GETINTERFACESTATUS(*PORT*)

ARGUMENT	TYPE	DESCRIPTION
interface	entier	L'interface à interroger.

DESCRIPTION

Renvoie une chaîne qui indique l'état de l'interface TCP/IP spécifiée. Reportez-vous au chapitre précédent Communications avancées pour en savoir plus sur le calcul de la valeur à placer dans le paramètre ***interface*** ainsi que sur l'interprétation de la valeur affichée.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

cstring.

EXEMPLE

```
EthernentStatus := GetInterfaceStatus(1)
```

GETMONTHDAYS(*A, M*)

ARGUMENT	TYPE	DESCRIPTION
a	entier	L'année à traiter, sous la forme de quatre chiffres.
m	entier	Le mois à traiter, de 1 à 12.

DESCRIPTION

Renvoie le nombre de jours du mois donné, représentant les années bissextiles, etc.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

entier.

EXEMPLE

```
Jours := GetMonthDays(2000, 3)
```


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

GETNETGATE(*PORT*)

ARGUMENT	TYPE	DESCRIPTION
port	entier	L'index du port Ethernet. Doit être égal à zéro.

DESCRIPTION

Renvoie l'adresse IP de la passerelle par défaut du port sous forme de chaîne de texte décimale avec des points.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

cstring.

EXEMPLE

```
passerelle := GetNetGate(0)
```

GETNETID(*PORT*)

ARGUMENT	TYPE	DESCRIPTION
port	entier	L'index du port Ethernet. Doit être égal à zéro.

DESCRIPTION

Renvoie une adresse MAC du port Ethernet sous forme de chaîne de texte de 17 caractères.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

cstring.

EXEMPLE

```
MAC := etNetId(0)
```


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

GETNETIP(*PORT*)

ARGUMENT	TYPE	DESCRIPTION
port	entier	L'index du port Ethernet. Doit être égal à zéro.

DESCRIPTION

Renvoie une adresse IP du port Ethernet sous forme de chaîne de texte décimale avec des points.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

cstring.

EXEMPLE

```
IP := GetNetIp(0)
```

GETNETMASK(*PORT*)

ARGUMENT	TYPE	DESCRIPTION
port	entier	L'index du port Ethernet. Doit être égal à zéro.

DESCRIPTION

Renvoie un masque d'adresse IP du port Ethernet sous forme de chaîne de texte décimale avec des points.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

cstring.

EXEMPLE

```
masque := GetNetMask(0)
```


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

GETNOW()

ARGUMENT	TYPE	DESCRIPTION
aucun		

DESCRIPTION

Renvoie l'heure et la date actuelles sous forme du nombre de secondes écoulées depuis le point de données du 1er janvier 1997. Cette valeur peut alors être utilisée avec d'autres fonctions d'heure/de date.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

entier.

EXEMPLE

```
t := GetNow()
```

GETNOWDATE()

ARGUMENT	TYPE	DESCRIPTION
aucun		

DESCRIPTION

Renvoie le nombre de secondes dans les jours qui se sont écoulés depuis le 1er janvier 1997.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

entier.

EXEMPLE

```
d : = GetNowDate()
```


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

GETNOWTIME()

ARGUMENT	TYPE	DESCRIPTION
aucun		

DESCRIPTION

Renvoie l'heure du jour en termes de secondes.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

entier.

EXEMPLE

```
t := GetNowTime()
```

GETUPDOWNDATA(*DONNÉES, LIMITE*)

ARGUMENT	TYPE	DESCRIPTION
données	entier	Une valeur source qui augmente régulièrement.
limite	entier	Le nombre de valeurs à générer.

DESCRIPTION

Cette fonction prend une valeur qui augmente régulièrement et la convertit en une valeur qui oscille entre 0 et ***limite***–1. Elle est généralement utilisée dans une base de données de démonstration pour générer une animation qui semble réaliste, souvent en passant **DispCount** comme le paramètre ***données*** par défaut pour que la valeur obtenue soit modifiée à chaque mise à jour de l'affichage. Si la fonction **GetUpDownStep** est appelée avec les mêmes arguments, elle renvoie une valeur qui indique la direction de la modification des données retournées par **GetUpDownData**.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

entier.

EXEMPLE

```
Données := GetUpDownData(DispCount, 100)
```


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

GETUPDOWNSTEP(*DONNÉES, LIMITE*)

ARGUMENT	TYPE	DESCRIPTION
données	entier	Une valeur source qui augmente régulièrement.
limite	entier	Le nombre de valeurs à générer.

DESCRIPTION

Voir la fonction **GetUpDownData** pour obtenir une description de cette fonction.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

entier.

EXEMPLE

```
Delta := GetUpDownStep(DispCount, 100)
```

GOTOPAGE(*NOM*)

ARGUMENT	TYPE	DESCRIPTION
nom	Page d'affichage	La page à afficher.

DESCRIPTION

Sélectionne ***nom*** de page à afficher sur l'écran du maître.

TYPE DE FONCTION

Cette fonction est active.

TYPE DE RETOUR

Cette fonction ne renvoie aucune valeur.

EXEMPLE

```
GotoPage(Page1)
```


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

GOTOPREVIOUS()

ARGUMENT	TYPE	DESCRIPTION
aucun		

DESCRIPTION

Entraîne le retour du HMI virtuel sur la page précédente qui était affichée.

TYPE DE FONCTION

Cette fonction est active.

TYPE DE RETOUR

Cette fonction ne renvoie aucune valeur.

EXEMPLE

```
GotoPrevious()
```

HIDEPOPUP()

ARGUMENT	TYPE	DESCRIPTION
aucun		

DESCRIPTION

Masque la popup qui était précédemment affichée à l'aide de **ShowPopup**.

TYPE DE FONCTION

Cette fonction est active.

TYPE DE RETOUR

Cette fonction ne renvoie aucune valeur.

EXEMPLE

```
HidePopup()
```


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

INTTOTEXT(*DONNEES, BASE, NOMBRE*)

ARGUMENT	TYPE	DESCRIPTION
données	entier	La valeur à traiter.
base	entier	La base numérique à utiliser.
nombre	entier	Le nombre de chiffres à générer.

DESCRIPTION

Renvoie la chaîne obtenue en mettant en forme ***données*** dans ***base***, créant ***nombre*** de chiffres. En principe, la valeur ne doit contenir aucun signe. Par conséquent, si une valeur avec signe est requise, utilisez **Sgn** pour décider d'ajouter un préfixe au signe négatif, puis utilisez **Abs** pour passer la valeur absolue à **IntToText**.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

cstring.

EXEMPLE

```
PortPrint(1, IntToText(Valeur, 10, 4))
```

ISDEVICEONLINE(*PERIPHERIQUE*)

ARGUMENT	TYPE	DESCRIPTION
périphérique	entier	Signale si le périphérique est en ligne.

DESCRIPTION

Signale si le périphérique est en ligne ou pas. Un périphérique est indiqué hors ligne si une erreur de communication de séquence répétée s'est produite. Lorsqu'un périphérique est en l'état hors ligne, il est régulièrement interrogé pour vérifier s'il est à nouveau en ligne.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

entier.

EXEMPLE

```
Okay := IsDeviceOnline(1)
```


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

LEFT(*CHAINE, NOMBRE*)

ARGUMENT	TYPE	DESCRIPTION
chaîne	cstring	La chaîne à traiter.
nombre	entier	Le nombre de caractères à retourner.

DESCRIPTION

Renvoie le premier ***nombre*** de caractères à partir de ***chaîne***.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

cstring.

EXEMPLE

```
AreaCode := Left(Téléphone, 3)
```

LEN(*CHAINE*)

ARGUMENT	TYPE	DESCRIPTION
chaîne	cstring	La chaîne à traiter.

DESCRIPTION

Renvoie le nombre de jours dans ***chaîne***.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

entier.

EXEMPLE

```
Taille := Len(Entrée)
```


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

LOG(*VALEUR*)

ARGUMENT	TYPE	DESCRIPTION
valeur	virgule flottante	La valeur à traiter.

DESCRIPTION

Renvoie le journal naturel de ***valeur***.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

virgule flottante.

EXEMPLE

```
Variable1 := log(5.0)
```

LOG10(*VALEUR*)

ARGUMENT	TYPE	DESCRIPTION
valeur	virgule flottante	La valeur à traiter.

DESCRIPTION

Renvoie le journal en base 10 de ***valeur***.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

virgule flottante.

EXEMPLE

```
Variable3 := log10(5.0)
```


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

MAKEFLOAT(*VALEUR*)

ARGUMENT	TYPE	DESCRIPTION
valeur	entier	La valeur à convertir.

DESCRIPTION

Réinterprète l'argument d'entiers comme une valeur en virgule flottante. Cette fonction n'effectue aucune conversion du type, mais prend le motif de bits stocké dans l'argument et suppose que plutôt que de représenter un entier, il représente une valeur en virgule flottante. Il peut être utilisé pour gérer des données à partir d'un périphérique distant pouvant avoir un type de données différent de celui qui est attendu par le pilote de communication.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

virgule flottante.

EXEMPLE

```
fp := MakeFloat(n);
```

MAKEINT(*VALEUR*)

ARGUMENT	TYPE	DESCRIPTION
valeur	virgule flottante	La valeur à convertir.

DESCRIPTION

Réinterprète l'argument en virgule flottante comme un entier. Cette fonction n'effectue aucune conversion du type, mais prend le motif de bits stocké dans l'argument et suppose que plutôt que de représenter une valeur en virgule flottante, il représente un entier. Il peut être utilisé pour gérer des données à partir d'un périphérique distant qui peut en effet avoir un type de données différent de celui qui est attendu par le pilote de communication.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

entier.

EXEMPLE

```
n := MakeInt(fp);
```


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

MAX(*A, B*)

ARGUMENT	TYPE	DESCRIPTION
a	entier/virgule flottante	La première valeur à comparer.
b	entier/virgule flottante	La seconde valeur à comparer.

DESCRIPTION

Renvoie le plus grand des deux arguments.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

entier ou **virgule flottante**, selon le type des arguments.

EXEMPLE

```
Plus grand := Max(Réservoir1, Réservoir2)
```

MEAN(*ELEMENT, NOMBRE*)

ARGUMENT	TYPE	DESCRIPTION
élément	entier/virgule flottante	Le premier élément du tableau à traiter.
nombre	entier	Le nombre d'éléments à traiter.

DESCRIPTION

Renvoie l'élément moyen du ***nombre*** d'éléments du tableau à partir d'***élément***.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

virgule flottante.

EXEMPLE

```
Moyenne := Mean(Données[0], 10)
```


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

MID(*CHAINE, POSITION, NOMBRE*)

ARGUMENT	TYPE	DESCRIPTION
chaîne	cstring	La chaîne à traiter.
position	entier	La position de démarrage.
nombre	entier	Le nombre de caractères à retourner.

DESCRIPTION

Renvoie ***nombre*** de caractères à partir de position ***position*** dans ***chaîne***, où 0 est la première position.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

cstring.

EXEMPLE

```
Echange := Mid(Téléphone, 3, 3)
```

MIN(*A, B*)

ARGUMENT	TYPE	DESCRIPTION
a	entier/virgule flottante	La première valeur à comparer.
b	entier/virgule flottante	La seconde valeur à comparer.

DESCRIPTION

Renvoie le plus petit des deux arguments.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

entier ou **virgule flottante**, selon le type des arguments.

EXEMPLE

```
Plus petit := Min(Réservoir1, Réservoir2)
```


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

MULDIV(*A, B, C*)

ARGUMENT	TYPE	DESCRIPTION
a	entier	Première valeur.
b	entier	Deuxième valeur.
c	entier	Troisième valeur.

DESCRIPTION

Renvoie **a*b/c**. Le calcul intermédiaire est effectué avec des entiers à 64 bits pour éviter les débordements.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

entier.

EXEMPLE

```
d := MulDiv(a, b, c)
```

NOP()

ARGUMENT	TYPE	DESCRIPTION
aucun		

DESCRIPTION

Cette fonction ne fait rien.

TYPE DE FONCTION

Cette fonction est active.

TYPE DE RETOUR

Cette fonction ne renvoie aucune valeur.

EXEMPLE

```
Nop()
```


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

OPENFILE(*NOM, MODE*)

ARGUMENT	TYPE	DESCRIPTION
nom	cstring	Le fichier à ouvrir.
mode	entier	Le mode d'ouverture du fichier… 0 = Lecture seule 1 = Lecture/Ecriture au début du fichier 2 = Lecture/Ecriture à la fin du fichier

DESCRIPTION

Renvoie un descripteur vers le fichier ***nom*** situé sur la carte CompactFlash. Cette fonction est restreinte à un maximum de quatre fichiers ouverts, à tout moment. La carte CompactFlash ne peut pas être démontée lorsqu'un fichier est ouvert. Notez que le système de remplissage utilisé sur la carte ne gère pas les longs noms de fichiers. Si des barres obliques inverses (backslash) sont ajoutées dans le chemin d'accès pour séparer les éléments du chemin, elles doivent être doubles conformément aux règles de Crimson sur les constantes de chaînes, telles que décrites dans le chapitre Ecriture d'expressions. Pour éviter cette complication, des barres obliques (slash) peuvent être utilisées à la place des barres obliques inverses (backslash). Par conséquent, il est inutile de les doubler. Notez également que cette fonction ne crée pas de fichiers qui n'existent déjà. Pour cela, appelez **`CreateFile()`** avant d'appeler cette fonction.

TYPE DE FONCTION

Cette fonction est active.

TYPE DE RETOUR

`entier`.

EXEMPLE

```
Fichierh := OpenFile("/LOGS/LOG1/01010101.csv", 0)
```

PI()

ARGUMENT	TYPE	DESCRIPTION
aucun		

DESCRIPTION

Renvoie *pi* comme nombre en virgule flottante.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

virgule flottante.

EXEMPLE

```
Echelle = Pi()/180
```


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

PLAYRTTTL(*MÉLODIE*)

ARGUMENT	TYPE	DESCRIPTION
mélodie	cstring	La mélodie à jouer en représentation RTTTL.

DESCRIPTION

Joue une mélodie à l'aide du signal sonore interne du maître. L'argument ***tune*** doit contenir la mélodie à jouer au format RTTTL (format utilisé par de nombreux téléphones portables pour les sonneries personnalisées). Des exemples de mélodies sont disponibles sur de nombreux sites Internet.

TYPE DE FONCTION

Cette fonction est active.

TYPE DE RETOUR

Cette fonction ne renvoie aucune valeur.

EXEMPLE

```
PlayRTTTL("TooSexy:d=4,o=5,b=40:16f,16g,16f,16g,16f.,16f,16g,16f,16g,16g#
.,16g#,16g,16g#,16g,16f.,16f,16g,16f,16g,16f.,16f,16g,16f,16g,16f.,16f,16
g,16f,16g,16g#.,16g#,16g,16g#,16g,16f.,16f,16g,16f,16g,32f.")
```

POPDEV(ELEMENT, NOMBRE)

ARGUMENT	TYPE	DESCRIPTION
élément	entier/virgule flottante	Le premier élément du tableau à traiter.
nombre	entier	Le nombre d'éléments à traiter.

DESCRIPTION

Renvoie la déviation standard du ***nombre*** d'éléments du tableau à partir d'***élément***, en supposant que les points de données représentent l'ensemble de la population étudiée. Si vous devez rechercher la déviation standard d'un échantillon, utilisez plutôt la fonction **StdDev**.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

virgule flottante.

EXEMPLE

```
Dev := PopDev(Données[0], 10)
```


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

PORTCLOSE(*PORT*)

ARGUMENT	TYPE	DESCRIPTION
port	entier	Ferme le port spécifié.

DESCRIPTION

Cette fonction est utilisée en association avec les pilotes de port brut TCP actifs ou passifs pour fermer le port sélectionné en fermant gracieusement la connexion qui est attachée au socket associé.

TYPE DE FONCTION

Cette fonction est active.

TYPE DE RETOUR

Cette fonction ne renvoie aucune valeur.

EXEMPLE

```
PortClose(6)
```

PORTINPUT(*PORT*, *DEBUT*, *FIN*, *DELAI*, *LONGUEUR*)

ARGUMENT	TYPE	DESCRIPTION
port	entier	Le Port brut à lire.
début	entier	Le caractère de début pour la correspondance, le cas échéant.
fin	entier	Le caractère de fin pour la correspondance, le cas échéant.
délai	entier	Le délai entre les caractères, exprimé en millisecondes, le cas échéant.
longueur	entier	Le nombre maximum de caractères à lire, le cas échéant.

DESCRIPTION

Lit une chaîne de caractères à partir du ***port*** indiqué par le port à l'aide des autres paramètres pour contrôler le processus d'entrée. Si ***début*** est autre que zéro, le processus commence en attendant que le caractère indiqué par ce paramètre soit reçu. Si ***début*** est zéro, le processus de réception commence immédiatement. Le processus se poursuit alors jusqu'à ce que l'une des conditions suivantes soit remplie…

- ***fin*** est autre que zéro et une ***fin*** de correspondance de caractères est reçue.
- ***délai*** est autre que zéro et cette période passe sans qu'aucun caractère ne soit reçu.
- ***longueur*** est autre que zéro et de nombreux caractères ont été reçus.

La fonction renvoie ensuite les caractères reçus, sans l'octet ***début*** ou ***fin***. Cette fonction est utilisée avec les pilotes de port brut pour mettre en œuvre des protocoles personnalisés à l'aide du langage de programmation de Crimson. Elle remplace la fonctionnalité RYOP qui se trouvait dans Edict.

TYPE DE FONCTION

Cette fonction est active.

TYPE DE RETOUR

cstring.

EXEMPLE

```
Trame := PortInput(1, '*', 13, 100, 200)
```


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

PORTPRINT(*PORT, CHAINE*)

ARGUMENT	TYPE	DESCRIPTION
port	entier	Le port brut sur lequel écrire.
chaîne	cstring	La chaîne de texte à transmettre.

DESCRIPTION

Transmet le texte contenu dans ***chaîne*** au port spécifié par ***port***. Le port doit être configuré pour utiliser un pilote de port brut comme un pilote de Raw Serial Port ou les pilotes Raw TCP/IP. Les données sont transmises et la fonction effectue un renvoi. Le pilote de port gère le protocole de transmission et le contrôle du transmetteur active les lignes de la façon requise.

TYPE DE FONCTION

Cette fonction est active.

TYPE DE RETOUR

Cette fonction ne renvoie aucune valeur.

EXEMPLE

```
PortPrint(1, "ABCD")
```

PORTREAD(*PORT, PERIODE*)

ARGUMENT	TYPE	DESCRIPTION
port	entier	Le port brut à lire.
période	entier	La période d'attente en millisecondes.

DESCRIPTION

Tente de lire un caractère à partir du port spécifié par ***port***. Le port doit être configuré pour utiliser un pilote brut comme un pilote de Raw Serial Port ou les pilotes Raw TCP/IP. Si aucune donnée n'est disponible dans la période de temps spécifiée, une valeur de –1 est retournée. La définition de ***période*** sur zéro entraîne le renvoi des données dans la file d'attente, mais empêche Crimson d'attendre que les données arrivent si aucune n'est disponible.

TYPE DE FONCTION

Cette fonction est active.

TYPE DE RETOUR

entier.

EXEMPLE

```
Données := PortRead(1, 100)
```


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

PORTWRITE(*PORT, DONNEES*)

ARGUMENT	TYPE	DESCRIPTION
port	entier	Le port brut sur lequel écrire.
données	entier	L'octet à transmettre.

DESCRIPTION

Transmet l'octet spécifié par ***données*** sur le port spécifié par ***port***. Le port doit être configuré pour utiliser un pilote brut comme un pilote de Raw Serial Port ou les pilotes Raw TCP/IP. Le caractère est transmis et la fonction effectue un renvoi. Le pilote de port gère le protocole de transmission et le contrôle du transmetteur active les lignes comme cela est requis.

TYPE DE FONCTION

Cette fonction est active.

TYPE DE RETOUR

Cette fonction ne renvoie aucune valeur.

EXEMPLE

```
PortWrite(1, 'A')
```

POSTKEY(*CODE, TRANSITION*)

ARGUMENT	TYPE	DESCRIPTION
code	entier	Code de la touche.
transition	entier	Code de transition.

DESCRIPTION

Ajoute une opération de touche physique à la file d'attente.

TYPE DE FONCTION

Cette fonction est active.

TYPE DE RETOUR

vide

EXEMPLE

```
PostKey(0x80 , 0)
```

CODE	TOUCHE
0x80	Touche d'affichage 1
0x81	Touche d'affichage 2
0x82	Touche d'affichage 3
0x83	Touche d'affichage 4
0x84	Touche d'affichage 5
0x85	Touche d'affichage 6
0x86	Touche d'affichage 7
0x90	Touche de fonction 1
0x91	Touche de fonction 2
0x92	Touche de fonction 3
0x93	Touche de fonction 4

CODE	TOUCHE
0x95	Touche de fonction 6
0x96	Touche de fonction 7
0x97	Touche de fonction 8
0xA0	ALARMES
0xA1	MUET
0x1B	EXIT
0xA2	MENU
0xA3	RAISE
0xA4	LOWER
0x09	SUIV
0x08	PREC

AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

CODE	TOUCHE
0x94	Touche de fonction 5

CODE	TOUCHE
0x0D	ENTER

TRANSITION	FONCTIONNEMENT
0	Afficher la touche du bas, puis la touche du haut
1	Afficher uniquement la touche du bas
2	Afficher uniquement la touche du haut
3	Afficher uniquement la répétition de touche

POWER(*VALEUR*, *PUISSANCE*)

ARGUMENT	TYPE	DESCRIPTION
valeur	entier/virgule flottante	La valeur à traiter.
puissance	entier/virgule flottante	La puissance à laquelle ***valeur*** doit être majorée.

DESCRIPTION

Renvoie ***valeur*** majorée de la puissance ***puissance***-ème.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

entier ou **virgule flottante**, selon le type de l'argument ***valeur***.

EXEMPLE

```
Volume := Power(Longueur, 3)
```


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

RAD2DEG(*THÊTA*)

ARGUMENT	TYPE	DESCRIPTION
thêta	virgule flottante	L'angle à traiter.

DESCRIPTION

Renvoie ***thêta*** converti de radians en degrés.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

virgule flottante.

EXEMPLE

```
Droite := Rad2Deg(Pi()/2)
```

RANDOM(*GAMME*)

ARGUMENT	TYPE	DESCRIPTION
gamme	entier	La gamme des valeurs aléatoires à produire.

DESCRIPTION

Renvoie une valeur pseudo-aléatoire entre 0 et ***gamme*-1**.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

entier.

EXEMPLE

```
Bruit := Random(100)
```


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

READDATA(*DONNEES, NOMBRE*)

ARGUMENT	TYPE	DESCRIPTION
données	tous	Premier élément du tableau à lire.
nombre	entier	Nombre d'éléments à lire.

DESCRIPTION

Demande à ce que ***nombre*** d'éléments des ***données*** d'un élément de tableau soient lus lors de l'analyse suivante des communications. Cette fonction est utilisée avec des tableaux qui ont été mappés sur des données externes et dont la communication est définie sur *Lecture Manuelle*. La fonction effectue un renvoi immédiat et n'attend pas la lecture des données.

TYPE DE FONCTION

Cette fonction est active.

TYPE DE RETOUR

Cette fonction ne renvoie aucune valeur.

EXEMPLE

```
ReadData(tableau1[8], 10)
```

READFILELINE(*FICHIER*)

ARGUMENT	TYPE	DESCRIPTION
fichier	entier	Descripteur de fichier tel que renvoyé par OpenFile.

DESCRIPTION

Renvoie une seule ligne de texte à partir du fichier.

TYPE DE FONCTION

Cette fonction est active.

TYPE DE RETOUR

cstring.

EXEMPLE

```
Texte := ReadFileLine(Fichierh)
```


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

RIGHT(*CHAINE, NOMBRE*)

ARGUMENT	TYPE	DESCRIPTION
chaîne	cstring	La chaîne à traiter.
nombre	entier	Le nombre de caractères à retourner.

DESCRIPTION

Renvoie le dernier ***nombre*** de caractères à partir de ***chaîne***.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

cstring.

EXEMPLE

```
Local := Right(Téléphone, 7)
```

SCALE(*DONNEES, R1, R2, E1, E2*)

ARGUMENT	TYPE	DESCRIPTION
données	entier	La valeur à mettre à l'échelle.
r1	entier	La valeur brute minimale enregistrée dans ***données***..
r2	entier	La valeur brute maximale enregistrée dans ***données***..
e1	entier	La valeur d'ingénierie correspondant à ***r1***.
e2	entier	La valeur d'ingénierie correspondant à ***r2***.

DESCRIPTION

Cette fonction met l'argument ***données*** à l'échelle de façon linéaire, en supposant qu'il contient des valeurs entre ***r1*** et ***r2*** et en produisant une valeur affichée entre ***e1*** et ***e2***. Le calcul interne est implémenté à l'aide d'entiers à 64 bits, évitant de ce fait les débordements qui pourraient se produire si vous avez tenté de mettre à jour de très grandes valeurs à l'aide des opérateurs mathématiques de Crimson.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

entier.

EXEMPLE

```
Données := Scale([D100], 0, 4095, 0, 99999)
```


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

SENDMAIL(*DESTINATAIRE, OBJET, CORPS*)

ARGUMENT	TYPE	DESCRIPTION
destinataire	entier	L'index du destinataire dans le carnet d'adresses de la base de données.
objet	cstring	La ligne d'objet requise dans le message électronique.
corps	cstring	Le corps du texte requis du message électronique.

DESCRIPTION

Envoie un message électronique à partir de l'interface d'opérateur. La fonction effectue un renvoi immédiat, après avoir d'abord ajouté le message électronique requis dans la file d'attente des messages du système. Le message est envoyé à l'aide du transport de messagerie approprié, tel que configuré dans la base de données.

TYPE DE FONCTION

Cette fonction est active.

TYPE DE RETOUR

Cette fonction ne renvoie aucune valeur.

EXEMPLE

```
SendMail(1, "Tester la ligne d'objet", "Tester le corps de texte")
```

SET(*ETIQUETTE, VALEUR*)

ARGUMENT	TYPE	DESCRIPTION
étiquette	Entier ou réel	L'étiquette à modifier.
valeur	Entier ou réel	La valeur à attribuer.

DESCRIPTION

Cette fonction définit l'étiquette spécifiée dans la valeur spécifiée. Elle diffère de l'opérateur d'attribution plus normalement utilisé car elle supprime toutes les écritures mises en file d'attente dans cette étiquette pour les remplacer par une écriture immédiate de la valeur spécifiée. Ainsi, elle est utilisée lorsque le comportement d'écriture normal de Crimson n'est pas requis.

TYPE DE FONCTION

Cette fonction est active.

TYPE DE RETOUR

Cette fonction ne renvoie aucune valeur.

EXEMPLE

```
Set(Tag1, 100)
```


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

SETLANGUAGE(*CODE*)

ARGUMENT	TYPE	DESCRIPTION
code	entier	La langue à sélectionner.

DESCRIPTION

Définit la langue actuelle du maître sur la langue indiquée par ***code***.

TYPE DE FONCTION

Cette fonction est active.

TYPE DE RETOUR

Cette fonction ne renvoie aucune valeur.

EXEMPLE

```
SetLanguage(1)
```

SETNETCONFIG(*PORT, ADRESSE, MASQUE, PASSERELLE*)

ARGUMENT	TYPE	DESCRIPTION
port	entier	L'index du port Ethernet. Doit être égal à zéro.
adresse	entier	L'adresse IP du port requise.
masque	entier	Le masque réseau du port requis.
passerelle	entier	La passerelle par défaut du port requise.

DESCRIPTION

Annule les paramètres de la base de données du port Ethernet. Les différents paramètres IP sont des entiers à 32 bits qui peuvent être formés de façon facultative à partir des chaînes, à l'aide de la fonction **`TextToAddr()`**. Notez que la définition des trois valeurs IP sur zéro réinitialise les paramètres du port sur les paramètres par défaut de la base de données. Notez également que l'unité doit être mise hors tension pour que les nouvelles valeurs soient prises en compte.

TYPE DE FONCTION

Cette fonction est active.

TYPE DE RETOUR

Cette fonction ne renvoie aucune valeur.

EXEMPLE

```
SetNetConfig(0,0,0,0)
```


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

SETNOW(*HEURE*)

ARGUMENT	TYPE	DESCRIPTION
heure	entier	La nouvelle heure à définir.

DESCRIPTION

Définit l'heure actuelle grâce à un entier qui représente le nombre de secondes qui se sont écoulées depuis le 1er janvier 1997. L'entier est en règle générale généré via les autres fonctions d'heure/de date.

TYPE DE FONCTION

Cette fonction est active.

TYPE DE RETOUR

Cette fonction ne renvoie aucune valeur.

EXEMPLE

```
SetNow(252288000)
```

SGN(*VALEUR*)

ARGUMENT	TYPE	DESCRIPTION
valeur	entier/virgule flottante	La valeur à traiter.

DESCRIPTION

Renvoie –1 si ***valeur*** est inférieure à zéro, +1 si elle est supérieure à zéro ou 0 si elle est égale à zéro.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

entier ou **virgule flottante**, selon le type de l'argument ***valeur***.

EXEMPLE

```
Etat := Sgn(Level)+1
```


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

SHOWMENU(*NOM*)

ARGUMENT	TYPE	DESCRIPTION
nom	Page d'affichage	Affiche la page qui apparaît comme menu contextuel.

DESCRIPTION

Affiche la page spécifiée comme menu contextuel. Cette fonction est uniquement disponible avec des unités activées, dotées d'écrans tactiles. Les menus contextuels sont affichés au-dessus de ce qui s'affiche à l'écran et sont alignés à gauche de l'écran.

TYPE DE FONCTION

Cette fonction est active.

TYPE DE RETOUR

Cette fonction ne renvoie aucune valeur.

EXEMPLE

```
ShowMenu(Page2)
```

SHOWPOPUP(*NOM*)

ARGUMENT	TYPE	DESCRIPTION
nom	Page d'affichage	La page à afficher comme popup.

DESCRIPTION

Affiche ***nom*** de page comme popup sur l'écran du maître. Le popup est centrée par rapport à l'écran et s'affiche dans la partie supérieure de la page existante. Le popup peut être supprimée en appelant la fonction **`HidePopup()`**. Il peut également être supprimé de l'écran si une nouvelle page est sélectionnée en appelant la fonction **`GotoPage()`** ou par une action du clavier définie de façon correcte.

TYPE DE FONCTION

Cette fonction est active.

TYPE DE RETOUR

Cette fonction ne renvoie aucune valeur.

EXEMPLE

```
ShowPopup(Popup1)
```


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

SIN(*THÊTA*)

ARGUMENT	TYPE	DESCRIPTION
thêta	virgule flottante	L'angle, en radians, à traiter.

DESCRIPTION

Renvoie le sinus de l'angle ***thêta***.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

virgule flottante.

EXEMPLE

```
yp := rayon*sin(thêta)
```

SLEEP(*PÉRIODE*)

ARGUMENT	TYPE	DESCRIPTION
période	entier	La période de mise en veille, exprimée en millisecondes.

DESCRIPTION

Met en veille la tâche en cours pendant le nombre de millisecondes spécifié. Cette fonction est en principe utilisée dans les programmes qui s'exécutent en arrière-plan ou qui implémentent des communications personnalisées à l'aide de pilotes de port brut. Il est déconseillé de l'appeler en réponse à des déclencheurs ou des touches activées.

TYPE DE FONCTION

Cette fonction est active.

TYPE DE RETOUR

Cette fonction ne renvoie aucune valeur.

EXEMPLE

```
Sleep(100)
```


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

SQRT(*VALEUR*)

ARGUMENT	TYPE	DESCRIPTION
`valeur`	entier/virgule flottante	La valeur à traiter.

DESCRIPTION

Renvoie la racine carrée de ***value***.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

`entier` ou **`virgule flottante`**, selon le type de l'argument ***valeur***.

EXEMPLE

```
Flux := Const * Sqrt(Entrée)
```

STDDEV(*ELEMENT, NOMBRE*)

ARGUMENT	TYPE	DESCRIPTION
élément	entier/virgule flottante	Le premier élément du tableau à traiter.
nombre	entier	Le nombre d'éléments à traiter.

DESCRIPTION

Renvoie la déviation standard du ***nombre*** d'éléments du tableau à partir d'***élément***, en supposant que les points de données représentent un échantillon de la population étudiée. Si vous devez rechercher la déviation standard de l'ensemble de la population, utilisez plutôt la fonction **PopDev**.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

virgule flottante.

EXEMPLE

```
Dev := StdDev(Données[0], 10)
```


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

STOPSYSTEM()

ARGUMENT	TYPE	DESCRIPTION
aucun		

DESCRIPTION

Arrête l'interface d'opérateur pour permettre à l'utilisateur de mettre à jour la base de données. Cette fonction est généralement utilisée lorsqu'une programmation série est nécessaire pour une unité dont le port de programmation a été affecté aux communications. L'appel de cette fonction ferme toutes les communications et permet une fois de plus au port de fonctionner comme un port de programmation.

TYPE DE FONCTION

Cette fonction est active.

TYPE DE RETOUR

Cette fonction ne renvoie aucune valeur.

EXEMPLE

```
StopSystem()
```

STRIP(*TEXTE, CIBLE*)

ARGUMENT	TYPE	DESCRIPTION
texte	cstring	La chaîne à traiter.
cible	entier	Le caractère à supprimer.

DESCRIPTION

Supprime toutes les occurrences d'un caractère donné à partir d'une chaîne de texte.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

cstring.

EXEMPLE

```
Texte := Strip("Mississippi", 's')
```

Le texte contient maintenant « Miiippi ».

AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

SUM(*ELEMENT, NOMBRE*)

ARGUMENT	TYPE	DESCRIPTION
élément	entier/virgule flottante	Le premier élément du tableau à traiter.
nombre	entier	Le nombre d'éléments à traiter.

DESCRIPTION

Renvoie la somme du ***nombre*** d'éléments du tableau à partir d'***élément***.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

entier ou **virgule flottante**, selon le type de l'argument ***valeur***.

EXEMPLE

```
Total := Sum(Données[0], 10)
```

TAN(*THÊTA*)

ARGUMENT	TYPE	DESCRIPTION
thêta	virgule flottante	L'angle, en radians, à traiter.

DESCRIPTION

Renvoie la tangente de l'angle ***thêta***.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

virgule flottante.

EXEMPLE

```
yp := xp * tan(thêta)
```


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

TESTACCESS(*DROITS, INVITE*)

ARGUMENT	TYPE	DESCRIPTION
droits	entier	Les droits d'accès requis.
invite	cstring	L'invite à utiliser dans la popup de connexion.

DESCRIPTION

Renvoie une valeur de **vraie** ou **faux** selon si l'utilisateur actuel dispose des droits d'accès définis par le paramètre ***droits***. Ce paramètre comprend un masque de bits qui représente les différents droits définis pour l'utilisateur, avec le bit 0 (c'est-à-dire le bit avec une valeur de 0x01) qui représente le Droit d'utilisateur 1, le bit 1 (c'est-à-dire le bit avec une valeur de 0x02) qui représente le Droit d'utilisateur 2, et ainsi de suite. Si aucun utilisateur n'est actuellement connecté, le système affiche une popup permettant de demander les informations d'identification de l'utilisateur à l'aide de l'argument ***invite*** pour indiquer la raison de l'affichage de la popup. La fonction est généralement utilisée dans des programmes qui effectuent plusieurs actions pouvant être soumises à la sécurité et interrompues par une popup de déconnexion. En exécutant cette fonction avant d'effectuer les actions, vous pouvez fournir une meilleure indication à l'utilisateur de la raison pour laquelle une connexion est nécessaire et vous pouvez en partie éviter que la sécurité n'échoue par une série d'opérations.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

entier

EXEMPLE

```
if( TestAccess(1, "Effacer toutes les données ?") ) {
       Data1 := 0;
       Data2 := 0;
       Data3 := 0;
       }
```

TEXTTOADDR(*ADRESSE*)

ARGUMENT	TYPE	DESCRIPTION
adresse	cstring	L'adresse sous forme décimale avec des points.

DESCRIPTION

Convertit une chaîne décimale avec des points en adresse IP à 32 bits.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

entier.

EXEMPLE

```
ip := TextToAddr("192.168.0.1")
```


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

TEXTTOFLOAT(*CHAINE*)

ARGUMENT	TYPE	DESCRIPTION
chaîne	cstring	La chaîne à traiter.

DESCRIPTION

Renvoie la valeur de ***chaîne*** en la traitant comme un nombre en virgule flottante. Cette fonction est souvent utilisée avec la fonction **Mid** pour extraire les valeurs des chaînes reçues à partir des Raw Serial Ports. Elle peut également être utilisée pour convertir d'autres valeurs de chaîne en nombres en virgule flottante.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

virgule flottante

EXEMPLE

```
Données := TextToFloat("3.142")
```

TEXTTOINT(*CHAINE*, BASE)

ARGUMENT	TYPE	DESCRIPTION
chaîne	cstring	La chaîne à traiter.
base	entier	La base numérique à utiliser.

DESCRIPTION

Renvoie la valeur de ***chaîne*** en la traitant comme un nombre de ***base***. Cette fonction est souvent utilisée avec la fonction **Mid** pour extraire les valeurs des chaînes reçues à partir des Raw Serial Ports. Elle peut également être utilisée pour convertir d'autres valeurs de chaîne en entiers.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

entier.

EXEMPLE

```
Données := TextToInt("1234", 10)
```


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

TIME(*H, M, S*)

ARGUMENT	TYPE	DESCRIPTION
h	entier	L'heure à coder, de 0 à 23.
m	entier	La minute à coder, de 0 à 59.
s	entier	La seconde à coder, de 0 à 59.

DESCRIPTION

Renvoie une valeur qui représente l'heure indiquée comme le nombre de secondes écoulées depuis minuit. Cette valeur peut alors être utilisée avec d'autres fonctions d'heure/de date. Elle peut également être ajoutée à la valeur générée par **`Date`** pour produire une valeur qui référence une heure et une date particulières.

TYPE DE FONCTION

Cette fonction est passive.

TYPE DE RETOUR

`entier`.

EXEMPLE

```
t := Date(2000,12,31) + Heure(12,30,0)
```

USERLOGOFF()

ARGUMENT	TYPE	DESCRIPTION
aucun		

DESCRIPTION

Entraîne la déconnexion du système de l'utilisateur actuel. Toutes les actions ultérieures qui nécessitent des droits d'accès sécurisés entraînent l'affichage de la popup de connexion permettant de saisir les informations d'identification.

TYPE DE FONCTION

Cette fonction est active.

TYPE DE RETOUR

Cette fonction ne renvoie aucune valeur.

EXEMPLE

```
UserLogOff()
```


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

USERLOGON()

ARGUMENT	TYPE	DESCRIPTION
aucun		

DESCRIPTION

Force l'affichage de la popup de connexion qui permet de saisir les informations d'identification. Normalement, vous ne devez pas utiliser cette fonction car Crimson vous invite à saisir vos informations d'identification lorsque n'importe quelle action qui nécessite des habilitations sécuritaires est effectuée.

TYPE DE FONCTION

Cette fonction est active.

TYPE DE RETOUR

Cette fonction ne renvoie aucune valeur.

EXEMPLE

```
UserLogOn()
```

WAITDATA(*DONNEES, NOMBRE, TEMPS*)

ARGUMENT	TYPE	DESCRIPTION
données	tous	Premier élément du tableau à lire.
nombre	entier	Nombre d'éléments à lire.
temps	entier	La période de délai requise en millisecondes.

DESCRIPTION

Demande à ce que ***nombre*** d'éléments des ***données*** d'un élément de tableau soient lus lors de l'analyse suivante des communications. Cette fonction est utilisée avec des tableaux qui ont été mappés sur des données externes et dont la communication est définie sur *Lecture Manuelle*. A la différence de la fonction **ReadData()**, cette fonction attend la durée spécifiée par le paramètre ***temps*** pour permettre la lecture des données. La valeur affichée est 1 si la lecture s'est terminée dans cette durée, sinon zéro.

TYPE DE FONCTION

Cette fonction est active.

TYPE DE RETOUR

entier.

EXEMPLE

```
état := WaitData(tableau1[8], 10, 1000)
```


AUDIN - 8, avenue de la malle - 51370 Saint Brice Courcelles - France
Tel : 03.26.04.20.21 - Fax : 03.26.04.28.20 - Web : http: www.audin.fr - Email : info@audin.fr

WRITEFILELINE(*FICHIER, TEXTE*)

ARGUMENT	TYPE	DESCRIPTION
fichier	entier	Descripteur de fichier tel que renvoyé par OpenFile.
texte	cstring	Texte à écrire dans le fichier.

DESCRIPTION

écrit une chaîne dans le fichier spécifié et renvoie le nombre d'octets qui ont été correctement écrits, notamment le retour chariot et les caractères de saut de ligne qui sont ajoutés à chaque ligne.

TYPE DE FONCTION

Cette fonction est active.

TYPE DE RETOUR

entier.

EXEMPLE

```
nombre := WriteFileLine(hFile, "Ecriture du texte dans le fichier.")
```