REFERENCE DE SIOC

REFERENCE DE SIOC

Ce guide est une traduction de l'aide du programme SIOC.
Il ne constitue pas en soi un turoriel mais est à utiliser comme une référence des commandes et fonctions pour la programmation de SIOC.
En annexe se trouvent des tableaux récapitulatifs.
(JJ Scohy).

Auteur
Manuel Vélez Campos
e-mail : manolo@arrakis.es

Traductions
Manuel Hernández-Peña M.
José Olliver

Testeurs
Fernando Brea
Pedro Bibiloni
José Olliver
Alberto Beaterio

Web
www.opencockpits.com

I- INTRODUCTION

Le système de simulation SIOC

Le système de simulation SIOC a été conçu pour couvrir les points que le software IOCards ne couvrait pas, recontrant des exigences qui émergent lorsque l'on va au-delà de la simple utilisation électronique des IOCards.

Avec ce software il est possible d'envisager tous types de simulations, définies par l'utilisateur, influençant le fonctionnement de l'électronique et aussi le fonctionnement d'un du simulateur ou des modules associés.

Le système utilise leprotocole IOCP pour envoyer et recevoir l'information, et pour accéder aux variables internes du système.

SIOC est constitué de :

- **Un serveur IOCP** capable de donner les entrées et sorties de n'importe quelle variable définie.

- Un module client IOCP pour accéder aux autres sytèmes (il peut s'agir de Flight Simulator utilisant IOCPserver.dll ou Xplane en utilisant le plugin XPLUIPC).

- un module client FSUIPC pour accéder à d'autres sytèmes (Flight Simulator, Project Magenta, etc).

- Un module d'interconnection aux IOCards, pour contrôler les cartes directement par le système.

- **Un programmeur de scripts** visuels pour rendre tous ces modules largement opérationnels, avec des fonctions spéciales, des formules mathématiques et conditionnelles, des timers, …

Tout le sysème est base sur la survenue d'**événements**.
Un événement pourrait se définir comme un groupe d'opérations qui se déroulent lorsqu'il se produit quelque chose.

Dans notre système, les événements peuvent se précipiter pour plusieurs raisons :

1) un changement d'état ou de valeur d'une variable interne.

2) quand on initie un temporisateur. C'est-à-dire un chronomètre qui à chaque intervalle de temps débute l'événement.

3) lorsqu'il se produit un changement d'état des entrées électroniques.

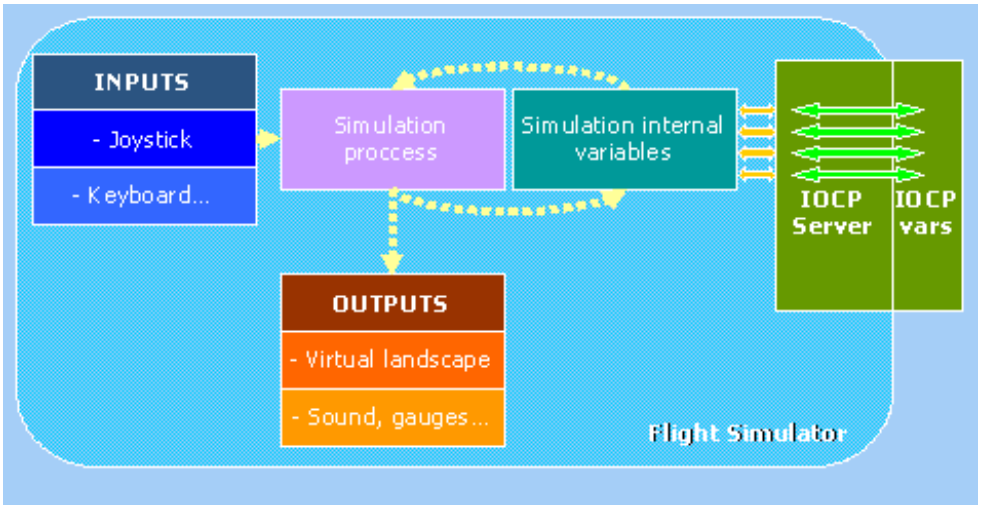
Mais tout se ramène toujours à la même chose : le numéro 1) : un changement de l'état ou la valeur d'une variable interne.

LE PROTOCOLE IOCP

Le protocole IOCP (protocole d'IOCards ou IOCards Protocol) est issu d'une nécessité de communication des cartes avec diférents logiciels sur différentes machines, en utilisant le protocole TCP/IP, rendant possible l'interconnexion dans des environnements locaux mais aussi par Internet.

Ce protocole présente de notables avances sur d'autres protocoles utilisés pour l'interconnexion des modules ou des cartes électroniques au simulateur de vol :

- Il est plus rapide.
- il consomme moins de resources.
- Il est basé sur des événements.
- Il pas le besoin d'applications de type WideFS pour être relié avec d'autres ordinateurs.
- Il est multi-plateforme, pouvant se relier à l'heure à part FSimulator, à X-Plane
- Il est totalement Freeware pour les environnements non-commerciaux.



REFERENCE DE SIOC

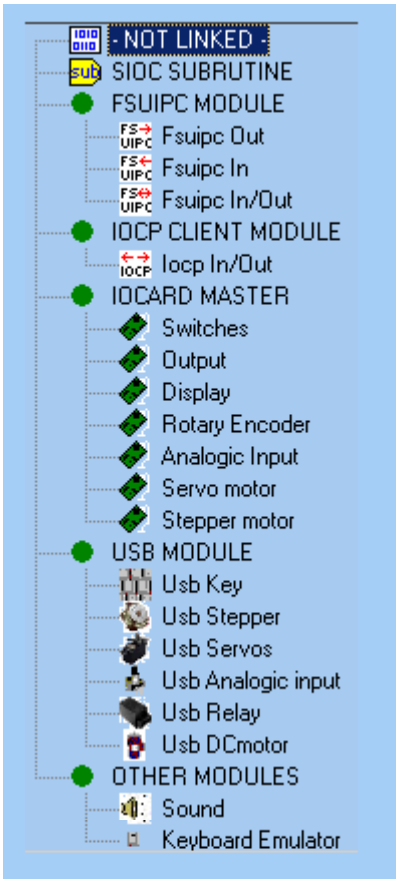
Modules IOCP

Il y a plusieurs modules de base pour lier les variables : le module FSUIPC, le module IOCP, le module d'accès aux OCards et les autres modules externes.

Egalement vous pouvez ne lier une variable à aucun module, en choisissant cette possibilité dans l'option correspondante du menu.
Dans ce cas cette variable fonctionnera avec SIOC sans que ses changements n'affectent aucun module.

La formule adoptée pour l'interaction avec les modules est toujours la même : la variable prend la valeur du résultat d'un événement initié par un module (auquel elle a été liée) et lance un script qui lui est associé, ou au contraire, en modifiant la valeur d'une variable, l'événement est initié et envoie au module l'information qui l' affectera de l'une ou l'autre manière comme nous pourrons le voir plus tard.

Par conséquent, nous différerons parmi différents types de liens : ceux qui envoient ou « sorties » (OUPUTS) , ceux qui reçoivent ou « entrées » (INPUTS), et ceux qui sont « bi-directionnelles » ou « entrées et sorties ».
Voici la relation des modules et des liens disponibles dans SIOC :



REFERENCE DE SIOC

VARIABLES IOCP

La programmation de SIOC est basée sur la définition de ses variables et de ses scritps associés. (événements).
Chacune des variables se distingue par son nombre qui peut aller de 0 à 9999.
De telle manière tout client qui se relie au serveur de SIOC, et a besoin de l'information de la variable #0134, fera référence à la variable que nous aurons définie avec ce nombre.
Chaque modification de la valeur sera aussi annoncée au client.

En outre, un ordre de la modification de la variable de la part du client, lancera l'événement correspondant et exécutera le script associé.

A côté de leur numéro d'identification, les variables peuvent avoir une description et une valeur initiale (facultatives).
Cette valeur sera la première assignée à la variable en commençant le programme.

Le contrôle des modules par SIOC se fait par les variables que nous lions aux modules.
Pour pouvoir accéder aux modules que SIOC gère, les variables ont une caractéristique spéciale : elles peuvent être LIEES à ces modules de telle manière qu'une modification de ces variables provoque une interaction avec le module auquel elles sont liées, ou au contraire, une action d'un des modules provoquera leur modification et donc, le lancement de l'événement associé aux variables.

Dès que nous définissons un certain type de lien, le programme attendra l'introduction des données du module auquel le lien est fait.
Par exemple pour lier une variable au module des commutateurs d'IOCards, automatiquement il faut préciser le numéro d'entrée et le type du commutateur.

Les variables stockent toujours une valeur entière dans la gamme -2147483648 à 2147483647, nous aurons toujours une conversion à faire pour stocker des valeurs décimales.

Par exemple, si nous devons garder une valeur avec 4 décimales, 1,0345, nous stockerons toujours cette valeur multipliée par 10.000, de manière à stocker 10345 pour respecter la nécessité d'utiliser des nombres réels, et nous nous diviserons toujours par 10.000 dans les opérations correspondantes.
Ce sont les caractéristiques du protocole d'IOCP.

VARIABLES INTERNES LOCALES

Lorsqu'on définit un script, il utillise plusieurs variables locales.
Elles sont défnies comme locales parce que leur valeur concerne seulement le script en exécution avec l'idée de stocker des valeurs provisoires.

Nous avons deux types de variables, les vraies et du Booléennés.

Les variables vraies stockent une valeur réelle, c'est-à-dire, des valeurs à décimales ou entières, positives ou négatives allant de 5.0 x 10^-324 à 1.7 x 10^308.
Il existe 3 variables vraies qui sont définies comme L0, L1 et L2.

Les variables Booléennes stockent la valeur d'une condition qui peut être fausse ou vraie.
Elles sont employées fondamentalement pour former des conditions puisqu'à partir de conditions simples dans les commandes conditionnelles on permet seulement une condition.
Il existe 3 variables booléennes qui sont définies comme C0, C1 et C2.

Par exemple, pour définir une condition de type « si L0>5 et L0 <10 », nous ferons une première assignation de type « C0 = L0>5, C1 = L0 <10, C0 = C0 ET C1 ».
De telle manière que C0 soit faux ou vrai selon les conditions précédentes

Par défaut, dans un script les variables vraies prennent la valeur 0, et les Booléennes la valeur fausse.

PROGRAMMES DE LA SUITE SIOC

SIOC comporte deux programmes et deux outils :

config_sioc Ce programme pour éditer et produire des scripts de SIOC avec une interface avancée d'aide à l'utilisateur.
Vous pouvez employer comme paramètre le dossier. ssi à ouvrir au début du programme.

config_sioc <compiled.ssi>

SIOC C'est le programme de gestion de SIOC. Le programme lit le script compilé en .ssi ou .ssc. et l'exécute, utilisant le serveur IOCP aussi bien que les différents modules de communications FSUIPC, IOCP et l'administration des IOCards.

sioc_compiler : ce programme réalise la compilation d'un script SIOC en fichier.
Les paramètres à utiliser sont :

sioc_compiler <text_file> <compiled.ssi> <log>

IOCP_Console Cet outil est fondamental pour pouvoir trier et savoir exactement quelles données sont négociées par le serveur IOCP. On utilise comme paramètres l'adresse IP et le port pour la connection.

IOCP_Console <IP address or name of the server> <port>

II - LE PROGRAMME SIOC.

Menu FICHIERS

New : Prepare un nouveau script vide où insérer variables et lignes de commandes.

Open : ouvre un fichier .ssi.

Save : enregistre le fichier actif dans le format .ssi.

Save as : enregistre le fichier .ssi actif à l aplace d'un autre fichier.

Export to TXT : Génère un fichier .txt (en langage SIOC script) qui pouura être importé plus tard.

Export to ASCII : Génère un fichier .txt (ASCII) qui peut être envoyé par e-mail ou publié dans les forums.

Export Codified : Génère un fichier .ssc, codé en 128 bits et peut être lu seulement par SIOC.

Import TXT : ouvre un fichier .txt (SIOC), générant le script correspondant, et montrant toute erreur trouvée.

Import ASCII : ouvre un fichier .txt (ASCII).

Exit : quitte le programme.

Menu EDIT

Insert Variable : par cette option du menu ou par un clic droit de la souris, nous pouvons insérer un espace consacré à la définition d'une variable juste avant la ligne actuelle. Une fois inséré cet espace, nous pouvons définir la variable par un double clic avec la souris.

New Variable : par cette option du menu ou par un clic droit de la souris, nous pouvons créer un espace consacré à la définition d'une variable juste à l'extrémité de la ligne existante. Une fois créé cet espace, nous pouvons définir la variable faisant par double clic avec la souris.

Insert Command : par cette option du menu ou par clic droit de la souris, nous pouvons insérer un espace dans la définition d'une ligne de commande juste avant la ligne actuelle.
Une fois inséré cet espace, nous pouvons définir la ligne correspondante par double clic avec la souris.
Il est important de choisir une variable où la ligne de commande, ou une ligne de la commande différente du même niveau, doit s'insérer.
Si la ligne choisie est IF ou ELSE, la nouvelle ligne sera créée à un niveau correspondant à ce IF ou ELSE.

New Command : par cette option du menu ou un clic droit de la souris, nous pouvons insérer un espace à la définition d'une ligne de commande à l'extrémité de la ligne de ce niveau.
Une fois inséré cet espace, nous pouvons définir la ligne correspondante par double clic avec la souris.
Il est important de choisir une variable où la ligne de commande, ou une ligne de la commande différente du même niveau, doit s'insérer.
Si la ligne choisie est IF ou ELSE, la nouvelle ligne sera créée à un niveau correspondant à ce IF ou ELSE.

Delete : Cette option éliminera une définition de variable ou une ligne de commande ainsi que le reste entier de lignes qui en dépendent.

Copy : pour copier une définition de variable ainsi que toutes lignes de commande qui en dépendent.

Paste : une fois qu'on a copié une définition de variable, cette option laisse insérera une copie de cette définition excepté que le nom de la variable sera omis, et son numéro sera le premier libre dans le script.

Menu SEE

Expand : développe tous les niveaux du script, montrant toutes les lignes.

Hide : cache toutes les lignes de commande, montrant seulement les définitions de variables.

En cliquant sur les signes +, nous pouvons développer ou cacher n'importe quelle partie du script.

Menu LANGUAGE

Spanish :Change la langue de l'interface en Espagnol.

English : Change la langue de l'interface en Anglais.

Menu COMPILER

Files : Définit un fichier .txt d'entrée et un fichier .ssi de sortie pour le processus d'édition et de compilation.

Run : Cette option exécute la compilation des fichiers définis précédemment. Le bouton COMPILER fait la même chose.

Toute erreur de compilation est montrée à l'écran.

Menu GROUPS

Avec cette option, l'utilisateur peut réunir un groupe de scripts .txt. Après compilation un seul fichier .ssi sera généré.

Toutes les lignes de commandes avec le numéro de variable 0000 seront regroupées en une seule variable 0000. Le reste des variables sera automatiquement renumérotées pour éviter les conflits.

Il faudra veiller à ne pas avoir de variables de même noms dans les différents fichiers du groupe, ou une erreur sera générée durant la compilation.

Files : .txt les fichiers seront définnis dans chauqe ligne de cette fenêtre. Ils doivent se situer dans le dossier de SIOC.

Run : exécute la compilation groupée des fichiers précédemment définis.

Menu UTILS

Ficheros 2.* a versión 3.* Cette option convertit des fichiers .ssi de la version SIOC 2.* à la version 3.*

Menu HELP

Pour accéder à cette aide (en Espagnol ou Anglais, comme spécifié).

About SIOC : Information sur la version actuelle de Config_Sioc version.

III - LE FICHIER SIOC.INI

Dans ce fichier vous pouvez définir les paramètres de SIOC :

Parametres généraux de SIOC :

IOCP_port : port du serveur IOCP.

IOCP_timeout : temps de réponse Max. des paquets IOCP.

Minimized : lorsque ce paramètre est Yes, SIOC se charge en fond. Une icône apparaît dans la barre de tâches.

toggle_delay : délai défini pour les opérations de basculement (utilisé dans plusieurs variables de Project Magenta).

CONFIG_FILE : Script utlisé par SIOC. Le fichier peut avoir les extensions suivantes :

.SSI : fichier compilé prêt à être exécuté.

.SSC : fichier compilé et encrypté.

.TXT : fichier non compilé. Dans ce cas, SIOC compile le script et l'exécute. Si un problème se produit, SIOC affiche un message d'erreur.

.LST : Liste de sources .TXT (un par ligne). SIOC charge le compilateur et unit tous les fichiers en un seul. SIOC exécute le résultat ou affiche une erreur en cas de problème.

Paramètres du module IOCARDS :

IOCard_disable : Si le paramètre est sur Yes, SIOC déconnecte le module IOCards.

On définit les cartes master en utilisant les paramètres :

MASTER=(device index),(Type),(Number of cards),(device number)

Device Index : Index de SIOC pour indiquer la carte utilisée.
Par défaut, vous ne devez pas utiliser le paramètre DEVICE dans le script SIOC si l'index utilisé est 0.

Type: définit le type de dispositif utilisé. choisir :

0 : émulateur de Master Card.
1 : Master Card connectée au port parallèle.
2 : Master Card connectée à un câble compatible.
3 : carte d'expansion port parallèle.
4 : carte d'expansion USB.

Number of cards = Nombre de Master Cards utilisées, 1 à 4 pour une Expansion Card (Lpt or USB), 1 pour une Master card ou Emulator.

REFERENCE DE SIOC

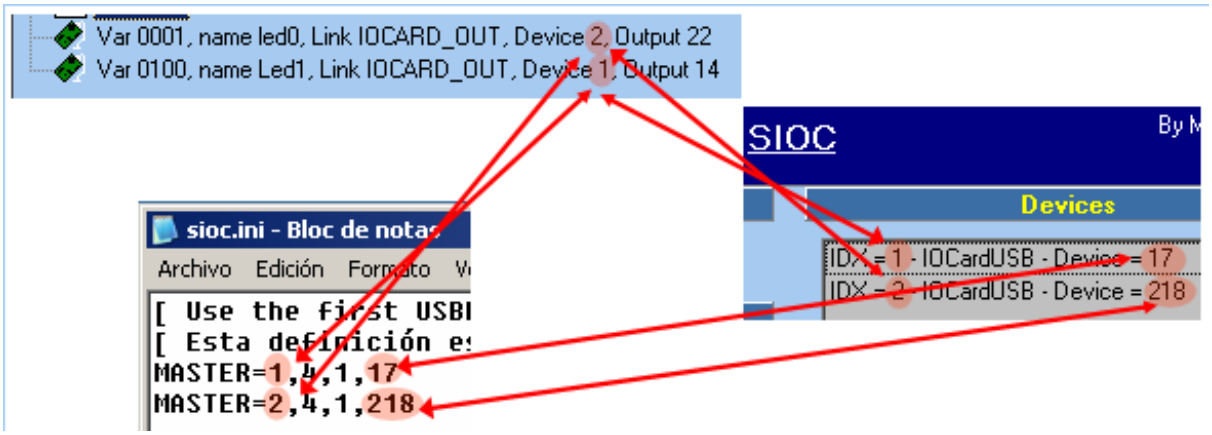
Device number = 0 pour Emulator ou pour utiliser la première carte détectée. Pour le port LPT (parallèle), l'adresse du port (par exemple $0378), pour un USB, le « device number » assigné.

Exemple pour une Master Card connectée au port LPT :
MASTER=0,1,1,$0378

Exemple de 2 USBExpansion connectées avec 3 & 2 Master Cards :
MASTER=0,4,3,22
MASTER=1,4,2,24

Exemple pour definir l'IOCard Emulator :
MASTER=0,0,1,0

Ce diagrame explique comment utiliser le paramètre "device" dans le script, et comment configurer SIOC.INI.



Pour d'autres cartes vous devez utiliser :

Nom de la carte=(Device Index),(Device Number)

USBStepper=(Device Index),(Device Number)
Pour les cartes USBStepper

USBKeys=(Device Index),(Device Number)
Pour les cartes USBKeys

USBServos=(Device Index),(Device Number)
Pour les cartes USBServos

USBRelays=(Device Index),(Device Number)
Pour les cartes USBRelays

USBDCmotor=(Device Index),(Device Number)
Pour les cartes USBDCmotor

Exemple de 2 USBServos ayant l'index 0& 1, et le device numbers 17 & 23
USBServos=0,17
USBServos=1,23

Pour les axes analogiques utilisez les cartes physiques autorisées. Si vous utilisez un axe de la carte USBServos avec le numéro 23, le device number pour USBAnalogicard devra être 23.

USBAnalogic=(Device Index),(Device Number)

Les index sont indépendants pour chaque type d'appareil.

Paramètres du module FSUIPC :

FSUipcdisable Si le paramètre est YES, le module FSUIPC est déconnecté.

FSUipcRefresh C'est le temps de rafraîchissement des entrées et sorties (en millisecondes). Vous pouvez le réduirepour de plus fréquents accès à FSUIPC mais cela signifie plus de charge pour l'ordinateur et un résultat peut être mauvais.

Paramètres des modules clients IOCP :

Sioc a deux modules clients pour le protocole IOCP : ils s'adressen tà un autre serveur IOCP que celui inclus dans SIOC.
Ces modules peuvent être connectés à deux différents serveurs dans deux ordinateurs différents.
Vous pouvez définir le client utilisé par le paramètre DEVICE dans les scripts SIOC.

IOCPini_delay : C'est le délai en millisecondes pour initier une connection du serveur vers les clients lorsque SIOC est initialise.

Ce délai est utilisé pour donner le temps à SIOC d'initaliser toutes les variables.

IOCPclient0_disable : sur Yes, SIOC désactive le module client IOCP #0.

IOCPclient1_disable : sur Yes, SIOC désactive le module client IOCP #1.

IOCPclient0_host : Définit l'adresse du serveur hôte lorsque le module client #0 doit se connecter.

IOCPclient1_host : Définit l'adresse du serveur hôte lorsque le module client #1 doit se connecter.

IOCPclient0_port : Définit le port du serveur lorsque le module client #0 doit se connecter.

IOCPclient1_port : Définit le port du serveur lorsque le module client #1 doit se connecter.

REFERENCE DE SIOC

Paramètres du module de SON :

SIOC a un module son pour différents bruits où vous pouvez commander le volume, la fréquence et la balance.

Sound_disable: Quand le paramètre est Yes, SIOC neutralise ce module.

Volume: Définir la valeur générale du volume de tous les sons utilisés [0-100].
Pour définir tout le son utilisé employer les paramètres suivants :

SOUND=(Wav file),(Frequency),(Volume),(Pan)

Wav file = ce dossier devrait être dans le même répertoire que votre SIOC. Si vous voulez jouer ce son en boucle continue, mettre* avant le nom de fichier.

Frequency = Frequence (100 à 100000, 0=original, -1=Default)

Volumen = Volume (0 à 100 -1= volume par défaut)

Pan = balance (-100 (Left) à +100 (Right) 0=center -1=Default)

Vous pouvez définir les sons requis avec une définition SON avec tous les paramètres.

L'index utilisé avec les scripts de SIOC est l'ordre de la définition de son, la première entrée de définition a l'index #1.

Par exemple :
Sound=*outermk.wav,-1,-1,-1

Paramètres du module EMULATION KEYS :

SIOC a un module d'émulation de touches pour lancer des commandes par touches dans la fenêtre active.
Il est important de s'assurer, si on désire envoyer les commandes à une fenêtre particulière, que c'est la fenêtre active, ou que SIOC est le programme en cours d'utilisation, au moyen de la définition WINDOW qui activera la fenêtre définie dans les paramètres.

WINDOW=(nom de la fenêtre)

Pour définir chaque commande de touche pour l'index utilisé dans les scripts SIOC, utiliser:

#(Index number used by SIOC script)= Key command

Exemple :
#1=\B\A

Vous pouvez définir l'utilisation de touches spéciales et utlliser ces lettres pour des fonctions spécifiques :

A = BKSP = #8
B = TAB = #9
C = ENTER = #13;
D = ESC = #27
E = F1 = #228

F = F2 = #229;
G = F3 = #230
H = F4 = #231
I = F5 = #232;
J = F6 = #233
K = F7 = #234
L = F8 = #235;
M = F9 = #236
N = F10 = #237
O = F11 = #238;
P = F12 = #239
Q = HOME = #240
R = END = #241;
S = UP = #242
T = DOWN = #243
U = LEFT = #244;
V = RIGHT = #245
W = PGUP = #246
X = PGDN = #247;
Y = INS = #248
Z = DEL = #249
1 = SHIFT_DN =#250;
2 = SHIFT_UP = #251
3 = CTRL_DN = #252
4 = CTRL_UP = #253
5 = ALT_DN = #254
6 = ALT_UP = #255 \ = \

Par exemple la séquence SHIFT+S peut être définie comme :
#1=\1S\2

Toutes les cléfs sont en majuscule. Si vous avez besoin des minuscules il faut placer avant le signe **<**

IV - IOC.EXE

SIOC est le programme qui execute les scripts SIOC, charge le serveur IOCP et se connecte aux autres modules.

Son fichier de configuration est sioc.ini et vous pouvez executer certaines operations en employant les boutons suivants :

EMPTY SCRIPT : Déclenche un RELOAD et utilise un fichier vide pour IOCPServer.

RELOAD : Initialise le système et recharge le script défini dans sioc.ini.

CONFIG : Charge le programme config_sioc.exe avecle script défini dans sioc.ini.

IOCPCONSOLE : Charge le programme IOCPconsole.exe en utilisant la definition de nom du script défini dans sioc.ini.

TRAY : SIOC est execute en tâche de fond, et une icône de contrôle est affichée dansl a barre de tâches.

EXIT : Déconnecte tous les modules et quitte le programme.

V - LANGAGE DE SCRIPT SIOC

CARACTERISTIQUES GENERALES

Li'dentifiant de langue, les variables, les constantes et les autres éléments sont écrits entre des espaces, colonnes, parenthèses ou tabulations.

Les différents niveaux sont indiqués par { et }

// peut être inséré à tout endroit pour indiquer un commentaire general ou particulier.

Une seule commande ou definition peut être écrite sur une ligne.

Il n'y a pas de difference entre majuscule et minuscule.

On fait référence à toute variable SIOC en écrivant V suivi de son numéro (par exemple V0001), ou en indiquant & suivi par son nom (par exemple &course).

Nous pouvons utiliser les variables internes avec les noms L0, L1, L2, C0, C1 et C2 (si la commande ou le type l'autorise).

DEFINITION DE VARIABLE

Var numéro Attribut paramètre Attribut paramètre (...)
{

paramètres de commande
paramètres de commande
paramètres de commande
...

}

numéro = entier de 0 à 9999

paramètre = information concernant l'attribut ou la commande.

Exemples:

Var 0001, Link FSUIPC_IN, Iffset $07F2, Length 2

Var 1002, Link IOCARD_ENCODER, Input 0, Aceleration 8, Type 1

Var 0086, Link IOCARD_DISPLAY, Digit 0, Numbers 5

Var 9023

Attributs valides :

Link Type de lien de la variable.
Type définnit des caractéristiques spéciales pour l'élément.
Offset numéro de variable IOCP ou offset FSUIPC.
Name nom symbolique associé à la variable.
Value valeur initiale pour une variable.
Length longueur de l'offset FSUIPC.

Input entrée.
Output sortie.
Digit premier chiffre d'un groupe d'affichage.
Acceleration acceleration d'un encodeur.
PosL position gauche pour la calibration.
PosC position centrale pour la calibration.
PosR position droite pour la calibration.
Device Index du périphérique mentionné par le lien de la variable.

TYPE DE LIEN ASSIGNE

SANS LIEN

Définit la variable, qui est prête à être utilisée.

Attributs :

Name : donne un nom symbolique à une variable.

Value : fixe une valeur initiale.

Exemple:

Var 0001
Var 0006, Name Heading, Value 180

FSUIPC_IN

La variable stockera la valeur de l'offset FSUIPC. Le script associé sera exécuté seulement si la valeur de la variable change.

Attributs :

Name : donne un nom symbolique à une variable.

Value : fixe une valeur initiale.

Offset FSUIPC : Longeur de Variable en bytes. Peut valoir 1, 2, 4 et 8.

Numbers : pour une valeur à 8 bytes de type FLOAT, cet attribut définit le facteur de division. Si nous mettons 1 et que le FLOAT est 12.3, la variable vaudra 123.

Type : en mettant 1, nous forçons la conversion d'un binaire en entier.

Exemple:

Var 0015, Link FSUIPC_IN, Offset $02CC, Length 8, Numbers 1

FSUIPC_OUT

Lien avecle module FSUIPC utilisant l'offset et la longueur.
Lorsque la variable change de valeur, cette nouvelle valeur est envoyée à FSUIPC.

Attributs :

Name : donne un nom symbolique à une variable.

Value : fixe une valeur initiale.

Offset FSUIPC : Longeur de Variable en bytes. Peut valoir 1, 2, 4 et 8.

Numbers : pour une valeur à 8 bytes de type FLOAT, cet attribut définit le facteur de division. Si nous mettons 1 et que le FLOAT est 12.3, la variable vaudra 123.

Type : en mettant 1, nous forçons la conversion d'un binaire en entier.

Exemple:

Var 0015, Link FSUIPC_OUT, Offset $02CC, Length 8, Numbers 1, Value 23

FSUIPC_INOUT

C'est la combination de FSUIPC_IN et FSUIPC_OUT. Les changements d'offsets de FSUIPC sont envoyés à la variable et les changements de variable sont envoyés à FSUIPC.

Name : donne un nom symbolique à une variable.

Value : fixe une valeur initiale.

Offset FSUIPC : Longeur de Variable en bytes. Peut valoir 1, 2, 4 et 8.

Numbers : pour une valeur à 8 bytes de type FLOAT, cet attribut définit le facteur de division. Si nous mettons 1 et que le FLOAT est 12.3, la variable vaudra 123.

Type : en mettant 1, nous forçons la conversion d'un binaire en entier.

Exemple :

Var 0015, Link FSUIPC_INOUT, Offset $02CC, Length 8, Numbers 1, Value 23

IOCP

Nous pouvons nous lier aux modules clients IOCP de SIOC.
Tout changement de la variable de destinatin IOCP est répercuté dans cette variable.

Attributs :

Name : donne un nom symbolique à une variable.

Value : fixe une valeur initiale.

Offset :Numéro de la variable de destination IOCP.

Device: Numéro du module client utilisé. Valeurs 0 ou 1 autorisées. (module client par défaut : 0).

Exemple:

Var 0015, Link IOCP, Offset 48, Device 1, Value 0

SUBRUTINE

Les variables avec ce lien sont converties en sous-routine. Le script associé sera exécuté automatiquement par la commande CALL.

Exemple:

Var 9999, Link SUBRUTINE

IOCARD_SW

Avec ce lien nnous pouvons lire les valeurs d'entrées de la carte Master et les configurer en boutons ou interrupteurs.
Si l'entrée est ON, la variable stockera 1; si l'entrée est OFF, la variable stockera 0.

Attributs :

Name : donne un nom symbolique à la variable.

Input : définit le numéro d'entrée lié à la variable.

Type : en réglant P la variable restera à 1 ou 0 jusqu'à une nouvelle activation de l'entrée correspondante. De cett emanière on peut simuler un interrupteur ou un bouton.

Exemple:

Var 0378, name SW1, Link IOCARD_SW, Input 12

IOCARD_OUT

Avec ce lien nous pouvons accéder aux sorties digitales de la carte Master.
Si la variable vaut 0, la sortie est désactivée; si la valeur est fixée à 1 la sortie est activée.

Attributs :

Name : donne un nom symbolique à la variable.

Output : définit le numéro de sortie lié à la variable.

Exemple:

Var 0398, name OUT1, Link IOCARD_OUT, Output 23

IOCARD_DISPLAY

Avec ce lien nous contrôlons la carte Displays.
La valeur de la variable est envoyée au groupe sélectionné de displays. Si la carte est une Display II, alors des caractères spéciaux peuvent être envoyés.

Les valeurs de variable de ces caractères spéciaux sont :

-999999 = affichage OFF.
-999998 = affichage "-"
-999997 = affichage "6"
-999996 = affichage "t"

-999995 = affichage "d"
-999994 = affichage "_"

Attributs :

Name : donne un nom symbolique à la variable.

Digit : établit le premier affichage du groupe. Le premier est le moins sgnificatif.

Numbers : règle le nombre de chiffres de l'affichage (par exemple, pour le cap ce sera 3). Il faut considérer qu'une indication "-" est un affichage supplémentaire.

Type : pour de futures options. Pas encore implémenté.

Exemple:

Var 0008, name heading, Link IOCARD_DISPLAY, Digit 1, Numbers 3

IOCARD_ENCODER

Avec ce lien nous contrôlons plusieurs types d'encodeurs rotatifs connetcés à la carte Master.
Pour chaque détente, la variable stocke la valeur 1 multipliée par le facteur d'accélération défini par l'attribut Acceleration. Après que le script ait été exécuté la valeur de la variable est remise à 0, donc le script n'est plus exécuté.
La valeur sera négative pour une rotation dans la direction opposée.

Attributs :

Name : donne un nom symbolique à la variable.

Input : définit la première engtrée de la carte Master où l'encodeur est connecté (un encodeur simulé utlise 3 entrées successives, les autres 2 entrée successives). Dans tous les cas seulement la première entrée est définie.

Acceleration : Coefficient à appliquer à chaque détente.

Type : il y a 3 types d'encodeurs:

0 (default) : interrupteur rotatif simulant un encodeur.

1 : encodeurs à deux phases de signal (nécessite la carte Encoders).

2: encodeurs type Gray directement connecté à la carte Master.

Exemple:

Var 0001, Link IOCARD_ENCODER, Input 11, Aceleration 6, Type 2

IOCARD_ANALOGIC

Avec ce lien nous pouvons lire les axes d'une carte analogue d'extension USB.
Les valeurs de ce convertisseur A/D sopnt stockées dans la variable.
Les valeurs vont de 0 à 255.

Attributs :

Name : donne un nom symbolique à la variable.

Input : règle l'axe (#1-4).

PosI : règle la position gauche (valeur normale à 0)

PosC : règle la position centrale (valeur normale à 127)

PosR : règle la position droite (valeur normale à 255)

Exemple:

Var 2032, name POT1, Link IOCARD_ANALOGIC, Input #1, PosL 1, PosC 128, PosR 255

IOCARD_SERVO

Avec ce lien nous contrôlons la carte IOCard Servos2. Cette carte contrôle 2 moteurs servo.
Ces cartes sont connectées à des sorties consécutives de la carte Master. Chaque servo utilise 8 sorties (10 dans le cas d'une carte à résolution à 10 bits).

La valeur de la variable est envoyée au servo après un processus de compensation dépendant des posisions gauche, centre et droite.
Les valeurs varient entre 0 et 255 (0 à 1023 dans le cas de la résolution à 10 bits).
Le servo qui est contrôlé est celui connecté aux sorties définies dans le paramètre Output.

Attributs :

Name :donne un nom symbolique à la variable.

Output :définit la première sortie à laquelle le servo est connecté.

Type : Defaut : 1 : résolution de 8 bits.
Pour une résolution de 10 bits résolution, nous mettrons 2.

PosI :règle la position gauche (valeur normale à 1)

PosC :règle la position centrale (valeur normale à 127)

PosR :règle la position droite (valeur normale à 255)

Exemple:

Var 1346, Link IOCARD_SERVO, Output 11, PosL 10, PosC 120, PosR 200

IOCARD_MOTOR

Avec ce lien nous contrôlons des moteurs pas à pas ou à courant continu connectés aux cartes correspondantes.
Ces cartes sont connectées à des entrées consécutives d'une carte: 2 pour les moteurs continus, 4 pour les moteurs pas à pas.

Lorsque la variable vaut 127, tout type de moteur est arrêté.
Pour des valeurs supérieures à 127, le moteur tourne dans une direction (plus haute est la valeur, plus vite il tourne).
Pour des valeurs inférieures à 127, le moteur tourne dans la direction opposée (plus basse est la valeur, plus vite il tourne).

Attributs :

Name : donne un nom symbolique à la variable.

Output : définit la première sortie à laquelle le moteur est connecté (2 sortie pour les moteurs DC et 4 sorties pour les pas à pas).

Acceleration : coéfficent d'accélération. 1 est la plus haute valeur, 255 la plus basse.

Type : différents types peuvent être utilisés simultanément.

Par défaut : un moteur pas à pas sera contrôlé.

C : un moteur DC (courant continu) sera contrôlé.

M: dans le cas d'un moteur pas à pas, il fonctionnera par demi-pas.

D: pour inverser la direction.

Exemple:

Var 9675, Link IOCARD_MOTOR, Output 30, Aceleration 12, Type MD

USB_KEYS

Avec ce lien nous accédons aux cartes USB_KEYS.
Chaque touche enfoncée est envoyée à la variable, qui prend la valeur correspondante et exécute le script associé et reprend la valeur 0.
Il est possible de mettre un lien au module émulateur de clavier en mettant le type sur K. De cette manière la touche enfoncée est envoyée par USB_KEYS comme défini dans le fichier SIOC.INI.

Attributs:

Name : donne un nom symbolique à la variable.

Value : donne une valeur initiale.

Device : c'est le numéro de la carte à laquelle la variable est assignée. Si c'est 0, le premier numéro disponible est pris en compte (par défaut). En cliquant sur le bouton on accède à la liste des cartes IOCards USB connectées. Dans ce cas nous pouvons sélectionner la carte dans la liste. SI elle n'est pas sélectionnée, un 0 est mis par défaut.

Type : K peut être indiqué pour préciser une simulation de clavier.

Exemple:

Var 0001, name Teclas, Link USB_KEYS, Device 123, Type K

USB_STEPPER

Avec ce lien nous pouvons accéder aux cartes USB_STEPPER.
Les valeurs de variable sont envoyées au contrôleur de moteur avec les paramètres spécifiques.

Angle peut varier de 0 à 359,99 degrés, avec 2 décimales. Les valeurs varieront de 0 à 35999 (valeur qui correspondt à 359,99 degrés).

Attributs :

Name : donne un nom symbolique à la variable.

Device : est le numéro de la carte où la varible est liée. Si elle est 0, le premier numéro disponible sera considéré. the first available number is considered (default).). En cliquant sur le bouton on accède à la liste des cartes IOCards USB connectées. Dans ce cas nous pouvons sélectionner la carte dans la liste. SI elle n'est pas sélectionnée, un 0 est mis par défaut.

Type : H fait travailler le moteur par demi-pas.

Output :définit le moteur correspondant (1-3).

PosI : règle la valeur de vitesse (0-255) (voir le manuel de la carte pour ajuster cette valeur).

PosC : règle la valeur de pas par tour ou calibration. A 0, le compte des pas par tour est fait automatiquement (0-65535).

PosD : règle le nombre maximum de pas par 1/10ème de secondes (0 à 255) (voyez le manuel de la carte pour ajuster cette valeur).

Exemple:

Var 0234, Link USB_STEPPER, Output 2, PosL 6, PosC 0, PosR 4, Type H

USB_SERVOS

Avec ce lien nous pouvons accéder aux cartes USB_SERVOS.
La valeur de la variable est envoyée au moteur servo après une compensation dépendant des positions gauche, centre et droite. Les valeurs varient de 0 à 1023.
La valeur 0 signifie déconnection.

Attributs :

Name : donne un nom symbolique à la variable.

Device : est le numéro de la carte où la varible est liée. Si elle est 0, le premier numéro disponible sera considéré. the first available number is considered (default).). En cliquant sur le bouton on accède à la liste des cartes IOCards USB connectées. Dans ce cas nous pouvons sélectionner la carte dans la liste. SI elle n'est pas sélectionnée, un 0 est mis par défaut.

Output : règle le moteur correspondant (1-6).

Type : pour options futures, pas encore implémenté.

PosI : règle la position gauche (valeur normale 1)

PosC : règle la position centre (valeur normale 512)

PosR : règle la position droite (valeur normale 1023)

Exemple:

Var 0078, Link USB_SERVOS, Device 23, Output 5, PosL 1, PosC 512, PosR 1023

USB_RELAYS

Avec ce lien nous accédons aux cartes USB_RELAYS.
Si la valeur est 0, le relais indiqué en sortie est mis dans une position. Si la valeur est 1, le relais est mis dans l'autre position, et la led correspondante de la carte s'allume.

Attributs :

Name : donne un nom symbolique à la variable.

Device : est le numéro de la carte où la varible est liée. Si elle est 0, le premier numéro disponible sera considéré. the first available number is considered (default).). En cliquant sur le bouton on accède à la liste des cartes IOCards USB connectées. Dans ce cas nous pouvons sélectionner la carte dans la liste. SI elle n'est pas sélectionnée, un 0 est mis par défaut.

Output : règle le relais correspondant (1-7).

Exemple:

Var 0587, Link USB_RELAYS, Device 123, Output 6

USB_ANALOGIC

Avec ce lien nous accédons axes analogiques des cartes USB_SERVOS, USB_RELAYS et USB_STEPPER.
La valeur du convertisseur A/D est envoyée à la variable correspondante. Elle peut varier de 0 à 255.

Attributs :

Name : donne un nom symbolique à la variable.

Device : est le numéro de la carte où la varible est liée. Si elle est 0, le premier numéro disponible sera considéré. the first available number is considered (default).). En cliquant sur le bouton on accède à la liste des cartes IOCards USB connectées. Dans ce cas nous pouvons sélectionner la carte dans la liste. SI elle n'est pas sélectionnée, un 0 est mis par défaut.

Input : définit l'axe correspondant (1-5 selon les axes sur la crte).

PosI : règle la position gauche (valeur normale 0)

PosC : règle la position centre (valeur normale 127)

PosR : règle la position droite (valeur normale 255)

Exemple:

Var 0378, Link USB_ANALOGIC, Device 254, Input 3, PosL 1, PosC 127, PosR 255

SOUND

Avec ce lien nous accédons au module de son.
La variable agira selon le TYPE réglé (S ou pas).
Les fichiers .wav sont à définir dans le fichier SIOC.INI, numérotés à partir de 1.

Si rien n'est défini pour TYPE, lorsque la valeur de la variable est différente de 0, le module de son ouvrira le fichier .wav correspondant (défini dans SIOC.INI) et le jouera.

Si le type S est spécifié, lorsque la variable est différente de 0, le module son stoppera et fermera le fichier .wav correspondant.

Attributs :

Name : donne un nom symbolique à la variable.

TYPE : S utlisé pour arrêter un son comme indiqué plus haut.

Exemple:

Var 0034, name ALARM, Link SOUND, Type S

KEYS

Avec ce lien nous accédons au module keyboard emulator.
La valeur des variables varie de 1 à 200. Chaque valeur correspondt à une séquence de touches préalablement définie dans SIOC.INI

Ces définitions dans SIOC.INI se font dans la forme :

#1=\B\A

où #1 est le numéro de réglage (si la variable prend cette valeur, la séquence de touche est envoyée) et\B\A est la séquence de touches.

Pour envoyer la séquence #1, nous faisons V0010 = 1, et automatiquement la valeur revient à 0 et la séquence peut à nouveau être renvoyée.

Il y a des codes spéciaux pour des touches spéciales (Ctrl, Caps, etc.).
Le code / sera indiqué avant une des lettres suivantes :

A = BKSP = #8;
B = TAB = #9;
C = ENTER = #13;
D = ESC = #27;
E = F1 = #228;
F = F2 = #229;
G = F3 = #230;
H = F4 = #231;
I = F5 = #232;
J = F6 = #233;
K = F7 = #234;
L = F8 = #235;
M = F9 = #236;

N = F10 = #237;
O = F11 = #238;
P = F12 = #239;
Q = HOME = #240;
R = END = #241;
S = UP = #242;
T = DOWN = #243;
U = LEFT = #244;
V = RIGHT = #245;
W = PGUP = #246;
X = PGDN = #247;
Y = INS = #248;
Z = DEL = #249;
1 = SHIFT_DN = #250;
2 = SHIFT_UP = #251;
3 = CTRL_DN = #252;
4 = CTRL_UP = #253;
5 = ALT_DN = #254;
6 = ALT_UP = #255;

\ = \

USB_DCMOTOR

Avec ce lien nous accédons à la carte USB_DCMOTOR.
La valeur de la variable est envoyée au contrôleur. Les valeurs varient de 0 (arrêté) à 127 (Max. speed) pour la rotation droite. Ajouter 128 pour la rotation gauche.
Il faut envoyer une première valeur pour démarrer la carte.

Attributs :

Name : donne un nom symbolique à la variable.

Device : est le numéro de la carte à laquelel la variable est assignée. Par défaut 0.

Output : règle le moteur correspondant (1-6).

Exemple:

Var 0078, Link USB_DCMOTOR, Device 2, Output 5

DEFINITION DES COMMANDES

ASSIGNMENT

Variable = Constante ou Variable

Variable = Constante ou Variable Opérateur Constante ou Variable

Assigne le résultat de l'opération à une variable IOCP ou interne.

Exemple:

```
{
  V0002 = V0008 * 3.14
  L2 = 3.8673
  L1 = V0001 AND 128
  C1 = L1 < 5
}
```

CALL

CALL Variable(sous-routine) Constante ou Variable

Exécute le script associé à la variable, et, si un paramètre est inclus, la variable prend cette valeur.

Le script est exécuté dans tous les cas.

Exemple:

```
{
  CALL V9888
  CALL V1001 326
  CALL V3004 L0
}
```

IF ELSE (CONDITION)

```
IF  Condition
{
  Commande
}
ELSE
{
  Commande
}
```

Les commandes suivant IF sont seulement exécutées si la condition est vraie; sinon, les commandes suivant ELSE seront exécutées.

La partie ELSE est facultative. Si elle est utilisée elle doit se trouver après IF.

Il est possible d'intriquer jusqu'à 100 IF.

La condition peut être une variable booléenne, ou 2 variables booléennes associées par AND / OR, ou des conditions entre variables et/ou des constantes, avec l'opérateur correspondant.

Exemple:

```
{
  IF L1 > 5
  {
    CALL V1000
  }
  ELSE
  {
    IF C2
    {
      L1 = L1 + 1
    }
  }
}
```

LISTE DES OPERATEURS

OPERATEUR	DESCRIPTION
+	Addition de deux variables ou constantes
-	Soustraction de deux variables ou constantes
*	multiplication de deux variables ou constantes
/	division de deux variables ou constantes
AND	Condition logique ET
OR	Condition logique OU
>	Condition si plus grand que
<	Condition si plus petit que
=	Condition si égale
>=	Condition si plus grand ou égal
<=	Condition si plus petit ou égal
<>	Condition si différent

REFERENCE DE SIOC

FONCTIONS

DEFINITION DES FONCTIONS

Variable = Fonction paramètre1 paramètre 2 paramètre 3

paramètre : variable, constante en nombre reél ou entier.

Chaque fonction a 1, 2 ou 3 paramètres maximum, et la valeur finale est assignée à la variable.

Exemple:

```
{
  V0002 = ROUND L0
  L2 = TOBCD V0005
  C1 = TESTBIT V1234 6
  V9888 = TIMER 100 5 10
}
```

ABS

Variable = ABS Paramètre

Assigne à la variable la valeur absolue du paramètre.

Paramètre : variable, constante en nombre reél ou entier.

Exemple :

```
  V0002 = ABS V0008
```

CHANGEBIT

Variable = CHANGEBIT Paramètre1 Paramètre2

Change le bit de variable indiqué dans le paramètre1, en la valeur du paramètre 2. Le Bit change à 0 si le paramètre2 est 0; à 1 si le paramètre2 est 1.

Paramètre1 : variable, constante en nombre reél ou entier. Bit à changer.

Paramètre2 : variable, constante en nombre reél ou entier. Mode de changement.

Exemple:

```
  V1354 = CHANGEBIT 3 V0034
```

CHANGEBITN

Variable = CHANGEBITN Paramètre1 Paramètre2

Change le bit de variable indiqué dans le paramètre1, en la valeur du paramètre 2. Le Bit change à 1 si le paramètre2 est 0; à 0 si le paramètre2 est 1.

Paramètre1 : variable, constante en nombre reél ou entier. Bit à changer.

Paramètre2 : variable, constante en nombre reél ou entier. Mode de changement.

Exemple :

V1354 = CHANGEBITN 3 V0034

CLEARBIT

Variable = CLEARBIT Paramètre1

Met à 0 le bit indiqué par le Paramètre1.

Paramètre1 : variable, constante en nombre reél ou entier. Bit à changer.

Exemple :

V0002 = CLEARBIT 3

COS

Variable = COS Paramètre

Donne à la variable la valeur du cosinus du paramètre (radians).

Paramètre : variable, constante en nombre reél ou entier.

Exemple :

V0002 = COS V0008

DELAY

Variable = DELAY Paramètre1 Paramètre2

Exécute le script après le délai de temps indiqué dans le paramètre 2 (en 1/10ème de secs) et assigne à la variable la valeur indiquée dans le Paramètre1.

Paramètre1 : Variable, constante en nombre reél ou entier. Valeur à indiquer.

Paramètre2 : Variable, constante en nombre reél ou entier. délai (1/10ème de secs).

Exemple:

V1354 = DELAY 57 10

DIV

Variable = DIV Paramètre1 Paramètre2

Divise Paramètre1 / Paramètre2. Le résultat est stocké dans la variable.

Paramètre1 : Variable, constante en nombre reél ou entier. Nombre à diviser.

Paramètre2 : Variable, constante en nombre reél ou entier.

Exemple:

V0032 = DIV V0034 2

FROMBCD

Variable = FROMBCD Paramètre1

Convetit la valeur du paramètre1 (BCD) en format décimal et stocke le résultat dans la variable. (format utilisé dans plusieurs offsets FSUIPC).

Paramètre1 : Variable, constante en nombre reél ou entier.

Exemple:

V2302 = FROMBCD V9233

LIMIT

Variable = LIMIT Paramètre1 Paramètre2 Paramètre3

Augment ou diminue la valeur de la variable par la valeur du paramètre 3. Si le résultat est supérieur au Paramètre2, alors la variable = Paramètre2. Si le résultat est inférieur au Paramètre1, alors la variable = Paramètre1.

Paramètre1 : Variable, constante en nombre reél ou entier. Valeur inférieure.

Paramètre2 : Variable, constante en nombre reél ou entier. Valeur supérieure.

Paramètre3 : Variable, constante en nombre reél ou entier. Incrément/Décrément.

Exemple:

V9302 = LIMIT 0 60000 V0456

LOGN

Variable = LOGN Paramètre1

Donne à la variable la valeur du logarithme naturel du paramètre.

Paramètre1 : Variable, constante en nombre reél ou entier.

Exemple :

V0002 = LOGN V0008

NOT

Variable = NOT Paramètre1

Donne à une variable booléenne la valeur opposée de celle indiquée par le Paramètre1.

Paramètre1 : Boolean variable.

Exemple:

C1 = NOT C0

MOD

Variable = MOD Paramètre1 Paramètre2

Calcule le reste de la division Paramètre1 / Paramètre2. La valeur est stockée dans la variable.

Paramètre1 : Variable, constante en nombre reél ou entier. Nombre à diviser.

Paramètre2 : Variable, constante en nombre reél ou entier.

Exemple:

V0032 = MOD V0034 2

POWER (PUISSANCE)

Variable = POWER Paramètre1 Paramètre2

Donne à la variable la valeur du Paramètre1 comme base et du Paramètre2 comme exposant

Paramètre1 : Variable, constante en nombre reél ou entier. Base

Paramètre2 : Variable, constante en nombre reél ou entier. Exposant

Exemples:

L0 = POWER V000 0.5

RANDOM

Variable = RANDOM Paramètre1 Paramètre2

La variable reçoit une valeur générée au hasard entre Paramètre1 et Paramètre2.

Paramètre1 : Variable, constante en nombre reél ou entier. Limite inférieure.

Paramètre2 : Variable, constante en nombre reél ou entier. Limite supérieure.

Exemple:

V1354 = RANDOM 0 9

ROTATE

Variable = ROTATE Paramètre1 Paramètre2 Paramètre3

Augmente ou diminue la valeur de la variable de la valeur du paramètre3. Pour une augmentation si le résultat est supérieur Paramètre2, alors variable = Paramètre1. Pour une diminution si le résultat est inférieur au Paramètre1, alors variable = Paramètre2.

Ce coportement cyclique est typique de jauges comme VOR OBS.

Paramètre1 : Variable, constante en nombre reél ou entier. Valeur inférieure.

Paramètre2 : Variable, constante en nombre reél ou entier. Valeur supérieure.

Paramètre3 : Variable, constante en nombre reél ou entier. Incrément/Decrément.

Exemple:

V9002 = ROTATE 1 360 V0034

ROUND

Variable = ROUND Paramètre1

Donne à la variable la valeur arrondie du Paramètre1.

Paramètre1 : Variable, constante en nombre reél ou entier.

Exemple:

V0002 = ROUND L1

SETBIT

Variable = SETBIT Paramètre1

La variable indiquée par le Paramètre1 est mise à 1.

Paramètre1 : Variable, constante en nombre reél ou entier. Bit à changer.

Exemple:

V0002 = SETBIT V0034

SETSOUND

Variable = SETSOUND Paramètre1 Paramètre2 Paramètre3

La fonction implique que soit précisée la variable locale assignée au son dans sioc.ini.

La fonction change les paramètres de Fréquence, Volume et Pan (balance) du son sélectionné. of

Paramètre1 : Variable ou constante (nombre entier). Fréquence.
(100 to 100000 0 = valeur initiale, -1 = défaut)

Paramètre2 : Variable ou constante (nombre entier). Volume.
(0 à 100, -1=défaut)

Paramètre3 : Variable ou constante (nombre entier). Valeur de balance

(-100 (gauche) à +100 (droit), 0=centre, -1=défaut)

Exemple:

```
Var 0002, Link SOUND, Type S

Var 0001, Link SOUND

Var 0000, Value 0
{
  V0001 = 1

  V0001 = 2
  L0 = 2
  L0 = SETSOUND -1 ,40 ,-1
  V0009 = -99
  V0009 = TIMER 100 ,1 ,3
}

Var 0009, Link SUBRUTINE
{
  L0 = 2
  L0 = SETSOUND -1 ,-1 ,V0009
  IF V0009 = 100
  {
    V0009 = TIMER -100 ,-1 ,3
  }
  IF V0009 = -100
  {
    V0002 = 2
  }
}
```

SIN

Variable = SIN Paramètre

Donen à la variable la valeur du sinus du paramètre (radians).

Paramètre : variable, constante en nombre reél ou entier.

Exemple :

```
  V0002 = SIN L1
```

TESTBIT

Variable = TESTBIT Paramètre1 Paramètre2

Si la variable est booléenne: la variable est mise sur vrai (faux) si le bit du paramètre1 indiqué par le paramètre2 est 1 (0).

Si la variable est un entier ou un réel : la variable est mise sur 1 (0) si le bit du paramètre1 indiqué par le paramètre2 est 1 (0).

Paramètre1 : Variable, constante en nombre reél ou entier. Nombre à tester.

Paramètre2 : Variable, constante en nombre reél ou entier. Nombre du bit.

Exemple:

C0 = TESTBIT V0008

TIMER

Variable = TIMER Paramètre1 Paramètre2 Paramètre3

Le script associé à la variable est exécuté périodiquement, comme indiqué dans le Paramètre3 ($1/10^{ème}$ de secs).

Pour chaque boucle (la première est exécutée après écoulement du délai indiqué par le Paramètre3) la valeur de la variable augmente/diminue comem indiqué par le Paramètre2.
Le processus cesse lorsque le valeur de la variable atteint le Paramètre1.

Paramètre1 : Variable, constante en nombre reél ou entier. Valeur finale.

Paramètre2 : Variable, constante en nombre reél ou entier. Incrément/Decrément.

Paramètre3 : Variable, constante en nombre reél ou entier. Récurrence.

Exemple:

V9000 = TIMER 1000 5 10

METTRE SCHEMA TUTORIEL

TOBCD

Variable = TOBCD Paramètre1

Convertit la valeur du paramètre1 en format BCD et le stocke dans la variable (ce format est utilisé par plusieurs offsets FSUIPC).

Paramètre1 : Variable, constante en nombre reél ou entier.

Exemple:

V2302 = TOBCD V9233

TOGGLE

Variable = TOGGLE Paramètre1

Déclenche un 'toggle' (setmet à 1 puis 0) pour le bit de variable indiqué par le paramètre1.

Paramètre1 : Variable, constante en nombre reél ou entier.

Exemple:

```
V0023 = TOGGLE 3
```

TRUNC

Variable = TRUNC Paramètre1

Enlève la partie décimale du paramètre1, le convertissant en nombre entier. La valeur est stockée dans la variable.

Paramètre1 : Variable, constante en nombre reél ou entier.

Exemple:

```
L1 = TRUNC L0
```

ANNEXE : TABLES DE RESUMES DES FONCTIONS ET COMMANDES.

Types de LINKs (liens)

MODULE	DEFINITION DE LINK	DESCRIPTION
MODULE FSUIPC	**FSUIPC_OUT**	Envoie les données aux offsets FSUIPC
	FSUIPC_IN	Reçoit les données des offsets FSUIPC
	FSUIPC_INOUT	Reçoit et envoie les données FSUIPC
MODULE CLIENT IOCP	**IOCP**	Envoie et reçoit les données de variables IOCP
MODULE IOCARDS	**IOCARD_SW**	Gère les switches des Iocards
	IOCARD_OUT	Active/désactive les sorties des Iocards
	IOCARD_DISPLAY	Envoie les chiffres aux Iocards display
	IOCARD_ENCODER	Reçoit les informations des encodeurs
	IOCARD_ANALOGIC	Reçoit les informations d'entrées analogiques
	IOCARD_SERVO	Fait bouger les servos Iocards
	IOCARD_MOTOR	Gère les moteurs DC pas à pas
SIOC	**SUBRUTINE**	Gère les variables en subroutine

ATTRIBUTS reconnus par SIOC

ATTRIBUT	DESCRIPTION
Link	Définit le type de liaison que doit avoir la variable
Type	Définit les caractéristiques spéciales de l'élément
Offset	Numéro de variable IOCP ou Offset FSUIPC
Value	Valeur initiale de la variable
Lenght	Longueur de l'Offset FSUIPC auquel la variable est liée
Input	Entrée de la carte master à laquelle la variable est liée
Output	Sortie de la carte master à laquelle la variable est liée
Digit	Premier digit de la carte display qui définit un nombre
Acceleration	Accélération sélectionnée pour un encodeur
Numbers	Chiffres à gérer de la carte display
PosL	Position gauche de calibration
PosC	Position centrée de calibration
PosR	Position droite de calibration

REFERENCE DE SIOC

COMMANDES reconnues par SIOC

COMMANDE	DESCRIPTION
Assignation	Pour assigner une valeur ou un calcul à une variable SIOC
Fonction	Fonctions à appliquer aux variables SIOC
Call	Lance un script associé à une variable se type subrutine et lui passe un paramètre
Condition If	Commandes soumises à une condition
Condition Else	Commandes exécutées si la condition n'est pas réalisée

OPERATEURS reconnus par SIOC

OPERATEUR	DESCRIPTION
+	Addition de deux variables ou constantes
-	Soustraction de deux variables ou constantes
*	multiplication de deux variables ou constantes
/	division de deux variables ou constantes
AND	Condition logique ET
OR	Condition logique OU
>	Condition si plus grand que
<	Condition si plus petit que
=	Condition si égale
>=	Condition si plus grand ou égal
<=	Condition si plus petit ou égal
<>	Condition si différent

FONCTIONS reconnues par SIOC

FONCTION	DESCRIPTION
Round	Arrondit à la valeur entière la plus proche
Trunc	Enlève les décimales et donne un nombre entier
Timer	Programme des événements périodiques selon un chrono
Setbit	Active le bit d'une variable
Clearbit	Désactive le bit d'une variable
Testbit	Teste si le bit d'une variable est actif
Not	Insère la valeur d'une variable booléenne (C0,C1,C2)
Rotate	Produit des incrémentations/décrémentations cycliques
ToBCD	Change une valeur décimale en format BCD
FromBCD	Change de BCD en valeur décimale
Toggle	Effectue une fonction toggle dans le bit d'une variable
Abs	Change à la valeur absolue
ChangeBit	Change un bit de la variable en fonction d'un paramètre
ChangeBitN	Idem avec une autre utilisation du paramètre
Limit	Augmente la variable en fonction d'une limite
Div	Calcule la division entière de deux nombres
Mod	Calcule le module (reste de la division) entre deux nombres