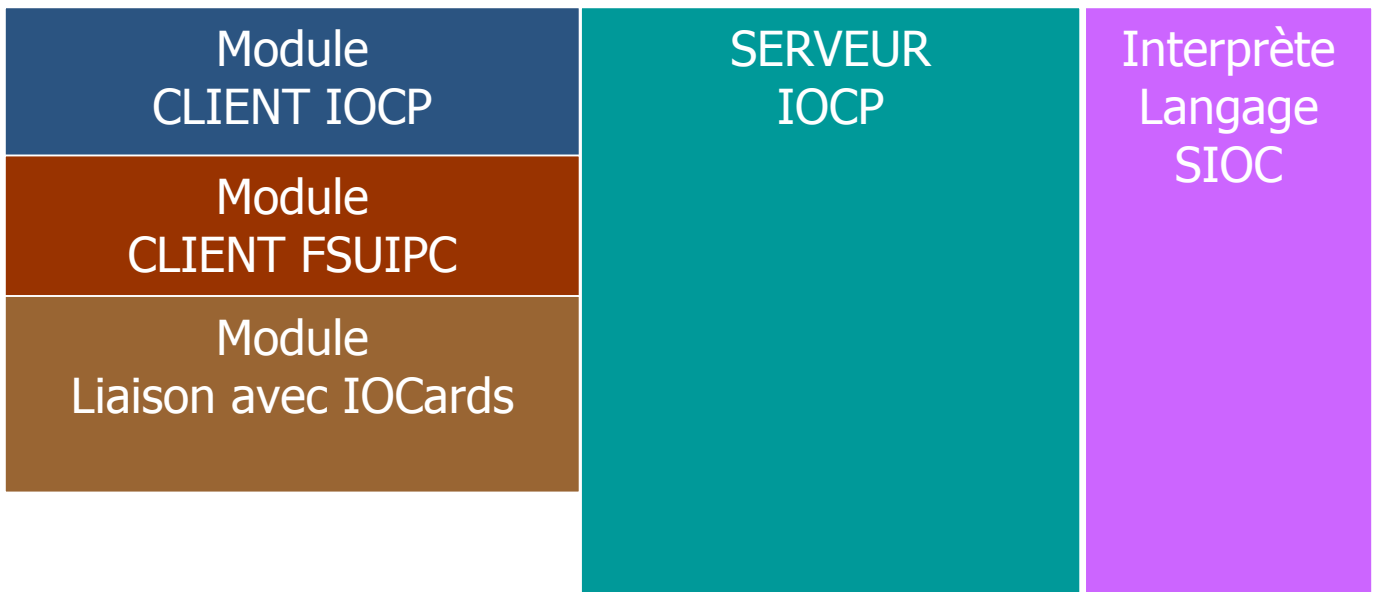


Concepts théoriques	2
Structure de SIOC	2
SIOC <-> IOCP	2
Système basé sur évènements	2
Outils de base du SIOC	3
Variables de SIOC	3
Pratique 1: Démarrer SIOC sans aucun programme chargé	4
Pratique 2: Accéder à SIOC par IOCP_Console	6
Concept de LINK (lien)	8
Langage SIOC	8
Pratique 3: Exemples d'accès aux données du simulateur par FSUIPC	9
Scripts associés à la variable	14
Exécution des scripts associés	15
Langage SIOC	15
Pratique 4: Exemple d'exécution des scripts associés.	16
Pratique 5: Connection avec les IOCards	22
Langage SIOC en text editor	27
Caractéristiques générales du langage des scripts	29
Définition des script associés à une variable SIOC	29
COMMANDES reconnues par SIOC	30
OPERATEURS reconnus par SIOC	30
Pratique 6: Connecter les IOCards au Simulateur	31
ANNEXE	33
Définition formelle du langage SIOC	33
Définition de script associé à une variable SIOC	33
Types de LINKs (liens)	34
ATTRIBUTS reconnus par SIOC	34
COMMANDES reconnues par SIOC	35
OPERATEURS reconnus par SIOC	35
Liaison d'une variable SIOC avec les différents modules	36
Fonction TIMER	40
Commande CALL	40

Concepts théoriques

Solution technique pour la gestion des simulateurs
 Gestion des communications, interactions avec le simulateur et avec l'électronique
 Élément structurel de la simulation conçu comme centre de contrôle

Structure de SIOC



SIOC <-> IOCP

SIOC est basé sur un protocole de communication IOCP
 SIOC a comme base un serveur IOCP interne et tout est basé sur la manipulation des variables IOCP.

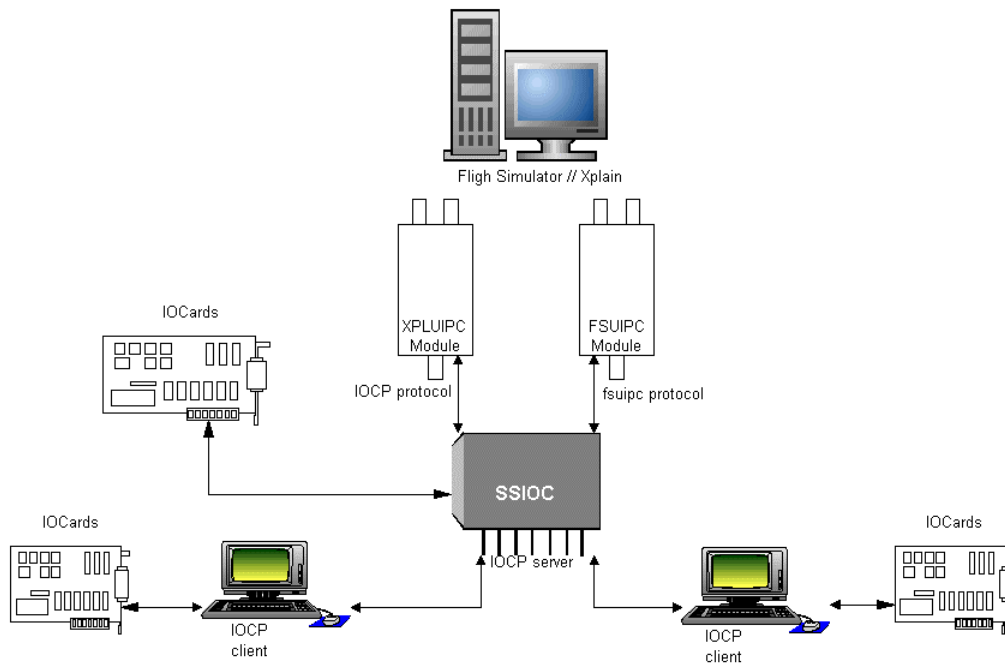
SIOC est basé sur des EVENEMENTS.

Système basé sur évènements

SIOC est surtout un Serveur IOCP
 Les évènements du SIOC se reflètent toujours sur ses variables internes (Celles du serveur IOCP interne).

Il y a trois types d'évènements:

- La modification d'une variable(interne ou externe).
- Changement d'états des entrées (digitales ou analogiques) de la IOCards.
- Activation d'un Timer.



Outils de base du SIOC

Config_SIOC : Pour créer des fichiers qui règlent le fonctionnement du SIOC.

SIOC : Programme principal. Serveur SIOC.

IOCP_Console : Client IOCP pour se connecter à un serveur IOCP qui peut être SIOC

Variables de SIOC

Une variable c'est un lieu où on gère un numéro.

SIOC gère 10.000 variables.

Ces variables sont numérotées de 0 à 9999.

Dans SIOC, toutes les variables gèrent un numéro entier avec signe.

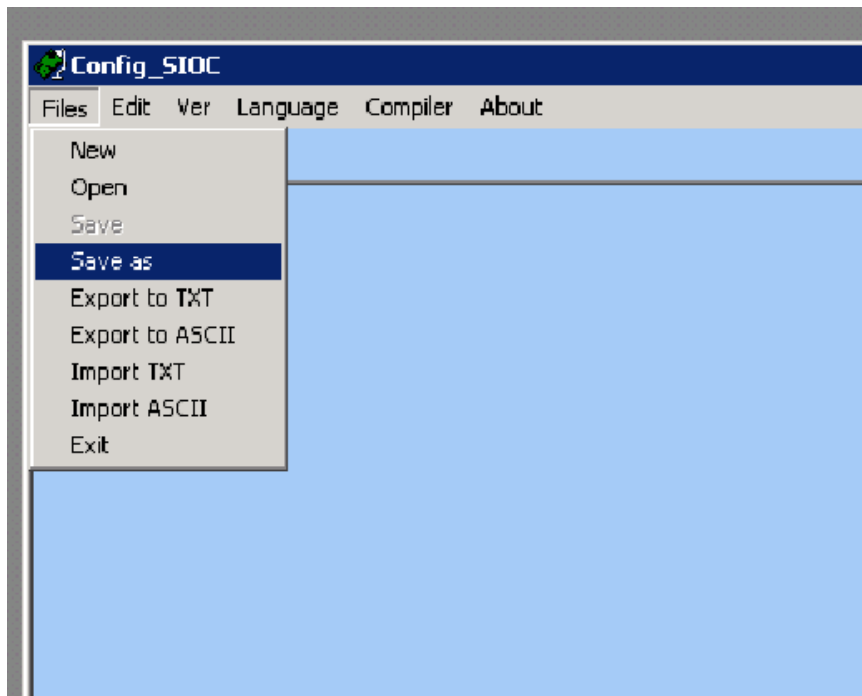
Ces variables peuvent être consultées ou modifiées par tous les programmes clients qui vont se connecter au SIOC.

La nouvelle valeur que prend une de ces variables est actualisée sur tous les clients connectés au SIOC.

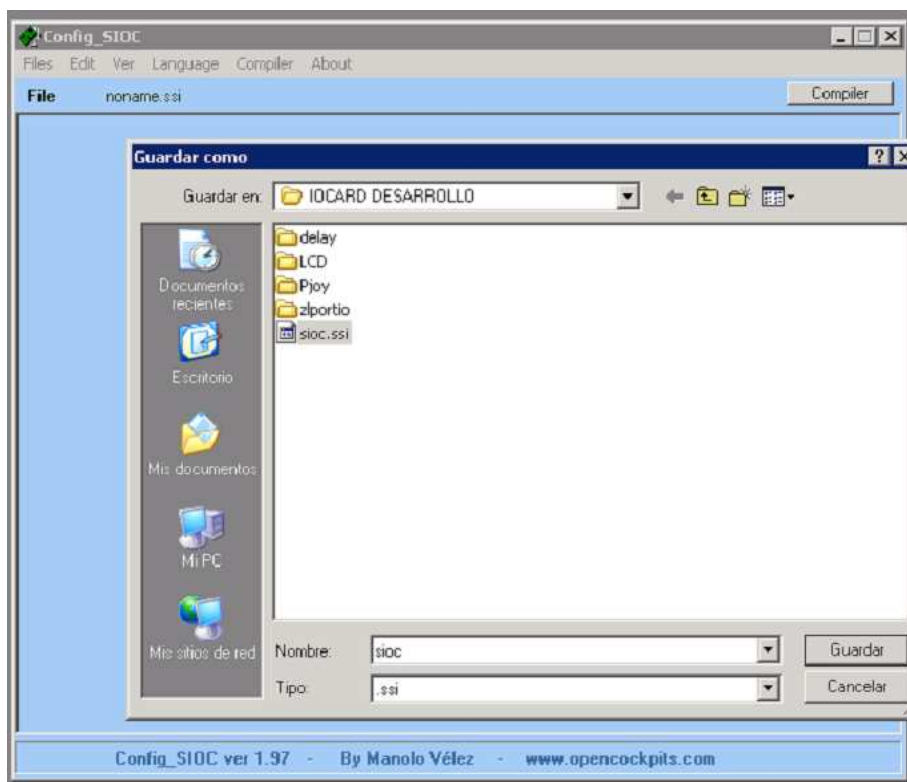
Pratique 1: Démarrer SIOC sans aucun programme chargé

Démarrer le programme Config_SIOC.exe

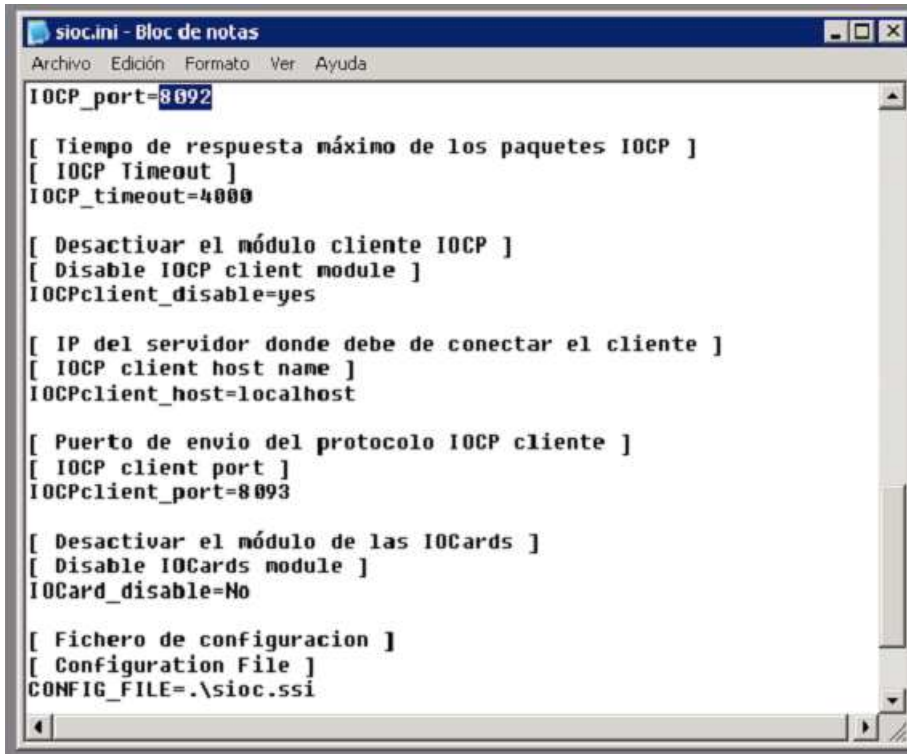
- Ouvrir le menu des fichiers et sélectionner sauver comme (save as)



Sélectionner le nom du fichier sioc.ssi et Ouvrir.

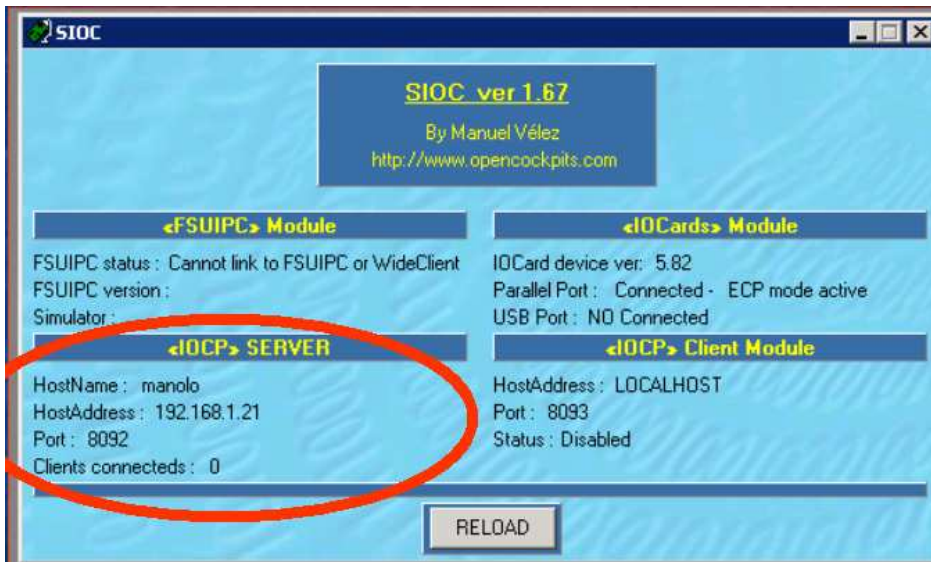


-Editer le fichier sioc.ini et s'assurer de sélectionner un port libre pour le serveur SIOC. Par défaut 8092.



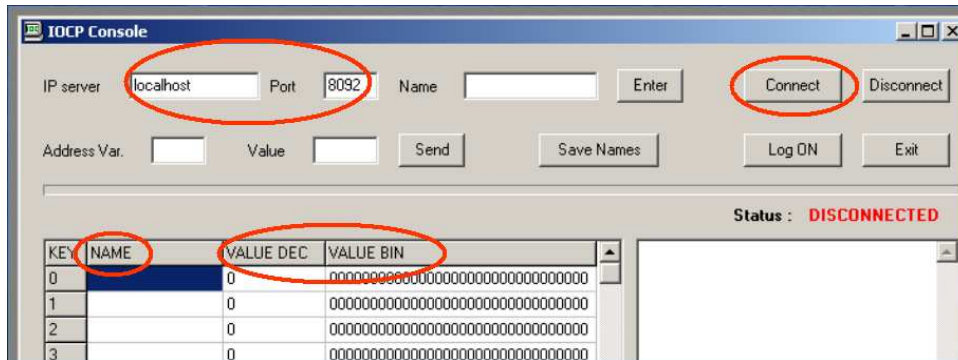
-Démarrer le programme SIOC.EXE

- S'assurer que le serveur est en fonction et attend la connection des clients.



Pratique 2: Accéder à SIOC par IOCP_Console

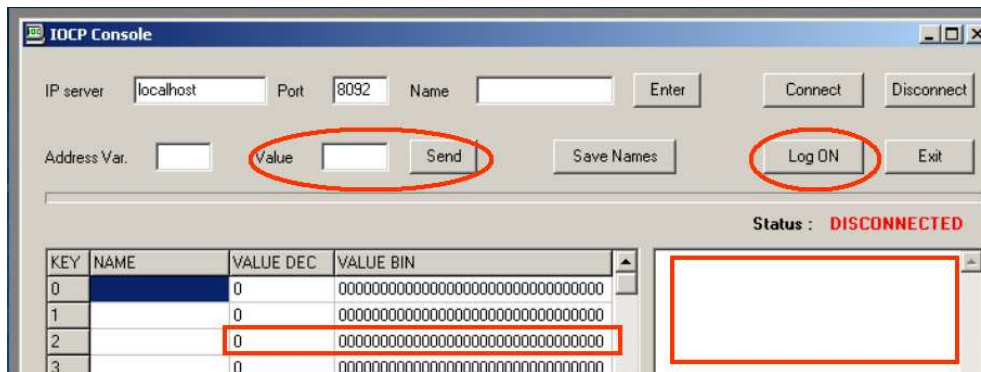
- Démarrer le programme IOCP_Console.exe
- Ecrire l'adresse IP et le port de SIOC
- Appuyer sur Connect
- On peut voir les valeurs des variables en décimale et binaire.
- On peut donner un nom à chaque variable qui nous aidera pour l'identification de sa fonction.

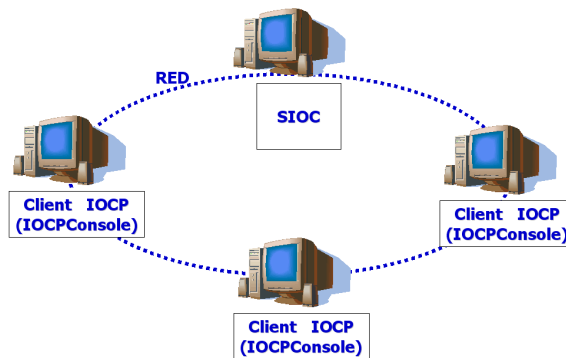
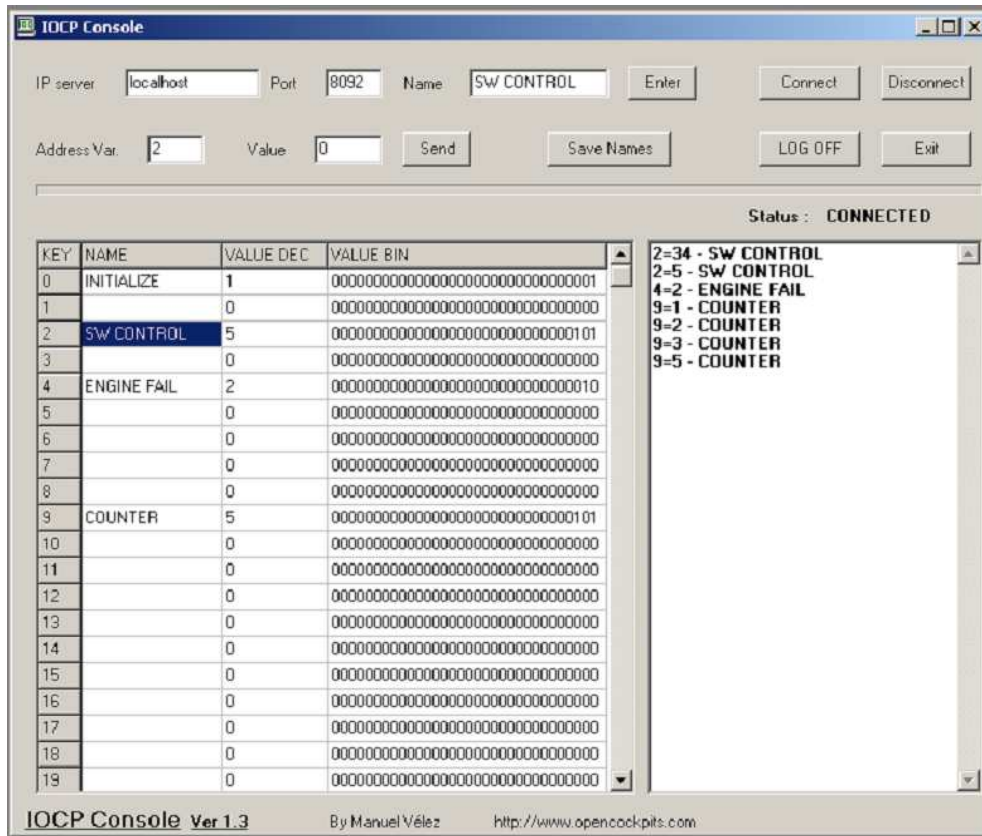


-On peut changer la valeur d'une variable en appuyant sur la ligne qui correspond à cette variable et on va mettre sa valeur dans la case value. Pour réaliser la modification il faut appuyer sur Send.

- Tous changements d'une variable SIOC, sera transféré dans la case correspondante.

- On peut activer le LOG en appuyant sur Log ON et tous changements de variables sera affiché dans la fenêtre

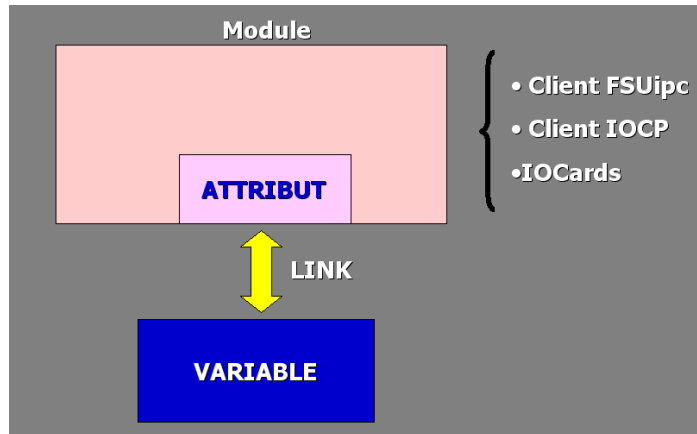




Concept de LINK (lien)

L'interaction de SIOC avec l'extérieur se fait par ses variables qui se définissent comme LINKS. Une variable liée (link) avec un module, envoie ou reçoit les données entre eux par la valeur des variables.

Un link est figé par une variable et en plus ce link peut avoir des attributions ou caractéristiques spéciales qui précisent comment la variable doit agir.



Langage SIOC

1.Langage de définition des variables.

On définit les caractéristiques des variables SIOC utilisées, ses attributions en relation aux LINK (liens) que nous avons assigné.

Types de LINKs (liens)

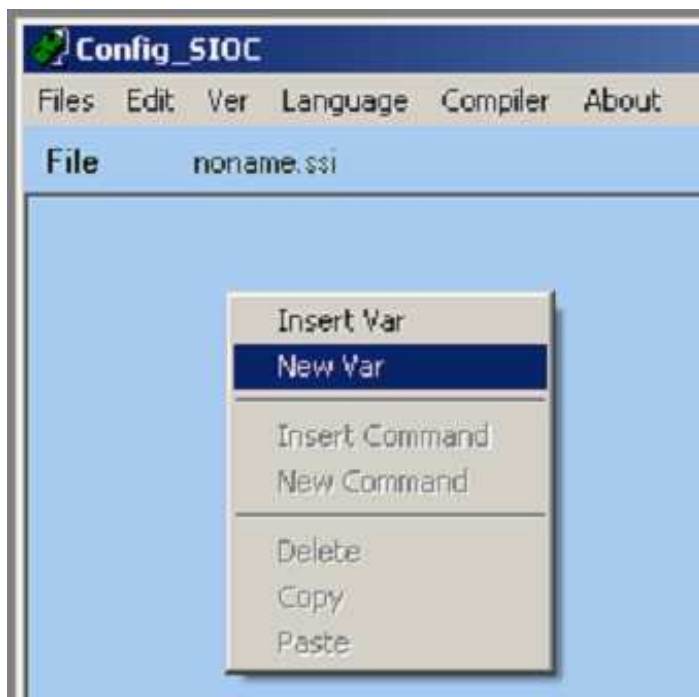
MODULE	DEFINITION DE LINK	DESCRIPTION
MODULE FSUIPC	FSUIPC_OUT	Envoie les données aux offsets FSUIPC
	FSUIPC_IN	Reçoit les données des offsets FSUIPC
	FSUIPC_INOUT	Reçoit et envoie les données FSUIPC
MODULE CLIENT IOCP	IOCP	Envoie et reçoit les données de variables IOCP
MODULE IOCARDS	IOCARD_SW	Gère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

Pratique 3: Exemples d'accès aux données du simulateur par FSUIPC

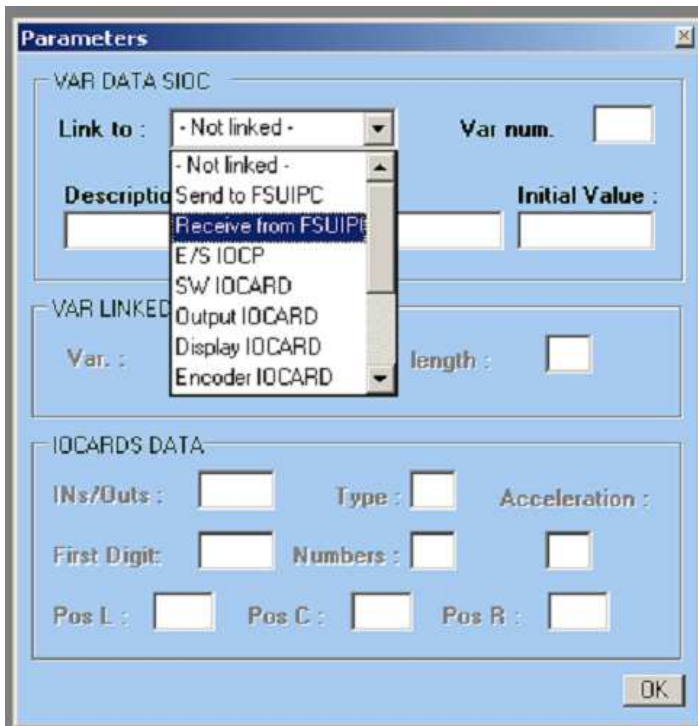
- Démarrer Config_SIOC
- Faire click à droite
- Appuyer sur Nouvelle Variable



Apparaîtra une ligne blanche
Double clic sur cette ligne



Apparaît une nouvelle fenêtre pour introduire les données de la variable que nous allons définir :choix du type de Link



Sélectionner le type de link choisi

Notre exemple est Recevoir de FSUIPC

Nous allons définir les données du IAS de l'Autopilot suivant la liste de variables FSUIPC SDK de Peter Downson :

07E2	2	Autopilot airspeed value, in knots	OK	OK
------	---	------------------------------------	----	----

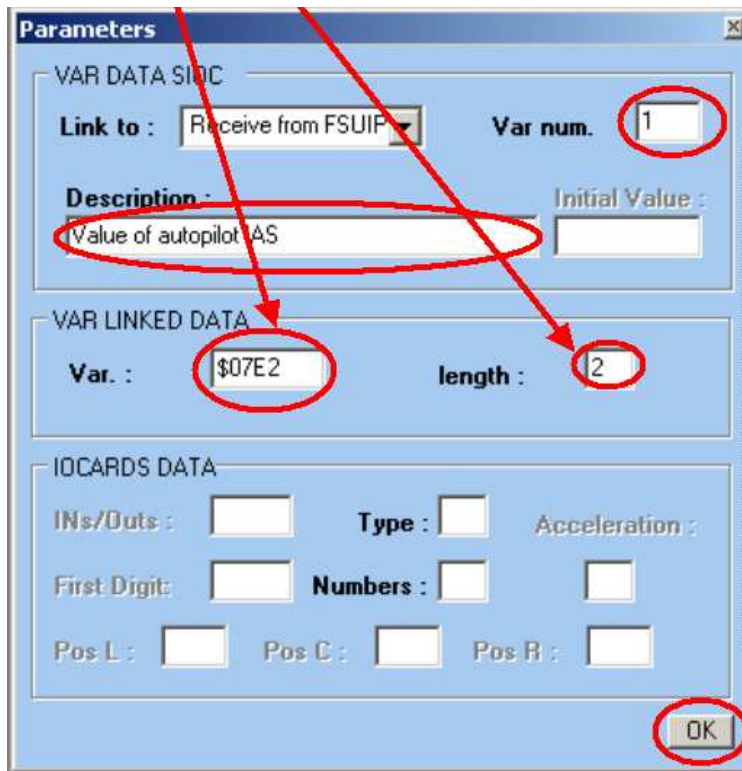
Introduction du Nombre de la variable.

Une description

Une valeur de l'offset

Sa longueur en bytes

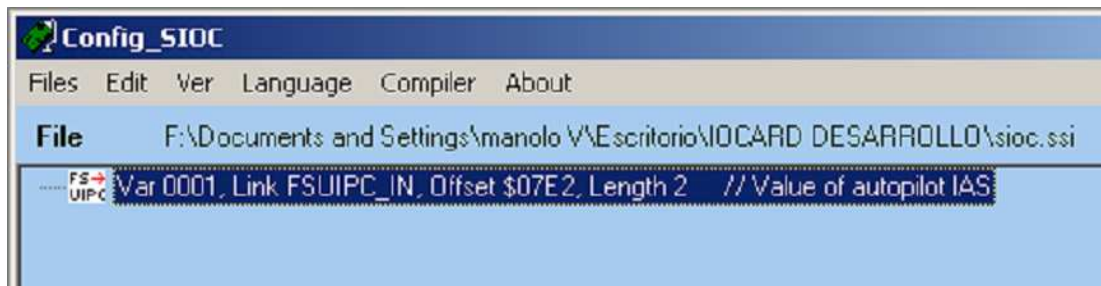
Appuyer sur OK



La ligne change et elle montre la définition qui correspond à ce link

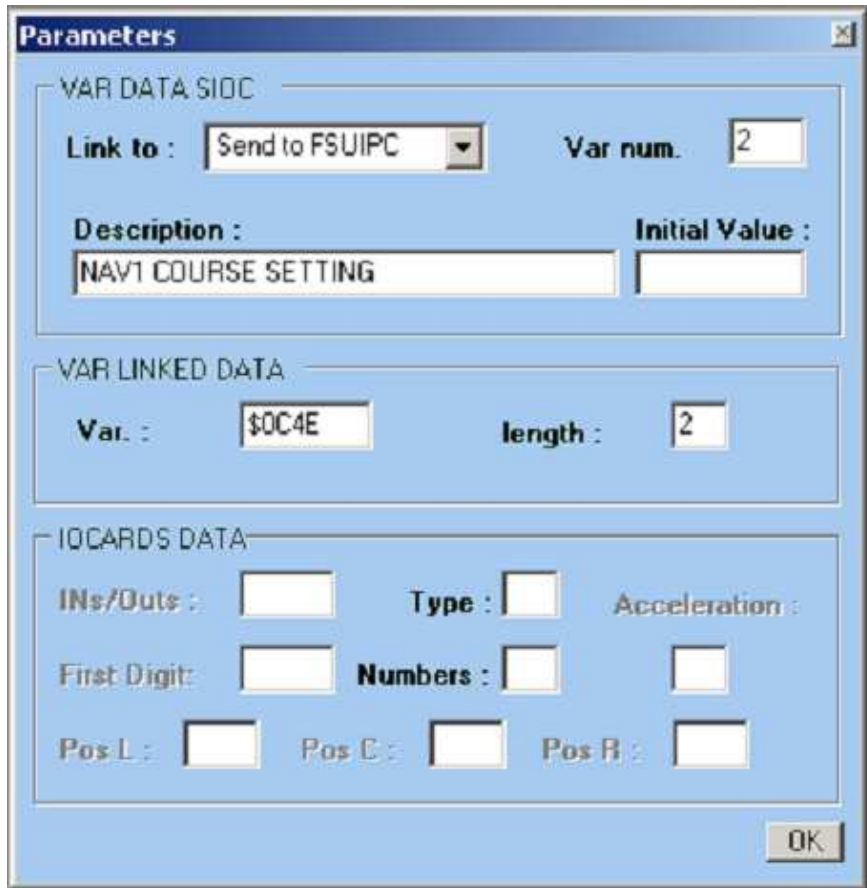
C'est une définition de la variable en langage SIOC.

Nous allons répéter cette opération, mais cette fois nous allons sélectionner un link Envoyer à FSUIPC, pour envoyer les données à Flight Simulator.

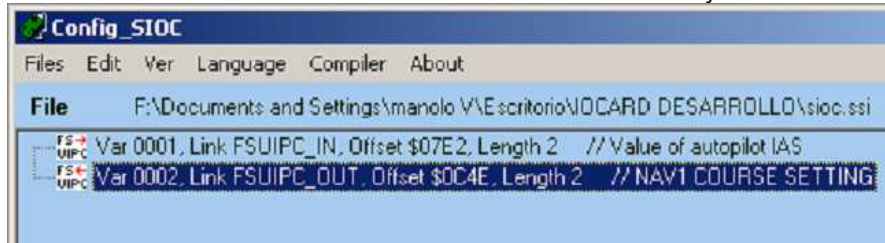


Nous allons sélectionner l'offset \$0C4E ou il y a la valeur du COURSE.

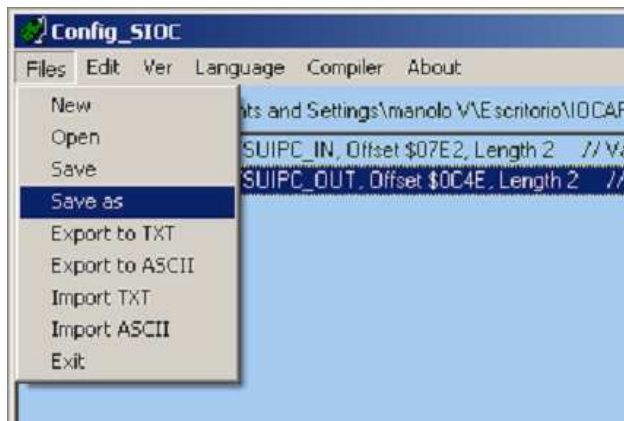
0C4E	2	NAV1 OBS setting (degrees, 0-359)	Ok	Ok
------	---	-----------------------------------	----	----



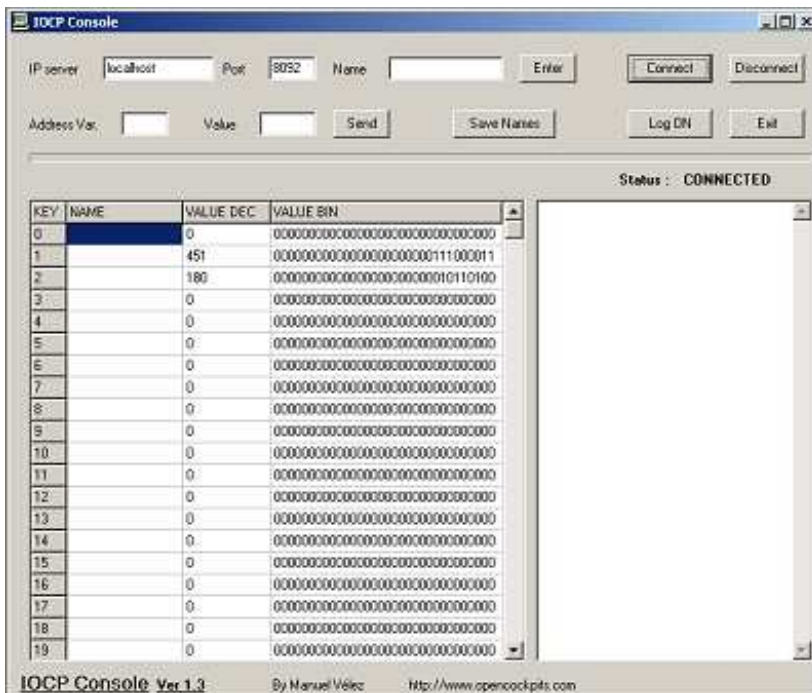
Nous avons terminé la définition des deux variables V0001 y V0002.



Appuyer sur le menu fichiers, sur sauver comme et nous utiliserons le nom sioc.ssi , celui par défaut de SIOC



Démarrer FightSimulator, SIOC et IOCPConsole ...



On peut voir que dans la variable V0001 apparaît la valeur du IAS/MACH

KEY	NAME	VALUE DEC	VALUE BIN
0		0	00000000000000000000000000000000
1	IAS/MACH <- IN	451	0000000000000000000000000111000011
2	COURSE -> OUT	0	00000000000000000000000000000000

1=451 - IAS/MACH <- IN



Toutes variations de l'IAS du simulateur sera reconnue par la variable V0001, et nous pourrons voir toujours la dernière valeur de la variable reflétée sur la console.

Tous changements de la variable seront reconnus en même temps dans la fenêtre du LOG, s'il est ouvert.

On peut changer la valeur de la variable V0002 en affichant la nouvelle valeur et appuyant sur Envoyer.

Automatiquement SIOC modifie la variable et envoie au FSimulateur la valeur qui va se refléter dans le MCP.

Address Var. Value

Status : CON

KEY	NAME	VALUE DEC	VALUE BIN
0		0	00000000000000000000000000000000
1	IAS/MACH <- IN	451	0000000000000000000000000111000011
2	COURSE -> OUT	180	000000000000000000000000010110100

1=451 - IAS/MACH <- IN
2=180 - COURSE -> OUT

Sur la console se reflétera le changement de la valeur de la variable.

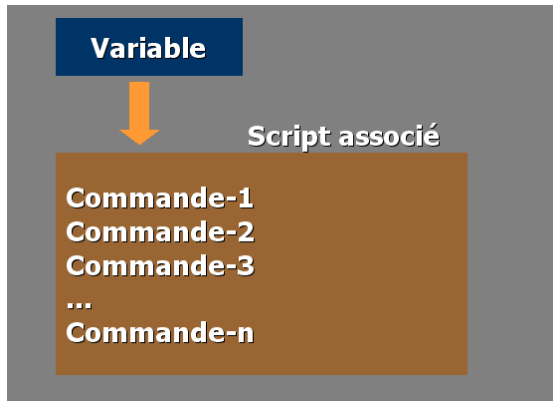


Scripts associés à la variable

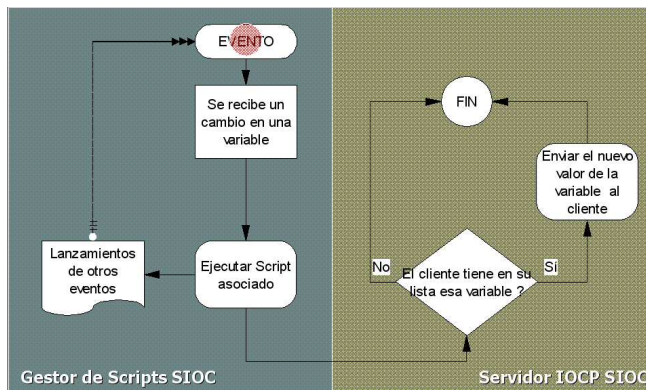
Chaque variable SIOC peut être associée à un script.

Le script c'est un complexe de commandes ou commandes d'exécution séquentielles.

Le script associé à une variable est exécuté lorsque cette variable modifie sa valeur.



Exécution des scripts associés



Langage SIOC

Langage de definition des variables.

On définit les caractéristiques des variables SIOC utilisées, et ses attributions en dépendance du LINK (lien) qu'on lui a assigné.

Langage des commandes d'exécution.

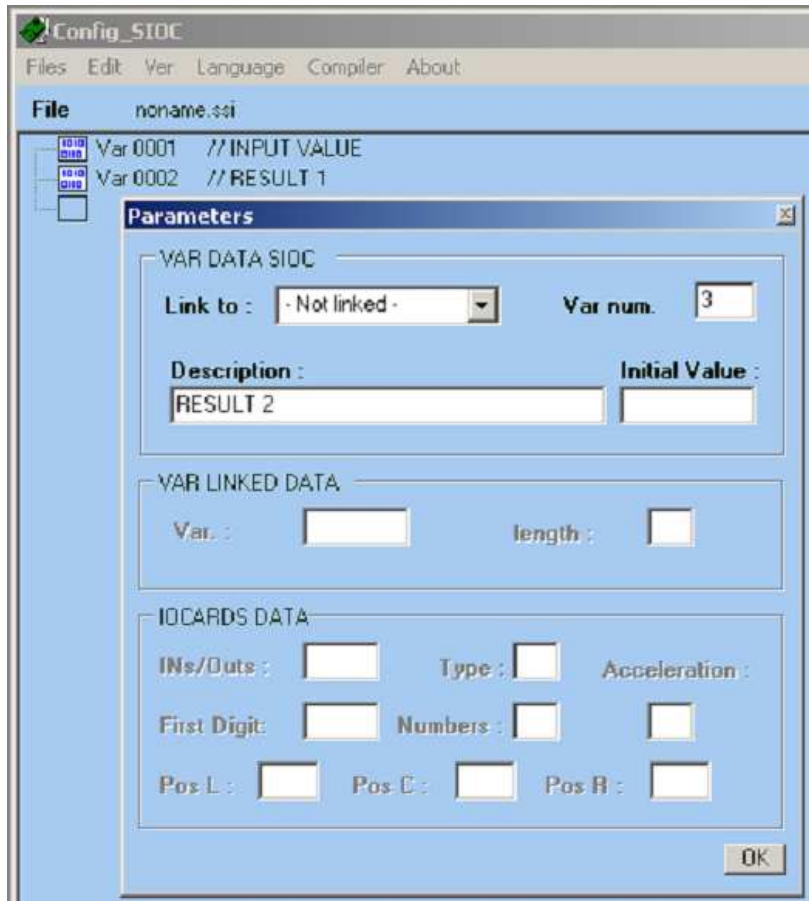
Composé de fonctions, assignations, conditions et autre genre de commandes qui sont exécutés en forme séquentielle par les variables définies.

Pratique 4: Exemple d' exécution des scripts associés.

Démarrer Config_SIOC.

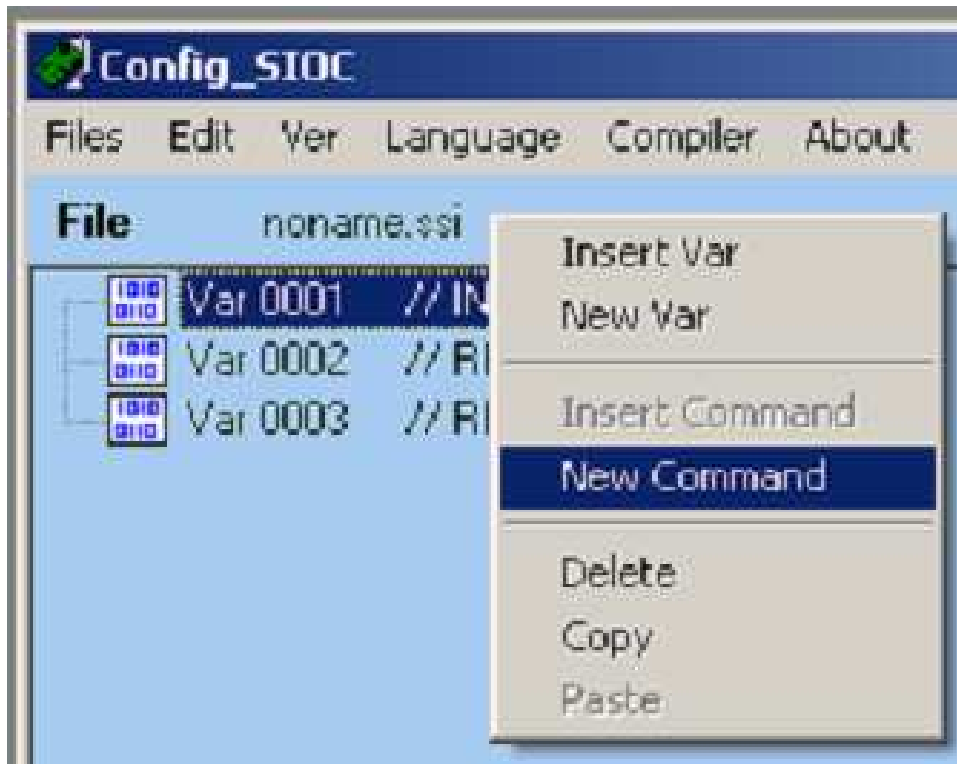
Créer 3 variables sans aucun Link.

Créer des lignes de commandes pour le script associé.

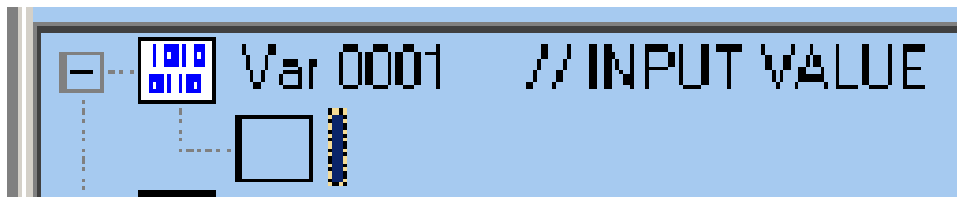


Pour insérer une commande dans le script associé à la variable, nous allons sélectionner la variable et appuyer sur bouton droit (ou par le menu Edit).

Sélectionner Nouvelle Commande.



Un carré vide s'affichera.



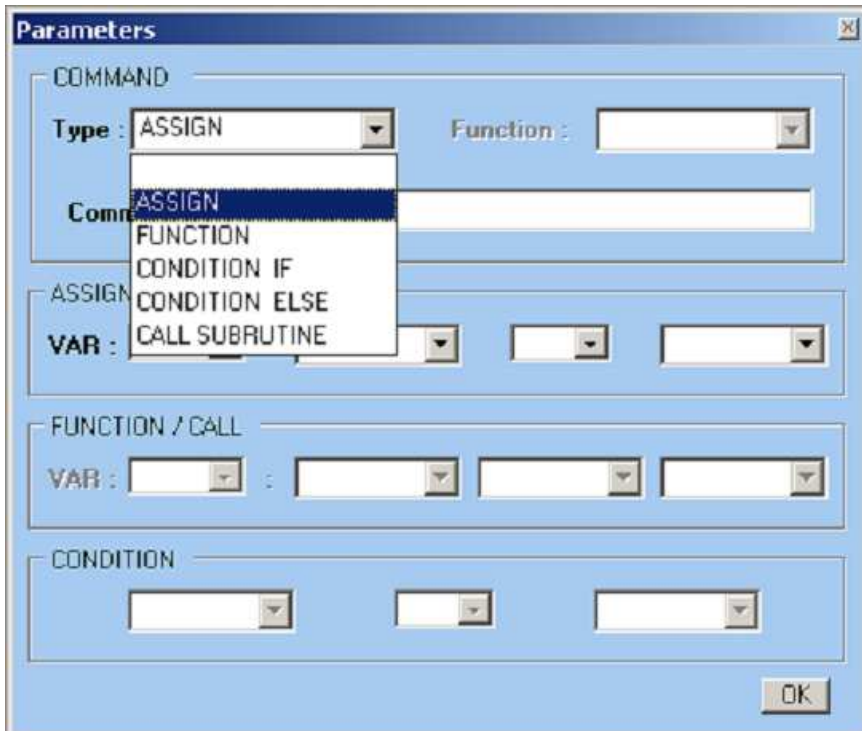
On peut répéter cette opération toutes les fois que nous voulons, pour créer ou insérer les commandes dont nous avons besoin.

On fait double click sur le carré vide pour faire apparaître le formulaire d'introduction des commandes



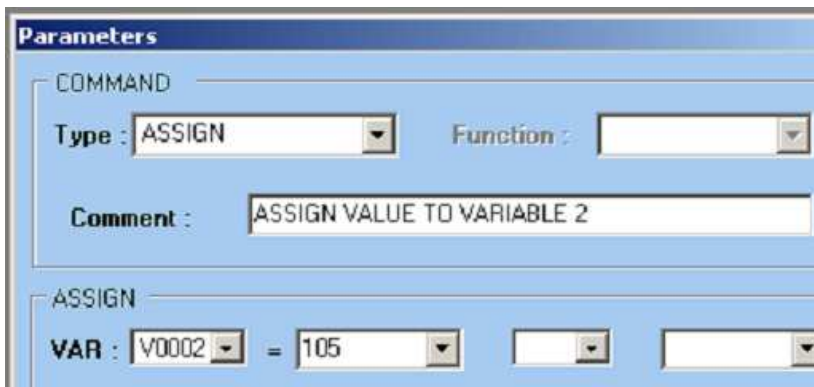
Le formulaire d'introduction des commandes nous permet d'introduire tous les données nécessaires.

Ouvrir les commandes et sélectionner ASSIGNATION



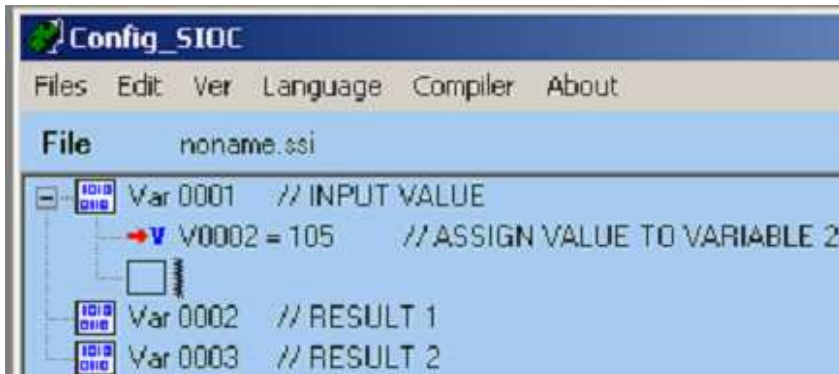
On peut insérer un commentaire pour cette commande.
Sélectionner la variable à la quelle nous allons assigner une valeur.

Introduire la valeur ou variable que nous voulons assigner et appuyer sur OK.



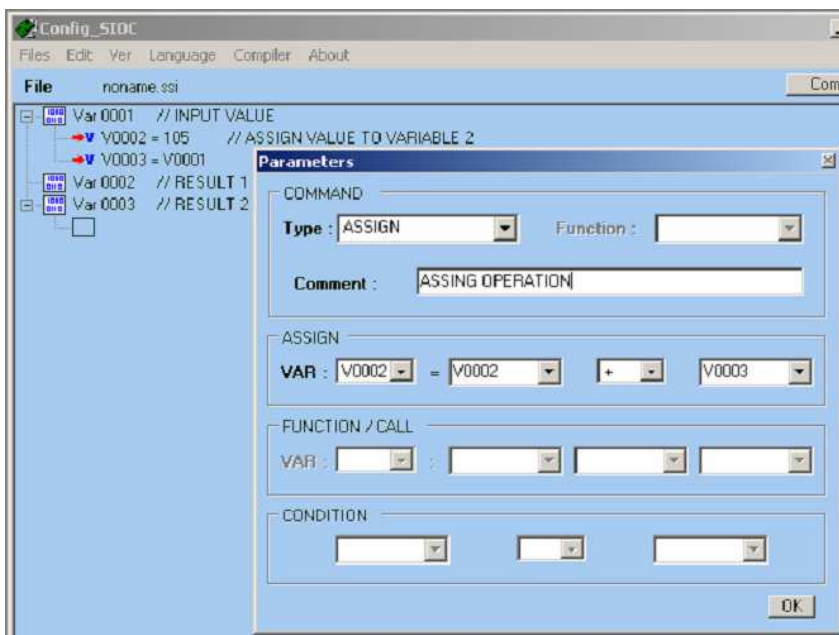
La ligne de la commande apparaîtra dessous de la variable avec l'icône correspondante a la commande.

Cette commande d'assignation fera que la variable V0002 prenne la valeur 105.

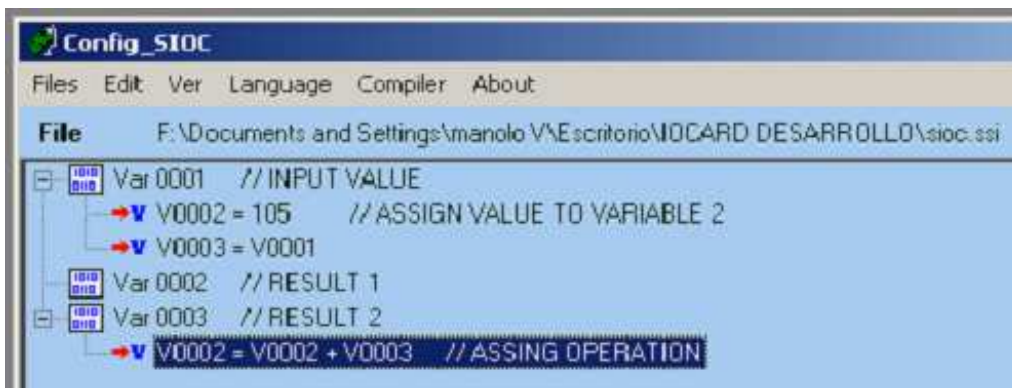


On fait la même chose avec la ligne suivante du script, mais cette fois on va assigner la variable V0001, pas une valeur.

Afficher une commande à la variable V0003 dans la quelle on assigne à la V0002 la valeur de la somme des variables V0002 et V0003.



Sauver la configuration dans le fichier sioc.ssi

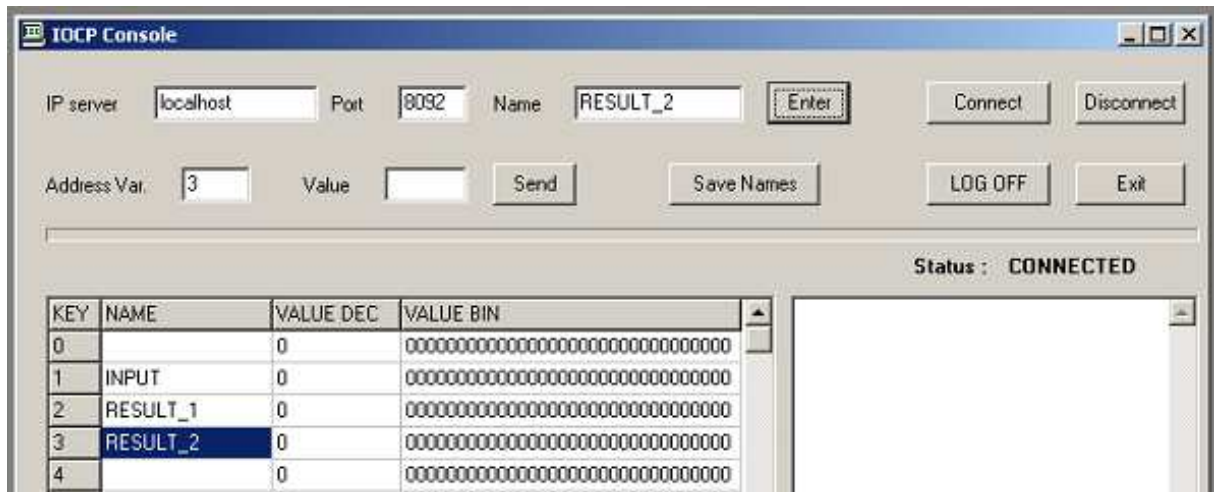


Démarrer SIOC

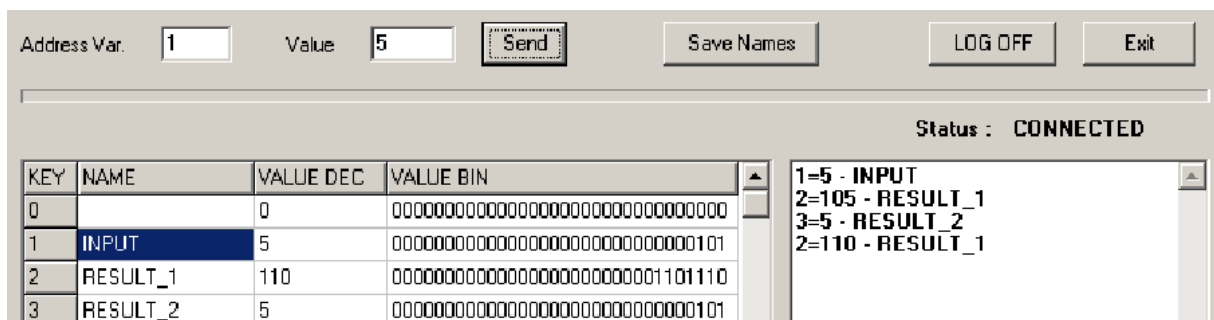
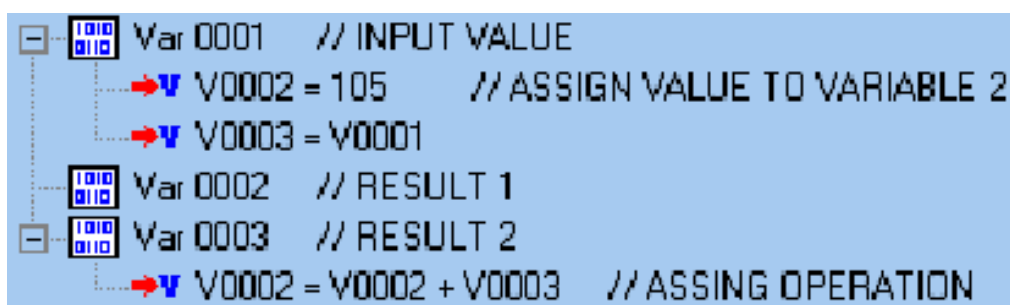
Démarrer IOCPConsole et connecter au SIOC



Mettre un nom à chaque variable utilisée
Activer le LOG



Modifier dans Console la valeur de la variable V0001, affichant la nouvelle valeur 5



Immédiatement vont apparaître les nouvelles valeurs des variables après que les scripts ont été exécutés.

Sur le LOG nous pouvons voir comment change la variable V0001 avec la nouvelle valeur 5.

A ce moment les scripts associés aux variables sont lancés

```

- [IO] Var 0001 // INPUT VALUE
  -> V V0002 = 105 // ASSIGN VALUE TO VARIABLE 2
  -> V V0003 = V0001
- [IO] Var 0002 // RESULT 1
- [IO] Var 0003 // RESULT 2
  -> V V0002 = V0002 + V0003 // ASSING OPERATION
  
```

```

Status : CONF
1=5 - INPUT
2=105 - RESULT_1
3=5 - RESULT_2
2=110 - RESULT_1
  
```

Après que la valeur de V0001 a changé, la première commande du script associé à la variable est exécutée.

Dans la commande on assigne la valeur 105 à la variable V0002.

La variable V0002 n'a pas de script associé, de conséquence elle va exécuter séquentiellement toutes les commandes qui sont incluses. Depuis ,sera exécutée la suivante commande du script associé à la variable V0001.

Cette commande assigne la valeur de la variable V0001 (qu'il est 5) à la variable V0003. Lorsque la valeur de la variable V0003 va changer, seront exécutées les commandes du script associé à cette variable.

Ce script assigne à la variable V0002 la valeur actuelle d'elle même (en ce moment 105) plus la variable V0003 (actuellement 5).

Pratique 5: Connection avec les IOCards

Fonctionnement du module IOCards :

LINK à IOCARD SW

Lorsqu'une entrée de la IOCard (pulseurs, interrupteurs...) passe à un état OFF, le module envoie un 0 à la variable qui est liée avec le link.



Lorsqu'une entrée de la IOCard passe à un état ON, le module envoie un 1.



LIEN A IOCARD OUT

Lorsque la variable prend la valeur 0, la sortie de la IOCard (leds, relais, ...) passe à un état OFF.

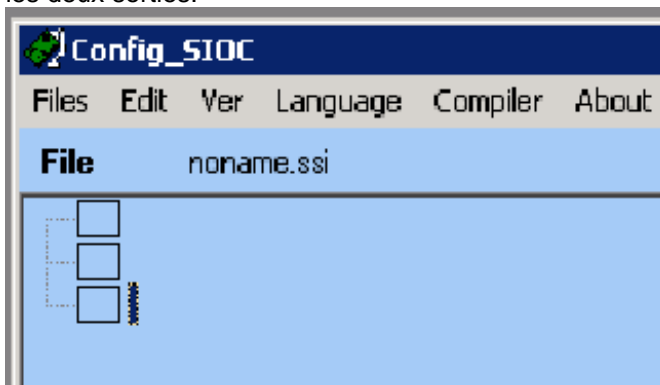


Si la variable prend la valeur 1, la sortie passe à un état ON.



Dans cet exemple nous allons connecter un interrupteur à l'entrée #8 de la carte Master et 2 leds à la sortie #15 et #16, de façon que lorsque nous allons connecter l'interrupteur, un des deux leds va s'allumer et l'autre va s'éteindre, et lorsque nous irons déconnecter, celui qui était allumé sera éteint et vice-versa.

Comme d'habitude nous allons démarrer Config_Sioc et nous allons créer 3 variables pour l'entrée et les deux sorties.



Double click sur la première variable et nous allons la définir.
Introduire les données et créer un link à IOCARD_SW
Faire click sur OK et définir la variable suivante.

Parameters

VAR DATA SIOC

Link to : Var num.

Description : Initial Value :

VAR LINKED DATA

Var. : length :

IOCARDS DATA

INs/Outs : Type : Acceleration :

First Digit: Numbers :

Pos L : Pos C : Pos R :

OK

Introduire les données et créer un link à IOCARD_OUT
 Faire clic sur OK et définir la variable suivante de la même façon pour le Led 2.

Parameters

VAR DATA SIOC

Link to : Var num.

Description : Initial Value :

VAR LINKED DATA

Var. : length :

IOCARDS DATA

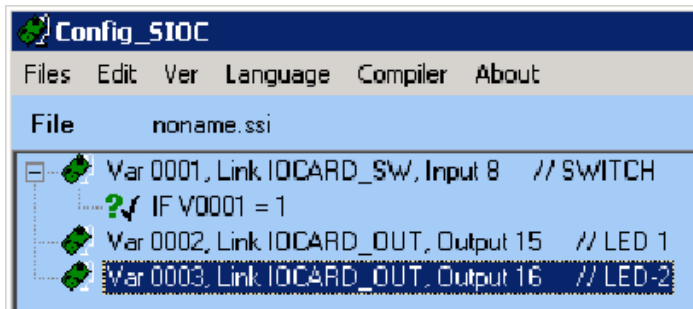
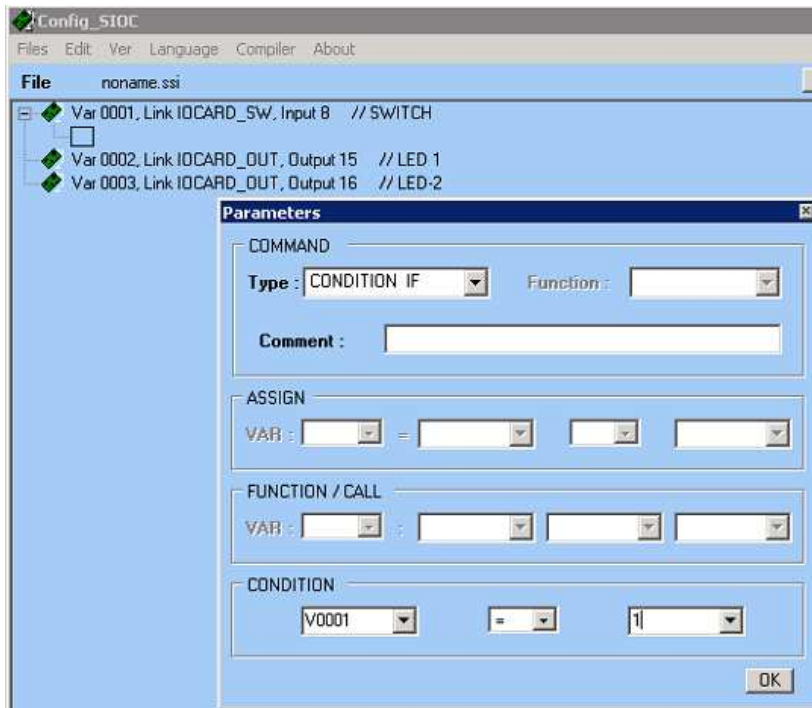
INs/Outs : Type : Acceleration :

First Digit: Numbers :

Pos L : Pos C : Pos R :

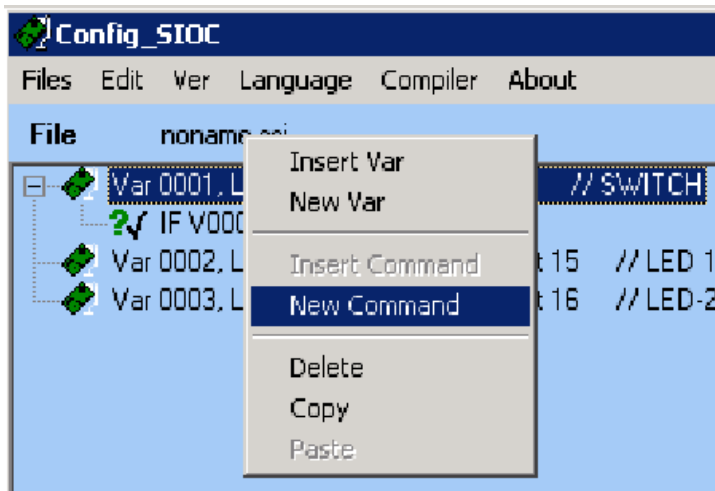
OK

Créer une ligne de commande et faire double click.
 Sélectionner type Condition IF.
 Dans la case CONDITION sélectionner V0001.
 Ouvrir les opérateurs de la condition et sélectionner '='.
 Dans la dernière case mettre la valeur 1.

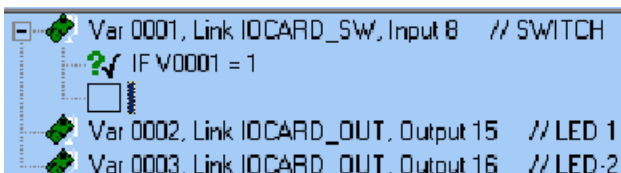


Sera affichée la commande de condition IF.

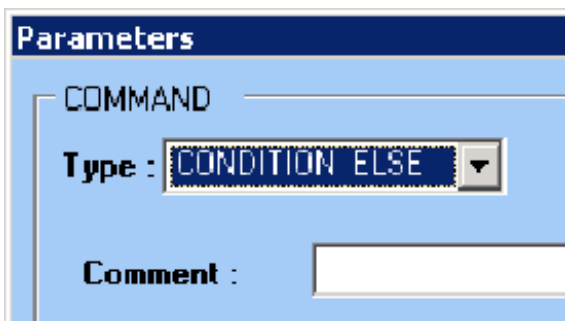
Pour créer d'autres lignes de commande, on ne peut pas se positionner sur la ligne de Condition IF ou ELSE, car c'est une ligne de niveau supérieur.
 Cliquer droit et Nouvelle Commande.



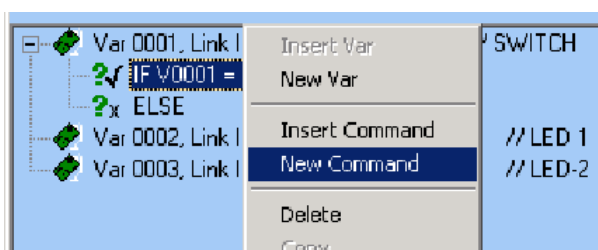
De nouveau double clic pour introduire la commande.



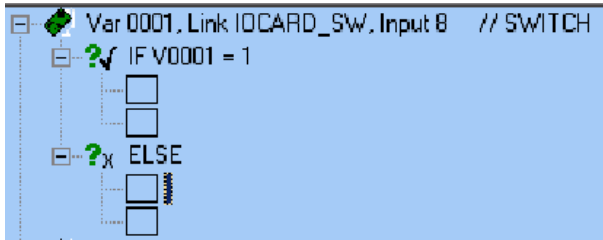
Sélectionner type Condition ELSE et appuyer OK car ce type n'admet pas d'autres paramètres.



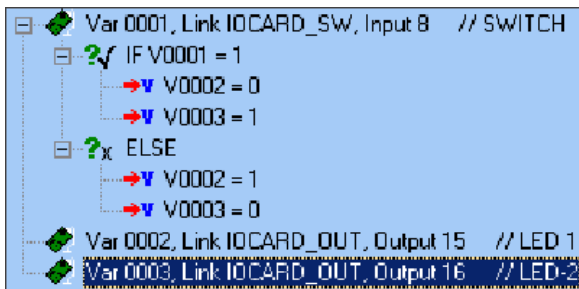
Se positionner sur la ligne de commande de la Condition IF et ELSE.
Appuyer le bouton droit et sélectionner Nouvelle Commande et créer 2 lignes pour chaque condition.



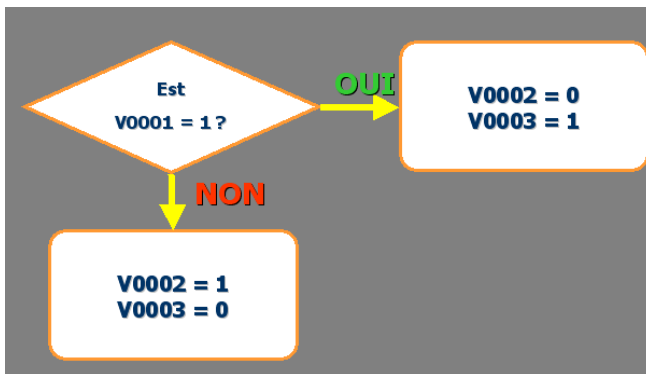
De conséquence les lignes que nous allons créer seront sur un autre niveau.
Pour indiquer dans quel moment elles devront être exécutées.



Définir les lignes de commandes avec les différentes assignments de façon que ce script va apparaître:



Lorsque V0001 change la valeur, la séquence d'exécution des commandes se déroulera en fonction de la valeur de la variable V0001 comme il est défini dans la commande de Condition IF.

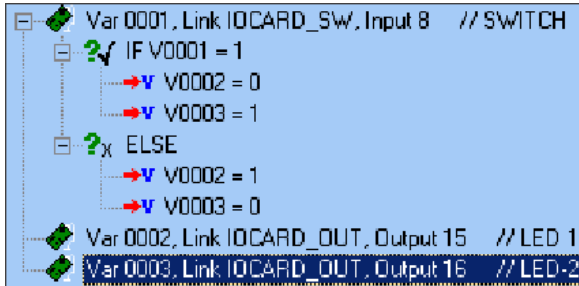


Langage SIOC en text editor

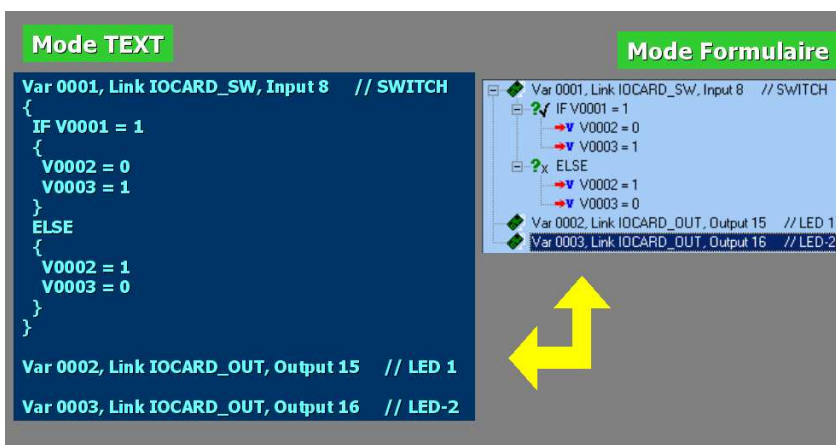
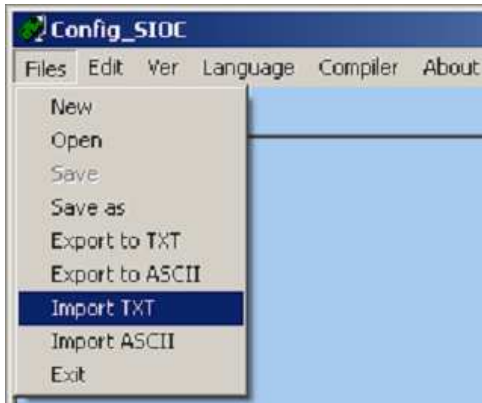
Toutes les configurations que nous sommes en train de réaliser par config_sioc, on peut également les réaliser avec un langage de texte écrit.

On peut utiliser un text editor et écrire le langage formel SIOC suivant les normes d'édition établies.

Le script de SIOC peut être créé comme on a vu précédemment.



Mais également, ce script peut être créé par un texte dans un editor (Bloc de notes, Word, ...) et depuis importé dans Config_Sioc ou il sera traduit en codes SIOC.

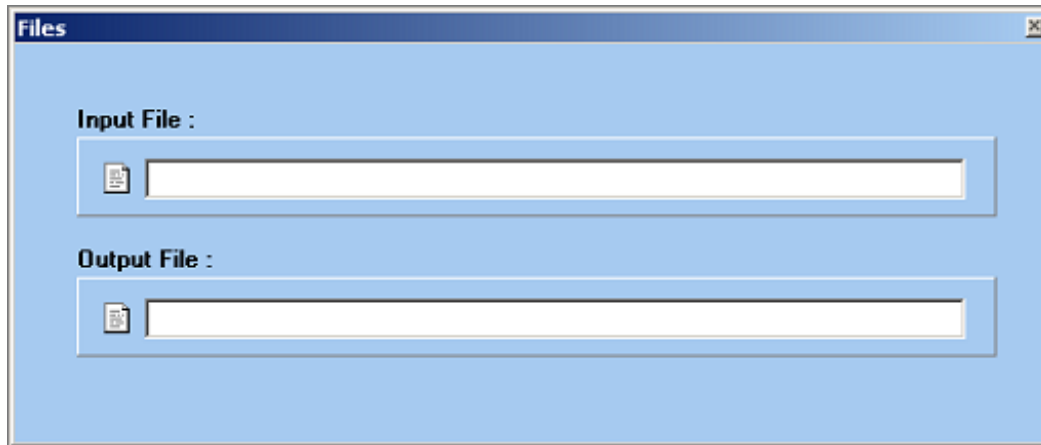


Si on travaille toujours en mode texte on peut utiliser l'option fichiers.



Soit le fichier text de input .TXT

Comme le fichier de output .SSI seront testés et générés par le code SIOC



Chaque fois qu'on appuie COMPILER, le Config_Sioc se charge de lire le fichier texte et de générer les codes SIOC.



Caractéristiques générales du langage des scripts

La séparation entre variables, constants, ou autres éléments est toujours un espace, virgule, parenthèse ou tab.

On utilise les parenthèses { } pour indiquer les différents niveaux.

On peut utiliser // au commencement de la ligne comme commentaire de text general, ou en fin d'une ligne de commande, comme commentaire particulier.

Une seule definition ou commande par ligne est admise.

Aucune différence entre majuscule et minuscule

Définition des script associés à une variable SIOC

```
Var Numéro 0-9999 Attribut paramètre Attribut paramètre ...
{
  Commande paramètres
  Commande paramètres
  .
  .
  .
}
```

exemple

```
Var 0001, Link FSUIPC_IN, Offset $07F2, Length 2
{
  V0002 = V0001 + 1
  CALL = V9999
  L1 = V0001 * 1.35
  L1 = ROUND L1
}
```

COMMANDES reconnues par SIOC

COMMANDE	DESCRIPTION
Assignment	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 subroutine 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

Opérateur	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

Pratique 6: Connecter les IOCards au Simulateur

Variables Internes Temporales

Pour la programmation du SIOC on peut utiliser différentes types de variables internes temporales, qui sont opérationnelles seulement à l'intérieur du script associé à une variable SIOC.

Ces variables peuvent être de 2 genre :

Réelles, qui se définissent comme L0, L1 y L2 et peuvent gérer valeurs décimales, entières, positives ou négatives dans le range 5×10^{-324} a 1.7×10^{308} .

Booléennes, qui se définissent comme C0, C1 y C2 et peuvent gérer valeurs d'une condition qui peut être vraie ou fausse.

On peut donner une référence à une variable SIOC dans nos scripts, avec la definition V et le numéro de la variable SIOC. Par exemple V0001

Dans cet exercice nous irons lire le N1 du moteur-1 de notre avion et nous irons l'afficher sur un display de 3 digits de 7 segments.

En premier il faudra trouver sur la liste des offsets le correspondant au N1 du moteur-1.

0898	2	Engine 1 Jet N1 as 0 – 16384 (100%), or Prop RPM (derive RPM by multiplying this value by the RPM Scaler (see 08C8) and dividing by 65536).
------	---	---

Deuxièmement nous devrions indiquer la formule pour calculer le pourcentage de N1 dans le range 0-100%, sera $N1 = \text{OFFSET} * (100 / 16384)$.

Finalement nous devrions considérer que FlightSimulator dans ses jauges ne peut pas ARRONDIR les calculs sans les TRANCHER, et pourtant nous figurons cette fonction

On démarre avec la definition de la variable de l'offset suivant les données que nous repérons sur la liste. Adresse \$0898 et longueur 2 :

Var 0001, Link FSUIPC_IN, Offset \$0898, Length 2 // ENGINE1 N1 0-16384 (100%)

Notre Display commence par le digit #0 de la IOCard Display et il a 3 digits :

Var 0002, Link IOCARD_DISPLAY, Digit 0, Numbers 3 // 3 DIGITS

Le module IOCARD_DISPLAY va trouver la valeur de la variable V0002 et va la passer directement au display suivant les digits définis dans Numbers.

De suite nous allons inclure les commandes que le script associé à la variable V0001 doit exécuter, avec les instructions de l'offset.

En premier nous irons utiliser une variable interne (L0) pour gérer la valeur temporelle du calcul du N1 suivant la formule établie :

L0 = V0001 * 0.00610352

0.00610352 c'est le résultat de 100 divisé par 16384.

Ensuite nous devons TRUNCER la valeur finale, et alors nous allons utiliser la fonction TRUNCER qu nous sera utile pour assigner la valeur à la variable V0002 qui fera apparaître l'information sur notre display.

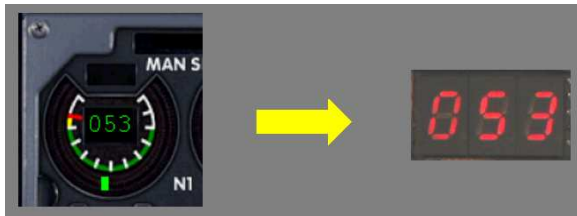
V0002 = TRUNC L0

Démarrer Config_Sioc et importer le fichier de text.
Sauver le fichier.SSI généré.
Démarrer FlighSimulator et SIOC

Notre programme apparaîtra comme ça :

```
Var 0001, Link FSUIPC_IN, Offset $0898, Length 2 // ENGINE1 N1 0-16384 (100%)
{
  L0 = V0001 * 0.00610352
  V0002 = TRUNC L0
}
Var 0002, Link IOCARD_DISPLAY, Digit 0, Numbers 3 // 3 DISPLAYS OF MY PANNEL
```

Le N1 du simulateur affichera sur notre display :



ANNEXE

Définition formelle du langage SIOC

Définition de script associé à une variable SIOC

```
Var Numéro 0-9999 Attribut paramètre Attribut paramètre ...  
{  
  Commande paramètres  
  Commaneo paramètres  
  .  
  .  
  .  
}
```

exemple

```
Var 0001, Link FSUIPC_IN, Offset $07F2, Length 2  
{  
  V0002 = V0001 + 1  
  CALL = V9999  
  L1 = V0001 * 1.35  
  L1 = ROUND L1  
}
```

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 locards
	IOCARD_OUT	Active/désactive les sorties des locards
	IOCARD_DISPLAY	Envoie les chiffres aux locards display
	IOCARD_ENCODER	Reçoit les informations des encodeurs
	IOCARD_ANALOGIC	Reçoit les informations d'entrées analogiques
	IOCARD_SERVO	Fait bouger les servos locards
	IOCARD_MOTOR	Gère les moteurs DC pas à pas
SIOC	SUBROUTINE	Gère les variables en sous-routine

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

COMMANDES reconnues par SIOC

COMMANDE	DESCRIPTION
Assignment	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 subroutine 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

Opérateur	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

Liaison d'une variable SIOC avec les différents modules

Var 0001

Définit la variable SIOC 0001, qui est prête à être utilisée

Var 0006 , Value 360 // heading

Ici nous assignons une valeur initiale et un commentaire

Var 1387, Link FSUIPC_OUT, Offset \$0BDE, Length 2

Liaison avec le module FSUIPC, offset \$0BDE de longueur 2.

Si la valeur de la variable 1387 change, elle sera envoyée à FSUIPC

Var 9341, Link FSUIPC_IN, Offset \$0C32, Length 4

La variable 9341 reçoit la valeur de l'offset \$0C32 et lancera son script associé si cette valeur est différente de celle qu'il a.

Liaison d'une variable SIOC avec les différents modules

Var 0009 , Link IOCARD_SW, Input 25, Type P

Liaison de la variable 0009 avec l'élément Switch du module IOCard , Qui deviendra 0 si l'entrée c'est OFF et 1 si l'entrée c'est ON.

Var 0006 , Link IOCARD_OUT, Output 54 // Led du F/D

Si la variable 0006 prend la valeur 1, la sortie IOCard 54 s'active, si Prend la valeur 0 la sortie se désactive.

Var 1387, Link IOCARD_DISPLAY, Digit 0, Numbers 3

La valeur de la variable, sera affichée sur les displays à partir de #0 et avec 3 chiffres. Si on veut utiliser des négatifs il faudra ajouter une chiffre.

Var 9341, Link IOCARD_ENCODER, Input 3, Accélération 1, Type 1

Chaque click de l'encoder enverra à la variable un valeur entre 1 et 1 plus le coefficient de accélération. L'encoder va se connecter à partir de l'entrée 3. La variable prend la valeur 0 sans provoquer un événement.

Liaison d'une variable SIOC avec les différents modules

Var 0009 , Link IOCARD_ANALOGIC, Input #2, PosL 1, PosC 127, PosR 255

La variable reçoit la valeur de l'entrée analogique 2, calibrée aux valeurs 1, 127 et 255 pour gauche, centre et droite.

Var 1209 , Link IOCARD_SERVO, Output 7, PosL 1, PosC 127, PosR 255

Le servo reçoit la valeur de la variable 1209, avec calibration.

Optionnellement on peut indiquer le type pour les servos de 10 bits.

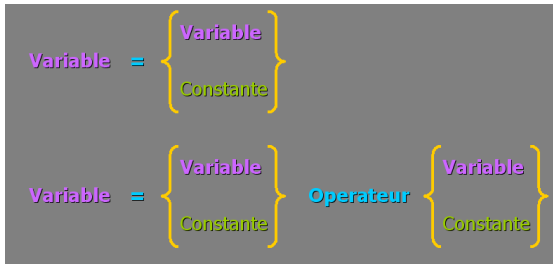
Var 1387, Link IOCARD_MOTOR, Ourput 187, Accélération 14, Type D

Suivant la variable, le moteur connecté sur la sortie 187 va tourner avec le coef. d' accélération 14, d'un coté si c'est <127 et de l'autre si c'est >127.

Var 9999, Link SUBROUTINE

La variable 9999 devient une SUBROUTINE, qui sera lancée , avec son script associé, automatiquement par la commande CALL.

Commande d' ASSIGNATION



exemple

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

Définition de FONCTIONS



paramètre peut être une variable ou une constante

Chaque fonction a 1, 2 ou 3 paramètres maxi et la valeur final est assignée à la Variable.

Exemple

```
{
  V0002 = Round L0
  L2 = ToBCD V0005
  C1 = TestBit V1234 5
  V9888 = Timer 100 5 10
}
```

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 incréments/décréments 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

Fonctions de SIOC

Variable = **Round** paramètre1

Arrondit la valeur à l'entier plus proche.

Paramètre 1 : Variable, constante réelle ou entière.

Variable = **Trunc** paramètre1

Tranche les décimaux du paramètre et le convertit en entier.

Paramètre1 : Variable, constante réelle ou entière.

Variable = **SetBit** paramètre1

Met à 1 le bit de la Variable affichée dans le paramètre

Paramètre1 : Variable, constante réelle ou entière.

Variable = **ClearBit** paramètre1

Met à 0 le bit de la Variable affichée dans le paramètre

Paramètre1 : Variable, constante réelle ou entière.

Variable = **Toggle** paramètre1

Fait un toggle (mettre à 1 et après à 0) à un bit de la variable. (Le bit indiqué par le paramètre).

Paramètre1 : Variable, constante réelle ou entière

Variable = **ToBCD** paramètre1

Change la valeur du paramètre1 au format BCD et l'assigne à la variable. (Ce format est utilisé souvent dans FSimulator).

Paramètre1 : Variable, constante réelle ou entière.

Variable = **Abs** paramètre1

Assigne à la variable la valeur absolue (positive) du paramètre.

Paramètre1 : Variable, constante réelle ou entière.

Variable = **FromBCD** paramètre1

Change la valeur du paramètre1 de BCD à décimale et il l'assigne à la variable.

Paramètre1 : Variable, constante réelle ou entière.

Variable = **TestBit** paramètre1

Assigne à la variable booléenne un vrai ou faux selon que le bit indiqué dans le paramètre est 1 ou non.

Paramètre1 : Variable, constante réelle ou entière.

Variable = **Not** paramètre

Assigne à la variable booléenne la valeur contraire à celle du paramètre1

Paramètre1 : Variable booléenne.

Variable = **Rotate** paramètre1 paramètre2 paramètre3

Donne de l'essor ou diminution à la variable, dans la valeur du paramètre 3. Si elle dépasse la valeur du paramètre 2, la variable revient à la valeur inférieure du paramètre 1, et le contraire si c'est une diminution, de façon à réaliser un essor/diminution cyclique, typique des systèmes comme l' OBS du VOR du simulateur, etc.

Paramètre1 : Variable, constante réelle ou entière. Valeur ou inférieure.

Paramètre2 : Variable, constante réelle ou entière. Valeur supérieure.
Paramètre3 : Variable, constante réelle ou entière. Essor/Diminution

Variable = ChangeBIT paramètre1 paramètre2

Change le bit de la variable, et il l'affiche dans le paramètre1 en fonction de la valeur du paramètre2.
A 0 si le paramètre2 est 0 et à 1 si c'est 1.

Paramètre1 : Variable, constante réelle ou entière. Bit qui doit changer
Paramètre2 : Variable, constante réelle ou entière. Moyen de changement

Variable = ChangeBITN paramètre1 paramètre2

Change le bit de la variable, et il l'affiche dans le paramètre1 en fonction de la valeur du paramètre2.
A 0 si le paramètre2 est 1 et à 1 si c'est 0.

Paramètre1 : Variable, constante réelle ou entière. Bit à changer
Paramètre2 : Variable, constante réelle ou entière. Moyen de changement

Variable = Limit paramètre1 paramètre2 paramètre3

Donne de l'essor ou diminue la variable, à la valeur du paramètre3. Si elle dépasse la valeur du paramètre 2, la variable s'affiche à la valeur du paramètre3, et si c'est inférieure au paramètre1, également s'affiche à la valeur du paramètre3.

Paramètre1 : Variable, constante réelle ou entière. Valeur inférieure.
Paramètre2 : Variable, constante réelle ou entière. Valeur supérieure.
Paramètre3 : Variable, constante réelle ou entière. Donne de l'essor/Diminue

Variable = Div paramètre1 paramètre2

Fait la division entière entre le paramètre1 et le paramètre2.

Paramètre1 : Variable, constante réelle ou entière
Paramètre2 : Variable, constante réelle ou entière

Variable = Mod paramètre1 paramètre2

Calcule le module ou reste de la division entre le paramètre1 et le paramètre2.

Paramètre1 : Variable, constante réelle ou entière
Paramètre2 : Variable, constante réelle ou entière

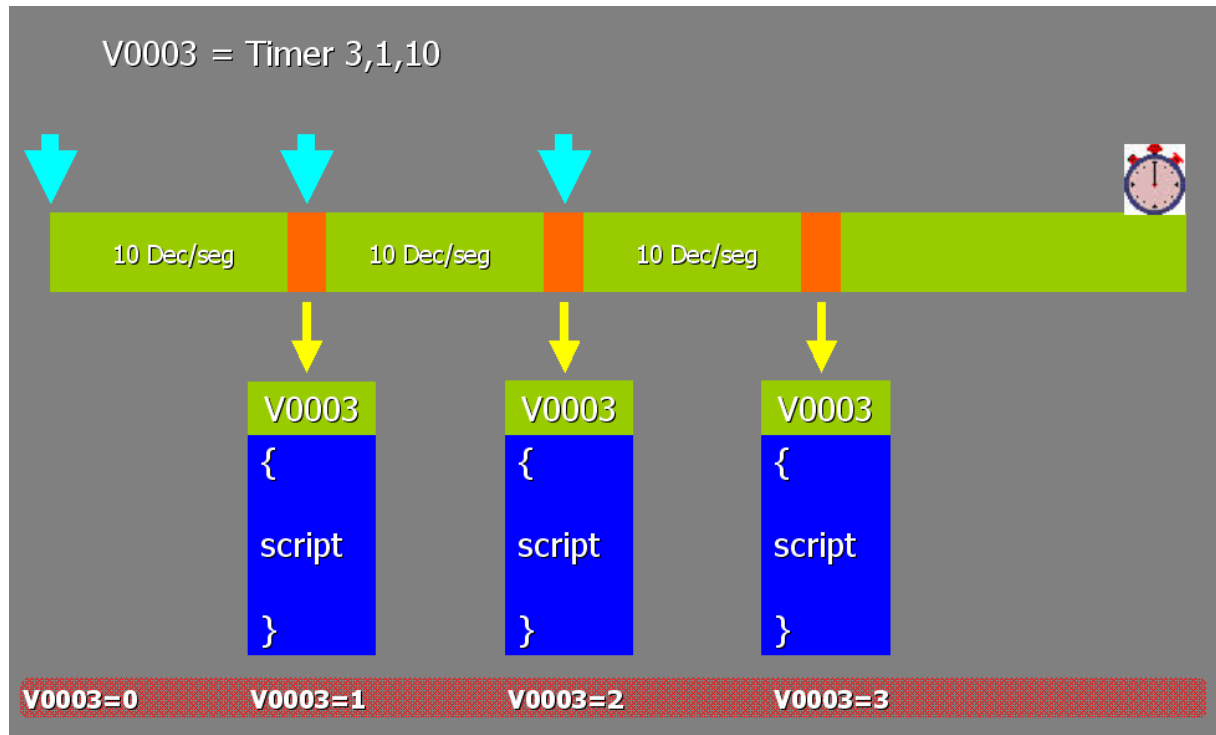
Variable = Timer paramètre1 paramètre2 paramètre3

Lance le script associé à la variable périodiquement, suivant l'indication du paramètre3 (en dixièmes de secondes).

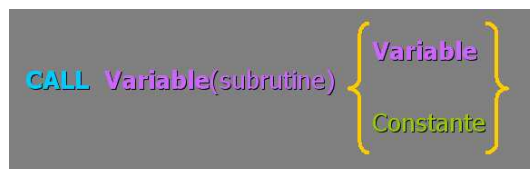
A chaque tour donne de l'essor ou diminue la variable suivant le paramètre 2, et s'arrête lorsque la variable rejoint la valeur indiquée par le paramètre 1.

Paramètre1 : Variable, constante réelle ou entière. Valeur finale.
Paramètre2 : Variable, constante réelle ou entière. Essor/Diminution.
Paramètre3 : Variable, constante réelle ou entière. Périodicité.

Fonction TIMER



Commande CALL



Exécute le script associé à la variable, et s'il y a un paramètre, la variable prend cette valeur. En tous cas, toujours, le script appelé sera exécuté.

Exemple

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

Commandes : condition IF et condition ELSE

```
IF CONDITION
{
commande
}
ELSE
```

```
{  
  commande  
}
```

Seront exécutées seulement les commandes qui sont entre parenthèses de la commande IF si la condition est satisfaite, si non seront exécutées les commandes entre parenthèses de la commande ELSE.

La commande de condition ELSE est facultative, et s'il y a, elle doit toujours se trouver après la condition IF à la quelle fait référence.

Il est possible d'avoir consécutivement jusqu'à 100 niveaux de commandes de condition.

La condition peut être une variable booléenne, union de 2 variables booléennes par AND / OR, ou conditions entre variables et/ou constantes réelles par les opérateurs correspondants

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