Serveur Profibus

Description

Le Serveur Profibus permet :

- L'interrogation cyclique ou acyclique d'équipements connectées à la carte DPC102 HI.
- A une vitesse comprise entre 93750 Bauds et 6 Mbits. (45450 Bauds réservé à Profibus PA).
- Le serveur de communication **rafraîchit cycliquement** une base de données de variables en mémoire.
- Le serveur communication puise dans une base de données de variables les valeurs à envoyer vers les équipements paramétrés.
- Le serveur de communication se place comme une **tache prioritaire** dans la barre de taches de Windows.
- Le serveur de communication fonctionne tout aussi bien sous Windows 95, 98, NT et 2000.
- Le serveur de communication peut être relié à des programmes écrit dans différents langages évolués par le jeu de fonctions contenues dans une DLL (VPLC.DLL) ou par un objet Automation (inprocserver) contenu dans VPLCOM.DLL.

Les différents programmes installés par le programme Setup.exe contenu sur le disque fourni sont :

ServDP.exe: Le configurateur Profibus.

Server.exe: Le serveur de communication final fonctionnant avec les paramétrages défini par le configurateur.

Visudyna.exe: Le programme de débug permettant de visualiser les variables évoluant au sein du serveur.

Recense.exe: Un utilitaire pour recenser ou dé-recenser l'objet COM dans la base de registre de Windows.

Autres Fichiers :

Visuln32.DLL	: Communication mémoire à mémoire avec le serveur .
Vplc.dll	: Contient les procédures et les fonctions de communication avec le serveur.
Vplcom.dll	: Objet COM (Inprocserver) de communication avec le serveur.
Arsdp.dll	: DLL utilisé par le serveur pour communiquer avec la carte DPC102 HI.
VarVPu.Duc	: Librairie pour Delphi 4 et Delphi 5. Contient les mêmes procédures et fonctions que celles implémentées dans la DLL Vplc.dll
Arsbt.sys	: Driver Windows **NT et 2000** pour accès à la carte DPC102HI.
Sous Répertoire Test	: Contient une application Visual I/O complète de test en communication avec le serveur.
Sous répertoire VB	: Contient une application complète en Visual Basic, permettant de lire et écrire des variables dans le serveur Profibus.

OPC

RecenseOPC.Exe	Programme de recensement du serveur OPC dans votre machine. A exécuter une seule fois après installation.
ARSOPC.EXE	Transforme votre serveur Profibus en serveur OPC.
DOPCExplorer.EXE	Utilitaire de test de connexion à un serveur OPC.

ServDP.exe

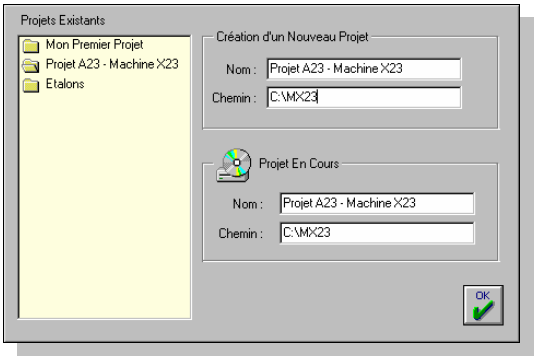
Ce programme permet le paramétrage des trames Profibus à envoyer cycliquement sur la liaison série :

Cliquez sur l'icône 

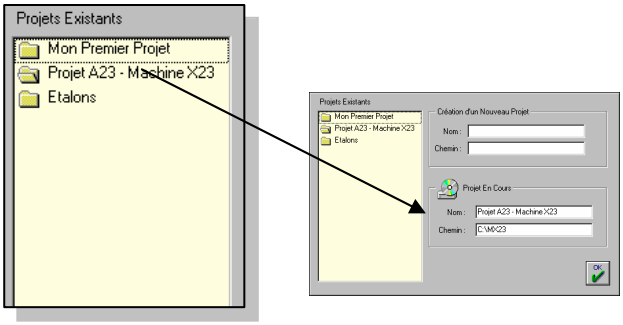
Première Etape

Au premier lancement, une fenêtre apparaît permettant de renseigner un nom de projet ainsi que la directorie dans lequel se trouve celui-ci :

Dans le champ **Nom**, frappez l'intitulé du projet. Ici Projet A23 - Machine X23.
Dans le champ **Chemin**, frappez le chemin complet sur le disque. Celui-ci contiendra tous les fichiers de votre application



Le nouveau projet créé apparaît dans la liste de gauche. A ce stade il n'est toujours pas validé. Double cliquez sur celui-ci pour ce projet nouvellement créé.



Deuxième Etape

Définition de la table image des communications

Après avoir défini un nouveau projet, vous devez définir les variables globales recevant les valeurs à transmettre vers les esclaves ou les variables recevant le résultat des communications en d'autres termes la table image Profibus.

Type de Variables dans le serveur Profibus

Boolean : Octet (True=1 False=0).
Byte : Octet 8 bits. (Bits de 0 à 7)
Word : Mot 16 bits. (Bits de 0 à 15).
Integer : Entier 32 Bits. (Bits de 0 à 31). Portée : -2147483648..2147483647
Real : Nombre en virgule Flottante. Portée : $1,9x10^{-4951}$.. $1,1x10^{4932}$
String : Chaîne de 255 caractères. (De 1 à 255).
Array : Tableau de type de variables définies ci-dessus.
Exemple : Array [0..12] of Boolean;

Adressage

Boolean

```
MBool   : Boolean;
MBool:=True; MBool:=False;      ou équivalent Mbool:=1; Mbool:=0;
```

Byte

```
By      : Byte;          By:=12;
```

Word

```
Wd      : Word;          WD:=12000;      WD:=$F012;
```

Integer

```
It      : Integer ;
It:=255; It:=$FF;        Affectation d'une valeur numérique;
It.0:=True; It.4:=False; Affectation d'un bit d'un Integer;
```

Real

```
RL      : Real;          RL:=-12.40;
```

String

```
ST      : String;        ST:='Visual PLC';ST:='C:\WinNt';
```

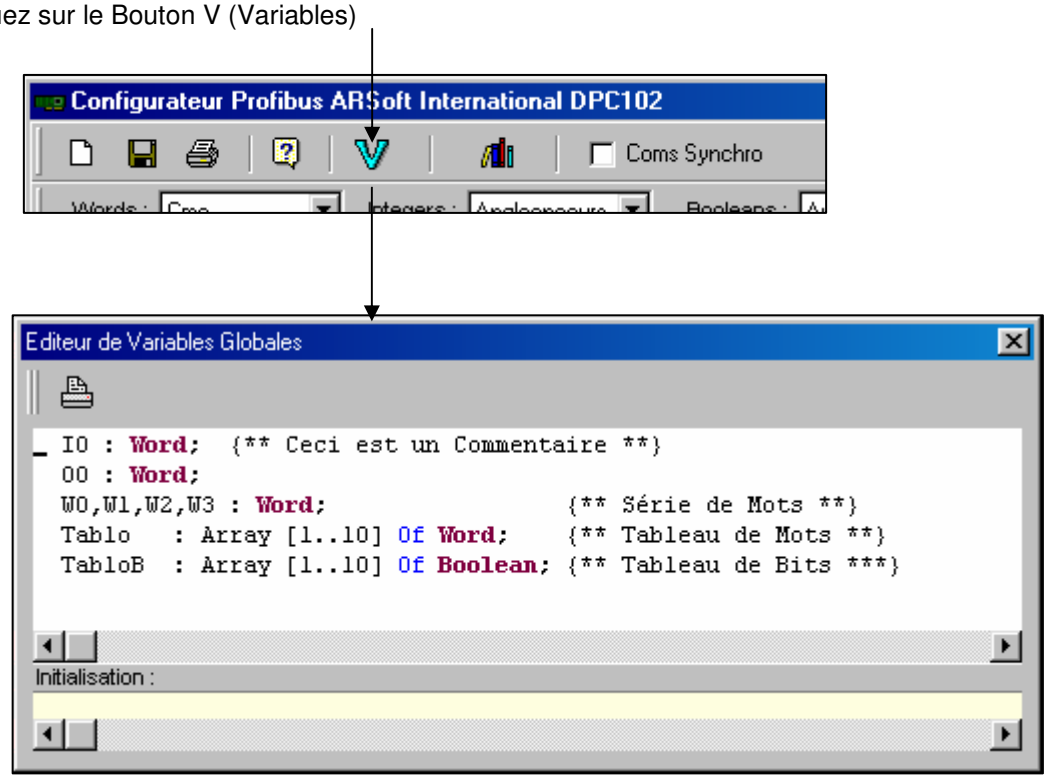
Array

```
AMBOOL: Array [0..10] of Boolean;AMBool[1]:=True;

ABY     : Array [1..50] of Byte;     ABY[2]:=34;

AWD     : Array [1..50] of Word;     AWD[2]:=12034;

AIT     : Array [1..30] of Integer;  AIT[4]:=-255;  AIT[5].0 :=True;

ARL     : Array [2..14] of Real;     ARL[2]:=-34.6789;

AST     : Array [0..6] of String;    AST[2]:='Bonjour';
```

Prenons comme exemple une table image correspondant à un bloc de 16 entrées physiques et à un bloc de 16 sorties physiques. Ces 16 Entrées physiques seront recopiées par le serveur Profibus dans la Variable **I0** et les 16 sorties seront affectées par le contenu de la variable **O0** à travers le serveur.

Exemple de déclaration :

Cliquez sur le Bouton V (Variables)



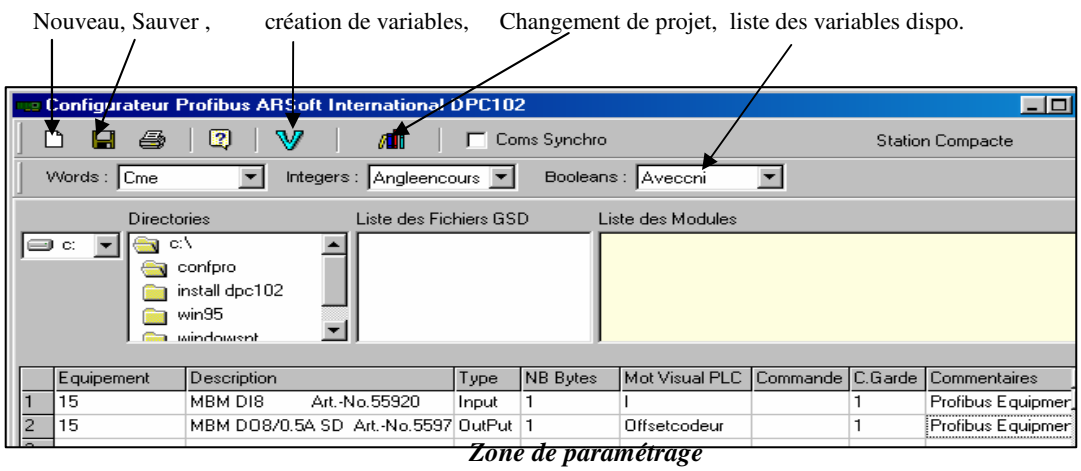
Ici I0,O0 est un Mot de 16 Bits. W0 à W3 sont 4 mots de 16 bits.
Tablo est un tableau de 10 Mots de 16 Bits. Tablob est un tableau de 10 Bits (Boolean).

Le choix des noms de variables constituant la table image est laissée libre à l'utilisateur. L'on peut imaginer des noms tel que CarteEntree1, Consigne1, Temperature10 Etc..

Les fonctions de la DLL ou de l'objet Automation auront besoin de ces noms pour adresser ces variables.

Possibilité de récupérer ou de forcer un bit d'un mot (ex : W0.0 ; CarteEnt.0 etc..)

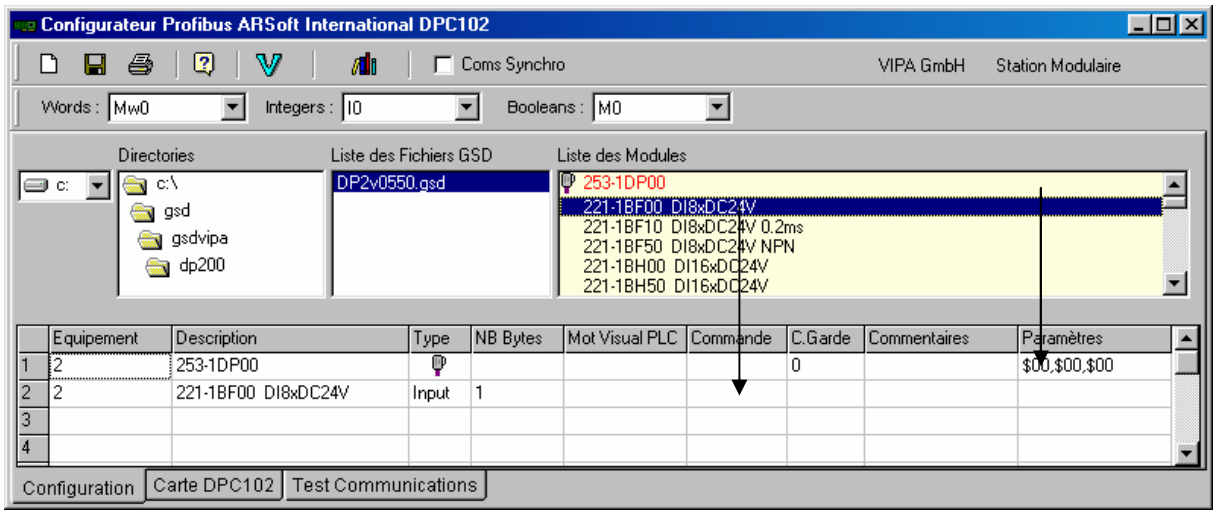
Vous ne pouvez pas paramétrer de communication, avant d'avoir défini des variables constituant la table image.



Zone de paramétrage

Exemple de paramétrage :

Choisissez un fichier **GSD** (d'extension .GSD) par navigation sur votre disque dur. La liste des modules apparaît dans la listbox de droite (Font jaune). Amenez **en premier** par Drag & Drop ou double clic une tête de station puis le type de module souhaité dans la grille.



En fonction du type de modules choisi, le nombre de bytes en Input et output s'affiche. Renseignant ainsi l'utilisateur sur le choix de la variable à inscrire dans la colonne Mot Visual PLC.

Equipement : Frappez le numéro de l'équipement (Roue codeuses sur la station profibus).
Description : Texte contenu dans le fichier GSD.
Type : INPUT (Entrées Profibus). OUTPUT (Sorties Profibus).
NB Bytes : Nombre d'Octets à lire (disponibles) dans l'équipement Profibus.

Mot Visual PLC : Premier mot de rangement des octets lus ou écrits dans l'équipement profibus concerné.

Commande : Si aucun bit n'est spécifié l'interrogation de l'équipement se fait cycliquement par la carte. Si un bit Visual PLC est spécifié l'interrogation de l'équipement concerné s'effectue lorsque ce bit (Booléen) est à 1. Lorsque la trame à été lue (l'équipement à répondu sans erreur) ce bit est remis à 0 (False) par la carte renvoyant ainsi un accusé de réception.

C.Garde : Chien de garde (WatchDog). Si aucune valeur n'est spécifiée, les sorties Profibus reste dans l'état même si plus aucune trame n'est envoyées par la carte. Si une valeur est spécifiée (x100ms) une trame doit être envoyée cycliquement par la carte avant d'atteindre cette valeur. Si ce n'est pas le cas remise à zéro des sorties et passage en Bus Fail par l'équipement concerné.

Commentaires : Vos commentaires.

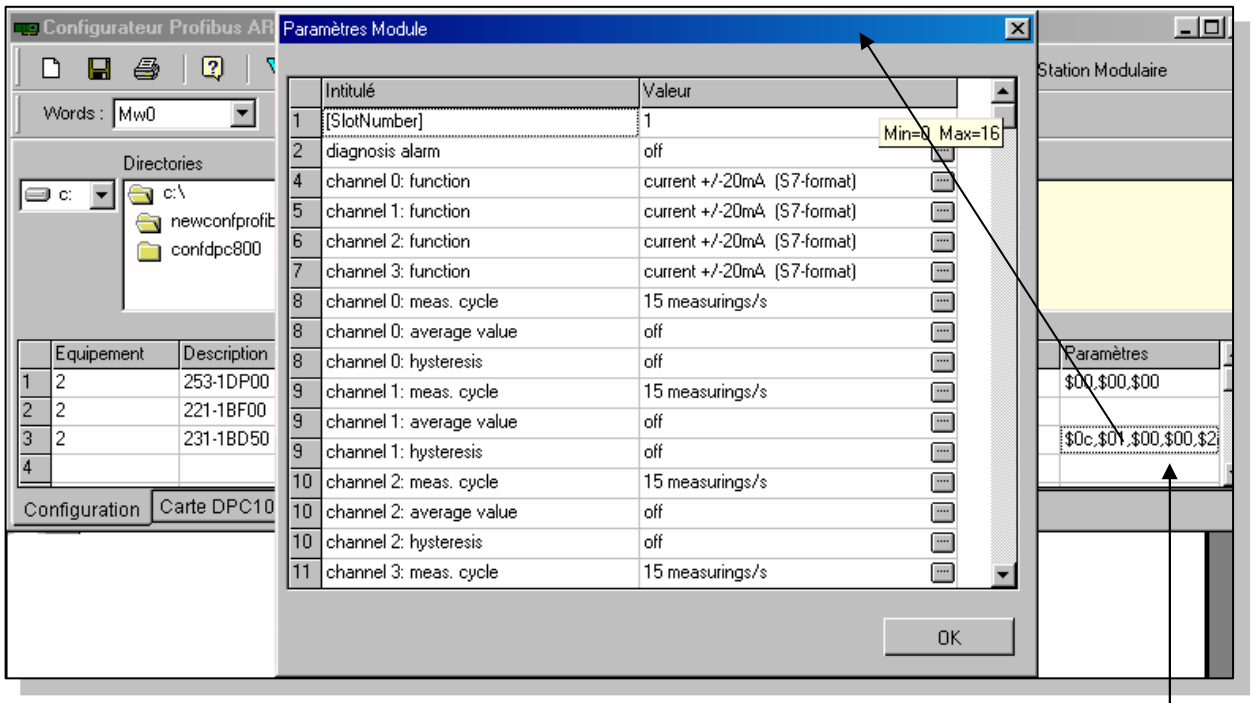
Paramètres : Permet de visualiser ou de modifier les paramétres d'initialisation de la station et des modules. En doublecliquant sur la cellule une boite de paramètres apparaît..

Case à cocher Coms Synchro : Si cette case est cochée, le driver généré est différent permettant **d'enchaîner** systématiquement toutes les trames profibus paramétrées toutes les 10 ms. Si cette case à cocher est décochée, les trames profibus seront envoyées une à une toutes les 10 ms.

Sauvez, le configurateur compile un driver par rapport à votre configuration et l'inclus dans le serveur de communication

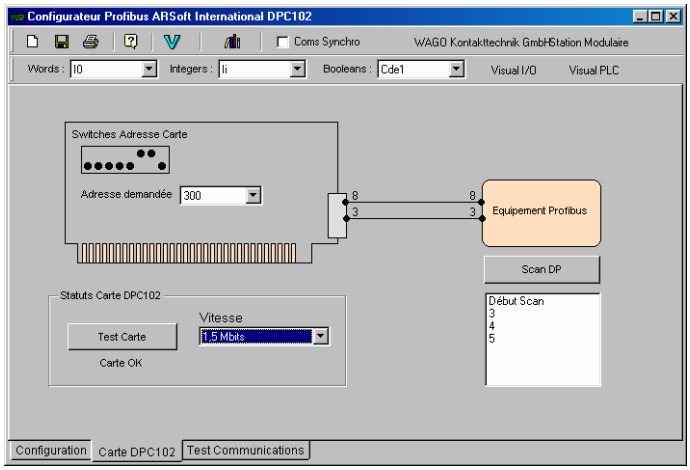
Modifier des paramètres Profibus.

Certaines cartes sont paramètrables (cartes entrées analogiques par exemple), le configurateur permet de régler les paramètres qui seront envoyés à l'initialisation.



Doublecliquez sur la cellule dans la colonne paramètres. Une fenêtre de réglage apparaît.

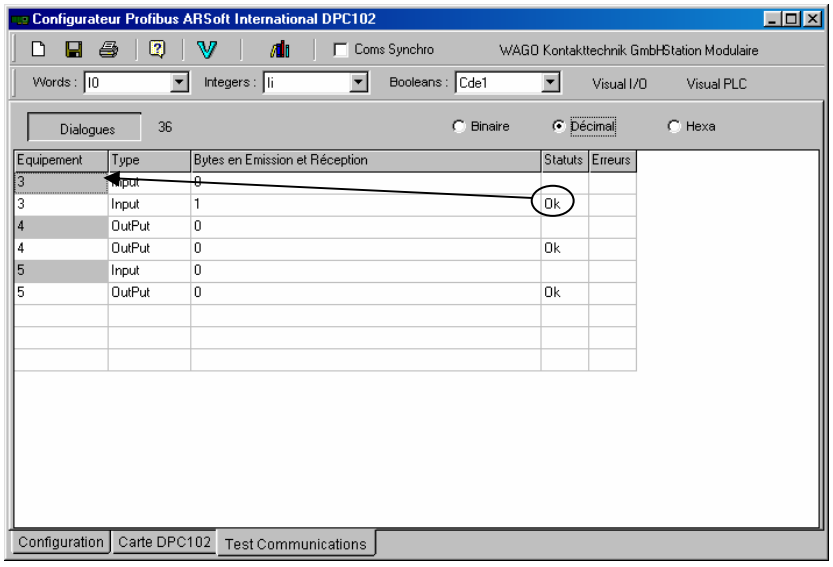
Test de la carte Profibus et ScanDP.



Cliquez sur le bouton **Test Carte** pour tester entièrement la carte profibus (Mode « loop Back »)
Cette commande indique la position des Switches sur la carte. Si la carte n'est pas détectée, il est probable que celle-ci soit en conflit d'adresse avec d'autres carte sur le bus. Choisissez une adresse dans le Combo 'Adresse demandée', le système vous indique alors la nouvelle position des Switches.

Scan DP : Permet de scanner les équipements connectés sur le bus Profibus. Du fait de l'auto-adaptation de la vitesse, les esclaves ne répondent pas forcément tout de suite.
Cette commande est à faire plusieurs fois.
Certains esclaves mémorisent en interne la dernière vitesse, il faut donc couper leur alimentation pour de vitesse.

Test de dialogue avec les équipements profibus connectés:



Cliquez sur le bouton Dialogues, une grille apparaît affichant l'état des octets d'échange lus et écrits dans les équipements configurés. Vous pouvez alors forcer certains en affectant une nouvelle valeur dans la cellule concernée.
Les cellules grisées dans la colonne Equipement, indiquent le premier Octet de l'équipement.

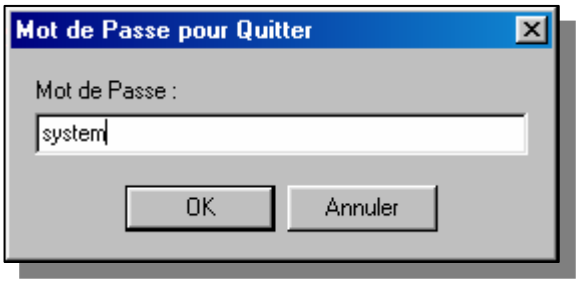
Il est évident que l'on ne peut que forcer que les Octets 'OutPut'.
Si les équipements répondent normalement, le texte Ok apparaît sinon le nombre d'erreurs détectées apparaît dans la colonne erreurs.

Ici le premier texte OK en partant du haut correspond à l'équipement N° 3.

Lancement du serveur de communication

Le serveur au lancement vient se positionner dans la barre de tache de windows indiquant ainsi sa présence.

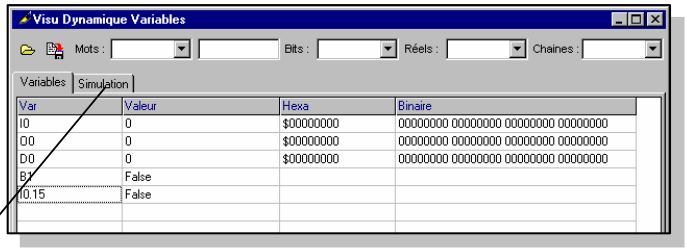
En double cliquant sur cette icône, une fenêtre apparaît permettant d'arrêter celui-ci. En cliquant sur le bouton Quit, une fenêtre demande un mot de passe apparaît :



Frappez le mot de passe **system** pour arrêter le serveur de communication

Visualisation dynamique des Variables du serveur

Ce programme permet d'explorer les variables déclarées évoluant dans le serveur de communication.
Une visualisation dynamique et un forçage en numérique, Hexadécimal et binaire est possible.
Cette liste peut être sauvegardée sur disque et sous un nom donné puis rappelé à tout moment. Permettant ainsi de constituer plusieurs pages de variables.



Simulateur

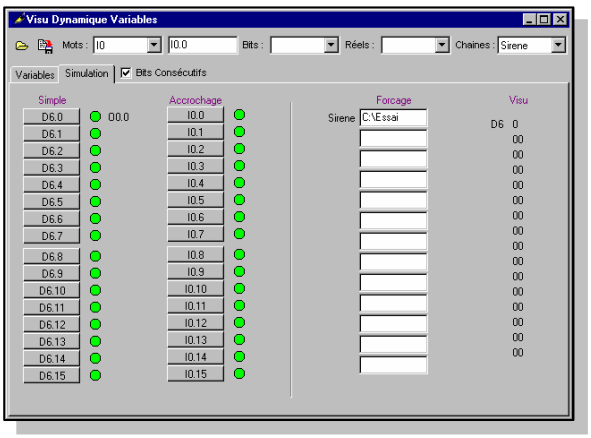
Un simulateur permet d'affecter à des boutons poussoirs simples ou à accrochages des variables booléennes.
Pour affecter un bouton ou un voyant, amenez par Drag & Drop du champ de saisie le nom de la variable concernée sur celui-ci.
Si la case à cocher 'Bits Consécutifs' est valide, les boutons suivants prennent automatiquement les noms des variables adjacentes.

Pour effacer l'affectation d'une variable à un bouton, effacer le texte dans le champ de saisie à droite du ComboBox Mots, puis amenez par drag & drop le texte vide sur le bouton, voyant ou autre élément de visualisation.

Note :
Pour effectuer un '**Drag & Drop**' Cliquez sur le texte avec le bouton gauche de la souris sans relâchez celui-ci, puis faites glissez le curseur de la souris sur l'élément cible souhaité et relâchez le bouton sur celui-ci.

Dans la colonne **Forçage**, des champs de saisie sont disponibles pour affecter directement des variables numériques et chaînes de caractères.
Dans la colonne **visu**, une simple visualisation de variables est possible sans forçage de celles-ci.

Les Combobox supérieurs, permettent de rappeler le nom des variables globales ainsi que le mécanisme de Drag & Drop.



Programmation

Le produit est livré avec 2 DLL : **VPC.DLL et VPLCOM.DLL**

VPLC.DLL est une DLL classique permettant l'interfacage avec différents langages évolués comme Delphi ou Visual C++.
Cependant du fait du passage dans les procédures et fonctions d'une String (Pointer sur caractères), il est impossible pour des langages comme Visual Basic de passer directement une chaîne. Les chaînes Visual Basic étant au format UNICODE incompatible avec les pointeurs sur chaînes de caractères.

il est alors possible de résoudre ce problème en utilisant VPLCOM.DLL qui est un objet COM et qui reçoit les chaînes UNICODE.

Pour Visual Basic Utilisez l'objet COM (VPLCOM.DLL) pour l'appel aux procédures et fonctions décrites ci-dessous. Cet objet est recensé automatiquement dans votre base de registres à l'installation du produit.

Pour une utilisation sous forme de DLL :
****
!! Attention respectez les minuscules dans les déclarations concernant le nom de la dll vplc.dll
!! Respectez aussi l'écriture (majuscule & minuscule) dans les noms de procédures et fonctions
****
Pour l'utilisation sous forme d'objet COM:
Les méthodes ainsi que leurs paramètres sont identiques.
Le respect des majuscules/Minuscules n'est pas nécessaire.

Prototypes et commentaires :

Function GetVpuInt (Name : PChar) : Integer;
-> récupère la valeur de la variable "Name" de type Integer ;

Procedure SetVpuInt (Name : PChar; Value : Integer);
-> affecte la valeur "Value" à la variable "Name" de type Integer ;

Function GetVpuWord (Name : PChar) : Word;
-> récupère la valeur de la variable "Name" de type Word ;

Procedure SetVpuWord (Name : PChar; Value : Word);
-> affecte la valeur "Value" à la variable "Name" de type Word ;

Function GetVpuBool (Name : PChar) : Boolean;
➔ récupère la valeur de la variable "Name" de type Boolean ;
➔ Possibilité de récupérer un bit d'un mot ex : GetVpuBool('W0.0')

Procedure SetVpuBool (Name : PChar; State : Boolean);
-> affecte la valeur "Value" à la variable "Name" de type Boolean ;

Function GetVpuByte (Name : PChar) : Integer;
-> récupère la valeur de la variable "Name" de type Byte ;

Procedure SetVpuByte (Name : PChar; Value : Byte);
-> affecte la valeur "Value" à la variable "Name" de type Byte ;

Function GetVpuString(Name : PChar) : PChar;
-> récupère la valeur de la variable "Name" de type String ;

Procedure SetVpuString(Name : PChar; Value : String);
-> affecte la valeur "Value" à la variable "Name" de type String ;

Function GetVpuReal (Name : PChar) : Single ;
-> récupère la valeur de la variable "Name" de type Real ;
!!Attention la valeur retournée est stockée sur 4 octêts
son étendue est limité par rapport aux variables réelles de Visual PLC stockées
sur 10 octêts (Extended).

Procedure SetVpuReal (Name : PChar; Value : Single);
-> affecte la valeur "Value" à la variable "Name" de type Real ;
!!Attention la valeur envoyée limite l'étendue par rapport aux variables réelles de Visual PLC stockées sur 10 octêts (Extended).

{**** Accès au Tableaux **********}
Function GetVpuAInt (Name : PChar; Element : Integer) : Integer;
-> récupère la valeur d'un tableau "Name" à la position "Element" de type Integer ;

Procedure SetVpuAInt (Name : PChar; Element : Integer; Value : Integer);
-> affecte la valeur "Value" à la position "Element" du tableau "Name" de type Integer ;

Function GetVpuAWord (Name : PChar; Element : Integer) : Word;
-> récupère la valeur d'un tableau "Name" à la position "Element" de type Word ;

Procedure SetVpuAWord (Name : PChar; Element : Integer; Value : Word);
-> affecte la valeur "Value" à la position "Element" du tableau "Name" de type Word ;

Function GetVpuABool (Name : PChar; Element : Integer) : Boolean;
-> récupère la valeur d'un tableau "Name" à la position "Element" de type Boolean ;

Procedure SetVpuABool (Name : PChar; Element : Integer; Value : Boolean);
-> affecte la valeur "Value" à la position "Element" du tableau "Name" de type Boolean ;

Function GetVpuAByte (Name : PChar; Element : Integer) : Integer;
-> récupère la valeur d'un tableau "Name" à la position "Element" de type Byte ;

Procedure SetVpuAByte (Name : PChar; Element : Integer; Value : Byte);
-> affecte la valeur "Value" à la position "Element" du tableau "Name" de type Byte ;

Function GetVpuAString(Name : PChar; Element : Integer) : PChar;
-> récupère la valeur d'un tableau "Name" à la position "Element" de type String ;

Procedure SetVpuAString(Name : PChar; Element : Integer; Value : PChar);
-> affecte la valeur "Value" à la position "Element" du tableau "Name" de type String ;

Function GetVpuAReal (Name : PChar; Element : Integer) : Single;
-> récupère la valeur d'un tableau "Name" à la position "Element" de type Real ;

Procedure SetVpuAReal (Name : PChar; Element : Integer; Value : Single);
-> affecte la valeur "Value" à la position "Element" du tableau "Name" de type Integer ;

{*** Accès à un Mot quel que soit son Type *****}
Function GetVpuValue (Name : PChar) : PChar;
-> récupère la valeur de la variable "Name" quel que soit son type et la renvoie sous forme de pointer sur chaîne de caractère (PChar);

{*** Fixe une valeur à n'importe quel variable *****}
Procedure SetVpuValue (Nom,Value : String);
Exemple : SetVpuValue ('Compteur','10');
Exemple : SetVpuValue ('Etat[1]','True');

{*** Fixe le chemin contenant le Fichier MVarglob.VPU *********}
Procedure SetPathVarglob(Path : String);
Exemple : SetPathVarglob('C:\Essai1\'); Attention mettre '\' à la fin

{**** Récupère l'adresse du premier mot spécifié *********}
Function GetVpuAddr (Nom : String) : Pointer;
Exemple : Adresse :=GetVpuAddr (Compteur) ; renvoie l'adresse mémoire de la variable Compteur

Procedure ClearOptimisation;
-> A chaque appel d'une fonction ou d'une procédure, l'adresse de la variable "Name" est stockée dans une table temporaire pour augmenter les performances d'accès aux variables de Visual PLC. Au bout d'un certain temps, cette table est plus importante que la table réelle des variables de Visual PLC ; il est donc nécessaire de réinitialiser la table temporaire à l'aide de cette procédure. Par exemple, utilisez cette fonction à chaque changement de page.

Utilisation sous forme de DLL

Programmation en Delphi :
déclarez tous les prototypes de fonctions et de procédures en var

VAR
Function **GetVpuInt** (Name : PChar) : Integer; StdCall; External 'vplc.dll';
Procedure **SetVpuInt** (Name : PChar; Value : Integer); StdCall; External 'vplc.dll';
Etc..

Utilisez les alors comme des procédures normales dans votre programme.

Implémentation en Visual Basic

Impossible passez par l'objet Automation VPLCOM.DLL

Implémentation en Visual C++ et C++Builder.

Utiliser les prototypes des fonctions de la DLL Vplc.dll ou utilisez un objet Com. Voir exemple fourni en Visual Basic.

Implémentation en Delphi et Visual Pascal

déclarez tous les prototypes de fonctions et de procédures en var

```
VAR
  Function  GetVpuInt   ( Name : PChar) : Integer; StdCall; External 'vplc.dll';
  Procedure SetVpuInt   ( Name : PChar; Value : Integer); StdCall; External 'vplc.dll';
Etc..
```

VarVpuD.pas pour Visual Pascal et Delphi

La librairie VarVpuD.pas est livrée afin de s'interfacer directement avec les variables globales dans le serveur de communication sans pour autant utiliser une DLL. Cette librairie se trouve dans la sous directorie d'installation Delphi. Recompilez cette librairie avec la version de Delphi correspondante.

Implémentation en C

```
#define       DWORD       unsigned int     // 32 bits
#define       BYTE        unsigned char    // 8 bits
```

DWord **GetVpuInt** (Char * Name) ;

Utilisez les alors comme des procédures normales dans votre programme

Utilisation de GetVpuAddr

Exemple :
Les mots déclarés dans les variables globales sont rangées consécutivement dans la mémoire du serveur. Il est possible de récupérer l'adresse du premier mot puis de lire ou d'écrire consécutivement des octêts

Exemple de variables globales déclarées.

```
Var
        Compteur        : Integer ;       {** 4 octets **}
        Etat            : Boolean         {**1 octet **}
        Eana            : Word ;          {** 2 octets **}
```

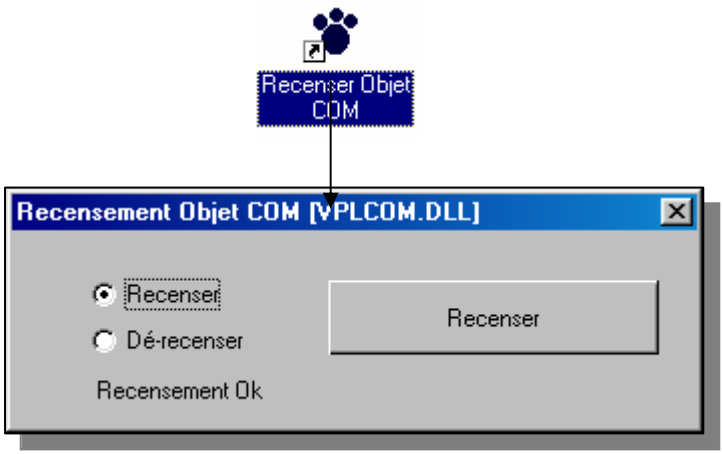
Exemple en Pascal (Visual Pascal ou Delphi)

```
Var
  Pt      : Pointer ;
  SaveCompteur: Integer ;
  SaveEtat      : Boolean ;
  SaveEana      : Word ;

Begin
 PT :=GetVpuAddr( Compteur) ;     {** récupère adresse premier mot dans le serveur ***}
 Move(PT,SaveCompteur,4+1+2) ;  {** Transfert mémoire serveur vers variables locales procédure **}
End ;
```

Utilisation sous forme d'objet COM

Le nom de l'objet est **VPLC.OBJ**

Avant de pouvoir utiliser l'objet Com (objet Automation) VPLCOM.OBJ, vous devez le recenser dans la base de registre de Windows. Cliquez sur l'icône recenser Objet Com



Choisissez le radio-bouton Recenser ou dé-recenser l'objet puis cliquez sur le bouton. Après un recensement correct, le texte *Recensement Ok* apparaît sous les 2 radio-boutons de gauche. Le recensement c'est alors correctement effectué. Quitter alors le programme l'objet COM est disponible.

Programmation en Visual Basic (voir aussi exemple complet fourni sur CD)

```
Private Sub Command1_Click()
Dim Svr As Object
Dim I   As Long
Set Svr = CreateObject("VPLC.OBJ")      '*** Create Object

Svr.SetVpuInt "W0", 123                 '*** Call Method SetVpuInt
I = Svr.GetVpuInt("W0")                 '*** Call Method GetVpuInt
Label1.Caption = Svr.GetVpuString("TS")  '*** Call Method GetVpuString

End Sub
```

Programmation en Delphi et Visual Pascal (voir aussi exemple complet fourni sur CD)
Exemple :

```
Var
  Svr : Variant;

Procedure TForm1.Button1Click(Sender: TObject);
Var
  I : Integer;
Const
  InitDone : boolean=False;
begin
  IF InitDone =False Then
  Begin
    Svr:=CreateOleObject('VPLC.OBJ');   //*** Create Object

    InitDone:=True;
  End;
  Svr.SetVpuInt('W0',-12);  //*** Call Method SetVpuInt
  I:=Svr.GetVpuInt('W0');               //*** Call Method GetVpuInt
  Label1.Caption:=IntToStr(I);
end;
```