En figure 6 la position du moteur à été étudiée en faisant changer la position de la position A à la position B.

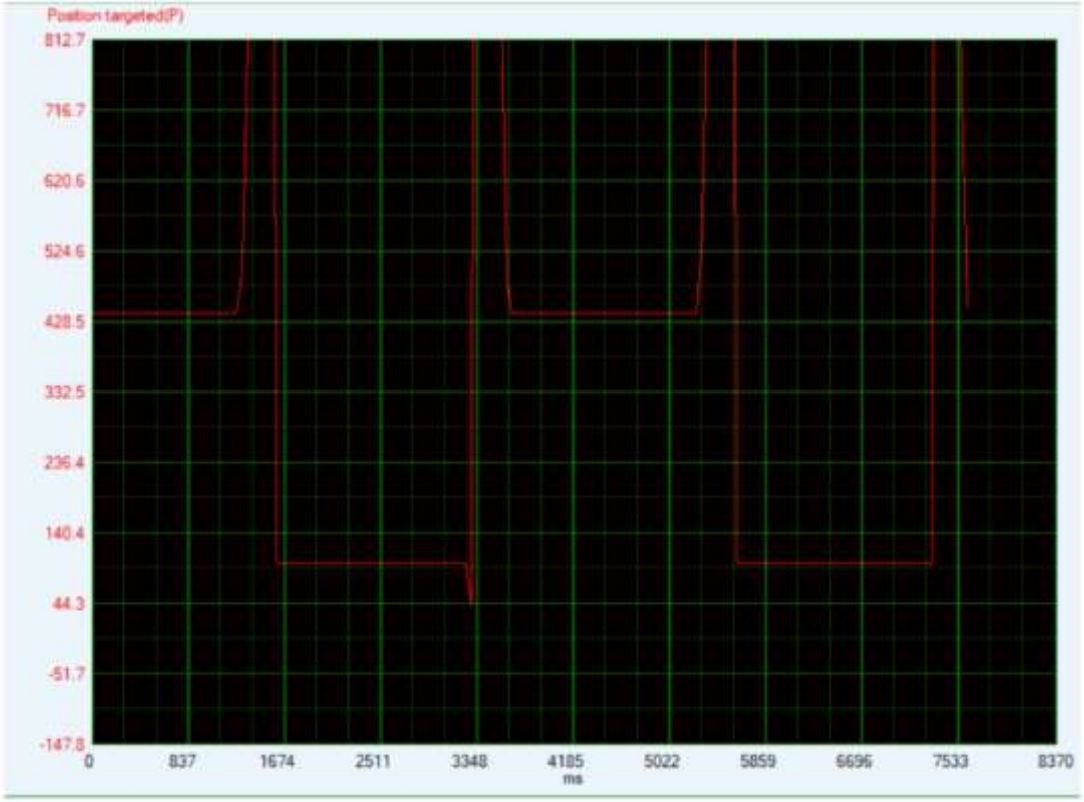


Figure 6 : Visualisation Motionstudio de la position du moteur variant du point A à B

1.4 Communication

Quand la communication entre deux objets est effectuée, il y a toujours deux éléments présents : **Tx et Rx.**
Donc Tx est une demande, et Rx est une réponse.

Communication Traffic

Exit | Stop | Clear | Save | Copy | Log | Stop on Error

No.	Time	Length	Tx/Rx	Packet
0	09:54:00.109	8	Tx	01 03 00 01 00 01 D5 CA
1	09:54:00.173	7	Rx	01 03 02 0C 80 BC E4
2	09:54:00.220	8	Tx	01 03 00 01 00 01 D5 CA
3	09:54:00.266	7	Rx	01 03 02 0C 80 BC E4
4	09:54:00.329	8	Tx	01 03 00 01 00 01 D5 CA
5	09:54:00.375	7	Rx	01 03 02 0C 80 BC E4
6	09:54:00.438	8	Tx	01 03 00 01 00 01 D5 CA
7	09:54:00.486	7	Rx	01 03 02 0C 80 BC E4
8	09:54:00.550	8	Tx	01 03 00 01 00 01 D5 CA
9	09:54:00.597	7	Rx	01 03 02 0C 80 BC E4

***Figure 7** :Capture d'écran de la communication sur Modbus Poll (Lecture d'impulsions par tour)*

Tx : Numéro d'esclaves 01, code fonction 0x03 Read, adresse 0x0001
Rx : Numéro d'esclaves 01, code fonction 0x03 Read et aprés donnée = valeur impulsions par tour

Chaque octet a son **start** et **stop.** Dans ces trames, le bit **faible est en premier.** La trame se décode de la manière suivante en comptant les **0** et les **1** et en obtenant les résultats présents dans la figure 8 avec **norme de communication RS232** (01 03 00 01….) et les bits de start et stop sont aussi représentés**.** Mais le système d'arrosage utilisera la norme de communication **RS485.**

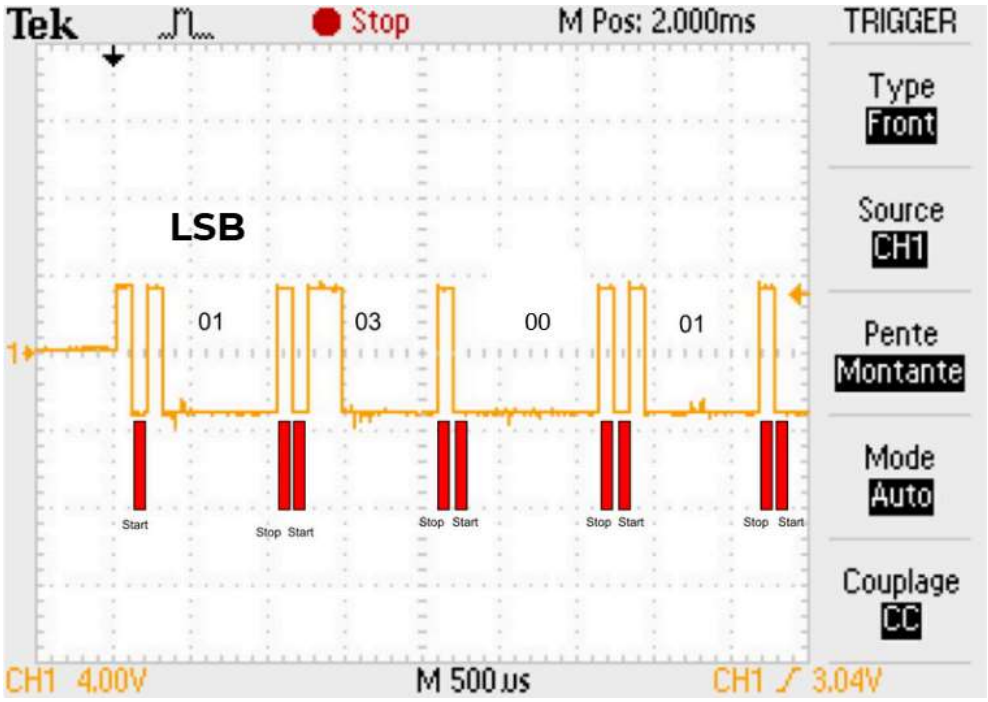


Figure 8 :Le trame RS232 de communication sur l'oscilloscope.

2- Présentation des programmes

2.1 Configuration mode de vitesses

Pour commander tout le système, il a fallu utiliser le logiciel CodeSys.
Pour l'utiliser il faut donc utiliser différentes variables.

Un tableau de variable Figure 9 à donc été réalisé pour bien les utiliser et les comprendre par la suite.

Nom	E/S	Type de la variable
Voyant 1 et 2	Sortie TOR	BOOL
Potentiomètre	Entrée analogique	WORD
Bouton 1	Entrée TOR	BOOL
Bouton 2	Entrée TOR	BOOL
Q	Interne(Entrée an)	REAL
Dosage2	Interne(Entrée an)	REAL
T	Interne	REAL

Figure 9 : Tableau de variable du programme.

Dans ce tableau de variable *Figure 9* il y a des variables externes qui sont présentes sur l'automate,mais aussi des variables internes permettant d'être dans des calculs ou directement modifiables sur l'interface finale.
Dans le programme il y avait toutes ces variables présentes :

```
PROGRAM PLC_PRG
VAR

    bp1visu : BOOL;
    bp2visu : BOOL;
    bp3visu : BOOL;
    bp4visu : BOOL;
    Q : WORD;
    Qt : TIME;
    Qt2 : TIME;
    mytime : TIME;
    T : REAL;
    T2 : REAL;
    hehe : TIME;
    hehe2 : TIME;
    dosage:REAL;
    dosage2:REAL;
    EM: BOOL;
    EA: BOOL;
    M1S : BOOL;
        M2S : BOOL;
END_VAR
```

Figure 10 : Définition des variables CodeSys.

Par la suite l'objectif va être de commander le moteur directement sur CodeSys et de tester les entrées/Sorties de l'automate mais aussi le moteur en le reliant au nombre correspondant à son adresse de fonction en décimal qui est ***16#4001.***

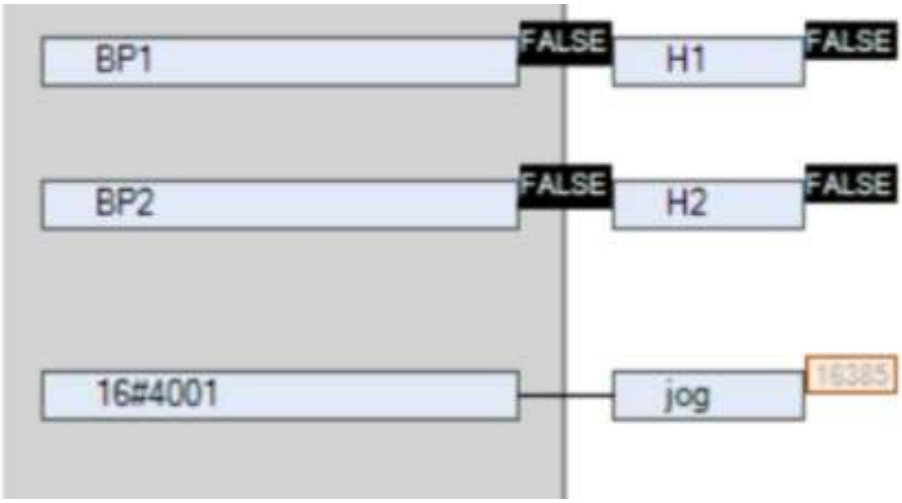


Figure 11 : Capture d'écran du programme de test du système.

Ensuite après avoir tout compris le programme en CFC dans la Figure 11 à été réalisé.

Le programme en SFC est composé tout d'abord de 3 conditions avec un **OU** (**Bouton 1,2 ou 3**). Ce programme permet au moteur de tourner selon trois modes à des vitesses différentes. Après avoir appuyé sur les boutons créés dans la visualisation, le moteur commence à tourner à la vitesse appropriée. La rotation continue pendant le temps imparti et le moteur s'arrête. De plus, si le bouton d'arrêt d'urgence est enfoncé pendant la rotation, le moteur s'arrêtera prématurément.

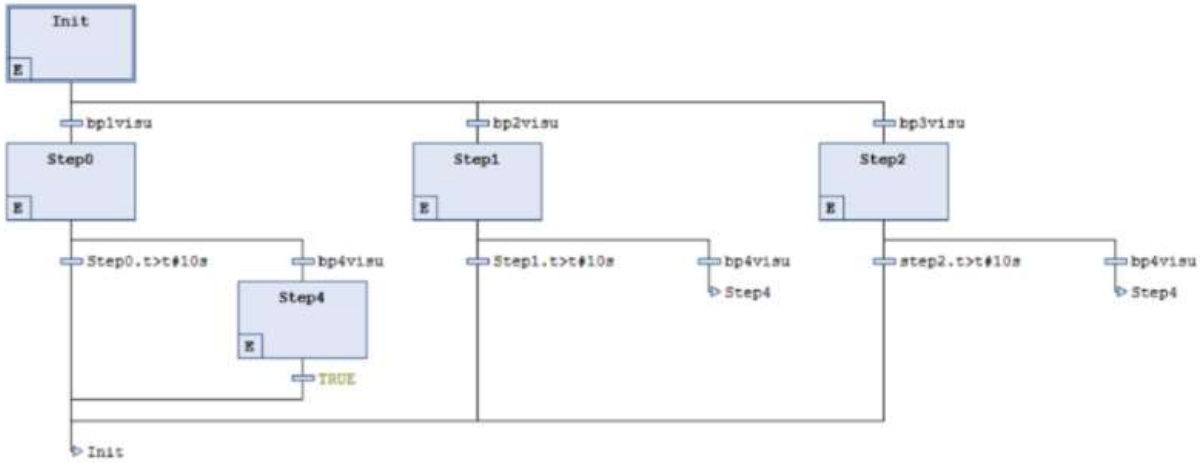


Figure 12 : Programme de fonctionnement du moteur avec 3 modes de vitesses..

Ces vitesses de rotation sont configurées dans Motion Studio. Ces paramètres sont appelés path. La configuration est dans la figure 13.

Control Parameters | Path Parameters | Manual | Parameter Manage

Path ID	Posiotion Mode	Position...	Speed(rpm)	Acceleration(...	Deceleration(...	Pause Time(...	S Code
0	0002H:_V,ABS,END	0	250	4500	500	0	0x00
1	0002H:_V,ABS,END	0	350	4500	500	0	0x00
2	0002H:_V,ABS,END	0	450	4500	500	0	0x00

Figure 13 : Configuration des path dans Motion Studio.

Afin de commander la commutation de **path**, une variable du même nom a été spécifiée dans **les entrées/sorties du d'esclave Modbus. L'adresse de fonction** de cette variable est **0x6002.** Pour modifier le chemin du moteur, il faut saisir des données dans cette variable. Par exemple, pour le **path 1**, cette adresse est **0x010 ou 16 en décimal**, 0x011 et 0x012 sont pour path 2 et 3. Pour **arrêter le moteur**, il faut écrire **64 en décimal** dans cette variable. Ces valeurs ont été trouvées dans la documentation du driver.

Pr8.02	0x6002	Trigger register	Write corresponding command to the 0x6002 to realize the selection and startup of each action. Write value 0x01P----P-segment positioning, (P indicates path NO. 0-15); Write value 0x020---- Homing; (edge triggered) Write value 0x021---- Set the current position as origin by manual; Write value 0x040---- E-stop; Read value 0x000---- Positioning is completed and new data can be received; Read value 0x01P, 0x020, 0x040---- Not responding to the command; Read value 0x10P---- Path is running; Read value 0x200---- Command is completed and waiting for positioning. Note: (P indicates path NO. 0-15);

Figure 14 : Documentation du driver.

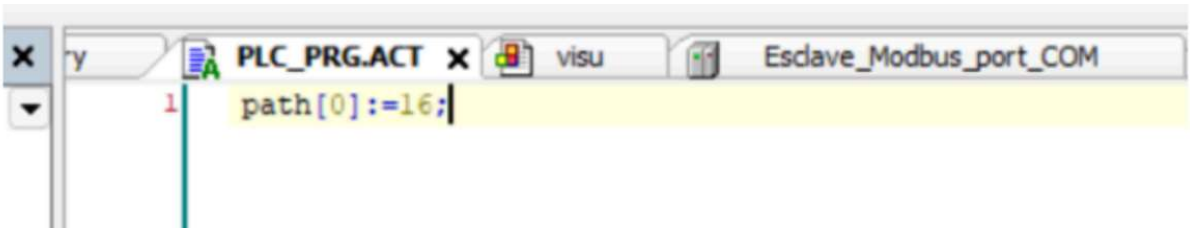
Général | Canal esclave Modbus | Modbus Esclave Init | ModbusGenericSerialSlave Mappage E/S

Rechercher | Filtrer: Afficher tous | Ajouter bloc fonctionnel pour canal E/S...

Variable	Mappage	Canal	Adresse	Type	Valeur par défaut	Unité	Description
path		Channel 0	%QW1	ARRAY [0..0] OF WORD			Write Single Register
		Channel 0[0]	%QW1	WORD	0		0x6002
		Bit0	%QX2.0	BOOL	FALSE		
		Bit1	%QX2.1	BOOL	FALSE		

Figure 15 : Déclaration de path dans CodeSys.

Et donc pour utiliser les différents modes dans chaques blocs après l'appui sur un des boutons, des actions d'entrées avec le bon numéro du "path" ont été initialisé pour choisir le mode de vitesse correspondant.



La visualisation à ensuite été ajoutée et reliée aux différentes variables créées auparavant dans le programme principale pour permettre de sélectionner un des **3 modes de vitesses.**
Et pour les reliés il a fallu aller dans les propriétés de l'objet faisant office de bouton et dans "basculer" ajouter la variable correspondante.
Dans la Figure 16, les 3 boutons bleus correspondent aux **mode 1,2,3** et le bouton rouge au bouton **d'arrêt d'urgence/stop.**

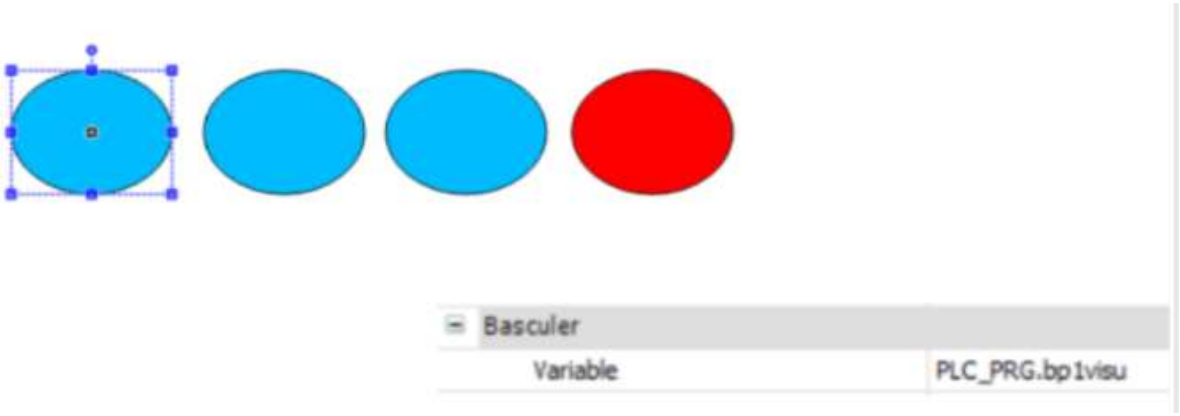


Figure 16 : Première visualisation du système avec boutons definissant le mode.

2.2 Calculs du programme Programme final et interface

L'objectif désormais est de **finaliser le programm**e en ajoutant une variable de temps qui va varier selon la **quantité,le dosage et la vitesse** et faire durer plus ou moins le moteur.

Pour se faire il a fallu tout d'abord réaliser différents calculs.

Tout d'abord la valeur du dosage à été récupéré et converti en L, et sachant que

1 mGallon = 3,8L il a fallu mettre la valeur max du dosage à 3,8L en faisant :

Dosage = $\frac{potentiomètre}{32761}$ * 3,8

(il faut diviser par 32761 car c'est la valeur analogique maximum obtenue par le potentiomètre de l'automate.).

Et ensuite le temps à été calculé en utilisant la formule suivante prenant en compte dans celle-ci le **dosage,la quantité, la vitesse et le débit par révolution.**

$$T = \frac{Dosage * Quantité}{Débit\ par\ révolution * Vitesse}$$

Le **débit par révolution** d'après la documentation de la pompe est de 1250µL ce qui est égal à = 1,25 mL il sera donc toujours **1,25** dans le calcul

Et dans le programme une autre variable à été créé s'appelant ***Qt*** permettant d'avoir le temps en variable **TIME** et donc de pouvoir l'utiliser comme une **temporisation** dans le programme.

Et tous ces calculs ont été ensuite ajoutés dans chacune des actions d'entrée permettant de définir un mode de vitesse comme il est observable en Figure 17.

```
path[0]:=16;

T := ((dosage * Q) / 1.25)/250;
Qt :=REAL_TO_TIME (T*1000);
hehe:=Qt/1000;
dosage := (pot/32761.0)*3.8 ;
```

Figure 17 : Capture d'écran de l'action d'entrée finale permettant de calculer le temps de durée de la pompe.

Et sur le programme la variable "**hehe**" est reliée à une visualisation permettant de voir le temps que va rester **actionné la pompe.**

Et la visualisation finale à été réalisé en 3 parties, la **commande globale**, la **commande pompe 1** et la **commande pompe 2.**
La **commande globale** va permettre de modifier les valeurs des **variables communes** à la pompe 1 et la pompe 2, tel que **la vitesse avec les 3 boutons et le bouton STOP** et la **quantité du tank de mélange avec le SLIDER** et la valeur de celui-ci qui sera affiché à droite.
La **commande pompe 1** va commander les variables unique à la pompe 1 tel que le **dosage** modifiable avec le potentiomètre de l'automate et sa **valeur de temps** sera visualisable à droite du dosage.
Et la **commande pompe 2** va aussi permettre de régler le **dosage** propre à la pompe 2 grâce à un **potentiomètre numérique**. Et possède aussi sa valeur de temps propre qui pourra être observée.