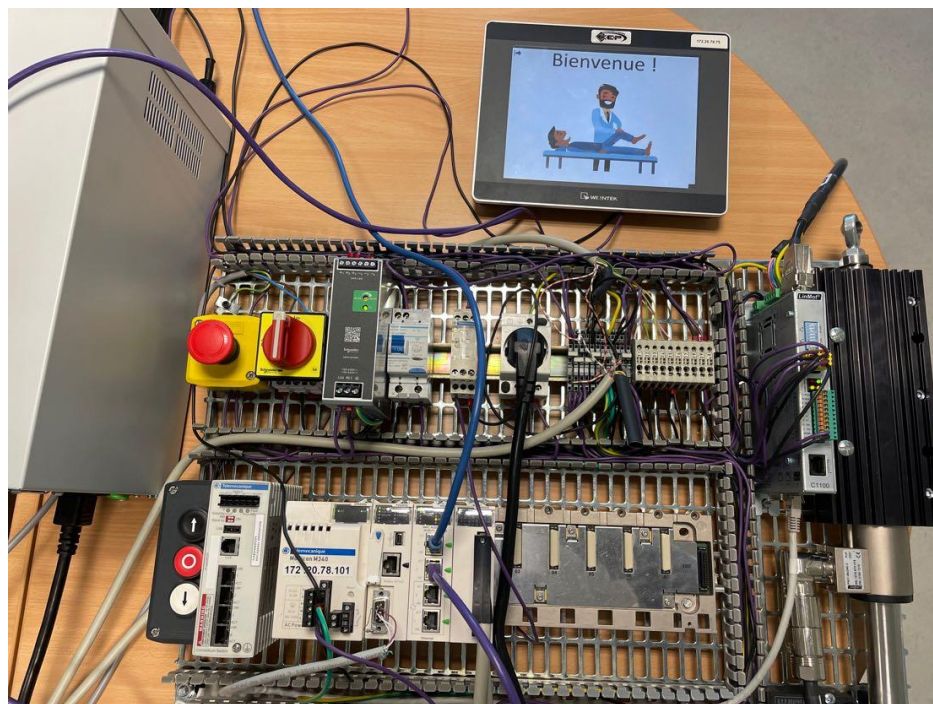


SAE Projet CANopen



Encadrant : Philippe Lucidarme

Groupe Aii : Elouan LUCAS, Titouan ROGER, Florian
GRELIER

Site d'Angers - Campus de Belle-beille :

IUT Angers-Cholet

4, boulevard Lavoisier - BP 42018

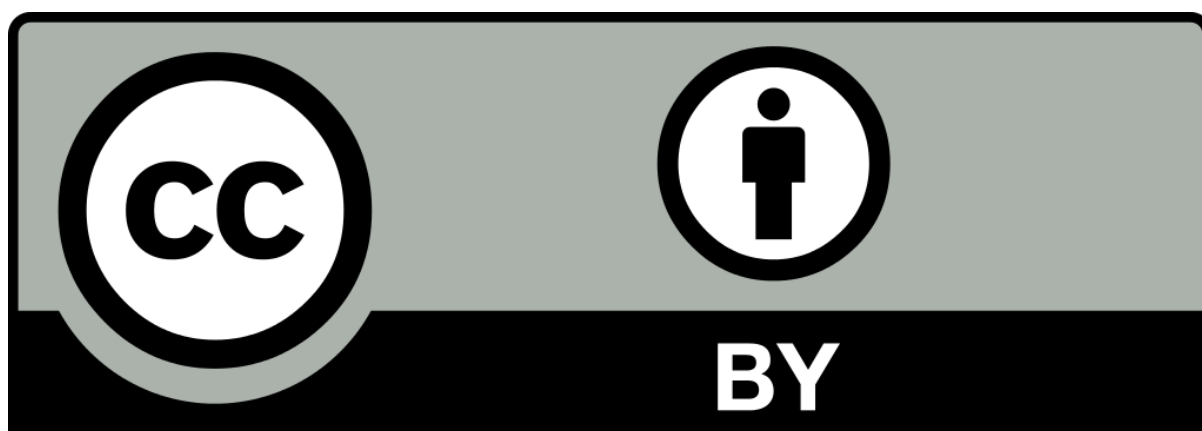
49016 - ANGERS Cedex

Tél. : 02 44 68 87 00

DÉCHARGE DE RESPONSABILITÉ ET LICENCE

Ce rapport présente le travail réalisé par un groupe d'étudiants dans le cadre d'un projet pédagogique. Les auteurs et l'Université d'Angers ne garantissent pas que l'information, les documents, la méthodologie et le matériel présentés dans ce document soient complets, conformes à l'état de l'art et exacts ni n'assurent en toutes circonstances la sécurité des biens, des personnes et des utilisateurs. Les auteurs et l'Université d'Angers ne seront pas tenus responsables des dommages éventuels qui pourraient résulter de l'utilisation du contenu du présent rapport.

Elouan LUCAS, Titouan ROGER, Florian GRELIER, auteurs du présent rapport, publions et divulguons celui-ci sous la Licence Creative Commons suivante « CC BY » pour le monde entier et pendant la durée légale de protection des droits d'auteur. Cette licence autorise la représentation, la reproduction, la modification, la création d'œuvres dérivées et l'utilisation y compris à des fins commerciales sous réserve de mentionner les noms et prénoms des auteurs.



INTRODUCTION.....	5
présentation du projet.....	5
cahier des charges.....	6
réalisation et présentation de la maquette ainsi que de ses différents composants....	8
COMPARATIF ENTRE DIFFÉRENTS BUS DE TERRAIN.....	15
bus TCP.....	16
ethernet IP.....	18
canopen.....	20
choix final de la liaison canopen.....	22
COMMUNICATION AVEC LE VARIATEUR DE VITESSE.....	23
description des étapes de configuration de la liaison.....	23
explication du fonctionnement de la communication.....	24
résolution des problèmes rencontrés.....	27
RÉALISATION DU PROGRAMME DE L'AUTOMATE.....	31
description des différentes étapes de programmation.....	31
explication du fonctionnement du programme.....	35
résolution des problèmes rencontrés.....	44
RÉALISATION DE L'IHM.....	45
présentation des choix de conception de l'ihm.....	45
description et explication des fonctionnalités de l'ihm.....	46
résolution des problèmes rencontrés.....	48
CONCLUSION ET PERSPECTIVES.....	49
bilan du projet et atteinte des objectifs.....	49
limites et perspectives d'amélioration.....	50
retour sur l'expérience et les compétences acquises.....	51
RÉSUMÉ.....	53
ANNEXES.....	54
annexe 1.....	54
annexe 2.....	58
annexe 3.....	59
annexe 4.....	61
annexe 5.....	63
annexe 6.....	64

REMERCIEMENT

Nous souhaitons prendre le temps de remercier toutes les personnes et organisations qui ont contribué à la réalisation de ce projet. Leur soutien, leur expertise et leur collaboration ont été essentiels pour mener à bien notre mission. Nous avons eu la chance de pouvoir compter sur leur aide et leur générosité tout au long de ce parcours. Nous tenons donc à exprimer notre gratitude envers tous ceux qui ont rendu ce projet possible et qui ont contribué à son succès.

Nous remercions :

- M. Lucidarme Philippe, notre enseignant référent, pour ses conseils et son suivi attentif tout au long de ce projet.
- M. Aubriot Thibault, notre professeur d'automatisme, pour avoir fourni le matériel et répondu à nos questions.
- Mme Gromellon Isabelle, notre professeur d'automatisme, d'être venue nous aider lors d'un problème que nous avons rencontré dans la liaison CANopen.
- M. Mercier, professeur à Polytech Angers, pour le prêt du matériel EtherCAT.
- Le personnel de LinMot pour avoir répondu à nos questions.
- Le personnel de Schneider pour nous avoir fourni le bon firmware.

En conclusion, nous remercions toutes les personnes et organisations qui ont contribué à la réussite de ce projet. Grâce à leur soutien et leur expertise, nous avons pu surmonter les difficultés et acquérir des compétences précieuses pour nos projets futurs. Nous leur exprimons notre profonde reconnaissance pour leur aide.

INTRODUCTION

PRÉSENTATION DU PROJET

Le domaine de l'automatisme industriel est en constante évolution, et la communication entre les différents composants est au cœur de cette transformation. Dans ce contexte, nous avons réalisé un projet d'étude supérieure visant à concevoir une maquette intégrant un automate, un variateur de vitesse et un moteur linéaire. L'objectif de cette maquette était de permettre une communication rapide et efficace entre ces différents éléments pour piloter le moteur linéaire.

Dans ce rapport, nous allons détailler les différentes étapes de la réalisation de cette maquette. Nous aborderons notamment le choix de la liaison de communication la plus adaptée pour permettre une transmission fluide et fiable entre les différents composants, ainsi que la description de la maquette et de ses composants. Nous expliquerons également en détail la configuration de la liaison CANopen et le fonctionnement de la communication avec le variateur de vitesse, en précisant les étapes de résolution des problèmes éventuellement rencontrés.

En outre, nous présenterons la réalisation du programme de l'automate et de l'interface homme-machine (IHM), avec une description des différentes étapes de programmation et du fonctionnement du programme. Nous expliquerons également les choix de conception pour l'IHM et les fonctionnalités développées pour une utilisation intuitive. Nous dresserons finalement un bilan du projet, en exposant les objectifs atteints, les limites rencontrées et les perspectives d'amélioration pour une utilisation encore plus performante de cette maquette dans un contexte industriel.

Enfin, nous conclurons en dressant un bilan de ce projet et en exposant les perspectives d'amélioration.

CAHIER DES CHARGES

Projet : Réalisation d'une maquette électrique pour le contrôle d'un moteur linéaire via une liaison CANOpen.

Objectifs :

- Mettre en place une maquette électrique comprenant un automate, un variateur de vitesse et un moteur linéaire pour le contrôle de ce dernier.
- Utiliser une liaison CANOpen pour assurer une communication efficace entre les différents composants.
- Contrôler le moteur en utilisant 3 boutons :
 - Le premier bouton est utilisé pour amener le moteur en position haute (BP haut);
 - Le deuxième bouton est utilisé pour ramener le moteur en position initiale (BP stop);
 - Le troisième bouton est utilisé pour amener le moteur en position basse (BSP bas).
- Contrôler le moteur linéaire via une IHM pour permettre une utilisation intuitive de la maquette.

Fonctionnalités de la maquette :

- Pilotage du moteur linéaire à partir de l'IHM;
- Modification des paramètres du variateur de vitesse à partir de l'IHM;
- Acquisition des données du moteur linéaire et affichage sur l'IHM.

Spécifications techniques :

- Automate schneider M340;
- Variateur de vitesse LinMot C1100;
- Liaison CANOpen;
- IHM KEP.

Livrables :

- Maquette électrique complète comprenant l'API, le variateur de vitesse, le moteur linéaire, la liaison CANOpen et l'IHM;
- Programme de l'API pour le contrôle du moteur linéaire via la liaison CANOpen;
- Programme de l'IHM pour le contrôle du moteur linéaire et l'affichage des données;
- Documentation technique détaillée pour la maquette et les programmes développés;

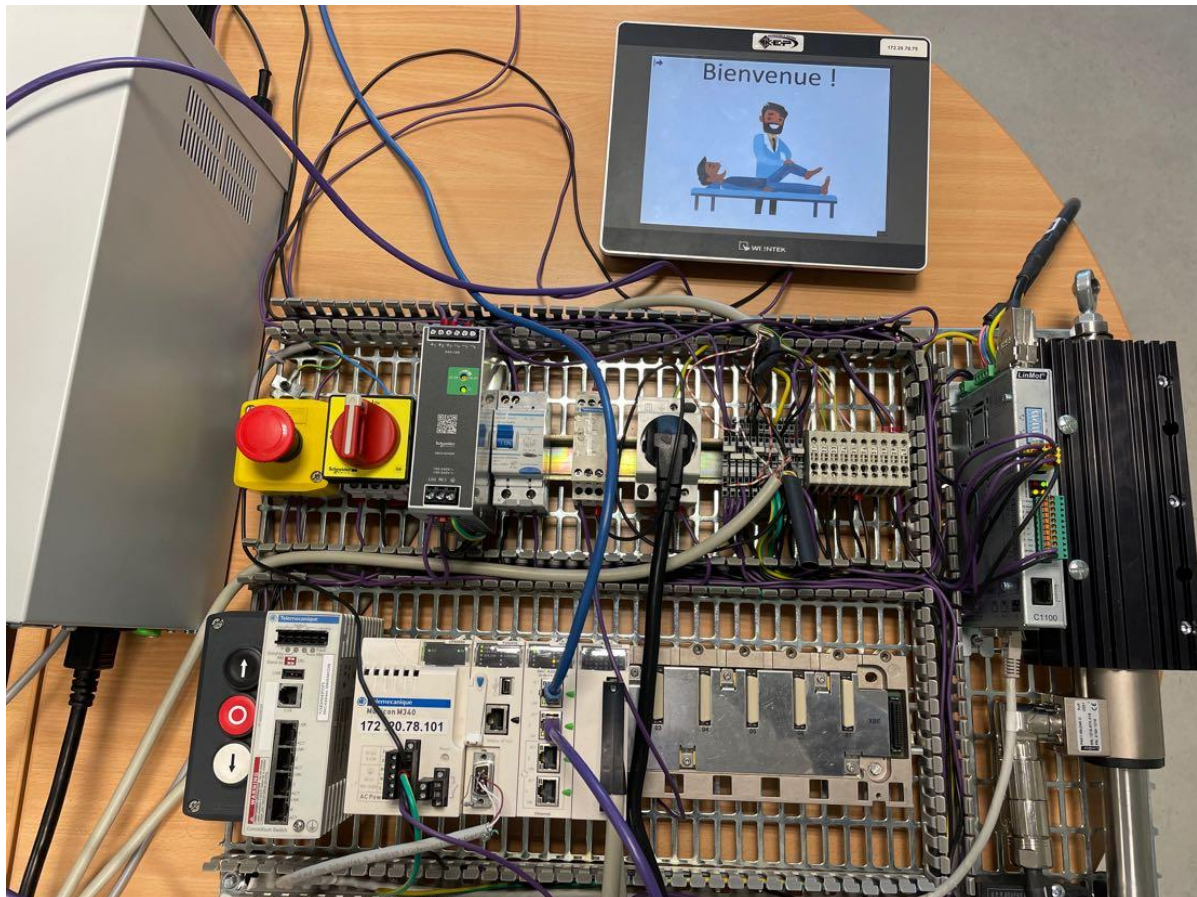
- Rapport de projet présentant les différentes étapes de réalisation, les choix effectués, les problèmes rencontrés et les solutions mises en place pour les résoudre.

Délais :

- Réaliser l'ensemble du cahier des charge pour une présentation le 16/05/2023;
- Rédaction du rapport de projet le 21/05/2023.

Après cela, nous avons consulté les documentations de chaque composant pour savoir comment les câbler et comment fonctionnent les entrées et sorties de l'automate. Cette étape nous a permis d'éviter les potentielles erreurs de câblage qui peuvent entraîner des problèmes par la suite.

Ensuite, nous avons procédé à la réalisation de la maquette en ayant pour objectif de faire un câblage le plus propre possible et de passer le maximum de câbles dans les goulottes. Nous avons pris le temps de réaliser un câblage soigné pour éviter tout problème électrique.



Finalement, nous avons réalisé le schéma électrique final de la maquette que vous pouvez retrouver en annexe 1.

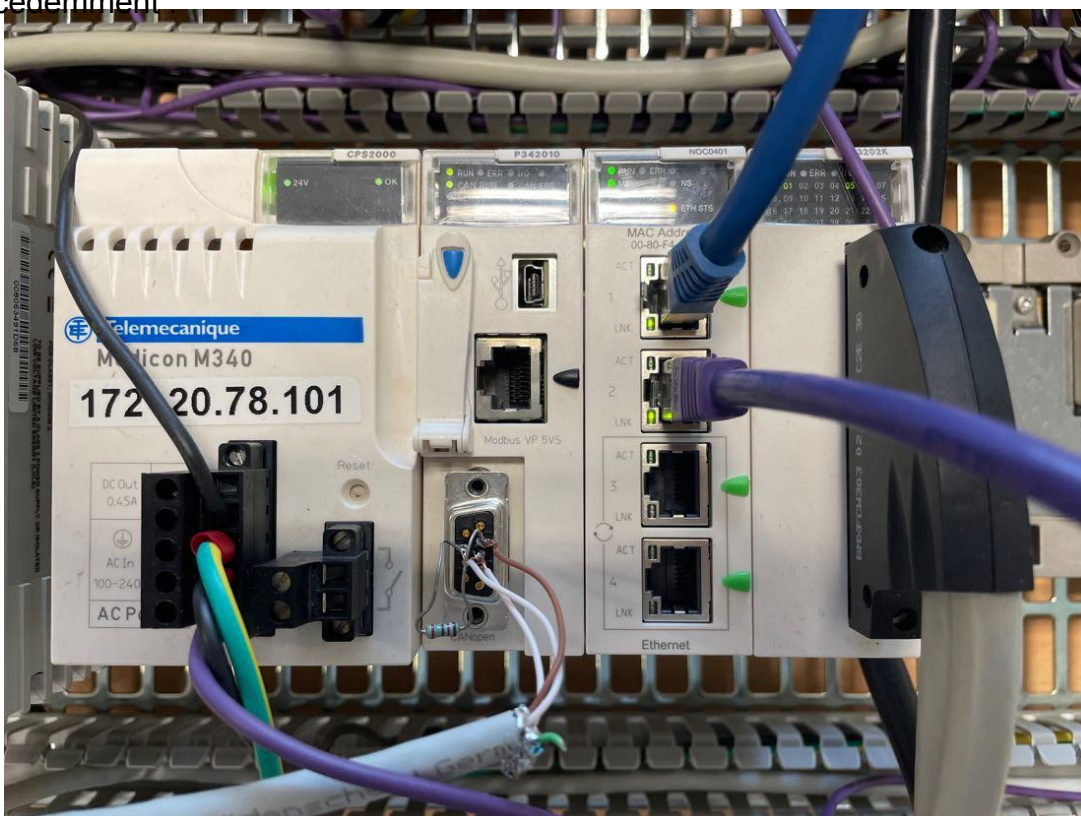
PRÉSENTATION DES COMPOSANTS :

Sur le schéma électrique, vous avez pu observer de nombreux petits composants tels que des boutons ou encore un sectionneur. Cependant, nous allons maintenant nous concentrer sur quatre composants clés : l'automate, l'IHM, le variateur de vitesse et le moteur.

L'AUTOMATE

L'automate programmable M340 est le cerveau de la maquette électrique. Il est équipé d'une alimentation qui transforme le courant 230V en 24V, ce qui permet de protéger les composants sensibles de la maquette. La CPU P342010 de l'automate a une prise CANopen qui permet de communiquer avec d'autres équipements industriels compatibles avec ce protocole. Cette fonctionnalité est essentielle pour le contrôle du moteur linéaire, car elle permet d'envoyer des ordres de mouvement précis à l'automate, qui les exécute ensuite via les sorties de l'interface homme-machine (IHM). De plus, le module Ethernet NOE0110 (TCP/IP) connecté à la CPU permet la liaison entre l'automate et le réseau Ethernet, permettant ainsi la communication à distance avec un ordinateur ou l'IHM. Enfin, le module d'entrée/sortie a pour rôle de transmettre des informations à l'automate à partir de boutons ou de fins de course.

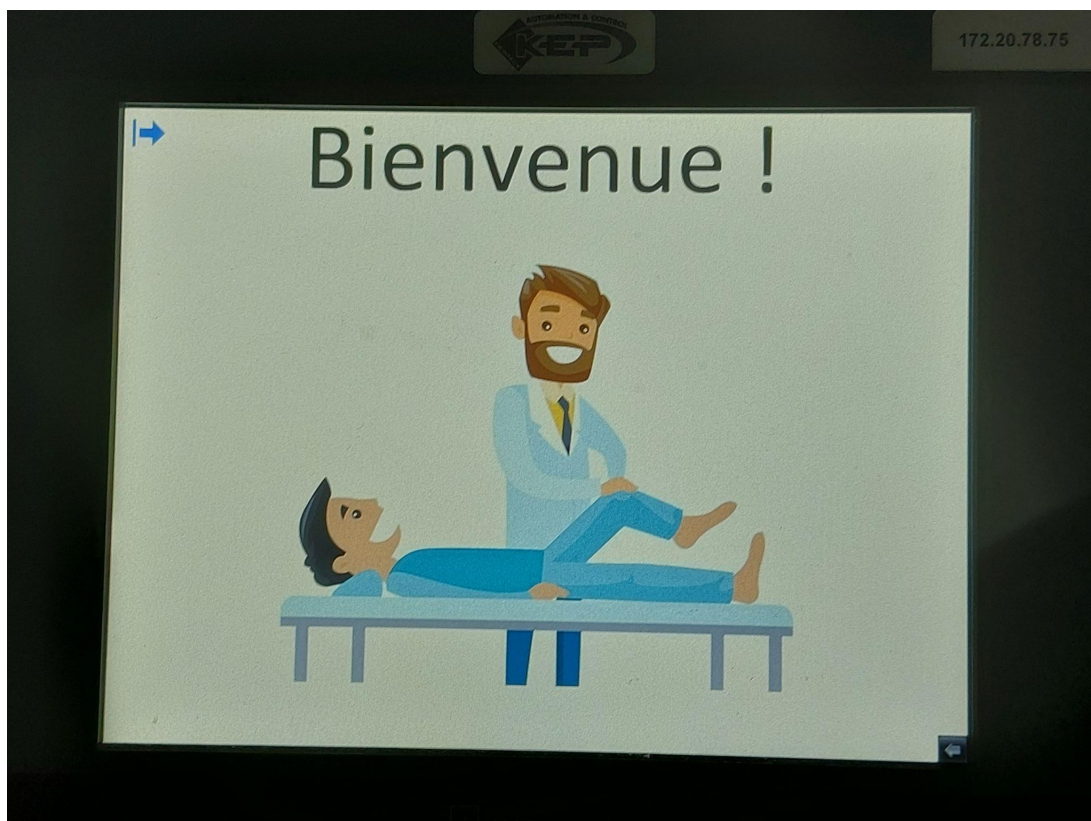
Ci- dessous une photo de l'automate et des modules expliqués précédemment :



L'interface homme-machine (IHM) est un composant clé de la maquette électrique, car elle permet à l'utilisateur d'interagir avec le système. L'IHM est un écran tactile qui affiche des informations sur le fonctionnement de la maquette et permet de régler les paramètres de contrôle du moteur linéaire. Elle est connectée à l'automate via un câble Ethernet, ce qui lui permet de communiquer avec l'automate et de recevoir des informations en temps réel sur l'état de la maquette.

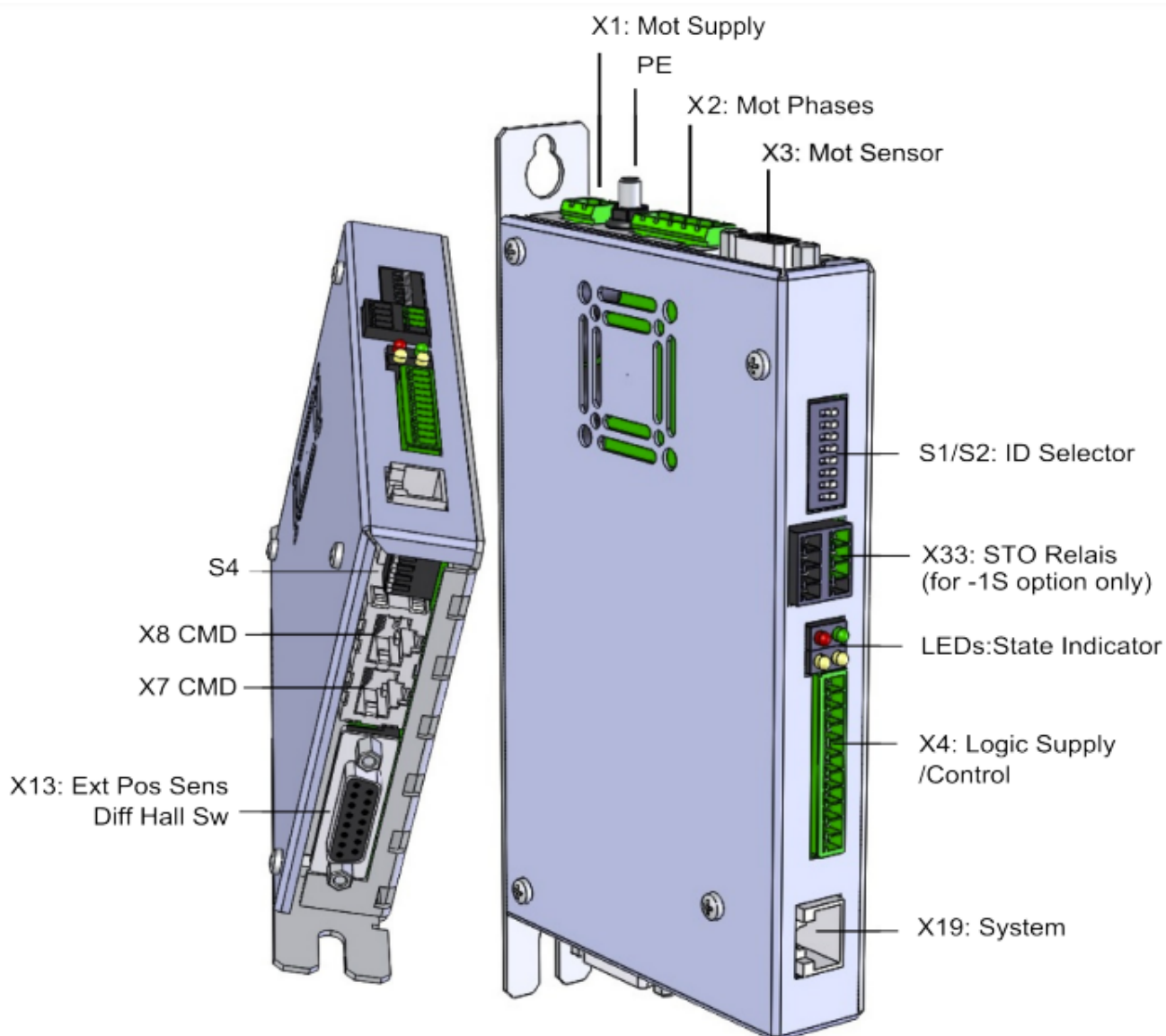
Grâce à l'IHM, l'utilisateur peut surveiller et ajuster la tension et la vitesse du moteur linéaire. L'écran affiche également la position de l'axe linéaire, ce qui permet à l'utilisateur de vérifier que le moteur se déplace correctement. En utilisant l'IHM, l'utilisateur peut ajuster les paramètres de contrôle du moteur linéaire, comme la vitesse de déplacement ou l'accélération. Cela permet de s'assurer que la maquette fonctionne de manière optimale et de garantir la sécurité de l'utilisateur.

Nous avons opté pour l'utilisation d'une IHM KEP pour notre maquette, car nous sommes familier avec son utilisation étant donné que nous l'étudions en cours.



Le variateur de vitesse est un élément crucial pour contrôler la vitesse du moteur linéaire dans la maquette électrique. Il est connecté à la sortie de l'automate programmable M340 et reçoit des signaux de commande qui lui permettent de réguler la tension envoyée au moteur. Le variateur de vitesse choisi pour cette maquette est le C1100 de la marque LinMot. Ce variateur est particulièrement intéressant car il permet la communication en RS232 et en CANopen, deux protocoles industriels couramment utilisés dans le domaine de l'automatisation. Cette fonctionnalité permet une intégration facile avec d'autres équipements industriels compatibles avec ces protocoles, ce qui facilite l'interconnexion de la maquette avec d'autres machines et systèmes de contrôle.

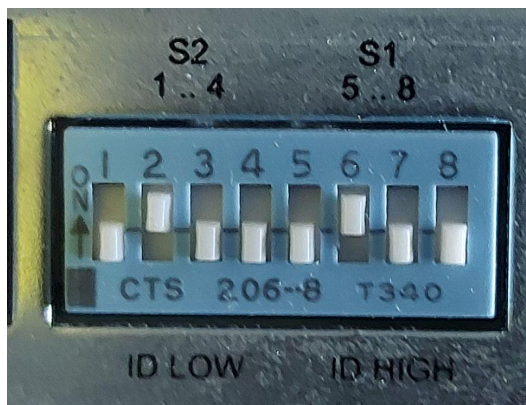
Cette image montre qu'il y a de nombreux branchements à effectuer pour la maquette électrique :



- X2 et X3 sont les alimentations fournies par le variateur ;
- X33 correspond aux relais de sécurité qui arrêtent le moteur en cas de problème ;
- X4 permet d'alimenter le variateur ;
- S1/S2 permet de définir l'utilisation du variateur grâce à une combinaison d'interrupteurs ;
- S4 permet de définir le moyen de communication ;
- X7 est l'entrée pour le protocole de communication CANopen ;
- X8 sert de liaison entre deux variateurs de vitesse.

Dans notre cas, nous avons branché le X33 comme indiqué dans l'annexe 1, folio 3.
Ci-dessous les combinaisons des interrupteurs S1/S2 et S4

Interrupteurs S1/S2



Interrupteurs S4



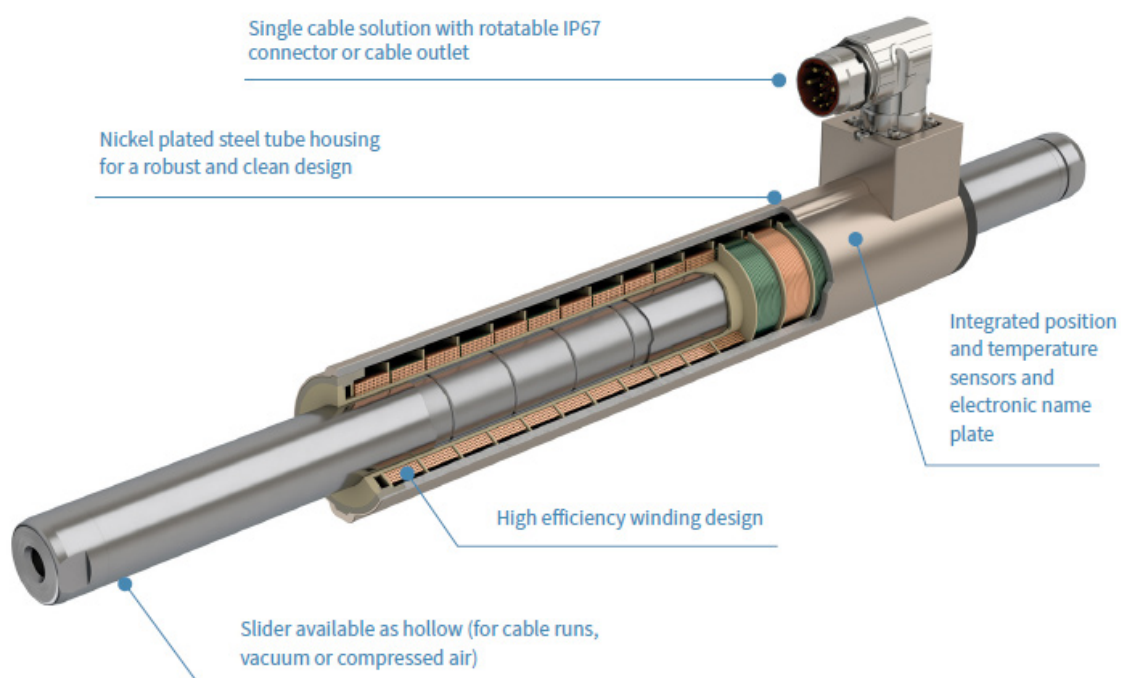
Un moteur linéaire est un type de moteur électrique qui convertit l'énergie électrique en mouvement linéaire. Contrairement à un moteur rotatif qui produit un mouvement circulaire, un moteur linéaire produit un mouvement linéaire directement sans avoir besoin d'un mécanisme de conversion.

Le fonctionnement d'un moteur linéaire repose sur le principe de l'interaction électromagnétique entre un champ magnétique produit par des aimants permanents et un champ magnétique variable produit par des bobines de cuivre. Les bobines de cuivre sont disposées le long d'une piste en fer doux, appelée "stator", tandis que les aimants permanents sont placés sur un chariot métallique, appelé "rotor", qui se déplace le long de la piste.

Lorsque du courant électrique est appliqué aux bobines de cuivre, cela crée un champ magnétique qui attire ou repousse les aimants permanents, selon l'orientation des pôles. Cette interaction magnétique fait avancer le rotor le long de la piste du stator, produisant un mouvement linéaire.

Le sens de déplacement du rotor est déterminé par la polarité de l'alimentation électrique. Si l'alimentation est inversée, le sens de déplacement du rotor sera inversé. La vitesse et l'accélération du rotor peuvent être contrôlées en ajustant la fréquence et la puissance de l'alimentation électrique.

En résumé, un moteur linéaire fonctionne en utilisant l'interaction électromagnétique entre des bobines de cuivre et des aimants permanents pour produire un mouvement linéaire. Ce type de moteur est utilisé dans une variété d'applications industrielles telles que les machines-outils, les robots, les systèmes de transport et les équipements de production.



COMPARATIF ENTRE DIFFÉRENTS BUS DE TERRAIN

Dans le cadre de notre projet, nous avons besoin d'un bus de terrain pour connecter notre automate et notre variateur de vitesse.

Commençons par définir ce qu'est un bus de terrain. Un bus de terrain est un ensemble de fils ou de connecteurs utilisé pour connecter différents composants d'un système électronique ou informatique. Il permet la transmission d'informations, d'alimentation électrique ou de données entre les composants.

L'utilisation d'un bus de terrain simplifie la conception des systèmes électroniques et améliore les performances et la fiabilité. Les bus de terrain peuvent être conçus pour utiliser différents protocoles de communication, tels que I2C, SPI, USB, Ethernet, etc. En choisissant le bon protocole pour le bus de terrain, nous pouvons maximiser les performances et minimiser les coûts de production.

L'objectif de notre projet est de contrôler un moteur linéaire. Nous avons besoin d'un automate qui donne les instructions de l'utilisateur au variateur de vitesse. Ce dernier alimente et transmet les ordres au moteur.

Pour la communication entre l'automate et le variateur, nous utilisons un bus de terrain. Nous avons plusieurs possibilités : le bus TCP, Ethernet IP et CANopen. Beaucoup de personnes font l'amalgame entre le bus TCP et l'Ethernet IP. Les principales différences entre les deux sont le protocole de communication utilisé et les capacités de traitement des volumes de données et des débits. Ethernet/IP offre également des fonctions supplémentaires telles que la découverte automatique des équipements, la configuration à distance et le diagnostic à distance.

Nous allons étudier les caractéristiques du bus TCP, Ethernet IP et CANopen afin de choisir le bus de terrain le mieux adapté pour atteindre notre objectif.

BUS TCP

DÉFINITION

Le bus TCP (Transmission Control Protocol) est un protocole de communication réseau utilisé pour transmettre des données sur Internet. Il fait partie du protocole Internet (IP) et est souvent utilisé avec l'application HTTP pour transmettre des pages web à partir d'un serveur web vers un navigateur client.

Une trame TCP est une unité de données transmise sur un réseau en utilisant le protocole TCP. Elle comprend une tête de trame TCP qui contient diverses informations sur la transmission de données, ainsi que les données proprement dites.

La tête de trame TCP contient les informations suivantes :

- Numéro de séquence : indique l'ordre des paquets de données transmis.
- Numéro d'acquittement : indique la dernière séquence de données reçue sans erreur.
- Données de contrôle : comprennent des indicateurs de l'état de la connexion, tels que la demande de connexion ou la demande de fermeture de connexion.
- Fenêtre : indique la taille maximale des données à transmettre sans acquittement.
- Vérification de somme de contrôle : utilisée pour détecter les erreurs de transmission.

Une trame TCP garantit la fiabilité de la transmission des données en vérifiant :

1- La vitesse de transmission des données en ajustant la taille du tampon de transmission pour éviter les problèmes de surcharge du réseau. Il faut savoir que la vitesse de communication du bus TCP dépend des caractéristiques du réseau et des équipements utilisés. En général, la vitesse de transmission des données sur un réseau TCP est de l'ordre de plusieurs mégabits par seconde (Mbps) à plusieurs gigabits par seconde (Gbps). Toutefois, la vitesse de communication réelle peut être limitée par les facteurs tels que la qualité de la connexion, le nombre d'équipements connectés sur le réseau, etc.

2- La présence de paquets perdus ou corrompus et en les retransmettant si nécessaire. Si un paquet se perd, le protocole TCP utilise des mécanismes de retransmission pour s'assurer que le paquet manquant est récupéré. Tout d'abord, TCP envoie un accusé de réception (ACK) pour tous les paquets reçus. Si un paquet n'est pas confirmé après un certain temps, TCP suppose que le paquet est perdu et demande une retransmission du paquet manquant à l'émetteur. Le paquet manquant est renvoyé et le processus de transmission continue à partir de celui-là.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Port Source 2 octets																Port destination 2 octets															
Numéro de séquence																															
Numéro d'acquittement																															
Taille de l'en-tête		Réservé	ECN / NS		CWR	ECE	URG	ACK	PSH	RST	SYN	FIN	Fenêtre																		
Somme de contrôle																Pointeur de données urgentes															
Options																							Remplissage								
Données																															

MATÉRIEL UTILISÉ

En général, les automates ou les variateurs de vitesse ne communiquent pas seulement en TCP mais en TCP/IP.

Pour les automates, il y a 2 marques qui peuvent être utilisées dans notre cadre scolaire les Siemens et les Schneider.

Pour les automates Siemens, il y a :

- S7-1500, ils communiquent via un module de communication PROFINET intégré, qui permet les échanges via ce protocole.
- S7-1200, ils peuvent également transmettre des données via TCP/IP en utilisant un module de communication PROFINET ou un module CP (Communication Processor) séparé.

En général, les automates Siemens équipés d'un module de communication PROFINET ou d'un module CP peuvent communiquer en bus TCP/IP.

Quant aux automates Schneider Electric, ceux qui supportent la communication en bus TCP disposent d'une carte de communication Ethernet intégrée ou disponible en option.

Voici quelques exemples d'automates Schneider Electric qui prennent en charge le bus TCP :

- Modicon M340
- Modicon M580
- Modicon Quantum
- Modicon Premium
- Modicon Momentum
- Unity Pro XL

Cependant, il est important de vérifier les spécifications techniques de l'automate sélectionné pour déterminer s'il prend en charge la communication en bus TCP.

Enfin, les variateurs de la série LinMot E1100 et E1200 de LinMot peuvent communiquer via TCP/IP. Ils sont équipés d'une interface Ethernet intégrée qui prend en charge le protocole de communication TCP/IP. Les variateurs LinMot de la série E1100 et E1200 peuvent être utilisés pour contrôler les moteurs linéaires de LinMot et peuvent être intégrés dans des systèmes de contrôle de mouvement et de processus industriels, qui utilisent des protocoles de communication basés sur TCP/IP.

ETHERNET IP

DÉFINITION

EtherNet/IP (Industrial Protocol) est un protocole de communication industriel basé sur Ethernet. Il est utilisé pour connecter des équipements industriels tels que des automates programmables (PLC), des actionneurs et des capteurs pour échanger des informations en temps réel.

EtherNet/IP est une version industrielle d'Ethernet qui ajoute des fonctionnalités pour la communication de contrôle industriel, telles que des fonctionnalités de découverte de dispositifs et une gestion des erreurs. Il utilise le protocole TCP/IP pour transmettre des données en toute sécurité sur un réseau Ethernet.

Une trame Ethernet IP comprend généralement les éléments suivants :

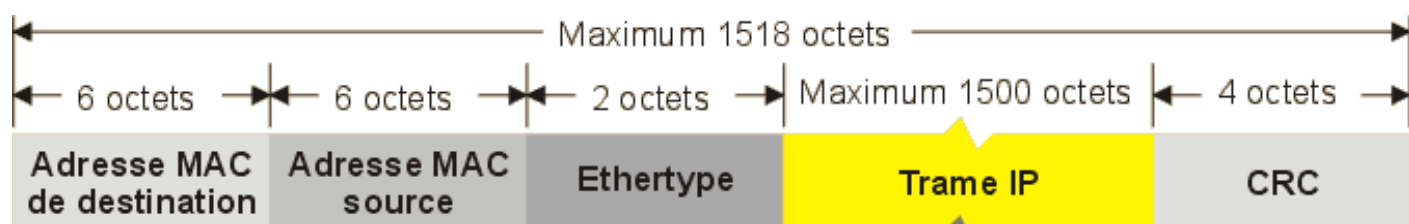
- En-tête Ethernet : Il comprend des informations sur la destination et la source de la trame, ainsi que la longueur totale de la trame.
- En-tête IP : Il comprend des informations sur la source et la destination IP pour la transmission de données sur Internet.
- En-tête UDP : Il comprend des informations sur les ports de source et de destination pour la transmission de données sur le réseau.
- Données utilisateur : Les données utilisateurs sont les informations réelles qui sont transmises entre les équipements.
- CRC : Le contrôle de redondance cyclique (CRC) est utilisé pour vérifier l'intégrité des données transmises.

EtherNet/IP est largement utilisé dans les systèmes de contrôle de processus et d'autres applications industrielles en raison de

1- Sa fiabilité : En effet, si un paquet ne parvient pas à destination, le protocole TCP (Transmission Control Protocol) utilisé pour la communication Ethernet/IP demande la retransmission du paquet manquant. Notons cependant que cela peut causer un délai supplémentaire dans la transmission des données.

2- Sa vitesse de transmission élevée : Les vitesses de communication typiques pour Ethernet/IP sont de 100 Mbps et 1 Gbps. Cependant, il est important de souligner que la vitesse de communication réelle dépend de la charge du réseau et des autres facteurs environnementaux, de la version d'Ethernet utilisée et de la qualité des équipements.

3- Sa flexibilité pour connecter différents types d'équipements.



MATÉRIEL UTILISÉ

Pour les automates, plusieurs marques, leaders du marché, peuvent être utilisées : Siemens et Omron. Pour les variateurs de vitesse, les variateurs de vitesse LinMot sont les seuls qui commandent un moteur linéaire.

Plusieurs automates de la marque Siemens prennent en charge la communication en Ethernet/IP :

- SIMATIC S7-1200
- SIMATIC S7-1500
- SIMATIC ET 200SP
- SIMATIC ET 200MP
- SIMATIC ET 200SP HA

Plusieurs automates Omron prennent en charge Ethernet/IP. Citons pour exemples :

- Les automates de la série NJ : ces automates haut de gamme sont conçus pour les applications complexes et prennent en charge Ethernet/IP ainsi que d'autres protocoles de communication tels que Modbus TCP, EtherCAT, etc.
- Les automates de la série NX1/NY1 : ces automates compacts prennent également en charge Ethernet/IP et sont conçus pour les applications de machine simple à moyenne complexité.
- Les automates de la série CP1 : ces automates économiques prennent en charge Ethernet/IP ainsi que d'autres protocoles de communication tels que Modbus TCP, EtherNet/IP, etc.

Il est important de vérifier les spécifications techniques de l'automate Siemens ou Omron sélectionné pour déterminer s'il prend en charge la communication en Ethernet/IP.

Pour les variateurs de vitesse, la série LinMot C1250 et C1260 peuvent être équipés d'une option de communication Ethernet/IP. Cela permet une communication bidirectionnelle avec des contrôleurs programmables. Les variateurs LinMot de la série C1250 et C1260 sont conçus pour les applications de contrôle de mouvement et peuvent être utilisés pour le contrôle de position, de vitesse et de force.

DÉFINITION

CANopen est un protocole de communication industriel qui permet la transmission en temps réel d'informations entre les différents composants d'une installation industrielle. Il est basé sur le bus CAN (Controller Area Network) et est utilisé pour les systèmes de contrôle de mouvement et de processus industriels.

CANopen utilise une structure hiérarchique pour organiser les données, ce qui simplifie la gestion et la maintenance du système. Chaque composant peut être identifié par un numéro d'identification unique, facilitant ainsi la gestion des communications entre les différents équipements.

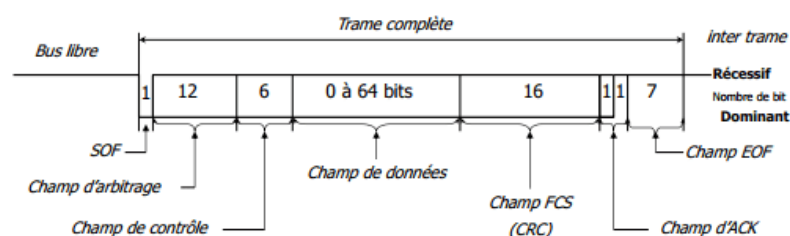
Les trames CANopen contiennent des informations sur les commandes de mouvement, les états des composants, les données d'entrée et les données de sortie. La trame peut également inclure des informations sur les erreurs et les diagnostics pour aider à la surveillance et à la maintenance du système.

- L'identificateur est un champ de 12 bits qui identifie le message CANopen. Il peut être utilisé pour classer les messages en différentes catégories, telles que les commandes de mouvement, les états des composants, les données d'entrée et les données de sortie.
- Les données sont un champ de 0 à 64 octets qui peuvent contenir des informations sur les commandes de mouvement, les états des composants, les données d'entrée et les données de sortie.
- Le CRC est un champ de 16 bits qui est utilisé pour vérifier l'intégrité des données transmises. Si une erreur est détectée dans les données, le composant qui reçoit le message peut ignorer les données erronées et demander une transmission complète du message.

Les trames CANopen sont transmises à une vitesse qui peut varier en fonction de la configuration et de la mise en œuvre de la couche physique CAN sous-jacente. Cependant, la norme CANopen spécifie des vitesses de communication standardisées pour certains débits de données couramment utilisés, notamment : 10 kbit/s, 20 kbit/s, 50 kbit/s, 125 kbit/s, 250 kbit/s, 500 kbit/s et 1 Mbit/s.

En CANopen, si une trame se perd, le protocole de reprise d'erreur intégré (Error Confinement Mechanism) entre en jeu pour gérer les erreurs de communication. Le mécanisme est conçu pour détecter et corriger les erreurs de transmission, telles que les trames perdues, les trames en double, les erreurs de bit et les erreurs de format. Si une trame est perdue ou corrompue, le protocole de reprise d'erreur tente de la renvoyer ou de la corriger en utilisant des mécanismes de répétition ou de retransmission.

En résumé, une trame CANopen est conçue pour transmettre de manière efficace, rapide et fiable des informations entre les différents composants d'un système industriel. Le dispositif comprend de plus, nous l'avons vu, un protocole de résolution d'erreur (transmission).



MATÉRIEL UTILISÉ

Pour les automates, nous avons identifié deux marques couramment utilisées dans notre cadre scolaire : Siemens et Schneider. En ce qui concerne les variateurs de vitesse, nous avons choisi d'utiliser les variateurs de vitesse LinMot, car ils sont les seuls à pouvoir commander un moteur linéaire.

Voici quelques exemples d'automates de la marque Siemens qui prennent en charge la communication en CANopen :

- SIMATIC S7-1200
- SIMATIC S7-1500
- SIMATIC ET 200SP
- SIMATIC ET 200MP

Voici quelques exemples d'automates Schneider Electric qui prennent en charge le protocole CANopen :

- Modicon M340
- Modicon M580
- Modicon Quantum
- Modicon Premium
- Modicon Momentum

Cependant, il est important de vérifier les spécifications techniques de l'automate utilisé pour déterminer s'il prend en charge la communication en CANopen.

Quant aux variateurs de la série LinMot, les modèles C1250, C1260 et C1100 peuvent être équipés d'une option de communication CANopen pour communiquer avec des dispositifs compatibles CANopen tels que les automates programmables industriels (API) et les capteurs. La communication bidirectionnelle permet la configuration et la surveillance des paramètres du variateur ainsi que la récupération de données de mesure en temps réel.

CHOIX FINAL DE LA LIAISON CANOPEN

TCP, Ethernet/IP et CANopen sont tous des protocoles de communication utilisés pour interconnecter des équipements industriels. Il existe cependant quelques différences entre les trois :

- TCP est un protocole de communication en mode connecté qui garantit la fiabilité et l'ordre d'arrivée des données.
- Ethernet/IP est un protocole de communication basé sur Ethernet qui permet la communication de données entre équipements industriels. Il est capable de transporter différents types de données, tels que des données de contrôle, des données d'E/S, des données de configuration, etc.
- CANopen, quant à lui, est un protocole de communication basé sur CAN (Controller Area Network) qui est largement utilisé dans l'automatisation industrielle et dans les applications d'ingénierie. CANopen est apprécié pour sa flexibilité, sa faible consommation d'énergie et sa capacité à prendre en charge des réseaux de taille variable.

Cette synthèse montre que chaque protocole a ses avantages et ses inconvénients. Le choix d'un protocole dépend des besoins spécifiques de l'application.

Dans le cadre de notre projet, nous avons choisi la liaison CANopen. Deux raisons principales justifient notre choix : la première est que c'était l'une des seules options de liaison que nous avions à notre disposition, et la seconde est sa vitesse d'exécution plus rapide que les autres.

De plus, CANopen est particulièrement adapté à notre application nécessitant des temps de réponse courts, une communication fiable et une faible consommation d'énergie. Il est également plus facile à configurer et à mettre en service, ce qui réduit le temps et les coûts de développement.

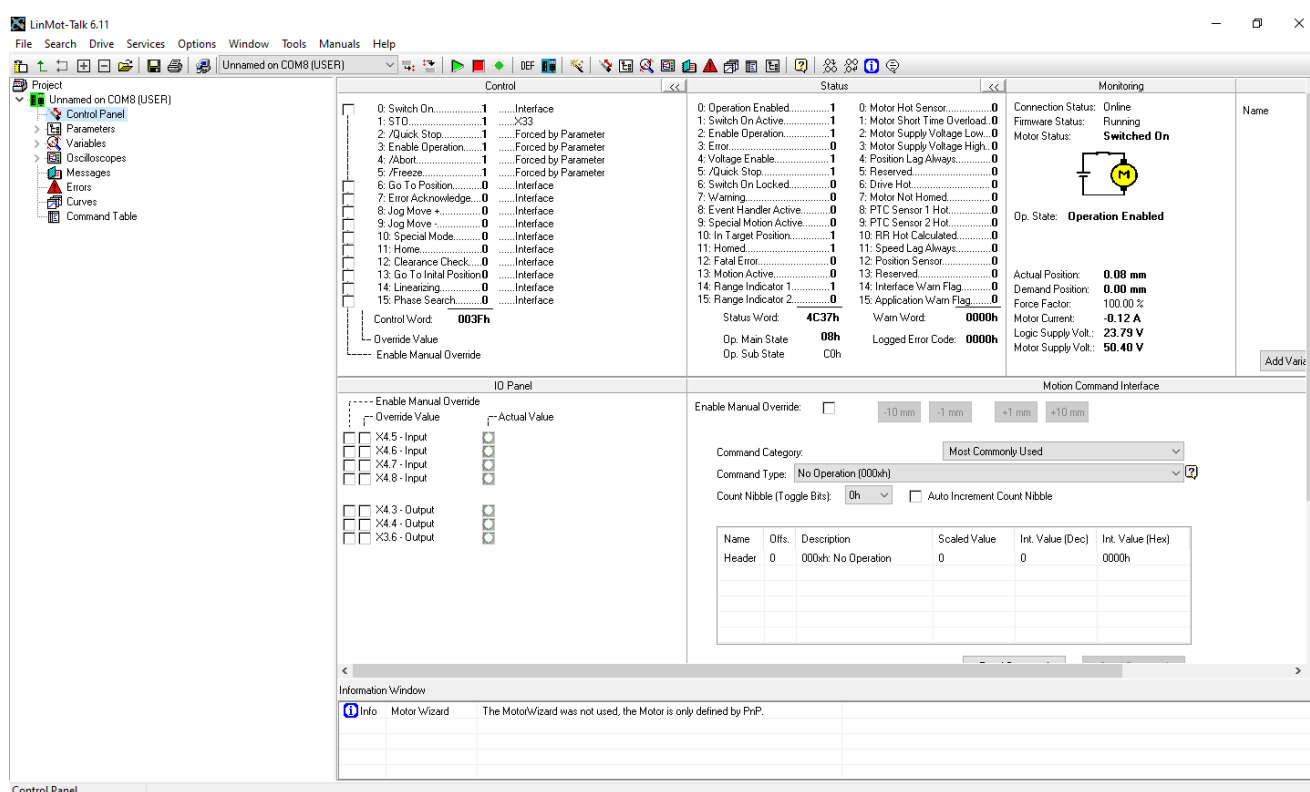
La liaison CanOpen est compatible avec plusieurs automates Siemens ou Schneider et variateurs LinMot. Le matériel à notre disposition est l'automate Schneider Electric M340 et un variateur de vitesse C1100.

COMMUNICATION AVEC LE VARIATEUR DE VITESSE

DESCRIPTION DES ÉTAPES DE CONFIGURATION DE LA LIAISON

La communication avec le variateur de vitesse a été réalisée en utilisant le protocole de communication CANopen. Avant de pouvoir établir une communication avec le variateur, il a fallu configurer la liaison CANopen entre l'automate et le variateur. Les étapes de configuration de la liaison sont décrites ci-dessous.

Tout d'abord, il a été nécessaire de configurer les paramètres de communication sur le variateur, tels que la vitesse de communication, l'adresse CANopen du variateur et les paramètres de transmission de données. Pour cela, il a fallu faire une communication entre le variateur de vitesse et l'application linmot. Grâce à cette application, nous avons pu faire bouger le moteur pour la première fois en envoyant des ordres via l'application. Voici la page de l'application :



Ensuite, sur l'automate, il a fallu créer une configuration de réseau CANopen avec les paramètres de communication correspondants, comme ci-dessous.

EXPLICATION DU FONCTIONNEMENT DE LA COMMUNICATION

En faisant la configuration du module CANopen, nous nous sommes rendu compte que lorsqu'on était dans le module sur contrôle expert, il fallait sélectionner des registres pour pouvoir écrire dedans. Nous avons donc cherché quel registre mettre dans la documentation et nous avons trouvé la composition suivante :

LinMot C1100 Linear Drive System with CANopen interface (C1100.eds)

C1100_xx_xx_xS_xxx
Voie 0

Configuration PDO Contrôle d'erreur Bootup Dictionnaire d'objets

Emission (%I) ☐ Masquer les PDO vides

PDO	Type d'é...	Inhibit Ti...	Event Ti...	Symbole	Adr. topo.	%M...	COBID	Index
☒ PDO 1	1	100	0				16#182	
☒ MCSW Statu...					%IW\3.2\0.0.0.22	%MW22		4F00:01
☒ MCSW State...					%IW\3.2\0.0.0.23	%MW23		4F00:02
☒ MCSW Actu...					%ID\3.2\0.0.0.2	%MW2		4F00:06
☒ PDO 2	1	100	0				16#282	
☒ MCSW Dem...					%ID\3.2\0.0.0.4	%MW4		4F00:08
☒ MCSW Dem...					%ID\3.2\0.0.0.0	%MW0		4F00:05
☒ PDO 3	1	100	0				16#382	
☒ MCSW War...					%IW\3.2\0.0.0.25	%MW25		4F00:04
☒ MCSW Logg...					%IW\3.2\0.0.0.24	%MW24		4F00:03
☐ PDO 4	1	100	0				-	

Réception (%Q) ☐ Masquer les PDO vides

PDO	Type d'é...	Inhibit Ti...	Event Ti...	Symbole	Adr. topo.	%M...	COBID	Index
☒ PDO 1	1						16#202	
☒ MCSW Contr...					%QW\3.2\0.0.0.28	%MW72		4F00:11
☒ MCSW MCH...					%QW\3.2\0.0.0.29	%MW73		4F00:12
☒ MCSW MCP...					%QD\3.2\0.0.0.0	%MW44		4F00:13
☒ PDO 2	1						16#302	
☒ MCSW MCP...					%QD\3.2\0.0.0.2	%MW46		4F00:15
☒ MCSW MCP...					%QD\3.2\0.0.0.4	%MW48		4F00:17
☒ PDO 3	1						16#402	
☒ MCSW MCP...					%QD\3.2\0.0.0.8	%MW52		4F00:1B
☒ MCSW MCP...					%QD\3.2\0.0.0.10	%MW54		4F00:1D
☐ PDO 4	1						-	

Fonction :
Default

En examinant la documentation et les registres nécessaires pour contrôler le variateur de vitesse, nous avons identifié les registres correspondant aux entrées (%I) et sorties (%Q). Étant donné que notre objectif était de faire fonctionner le moteur, nous avons concentré notre attention sur les registres de sortie (%Q). Cependant, il était important de noter que chaque partie (%I et %Q) était subdivisée en quatre sous-parties, même si nous n'avons utilisé que trois de ces sous-parties dans notre projet.

Dans chaque sous-partie, nous avons dû sélectionner deux ou trois registres différents pour contrôler différents aspects du variateur de vitesse. Heureusement, l'automate assigne automatiquement une adresse mémoire à chaque registre, ce qui nous a évité d'avoir à configurer manuellement chaque registre.

Pour mieux comprendre la communication entre l'automate et le variateur de vitesse, nous allons vous expliquer les registres que nous avons utilisés. Dans la partie "Émission", le registre principal est le "Status Word". Il sert à connaître l'état du moteur et du variateur, et permet de détecter les défauts ou les manques de courant dans le moteur. Dans l'automate, ce registre est représenté par le mot %MW22.

Dans la partie "Réception", nous avons utilisé plusieurs registres. Le premier est le "Control Word", qui permet de contrôler les fonctions de base du moteur, tel que le retour à la position initiale. Les autres registres ont pour objectif de contrôler le moteur de manière plus avancée. Dans notre projet, nous avons utilisé les registres suivants :

- %MW73 : en écrivant une valeur dans ce registre, on indique à quelle fonction du variateur on souhaite accéder. Toutes les informations liées à ce registre sont disponibles dans la documentation du variateur, mais dans notre cas, nous écrivons dans la fonction qui a le numéro 901.
- %MW44 : en écrivant une valeur dans ce registre, on indique à quelle position on souhaite envoyer le moteur.
- %MW45 : en écrivant une valeur dans ce registre, on corrige la vitesse du moteur.
- %MW46 : en écrivant une valeur dans ce registre, on indique la vitesse d'accélération souhaitée.
- %MW47 : en écrivant une valeur dans ce registre, on indique la vitesse de décélération souhaitée.

Ci-dessous, vous pouvez voir la partie de la documentation qui nous a permis de comprendre quelle valeur envoyer pour faire fonctionner correctement le moteur.

Il est important de noter que, pour écrire dans ces registres, il faut les sélectionner dans le module de contrôle expert de l'automate. Cette étape a été nécessaire pour pouvoir établir une communication efficace entre l'automate et le variateur de vitesse via le protocole CANopen.

3) Execute Motion Command : VAI 16Bit Go To Pos (090xh)

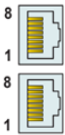
CMD Header	→			0901h
CMD Slave Header	→			0901h
Par Byte 0...1	→	Target Position :	50mm	01F4h
Par Byte 2...3	→	Maximal Velocity :	1m/s	03E8h
Par Byte 4...5	→	Acceleration :	10m/s ²	0064h
Par Byte 6...7	→	Deceleration :	10m/s ²	0064h

L'utilisation des registres a été une étape importante dans la communication avec le variateur de vitesse. Il a fallu sélectionner les bons registres pour envoyer les commandes de contrôle du moteur. La documentation du variateur a été une aide précieuse pour comprendre les différentes fonctions des registres et leur utilisation dans le contrôle du moteur. Il était important de noter que chaque registre avait une taille de 8 octets, alors que dans certains cas, nous devions écrire dans seulement 2 octets. Ainsi, en sélectionnant les registres appropriés et en utilisant les mots de mot distants correspondants tels que %MW44, %MW45 et %MW46, nous avons pu envoyer des commandes pour contrôler la position, la vitesse d'accélération et la vitesse de décélération du moteur.

RÉSOLUTION DES PROBLÈMES RENCONTRÉS

La première importation de notre projet Control Expert dans l'automate n'a pas pu aboutir. Le logiciel nous a remonté l'information qu'il y avait un problème de compatibilité. Dans le logiciel, nous avons défini nos équipements ainsi que leur version. L'automate ModicomM340 était alors configuré en version 2, or, le nôtre était en version. Nous avons alors juste changé la configuration sur Control Expert afin de pouvoir importer le projet.

Après avoir installé le fichier EDS du variateur LinMot C1100, le logiciel Control Expert ne détectait pas de connexion via CANopen avec le variateur. Avec l'aide de M. AUBRIOT, nous avons vérifié le câblage de notre connectique DB9.

X7 - X8		RS485 / CAN (ON GP DRIVES ONLY)	
	RJ-45	Nr	
		1	RS485_Rx+
	RJ-45	2	RS485_Rx-
		3	RS485_Tx+
	RJ-45	4	NC
		5	GND (1k Ohm to GND)
	RJ-45	6	RS485_Tx-
		7	CAN_H
	RJ-45	8	CAN_L
		Case	Shield

- » Use twisted pair (1-2, 3-6, 4-5, 7-8) cable for wiring.
- » The built in CAN and RS485 terminations can be activated by S4.2 and S4.3.
- » X7 is internally connected to X8 (1:1 connection)

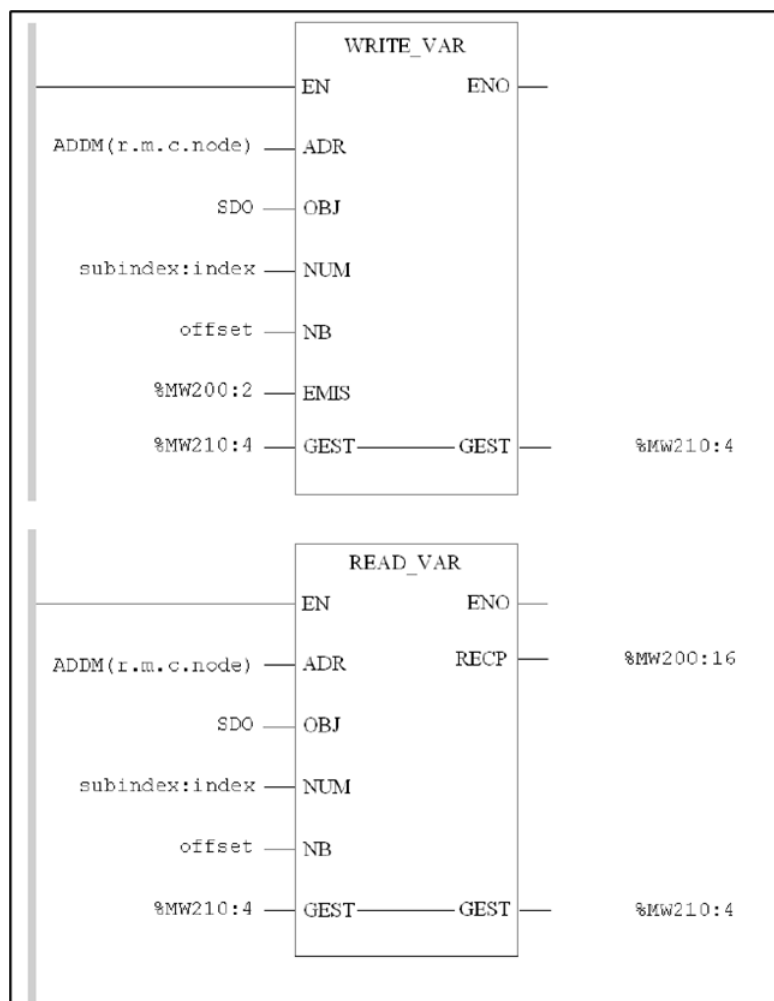
(cf. documentation variateur LinMot C1100, page 13)

À l'aide de la documentation du variateur, nous avons d'abord remarqué que nous nous étions trompés de câble. Il nous fallait connecter la masse (5, GND) et nous avons seulement besoin des câbles 7 et 8. Respectivement CAN_H et CAN_L. Nous avons alors ressoudé notre connectique comme présentée sur la photo.



M. LUCIDARME nous a parlé d'un des fournisseurs français du matériel LinMot, il s'agit de Transtechnik. Il nous a alors conseillé de contacter leur SAV qui serait capable de nous aider. Le SAV de la société nous a répondu qu'il ne se servait plus de la liaison CANopen depuis un long moment. Il n'a donc pas pu nous aider sur ce problème.

M. AUBRIOT est ensuite venu nous aider. Il a alors émis l'hypothèse que nous n'avions pas installé le bon firmware sur le variateur. Il nous manquerait un firmware permettant la communication CANopen. Il nous a ensuite dit que nous pouvions générer ces requêtes via les blocs WRITE_VAR et READ_VAR, disponible avec l'automate M340.

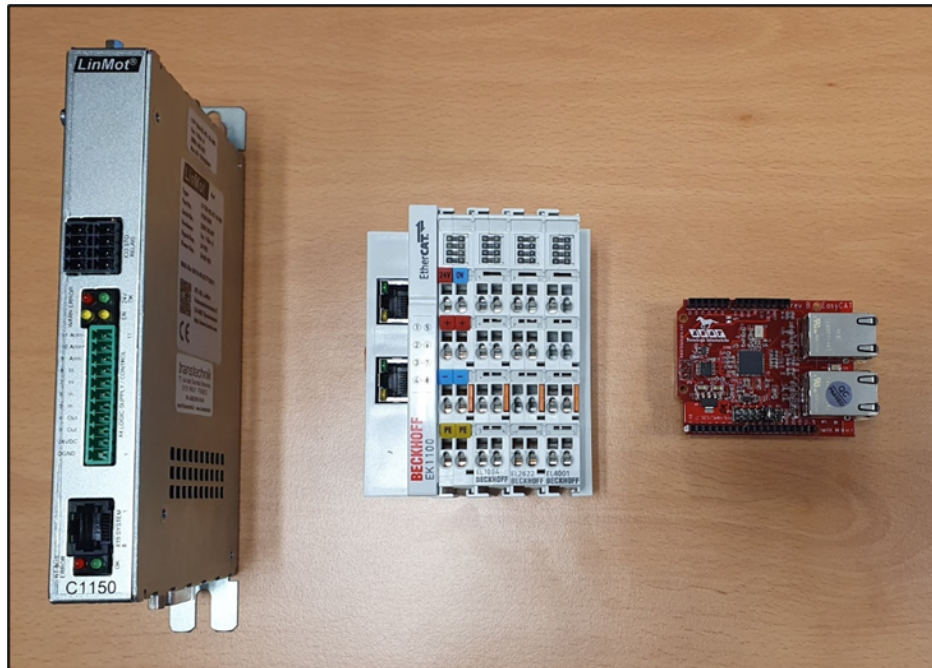


(cf. documentation Modicom M340 avec Unity Pro, page 167)

Mais cette solution étant très fastidieuse, l'un d'entre nous s'est mis à travailler sur cette solution pendant que les deux autres cherchaient d'autres solutions. M. LUCIDARME nous a parlé de la communication via EtherCAT.

LIAISON EN ETHERCAT

M. LUCIDARME a essayé de contacter un de ses collègues, M. Franck MERCIER, professeur à Polytech Angers. Celui-ci a déjà essayé une communication via EtherCAT il y a quelques années, mais n'a pas eu le temps d'aller au bout de ses recherches. Nous sommes allés voir M. MERCIER à Polytech afin qu'il nous parle de ces recherches. Il nous a aussi donné tout le matériel nécessaire.



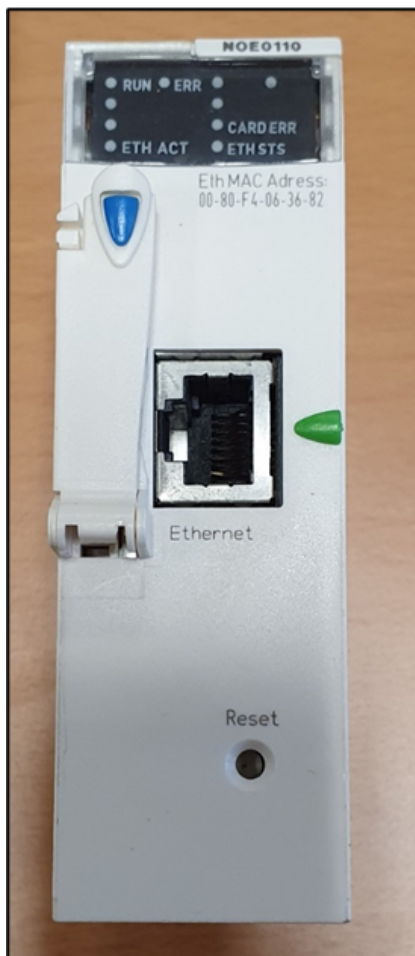
Nous nous sommes alors renseignés sur comment fonctionnent ces trois appareils ensemble mais nos recherches n'ont pas abouties. Le fonctionnement de cette communication est assez complexe à mettre en place et nous devons avancer sur notre projet.

En parallèle de nos recherches, M^{me} GROMELLON a contacté un ancien intervenant à l'IUT en poste chez Schneider Electric, M. GUITTET. Dans l'un de ces mails qui nous a été transféré par M^{me} GROMELLON, il évoquait un problème de correspondance entre la CPU et le firmware.

INSTALLATION DU BON FIRMWARE

Ce n'est qu'après-avoir essayé toutes les solutions évoquées plut tôt, qui n'ont pas abouti, que nous avons essayé de mettre à jour le firmware de la CPU. Nous avons d'abord cherché la version 2 du firmware sur Internet ainsi que sur le site de Schneider. Mais nous n'avons rien trouvé. M^{me} GROMELLON a alors envoyé un mail à l'employé de chez Schneider, nous avons alors pu recevoir la bonne version du firmware.

Au début du projet, nous possédions un module Ethernet NOE0110, mais celui-ci est seulement compatible avec la version 1 du firmware. Nous avons alors changé de module pour un plus récent et compatible avec notre version. Nous avons alors mis à la place un module NOC0401.



À l'aide de ce nouveau module, nous avons enfin pu faire communiquer tous nos matériels entre eux. Nous avons alors pu écrire dans les registres du variateur LinMot et commencer à faire bouger le moteur via Control Expert.

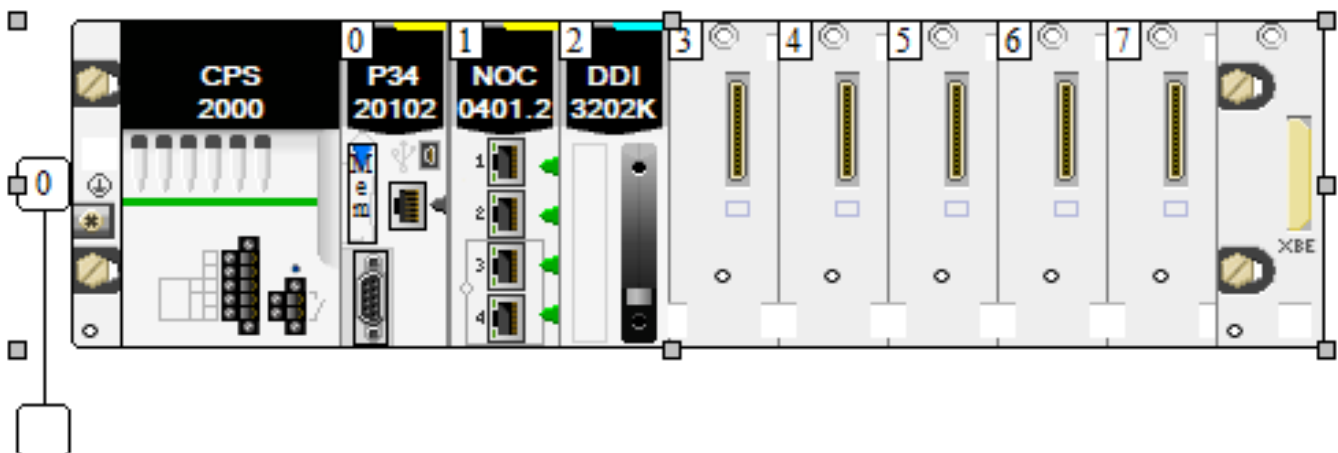
RÉALISATION DU PROGRAMME DE L'AUTOMATE

DESCRIPTION DES DIFFÉRENTES ÉTAPES DE PROGRAMMATION

Avant de commencer à vous parler avec des termes techniques complexes, nous allons vous expliquer quelques termes préliminaires. Tout d'abord, pour notre projet, nous utilisons un logiciel spécialement conçu pour les équipements Schneider, appelé "Control Expert". Ce logiciel offre plusieurs méthodes de programmation, mais nous avons choisi de coder principalement en utilisant deux méthodes : le Grafcet et le Ladder. Nous expliquerons ces méthodes au fur et à mesure de notre rapport.

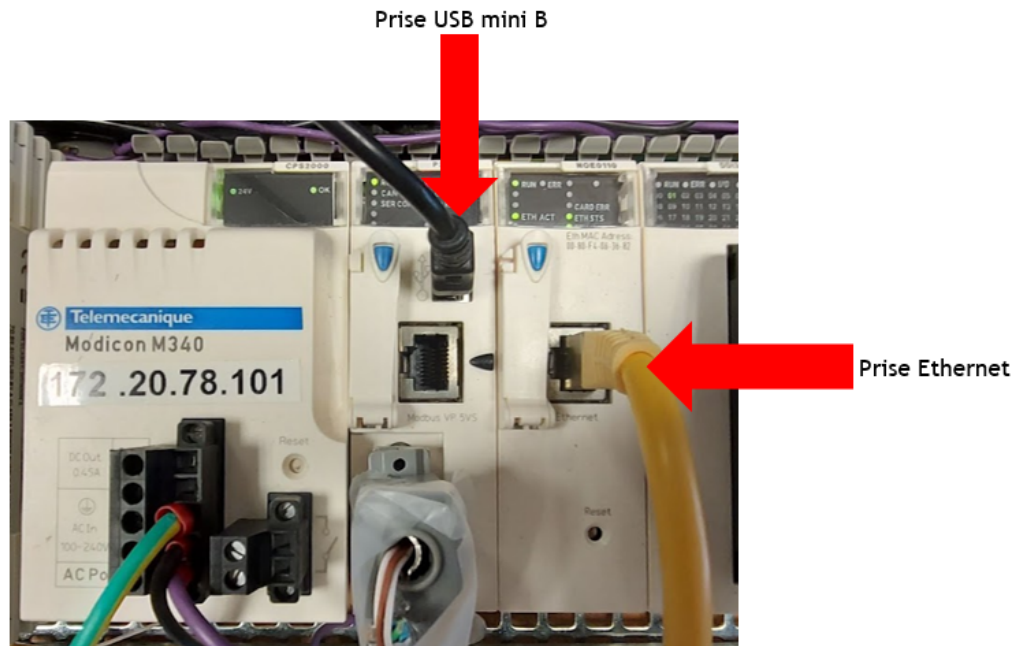
CONFIGURATION MATERIEL

Pour commencer à réaliser le programme dans l'automate, il est essentiel de configurer le matériel nécessaire. Pour cela, nous avons consulté le catalogue de Schneider pour trouver tous les composants requis pour notre automate, en commençant par la CPU P342010 et en allant jusqu'au module Ethernet NOE0110 (TCP/IP), comme illustré ci-dessous :



Après avoir sélectionné les composants nécessaires, nous avons connecté l'automate à un nouveau programme vide afin d'éviter que les anciennes conditions actives ne déclenchent une sortie sans avertissement préalable. Pour ce faire, il faut utiliser un câble USB mini B pour connecter les configurations réseau, puis se connecter à l'automate via un câble Ethernet.

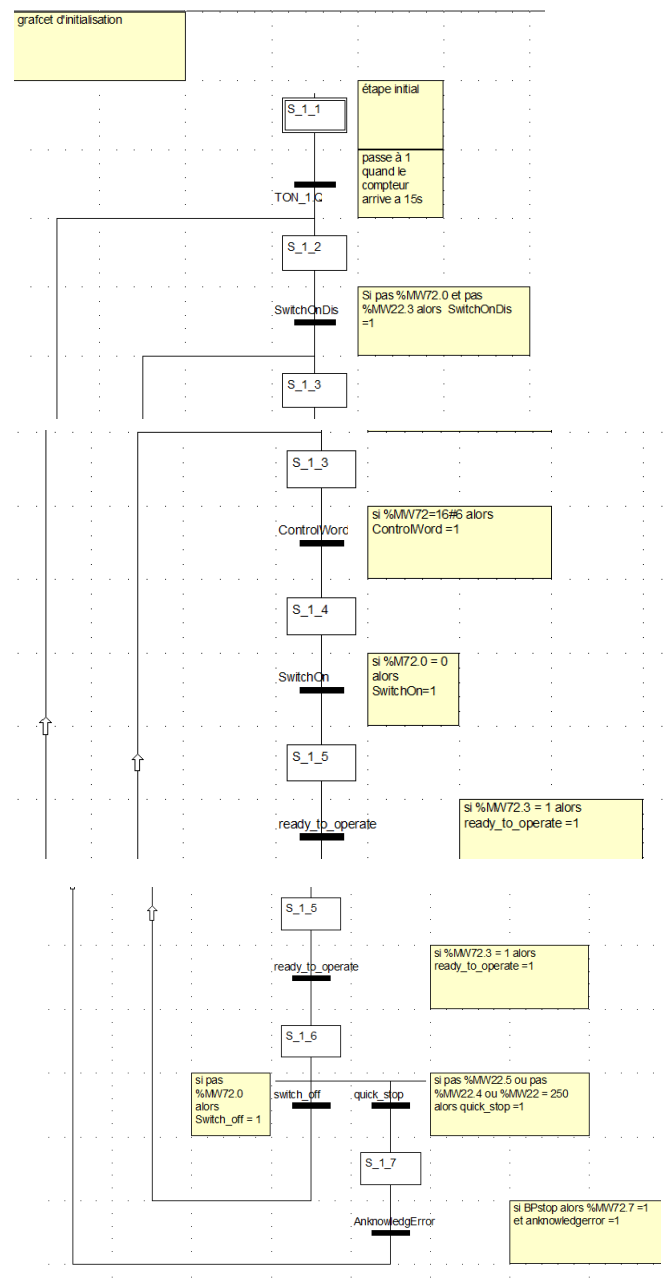
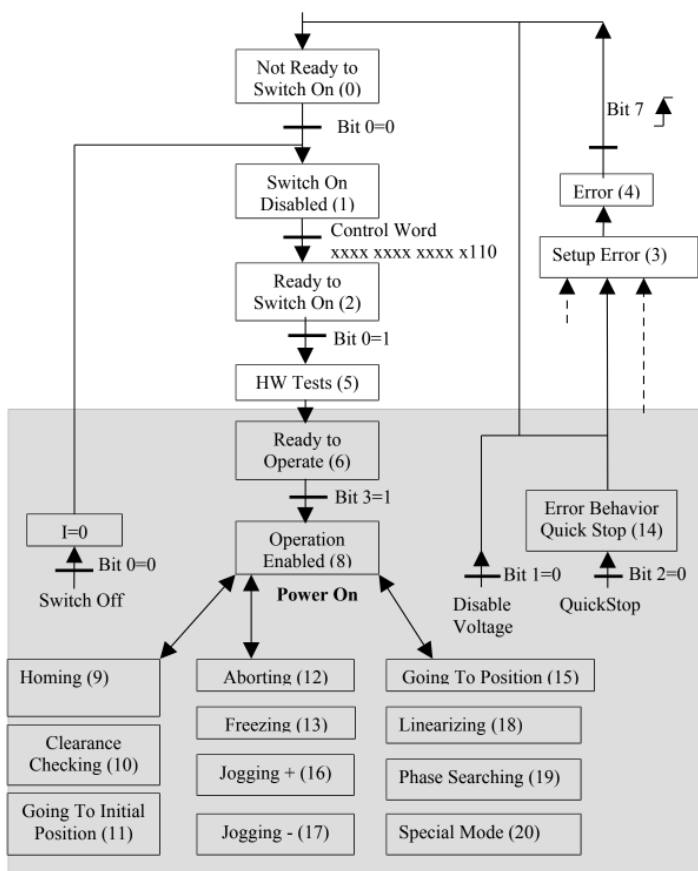
Sur la photo ci-dessous, vous pouvez voir le câble USB mini B et le câble Ethernet correctement branchés :



Il est important de noter que lors de la première utilisation, il est crucial de connecter l'automate à un programme vide et de bien configurer tous les composants matériels pour assurer un fonctionnement optimal et éviter tout dysfonctionnement. Une fois la configuration matérielle terminée, nous avons pu passer à la programmation de l'automate.

PROGRAMME DANS L'AUTOMATE

Lors de nos recherches approfondies sur la liaison CANopen entre le variateur de vitesse et l'automate, nous avons constaté qu'une procédure d'initialisation était nécessaire pour établir une bonne liaison entre les deux. Nous avons découvert cette procédure dans la documentation du variateur de vitesse. Ainsi, nous avons utilisé ces informations pour réaliser un grafcet d'initialisation conforme aux recommandations. Les deux grafkets correspondants sont présentés dans l'annexe 2 pour le grafcet de la documentation et dans l'annexe 3 pour le grafcet que nous avons réalisé.



En suivant ces procédures, nous avons réussi à établir une liaison stable entre le variateur de vitesse et l'automate, ce qui nous a permis de poursuivre la suite du projet. Grâce à cette liaison stable, nous avons pu mettre en place le fonctionnement du grafcet. Pour cela, nous avons élaboré un ladder détaillant les actions à effectuer à chaque étape ainsi que les transitions entre celles-ci. Chaque étape du grafcet représente un état dans lequel le système automatisé se trouve, tel que le moteur en marche ou l'arrêt d'urgence activé.

Chaque étape est symbolisée par un rectangle vertical, dans lequel on peut indiquer une action à réaliser, comme l'activation d'une sortie ou l'attente d'un événement. Les étapes sont reliées par des transitions, qui représentent les passages d'un état à un autre. Une transition peut être activée par un événement, comme l'appui sur un bouton, ou par la condition remplie dans l'étape précédente. Les transitions sont symbolisées par des traits noirs perpendiculaires au trait qui partent de l'étape courante pour arriver à l'étape suivante. Le ladder est une représentation graphique utilisée pour programmer les automates, qui permet de visualiser les différentes étapes du processus automatisé ainsi que les actions à effectuer à chaque étape et les événements déclencheurs pour passer d'une étape à l'autre. Nous reviendrons plus en détail sur ce point dans la partie suivante.

Enfin, la deuxième partie du programme est liée à l'interface homme-machine (IHM). Nous avons réalisé un ladder permettant de modifier le message envoyé par l'IHM pour que l'automate puisse le comprendre.

La dernière partie du programme permet de rentrer les bonnes valeurs au bon endroit puis de les envoyer au variateur de vitesse.

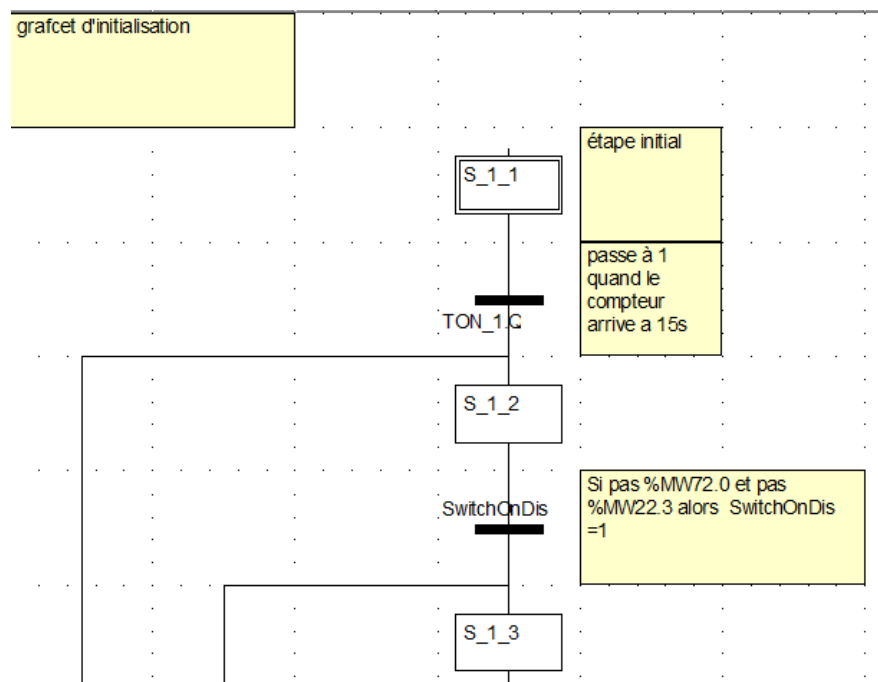
EXPLICATION DU FONCTIONNEMENT DU PROGRAMME

Nous venons de vous présenter la hiérarchie du programme et nous allons maintenant entrer dans les détails pour que tout le monde puisse comprendre son fonctionnement. Comme expliqué précédemment, la première partie concerne le grafcet d'initialisation de la liaison CANopen. Ensuite, nous allons vous expliquer la partie du programme qui permet la communication entre l'IHM et l'automate. Enfin, nous vous présenterons la dernière partie du programme qui est chargée de faire bouger le moteur.

GRAFCET D'INITIALISATION

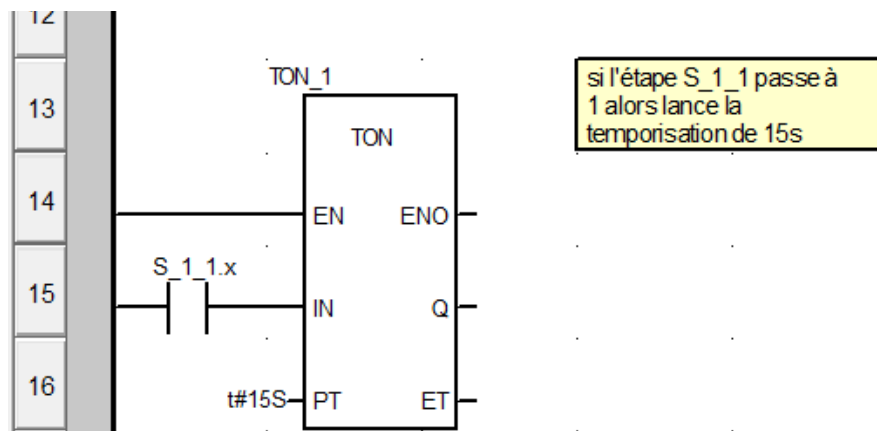
Pour bien comprendre ce grafcet, nous allons tout d'abord expliquer les transitions qui permettent de passer d'une étape à l'autre, puis nous détaillerons les actions réalisées à chaque étape. Vous pouvez retrouver toutes les images de grafcet dans l'Annexe 3 et toutes les transitions dans l'Annexe 4. N'hésitez pas à consulter ces annexes pour une meilleure compréhension du programme.

LES TRANSITIONS

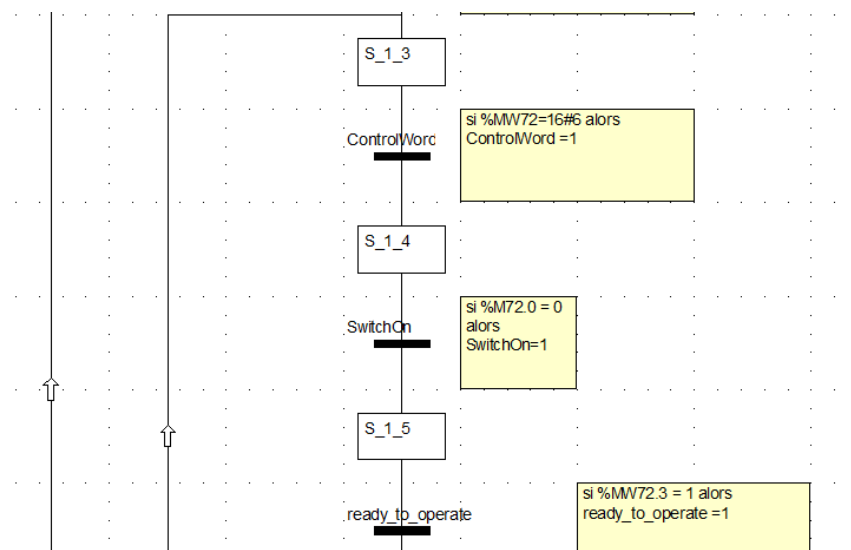
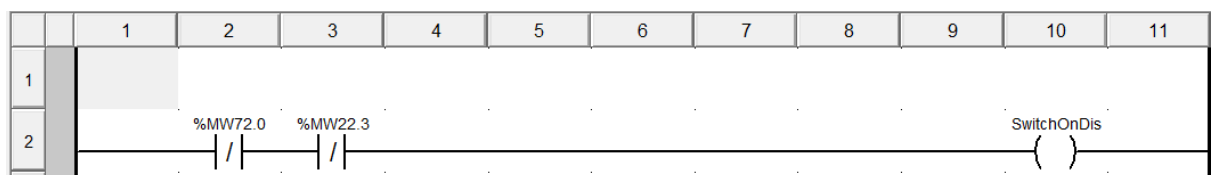


L'objectif de l'étape S_1_1 est d'être la première étape exécutée à chaque fois que l'automate est démarré. Pour passer de l'étape S_1_1 à S_1_2, il faut que la temporisation de 15 secondes soit passée, ce qui signifie que la condition

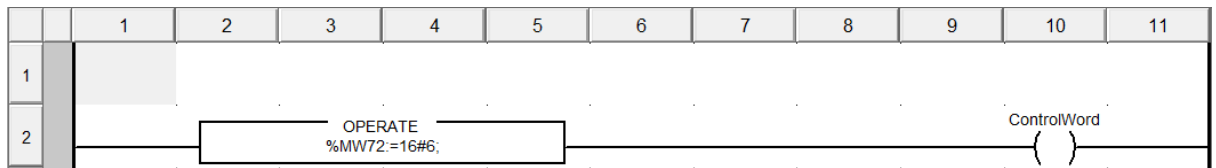
"TON_1.Q" doit être vraie. Pour activer la temporisation, il faut être dans l'étape S_1_1. En écrivant "TON_1.Q", on regarde la sortie Q de la temporisation et quand cette sortie passe à 1, la condition devient vraie.



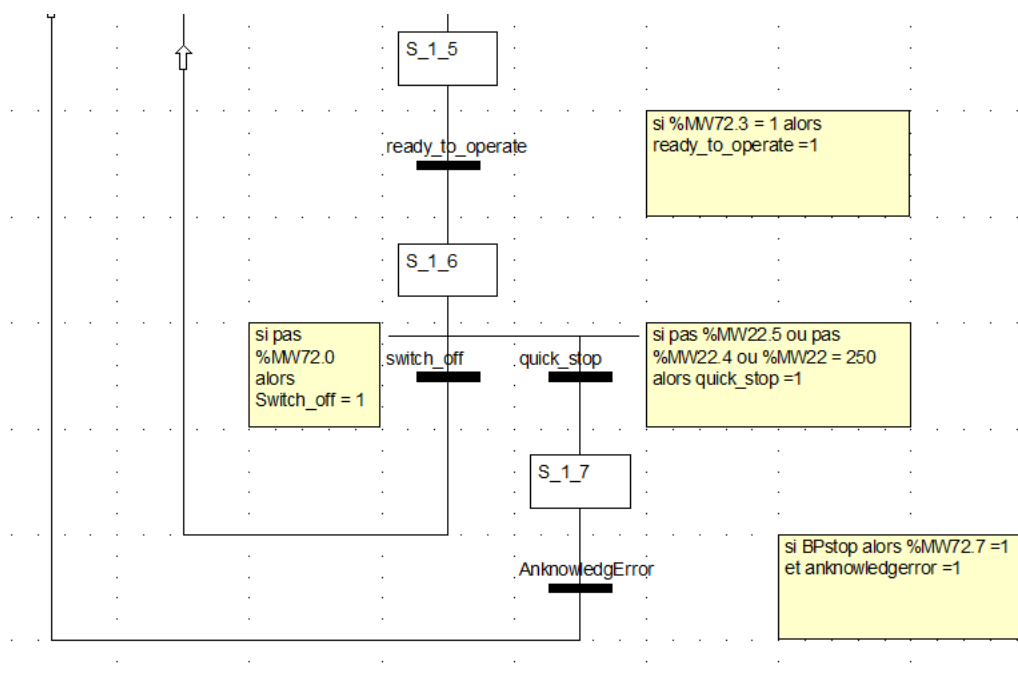
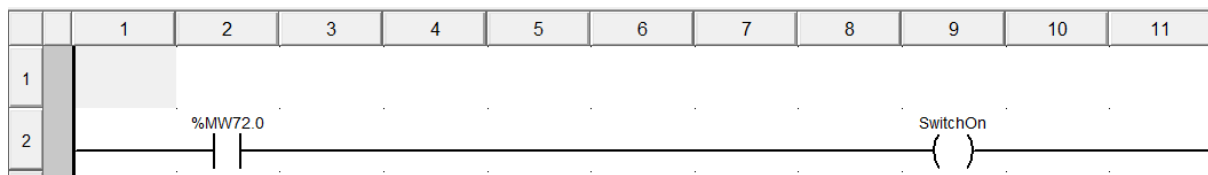
Pour passer de l'étape S_1_2 à S_1_3, la sortie %MW72.0 et %MW22.3 ne doivent pas être à 1, ce qui fait passer la transition "SwitchOnDis" à 1. Si %MW72.0 est égal à 0, cela signifie qu'il n'y a pas de courant, et si %MW22.3 est égal à 0, cela signifie qu'il n'y a aucun défaut.



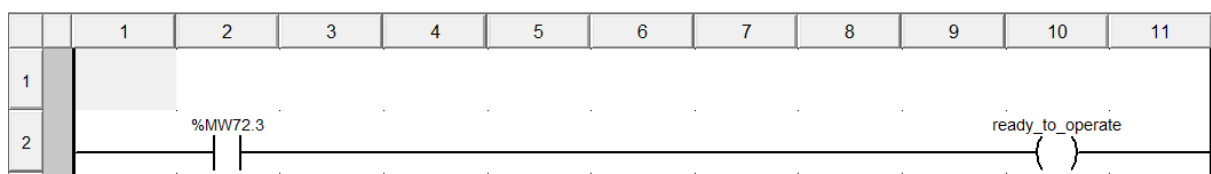
Pour passer de l'étape S_1_3 à S_1_4, la transition "ControleWord" doit passer à 1, ce qui nécessite que %MW74 soit égal à la valeur 16 en hexadécimal. Cela permet d'activer la possibilité du "Quick Stop" et "Voltage Enable".



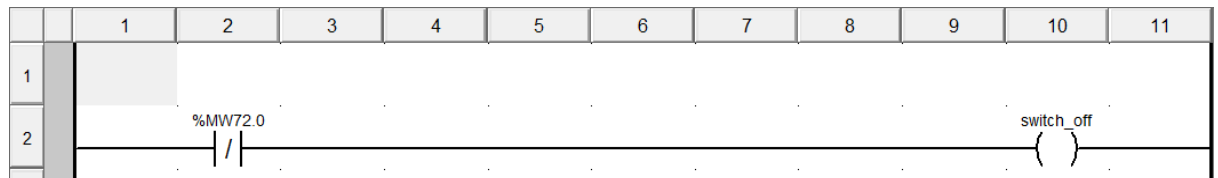
Pour passer de l'étape S_1_4 à S_1_5, la transition "SwitchOn" doit passer à 1, ce qui nécessite que %MW72.0 soit égal à 1. Cela permet de vérifier que le courant est bien à 0.



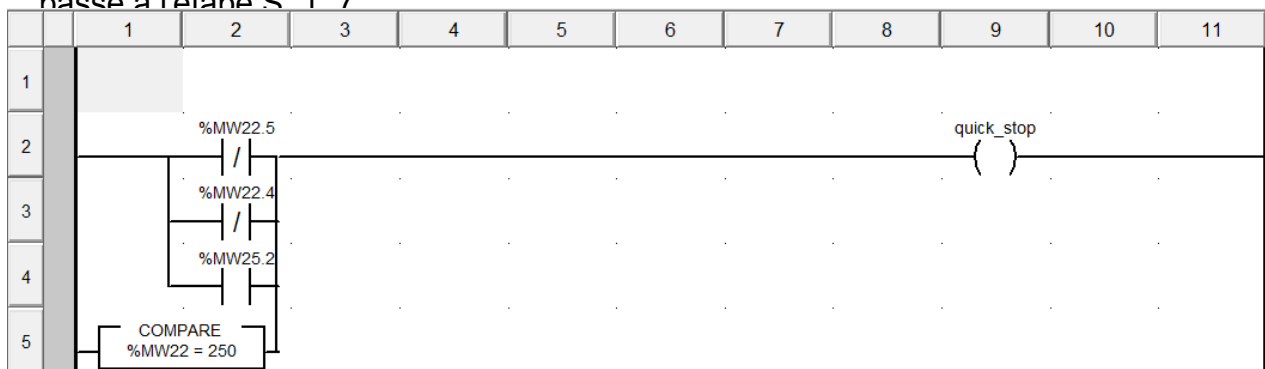
Pour passer de l'étape S_1_5 à S_1_6, la transition "ready_to_operate" doit passer à 1, ce qui nécessite que %MW72.3 soit vrai. Cela signifie que la commande de courant a été activée.



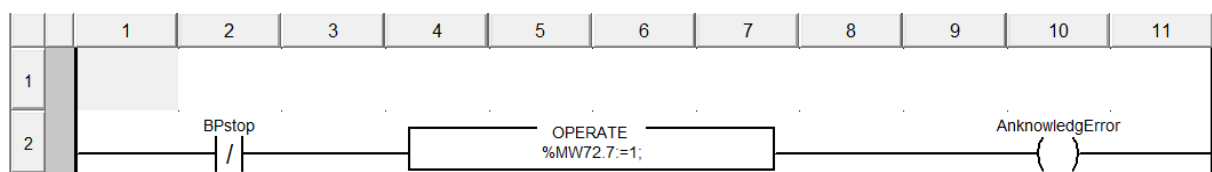
L'étape S_1_6 correspond à l'étape où le moteur est en fonctionnement normal, c'est-à-dire que c'est l'étape où le programme reste jusqu'à ce qu'il y ait un défaut ou que le moteur soit éteint, ce qui fait passer la transition "switch_off" à 1. La deuxième possibilité de sortie de l'étape S_1_6 est qu'il y ait un défaut, donc la transition "quick_stop" passera à 1. Pour la première possibilité pour sortir de l'étape S_1_6, il faut que la transition "switch_off" soit à 1, ce qui nécessite que %MW72.0 ne soit pas à 1. Cela signifie que le courant passe à 0. Si cette transition passe à 1, on revient directement à l'étape S_1_3.



La deuxième possibilité est que la variable "quick_stop" soit égale à 1, ce qui indique que le bouton d'arrêt d'urgence a été enclenché. Pour que cette transition soit activée, les variables %MW22.4 et %MW25.2 doivent être inactives, ce qui signifie qu'il n'y a plus de tension et que le moteur ne peut plus fonctionner. De plus, la variable %MW22.5 doit être égale à 0, sinon cela signifie que le bouton d'arrêt d'urgence n'a pas été correctement enclenché. Si cette transition est activée, on passe à l'étape S_1_7.

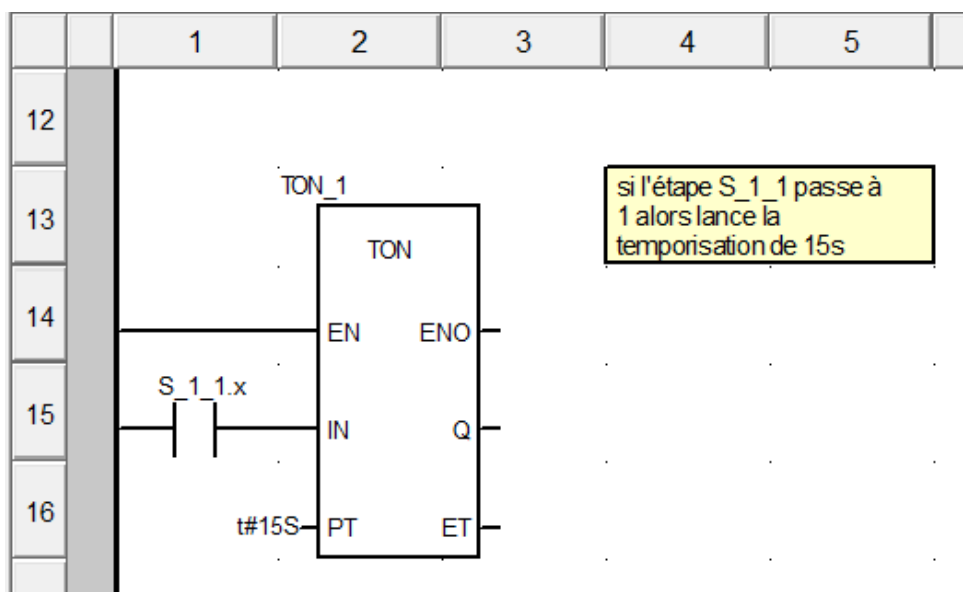


Dans l'étape S_1_7, la transition "AcknowedgErreur" doit être activée pour passer à l'étape suivante. Cette transition est activée lorsque le bouton stop est appuyé, ce qui indique que l'utilisateur a pris connaissance de l'erreur. Si cette transition est activée, on revient à l'étape S_1_2.

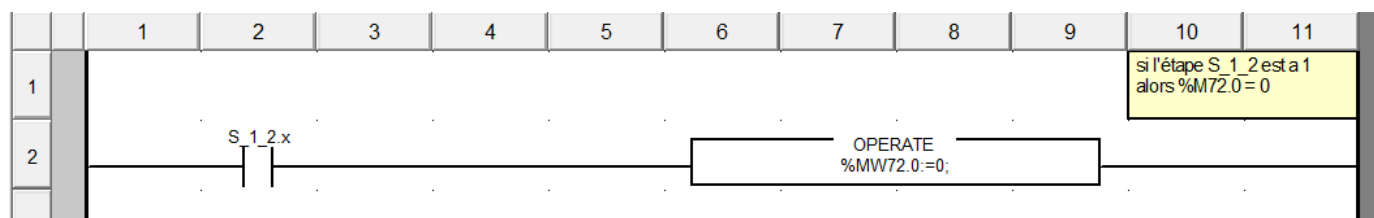


Maintenant que nous avons expliqué chaque transition, nous allons vous présenter les actions effectuées pour chaque étape. Lorsqu'une étape est active, la variable correspondante passe à 1, ce qui permet de réaliser des actions spécifiques.

Ci-dessous, nous prenons l'exemple de l'étape S_1_1, qui lance une temporisation lorsque activée. Les images associées à chaque étape sont disponibles en annexe 5 pour faciliter la compréhension.



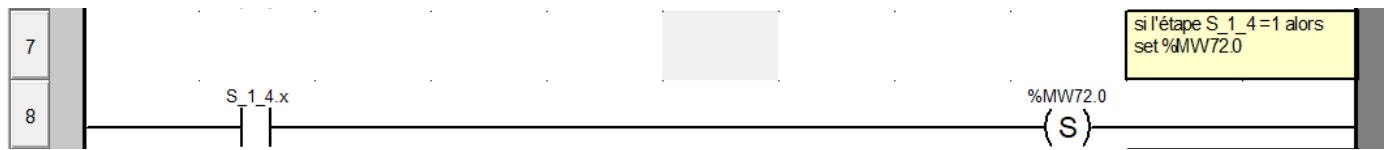
Pour l'étape S_1_2, comme indiqué dans l'image, le bit %MW72.0 est mis à 0, ce qui éteint le moteur.



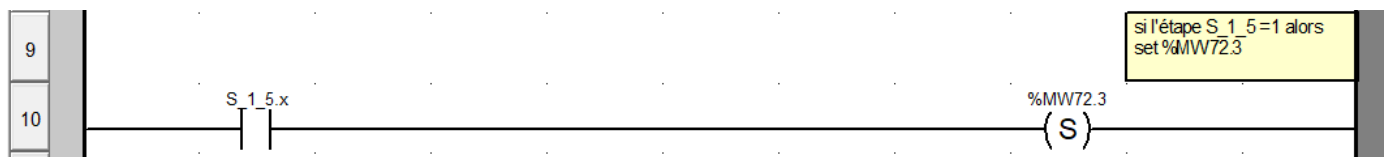
Pour les étapes S_1_3 et S_1_7, le bit %MW72.0 est réinitialisé, ce qui arrête le moteur.



Pour l'étape S_1_4, le bit %MW72.0 est activé, ce qui met le moteur en marche.



Enfin, pour l'étape S_1_5, le bit %MW72.3 est activé, ce qui permet d'activer les commandes de mouvement.



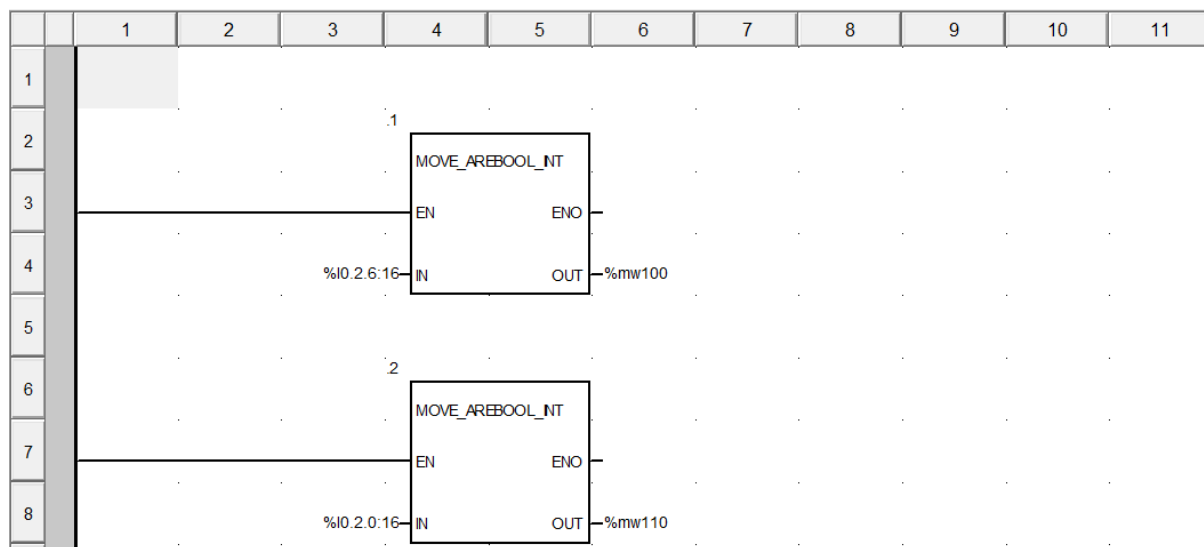
Après avoir présenté le grafcet d'initialisation, nous allons maintenant explorer la partie du programme dédiée à la communication entre l'automate et l'IHM. Pour cela, nous utilisons un bloc spécifique appelé "MOVE_AREEBOOL_INT".

Ce bloc fonctionne de manière simple : dans la section "IN", nous indiquons une valeur entière, tandis que dans la section "OUT", nous attribuons une autre valeur entière dans l'automate. Par exemple, nous avons utilisé "%I0.2.6:16" dans la section "IN" et "%MW100" dans la section "OUT".

Lorsque nous souhaitons lire une variable provenant de l'IHM, nous consultons les bits de "%MW100".

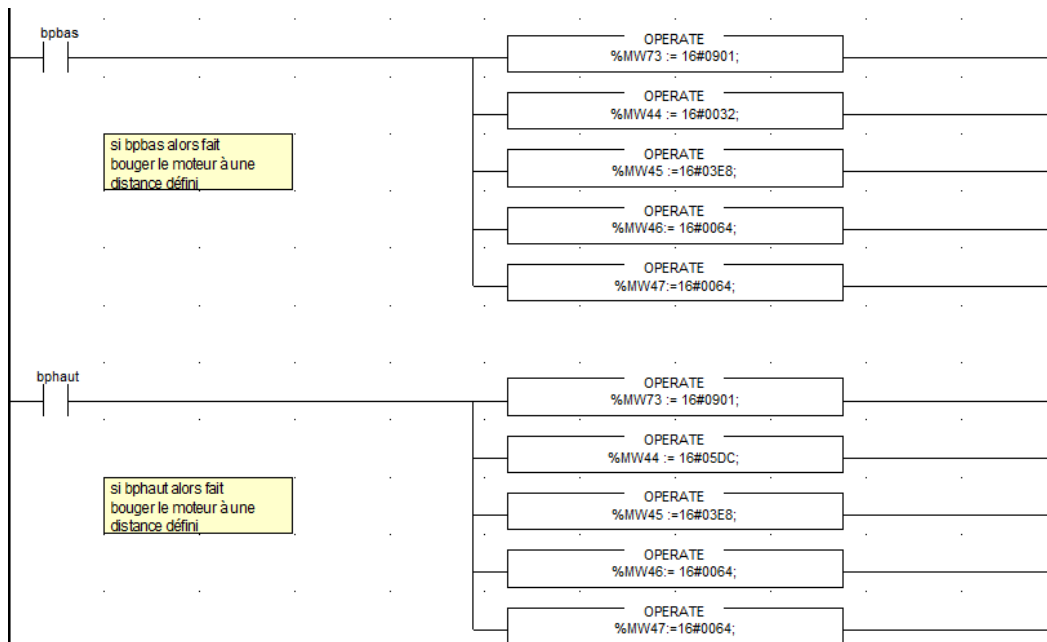
Dans notre programme, nous avons utilisé deux instances du bloc "MOVE_AREEBOOL_INT" car nous avons besoin de 20 sorties au total. Chaque instance du bloc permet de gérer 10 sorties distinctes. Ainsi, nous avons configuré la première instance pour les sorties de %MW100 à %MW109, et la deuxième instance pour les sorties de %MW110 à %MW119. Cela nous a permis d'avoir suffisamment de sorties pour répondre aux besoins de notre application.

Ci-dessous une image du programme réalisé :

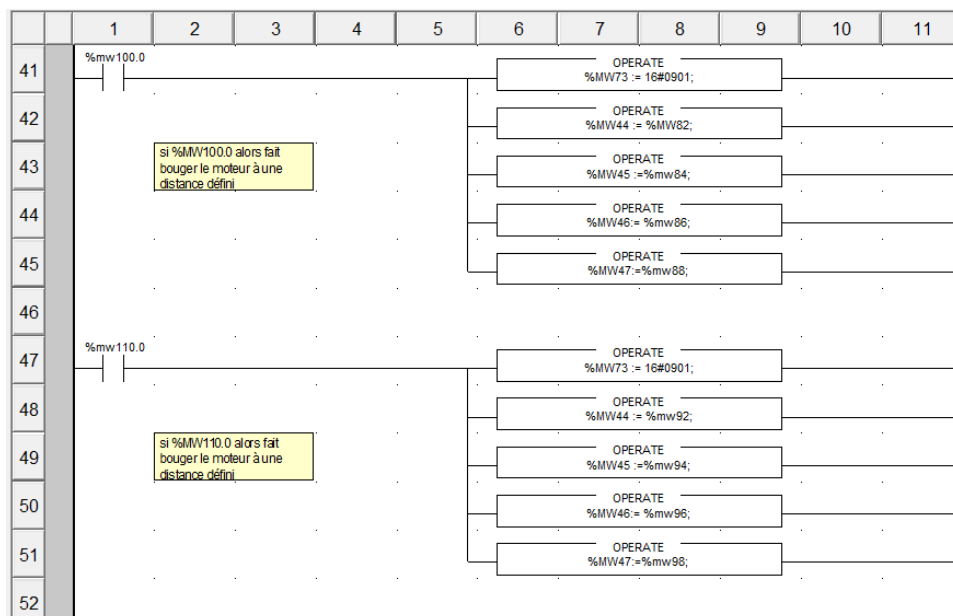


CONTRÔLE DU MOTEUR

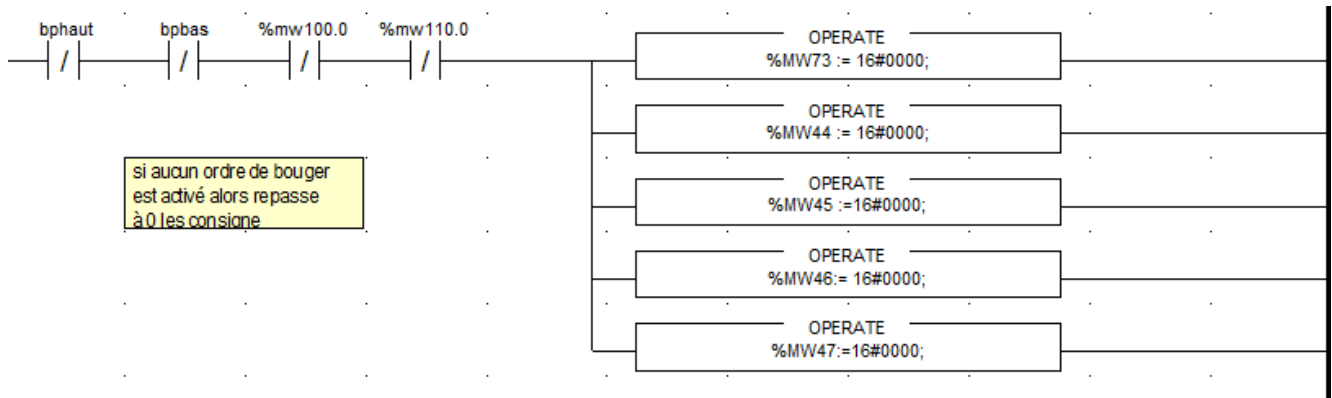
Nous allons maintenant aborder la partie du programme qui contrôle le moteur à travers le ladder. Comme expliqué précédemment, le contrôle du moteur nécessite l'interaction de plusieurs registres. Les images présentées dans l'annexe 6 illustrent les actions à effectuer sur les différentes variables en fonction de l'action entreprise. Par exemple, lorsque le bouton bas est activé, cela entraîne l'envoi de chiffres dans des variables spécifiques.



Le programme pour les boutons sur l'IHM fonctionne de la même manière, à la différence près que l'utilisateur remplit une variable au lieu d'utiliser des chiffres prédéfinis. Le code correspondant est visible ci-dessous.



Afin que le système fonctionne correctement, il est nécessaire d'envoyer la valeur 0 dans le registre pour effacer son contenu, chaque fois qu'aucun bouton n'est activé, comme cela est illustré dans l'exemple ci-dessous.



La dernière partie du programme qui fait bouger le moteur a pour but de l'amener à sa butée. Cela sert à set sa position initial. Pour faire cela il faut activer le bit `%MW72.11`. Comme vous pouvez le voir ci-dessous il y a 2 instructions qui peuvent arriver à cet adjectif: le 1er est le bouton stop et le 2eme est un bouton sur l'IHM.



RÉSOLUTION DES PROBLÈMES RENCONTRÉS

Pendant la phase de programmation, nous avons rencontré deux problèmes liés à la communication réseau. Le premier était lié à la liaison CANOpen, comme expliqué précédemment dans la section sur la communication en utilisant ce protocole. Le deuxième problème était lié à la liaison Ethernet, nécessaire pour attribuer une adresse IP à l'automate et lui permettre de communiquer avec l'IHM et l'ordinateur.

Pour changer l'adresse IP de l'automate, nous avons simplement modifié les paramètres dans la configuration matérielle du module de communication Ethernet, en y insérant l'adresse IP souhaitée, puis nous avons transféré cette configuration matérielle à l'automate à l'aide d'un câble USB mini B.

Cependant, nous avons constaté que le module de communication ne réagissait pas et n'attribuait pas l'adresse IP souhaitée. En consultant la documentation du module, nous avons découvert qu'une manipulation était nécessaire pour le faire fonctionner correctement. Vous pouvez voir ci-dessous la partie de la documentation qui nous a aidé à résoudre ce problème.

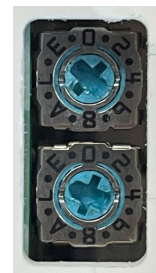
Les paramètres de ces deux commutateurs rotatifs s'ajoutent pour spécifier la source de l'adresse IP du module, comme suit :

Paramètre	Description
Commutateur supérieur :	
0...9	Valeur des dizaines pour la définition du nom de l'équipement (0, 10, 20...90). Elle est ajoutée à la valeur des unités du commutateur inférieur afin d'identifier le module auprès d'un serveur DHCP.
A...F	Valeur des dizaines pour la définition du nom de l'équipement (100, 200, 300...150). Elle est ajoutée à la valeur des unités du commutateur inférieur afin d'identifier le module auprès d'un serveur DHCP.
Commutateur inférieur :	
0...9	Valeur des unités pour la définition du nom de l'équipement (0, 1, 2 à 9). Elle est ajoutée à la valeur des dizaines du commutateur supérieur afin d'identifier le module auprès d'un serveur DHCP.
A + B	BootP ¹ : le module obtient son adresse IP auprès d'un serveur BootP.
C + D	Stored ¹ : le module utilise l'adresse IP configurée dans l'application.
E + F	Clear IP ¹ : le module utilise son adresse IP par défaut.
1. Le commutateur inférieur détermine seul le comportement du module. Le commutateur supérieur n'est pas pris en compte.	

Comme vous pouvez le voir il fallait identifier si on voulait le mettre en DHCP ou en IP fixe.

Nous avons choisi de passer l'automate en IP fixe. Par conséquent, le commutateur supérieur ne nous est pas utile on peut donc le laisser sur n'importe quelle position.

Pour le commutateur inférieur, nous l'avons placé sur C ou D dans le mode "stored" pour que le module acquiert l'adresse IP définie dans contrôle expert.



RÉALISATION DE L'IHM

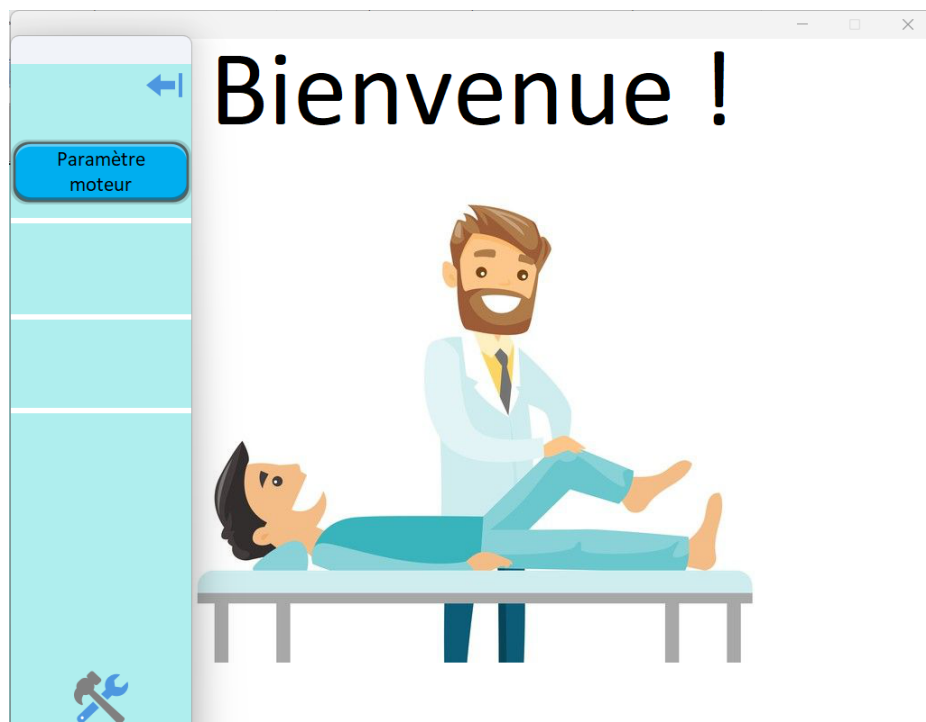
PRÉSENTATION DES CHOIX DE CONCEPTION DE L'IHM

Dans le cadre de ce projet, l'IHM (Interface Homme-Machine) est une IHM KEP permettant à l'utilisateur de contrôler et surveiller la maquette. L'IHM est conçue pour être simple et efficace, facile à comprendre et à utiliser, similaire à l'utilisation d'un site web. Les utilisateurs disposent de différents niveaux d'autorisation pour accéder à différentes fonctionnalités de l'IHM. Les utilisateurs de niveau 1 peuvent gérer la vitesse de déplacement et l'emplacement de la maquette, tandis que les utilisateurs de niveau 2, qui sont les personnes de maintenance, peuvent également régler la vitesse d'accélération et de décélération.

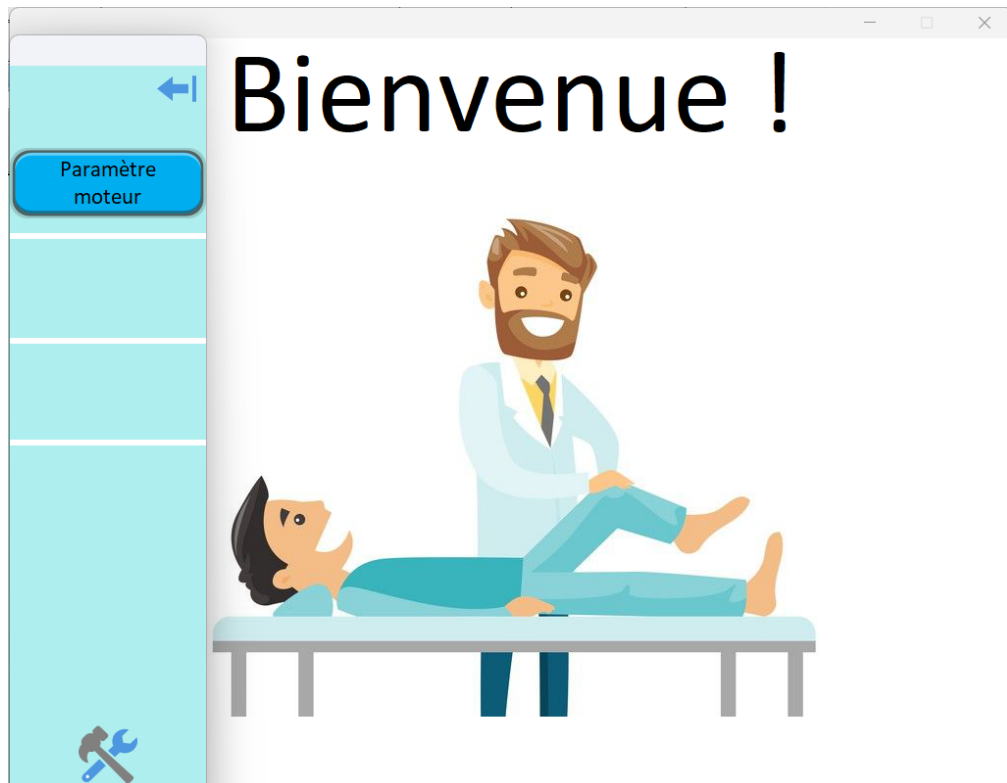
Les choix de conception de l'IHM ont été guidés par l'objectif d'optimiser l'expérience utilisateur. L'IHM est dotée d'une interface tactile pour une utilisation pratique et intuitive. Les informations de fonctionnement de la maquette peuvent être facilement visualisées et les paramètres de contrôle du moteur linéaire peuvent être ajustés en temps réel.

L'IHM est connectée à l'automate via un câble Ethernet. La conception de l'IHM a été réalisée en prenant en compte la compatibilité avec le variateur de vitesse linmot C1100 et la liaison CANopen.

En résumé, les choix de conception de l'IHM ont été guidés par la simplicité, l'efficacité et l'expérience utilisateur optimisée. La conception de l'IHM permet une utilisation pratique et intuitive de la maquette, avec différents niveaux d'autorisation pour les différents utilisateurs



Comme expliqué précédemment, l'IHM est dotée de deux niveaux d'autorisation permettant un contrôle du moteur adapté aux besoins des utilisateurs. Le premier niveau offre une interface simple et conviviale, permettant aux utilisateurs de se concentrer sur les commandes sans se soucier des détails techniques. Pour accéder à ces commandes, il suffit d'appuyer sur le bouton "Paramètres moteur" situé ci-dessous.



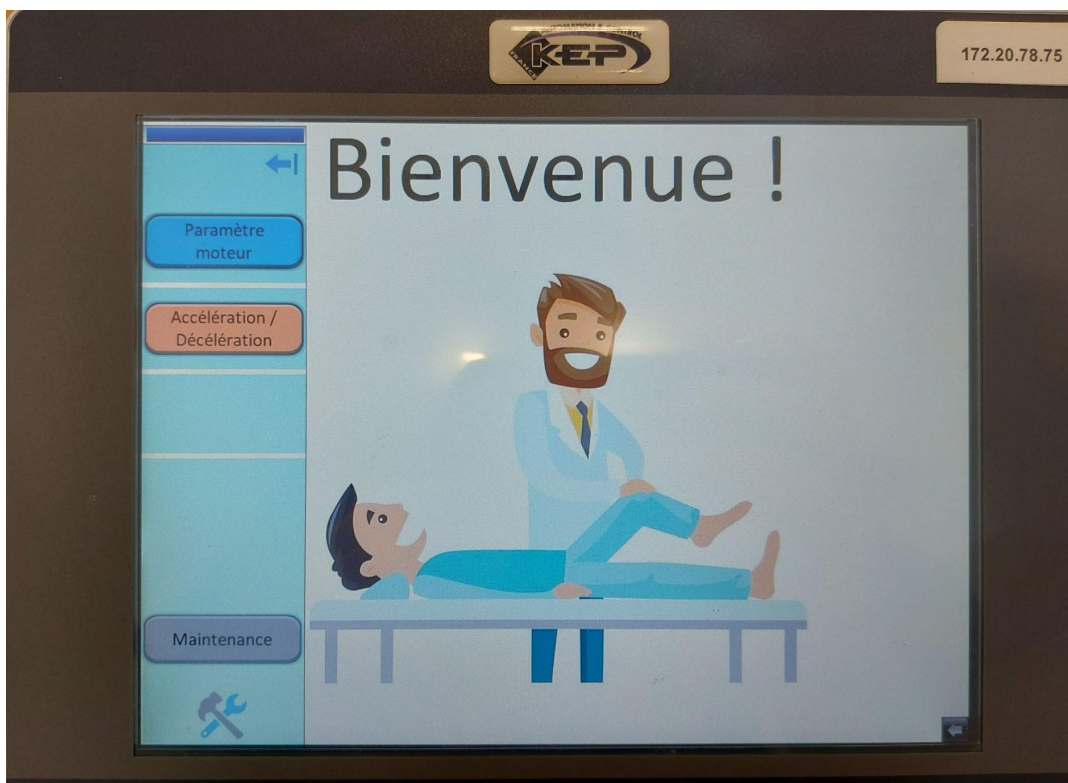
En appuyant sur le bouton "Paramètres moteur", une fenêtre s'ouvre, offrant les options de configuration suivantes :

- Un interrupteur permettant de ramener le moteur à sa position initiale.
- Un afficheur permettant de régler la vitesse du moteur.
- Un afficheur permettant de définir la position basse du moteur.
- Un afficheur permettant de définir la position haute du moteur.
- Un bouton de validation pour envoyer toutes les données saisies à l'automate.
- Un bouton permettant d'accéder au code nécessaire pour obtenir l'autorisation de maintenance.

Pour les utilisateurs disposant des autorisations de maintenance, l'accès aux commandes supplémentaires se fait en appuyant sur le bouton situé en bas à gauche de l'IHM. Une fenêtre contextuelle apparaît, vous permettant de saisir un titre et un code. Une fois ces informations fournies, vous accédez à une page offrant les mêmes options que précédemment, mais avec deux boutons supplémentaires :

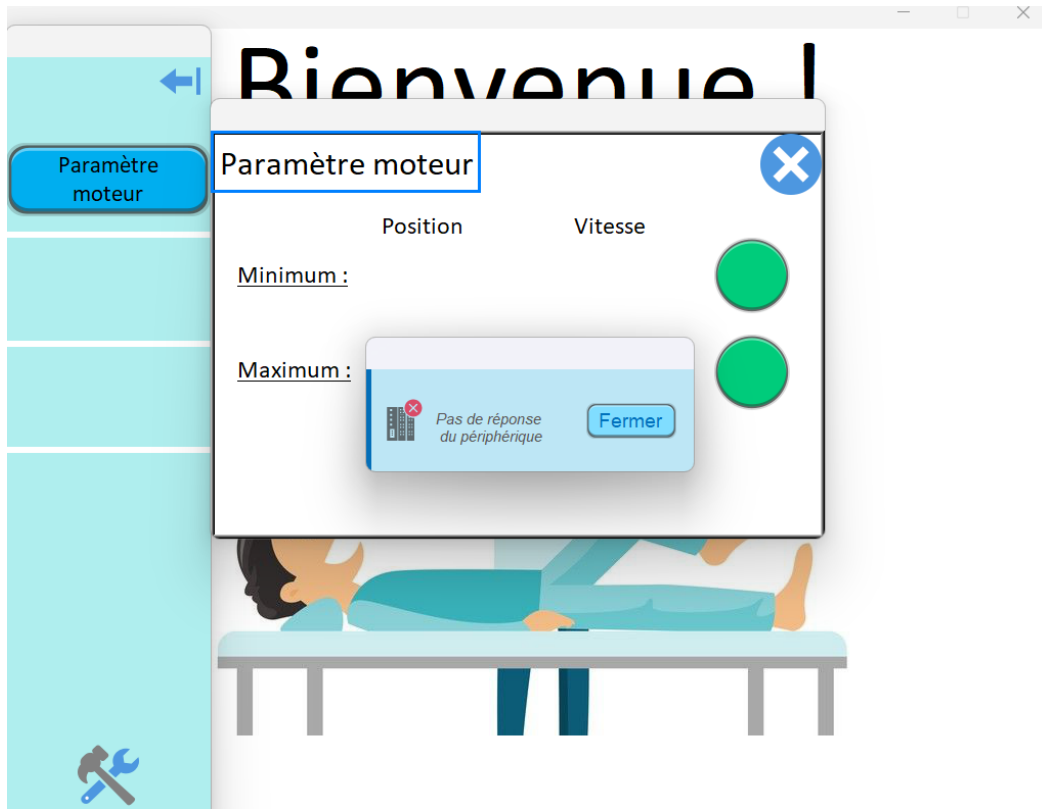
- "Accélération/Décélération" : Permet de modifier la vitesse d'accélération et de décélération à l'aide d'un afficheur.
- "Maintenance" : Offre un espace dédié au changement d'autres paramètres pour le moteur.

Ces fonctionnalités supplémentaires permettent aux utilisateurs de maintenance d'effectuer des réglages plus avancés et d'optimiser les performances du moteur.



RÉSOLUTION DES PROBLÈMES RENCONTRÉS

Lors des tests pour vérifier le bon fonctionnement du système, un message d'erreur est apparu.



Nous avons identifié ce problème lors de notre dernière séance de travail en autonomie et malheureusement, nous n'avons pas réussi à trouver une solution. Cependant, nous avons envisagé différentes pistes de solution telles que des erreurs dans la liaison entre l'automate et l'IHM, ou encore des erreurs dans les appellations de variables pouvant provoquer une mauvaise communication. Étant donné le manque de temps, nous n'avons pas été en mesure de résoudre ce problème.

CONCLUSION ET PERSPECTIVES

BILAN DU PROJET ET ATTEINTE DES OBJECTIFS

L'objectif du projet était de concevoir un système permettant de faire fonctionner un moteur linéaire à l'aide d'un pupitre de boutons et d'une interface homme-machine. Pour y parvenir, plusieurs étapes étaient nécessaires, notamment la conception d'une maquette contenant un automate, un variateur de vitesse, un moteur linéaire et une IHM, ainsi que la mise en place de la communication entre le variateur de vitesse et l'automate via une liaison CANopen. Le développement du programme pour l'automate et l'IHM, ainsi que la communication entre ces deux éléments, étaient également des éléments clés du projet.

Malgré les efforts déployés et la réalisation de la plupart des objectifs, nous avons malheureusement rencontré plusieurs obstacles qui ont eu un impact négatif sur la progression du projet. Le manque de temps ainsi que des problèmes techniques ont entraîné des retards significatifs, ce qui nous a empêchés de finaliser la communication entre l'IHM et l'automate. Malgré ce contretemps, nous avons réussi à atteindre plusieurs objectifs clés du projet et nous sommes fiers de ce que nous avons accompli.

LIMITES ET PERSPECTIVES D'AMÉLIORATION

LES LIMITES

Nous avons rencontré plusieurs problèmes, comme décrit précédemment, mais nous avons réussi à les résoudre un à un. Cependant, il nous reste encore deux problèmes à résoudre que nous n'avons pas réussi à résoudre pour le moment.

Le premier est lié à la communication entre l'IHM et l'automate. Nous avons plusieurs pistes pour identifier et résoudre ce problème. La première possibilité est que nous avons mal fait la liaison entre l'automate et l'IHM. La deuxième possibilité est que nous avons utilisé une mauvaise appellation de variable, ce qui peut provoquer une mauvaise communication.

Le deuxième problème concerne le variateur de vitesse qui ne se stabilise pas lorsqu'on allume l'automate en même temps. Nous avons essayé plusieurs tests, tels que l'utilisation d'une temporisation avant l'envoi des ordres par l'automate, mais cela n'a rien donné. Le seul moment où nous avons réussi à allumer le variateur sans problème était lorsque l'automate n'était pas relié. Nous avons donc trouvé une solution consistant à mettre une temporisation à l'allumage, ce qui signifie que nous allumons d'abord le variateur et 15 secondes plus tard, l'automate s'allume également. Malheureusement, nous n'avons pas pu tester cette idée en raison du manque de temps et de matériel.

LES PERSPECTIVES

Le projet présente de nombreuses perspectives intéressantes. Nous avons exploré deux projets potentiels qui pourraient bénéficier de l'utilisation d'un moteur linéaire et d'une liaison CANopen.

Le premier projet vise à concevoir une chaise de rééducation pour les kinésithérapeutes équipée d'un moteur linéaire. Ce dispositif permettrait de bouger la jambe du patient sans que le kiné ne sollicite excessivement son dos. Grâce à l'IHM, les kinésithérapeutes pourront contrôler la vitesse et la position du vérin en toute sécurité, offrant une expérience de rééducation plus efficace et moins douloureuse pour les patients.

Le deuxième projet est axé sur la liaison CANopen, une technologie de communication connue pour sa rapidité. Il s'agira de développer une solution de commande de moteur en utilisant la technologie CANopen. L'objectif sera d'explorer les différentes options pour optimiser les performances et la fiabilité du système, en vue d'une utilisation pour des jambes articulées ou d'autres applications similaires.

RETOUR SUR L'EXPÉRIENCE ET LES COMPÉTENCES ACQUISES

Tout au long de ce projet, nous avons développé de nombreuses compétences, notamment en termes de compétences techniques, de compétences en résolution de problèmes, de compétences en gestion de projet et de compétences en analyse et en synthèse. Pour chacune de ces compétences, nous avons identifié les points clés qui nous ont aidés à avancer et à progresser. Nous avons également fourni des exemples concrets pour illustrer notre développement de ces compétences et expliqué comment ces expériences ont influencé notre approche et notre perspective dans le futur.

COMPÉTENCES TECHNIQUES

Pour la première compétence, nous avons acquis des connaissances approfondies sur la mise en place d'une liaison CANopen. Au début du projet, nous n'avions aucune idée de son fonctionnement, mais grâce à nos recherches, nous avons appris tous les tenants et aboutissants de cette liaison. De plus, nous avons également appris à commander un moteur linéaire via un variateur de vitesse en comprenant qu'il ne suffisait pas d'envoyer une simple valeur, mais qu'il fallait envoyer une séquence d'informations pour que le moteur puisse fonctionner correctement. Nous avons également compris l'importance d'agir sur des registres précis pour faire bouger le moteur.

En outre, nous avons également développé des compétences déjà acquises, telles que la programmation d'un automate Schneider via le logiciel Control Expert et la programmation d'une IHM via le logiciel Easy Builder. Nous avons ainsi été en mesure d'améliorer notre compréhension et notre maîtrise de ces outils de programmation.

COMPÉTENCES EN RÉOLUTION DE PROBLÈME

Tout au long de ce projet, nous avons été confrontés à divers problèmes techniques et de communication entre les différents composants du système, que ce soit entre l'automate et le variateur de vitesse, ou entre l'automate et l'IHM. Cependant, grâce à nos recherches sur les documentations, les forums et en communiquant avec les constructeurs pour obtenir de l'aide, nous avons réussi à résoudre la plupart des problèmes. Nous avons également posé de nombreuses questions à notre professeur référent pour mieux comprendre les raisons de ces dysfonctionnements. Cette expérience nous a permis de développer des compétences telles que l'autonomie dans la résolution de problèmes techniques, la capacité de communiquer efficacement avec les constructeurs pour résoudre les problèmes, ainsi que de l'ingéniosité pour trouver des solutions innovantes aux problèmes rencontrés.

COMPÉTENCES EN GESTION DE PROJET

En travaillant ensemble sur ce projet, nous avons développé des compétences clés en matière de collaboration, de planification et d'organisation. Nous avons appris à communiquer efficacement, à répartir les tâches en fonction des compétences de chacun et à coordonner nos efforts pour atteindre les objectifs du projet. Nous avons également appris à gérer les contraintes de temps et de ressources, à prendre des décisions en équipe et à rester flexibles face aux changements imprévus. En somme, ce projet nous a permis de développer nos compétences en travail d'équipe et en gestion de projet.

COMPÉTENCES EN ANALYSE ET EN SYNTHÈSE

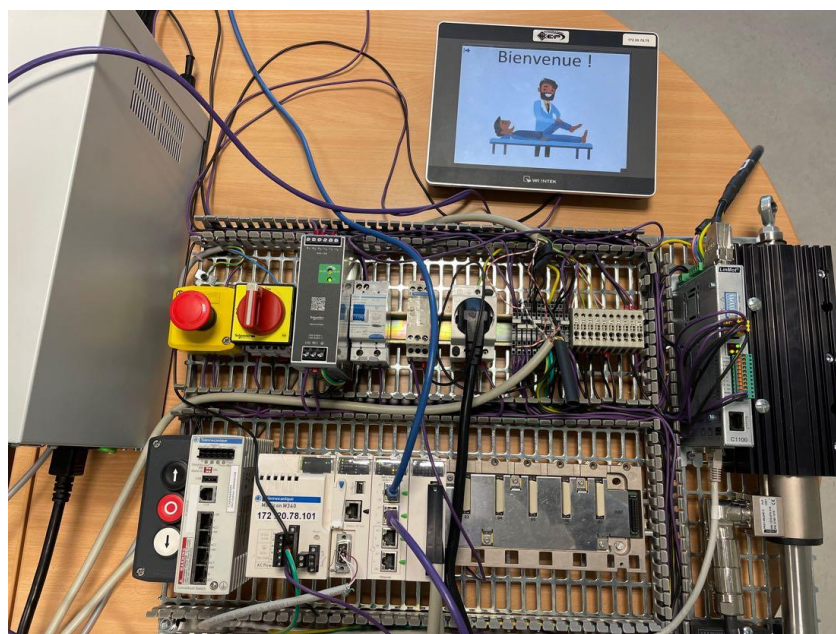
Nous avons mis en place une méthodologie de travail rigoureuse pour mener à bien ce projet. Nous avons commencé par une analyse approfondie des besoins et des contraintes du projet, ainsi que des spécifications techniques et des exigences du client. Nous avons ensuite synthétisé ces informations pour créer une solution appropriée pour le projet. Nous avons travaillé en étroite collaboration pour définir les différentes étapes du projet, répartir les tâches et assurer une bonne coordination entre les membres de l'équipe. Nous avons également régulièrement évalué l'avancement du projet et effectué des ajustements lorsque cela était nécessaire pour garantir la réussite du projet dans les délais impartis.

En conclusion, ce projet nous a permis de développer un large éventail de compétences techniques et non techniques. Nous avons acquis une expertise en conception, réalisation et mise en œuvre de systèmes impliquant l'automatisme, la programmation, l'interface homme-machine et la communication. Nous avons également développé des compétences en gestion de projet, en résolution de problèmes ainsi qu'en analyse et synthèse d'informations.

RÉSUMÉ

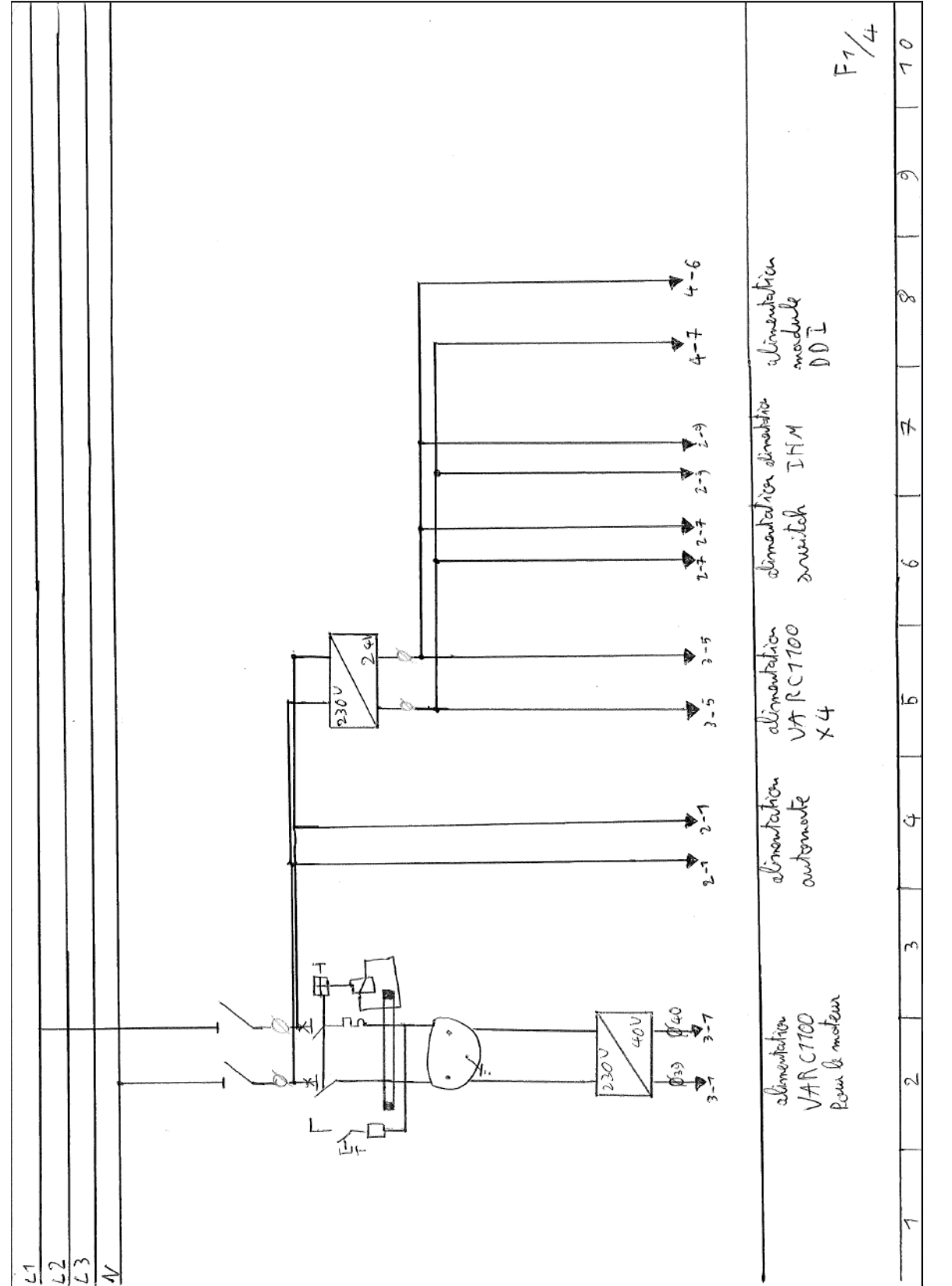
Au cours de ce projet, nous avons travaillé en équipe pour concevoir, réaliser et mettre en œuvre un système complet comprenant un moteur linéaire, un automate, un variateur de vitesse et une interface homme-machine (IHM). Nous avons acquis des compétences techniques en explorant la communication CANOpen, en programmant l'automate et en développant l'IHM à l'aide d'outils tels que Contrôle Expert et Easy Builder. Malgré les obstacles techniques et les difficultés de communication entre les composants, nous avons démontré notre autonomie en menant des recherches, en posant des questions pertinentes et en trouvant des solutions créatives pour résoudre les problèmes. Grâce à une gestion efficace du projet, nous avons su gérer le temps, les ressources et les contraintes afin de respecter les délais et d'atteindre les objectifs fixés. En résumé, ce projet nous a permis de développer des compétences techniques avancées, de renforcer notre capacité à résoudre des problèmes complexes et de perfectionner nos aptitudes en gestion de projet, ce qui sera bénéfique pour nos futures carrières professionnelles. Voici le résultat final du projet.

During this project, we worked as a team to design, build, and implement a complete system consisting of a linear motor, a PLC, a variable speed drive, and a human-machine interface (HMI). We acquired technical skills by exploring CANOpen communication, programming the PLC, and developing the HMI using tools such as Control Expert and Easy Builder. Despite technical challenges and communication issues between components, we demonstrated our autonomy by conducting research, asking relevant questions, and finding creative solutions to solve problems. Through effective project management, we managed time, resources, and constraints to meet deadlines and achieve project goals. In summary, this project allowed us to develop advanced technical skills, strengthen our ability to solve complex problems, and improve our project management skills, which will be beneficial for our future careers. Here is the final result of the project.



ANNEXES

ANNEXE 1



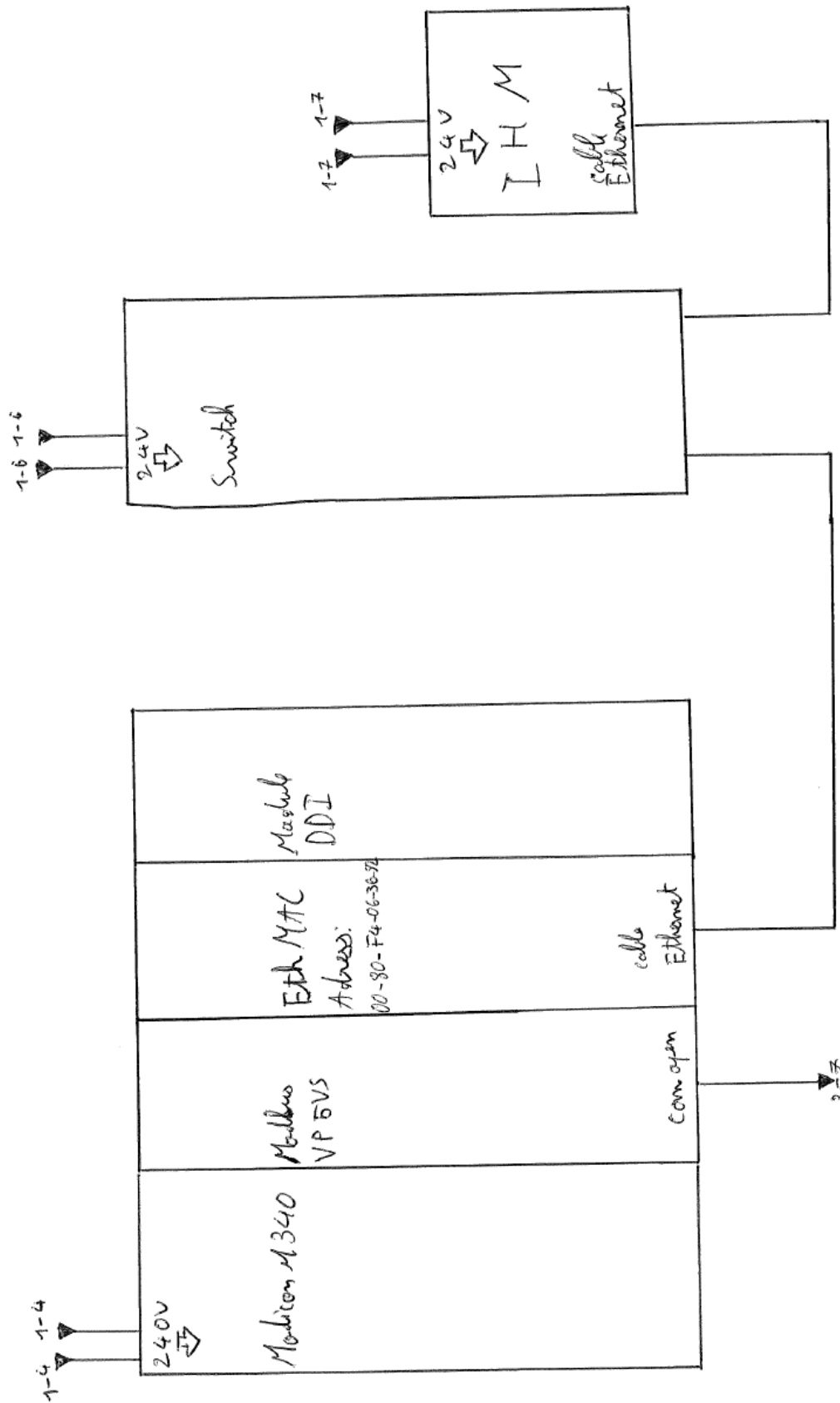
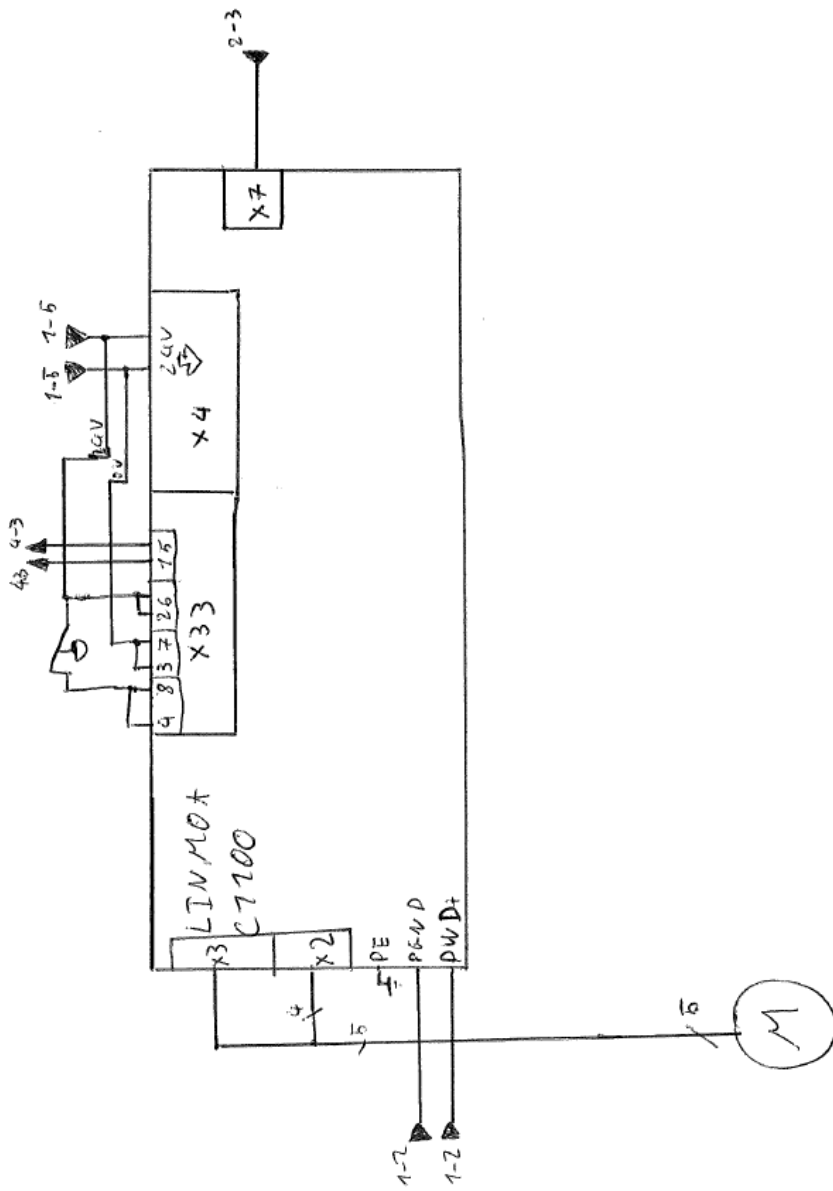


Diagram
(on other
sheet VAR
(X7))

F2/4

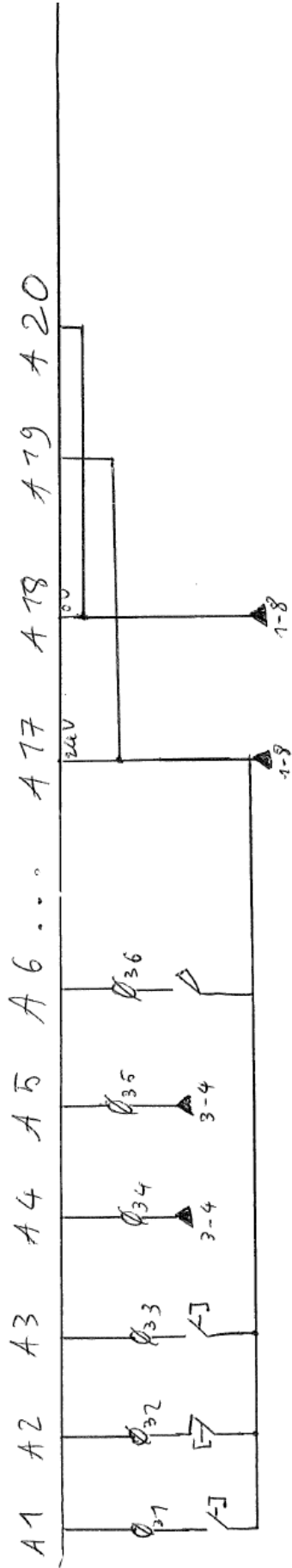
1	2	3	4	5	6	7	8	9	10
---	---	---	---	---	---	---	---	---	----



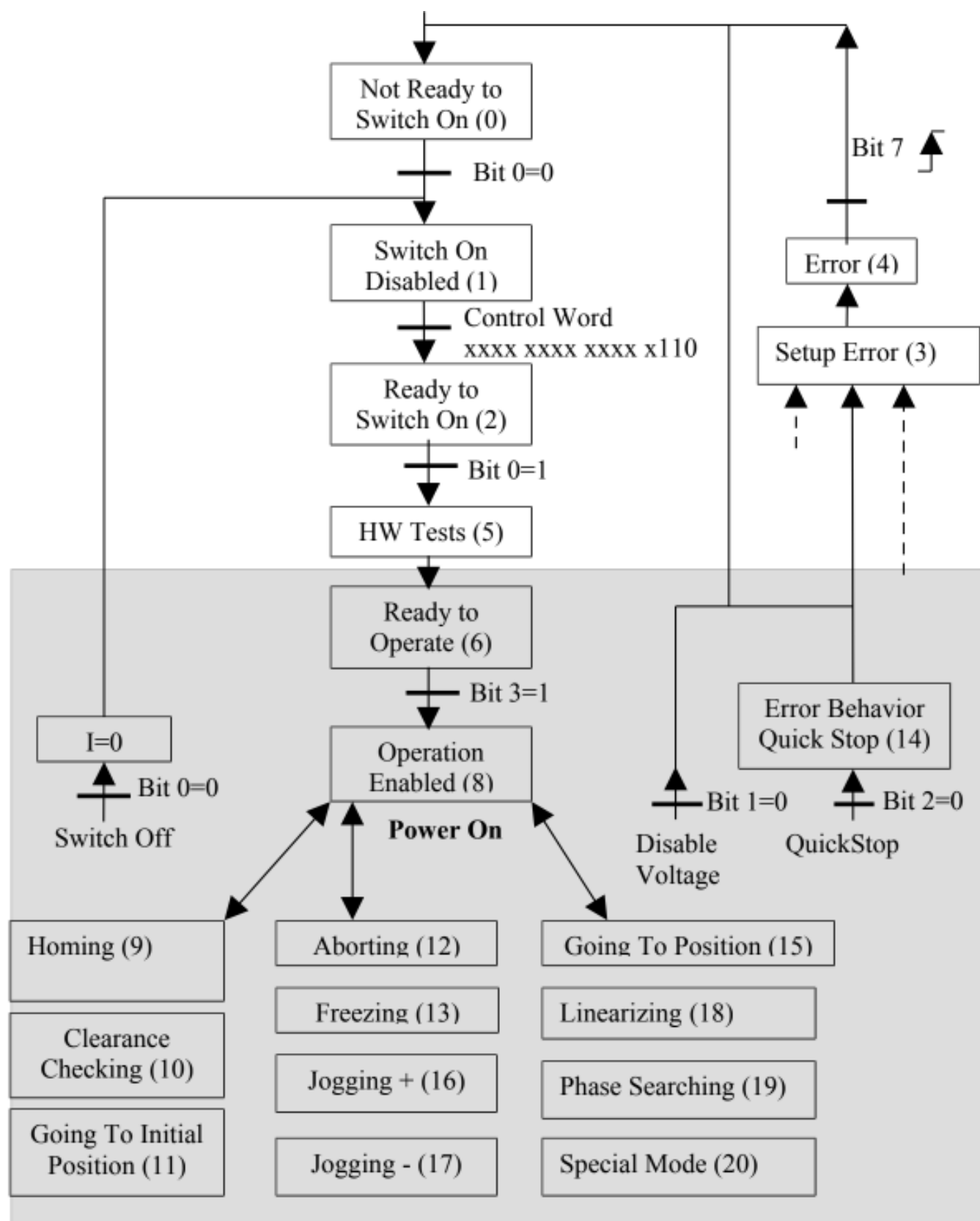
F3/4

1	2	3	4	5	6	7	8	9	10
---	---	---	---	---	---	---	---	---	----

Module entrée automate

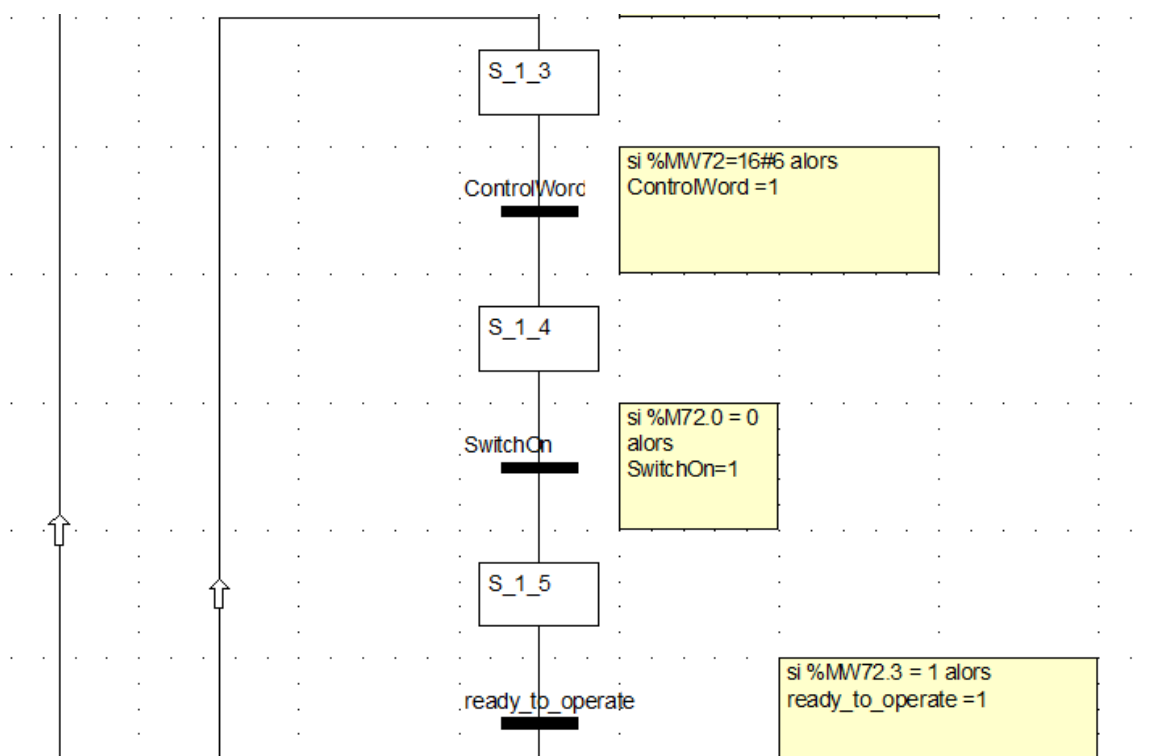
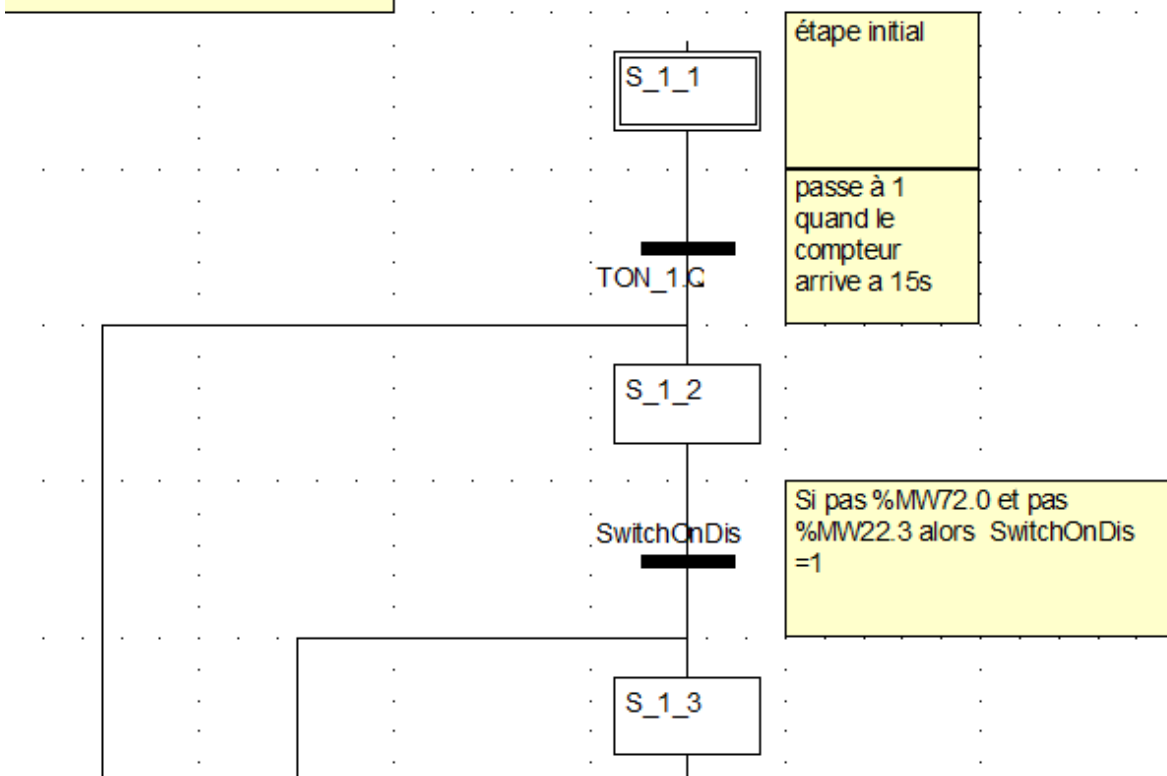


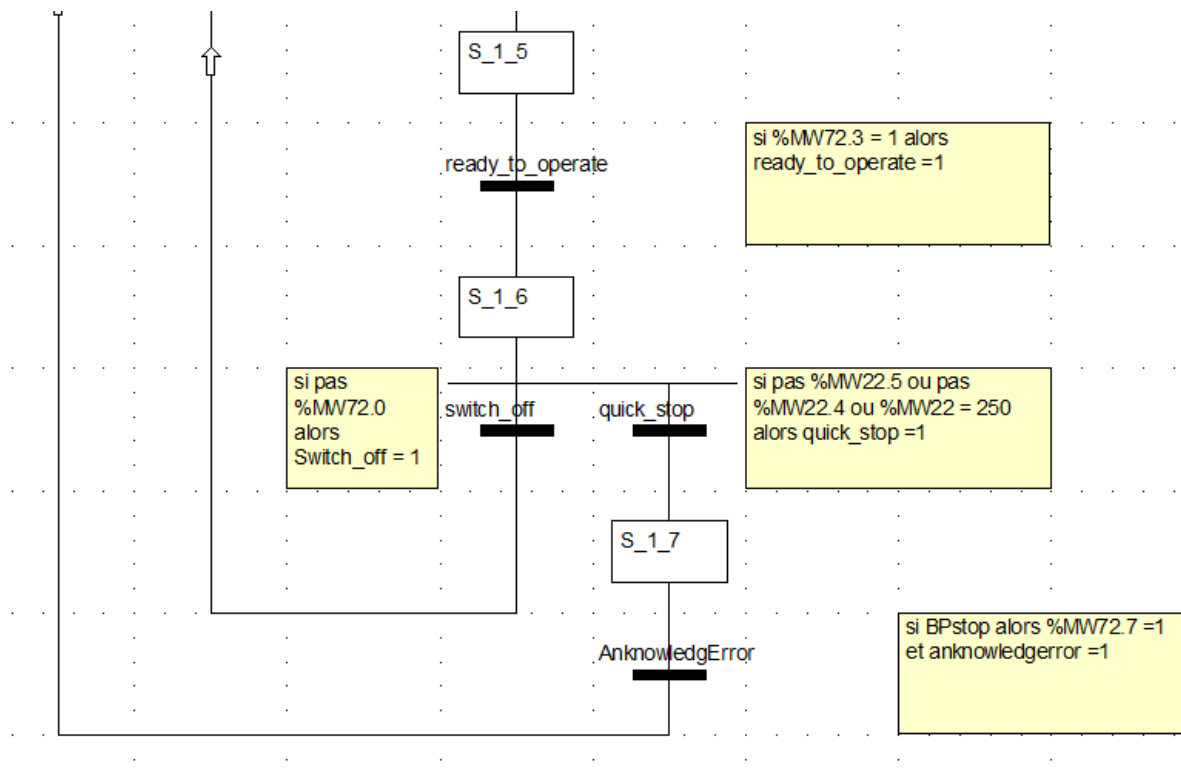
F4/4									
1	2	3	4	5	6	7	8	9	10



ANNEXE 3

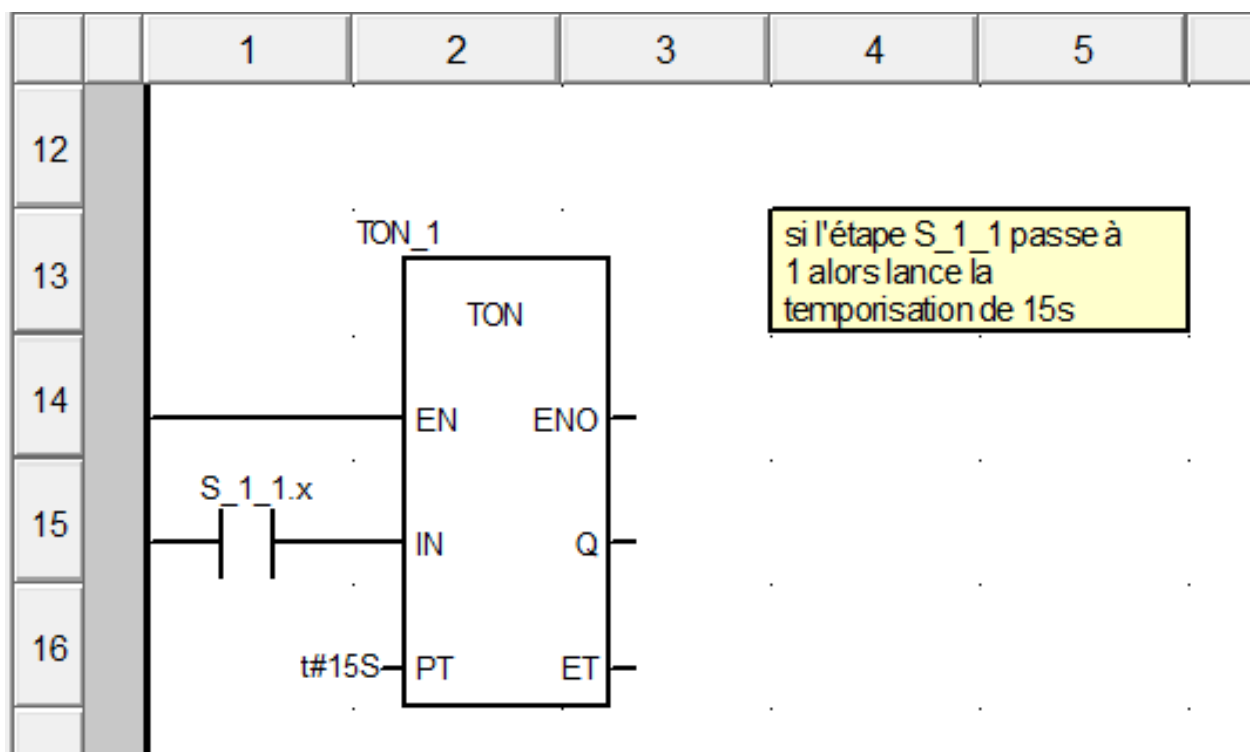
grafcet d'initialisation



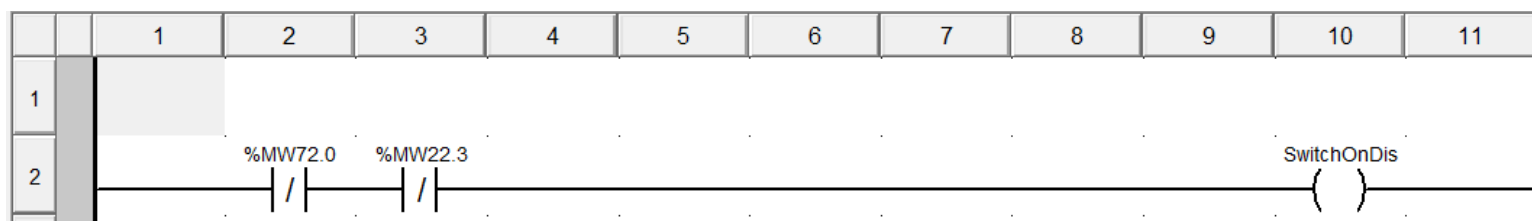


ANNEXE 4

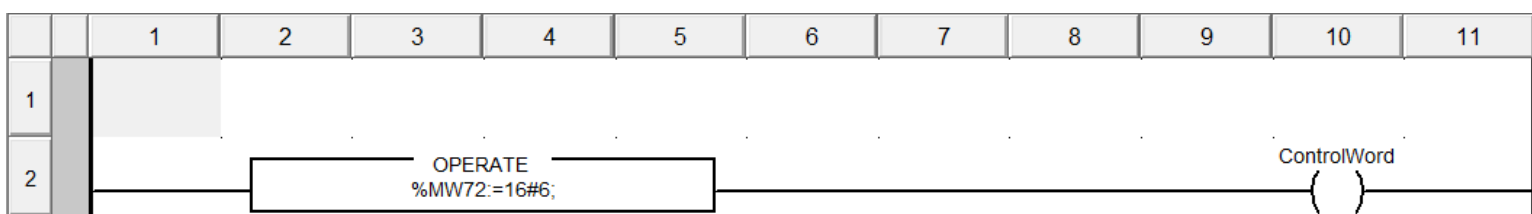
transition entre S_1_1 et S_1_2



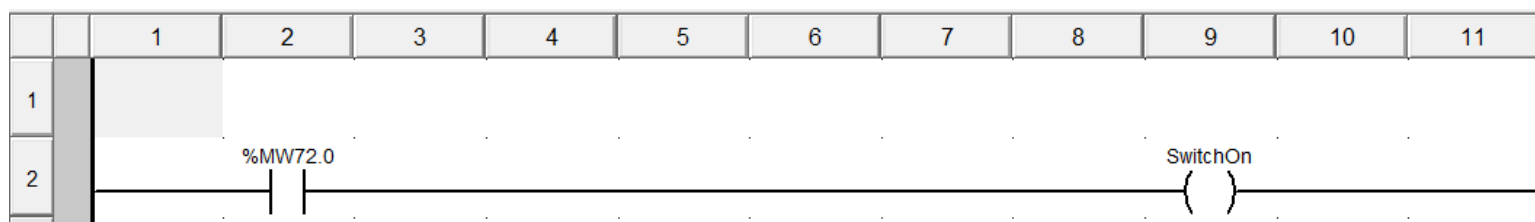
Transition entre S_1_2 et S_1_3



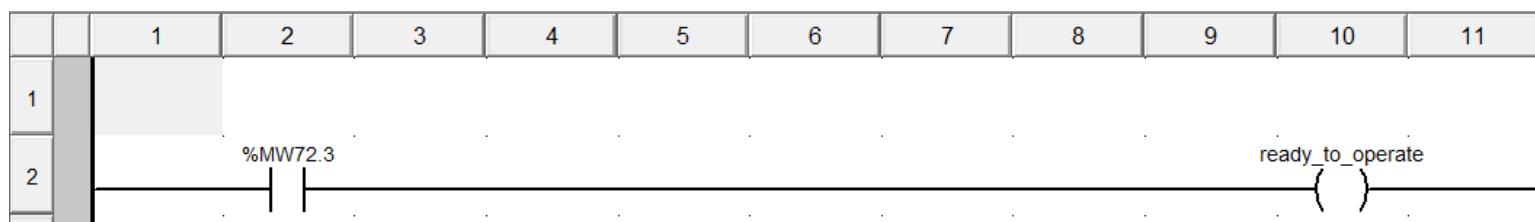
Transition entre S_1_3 et S_1_4



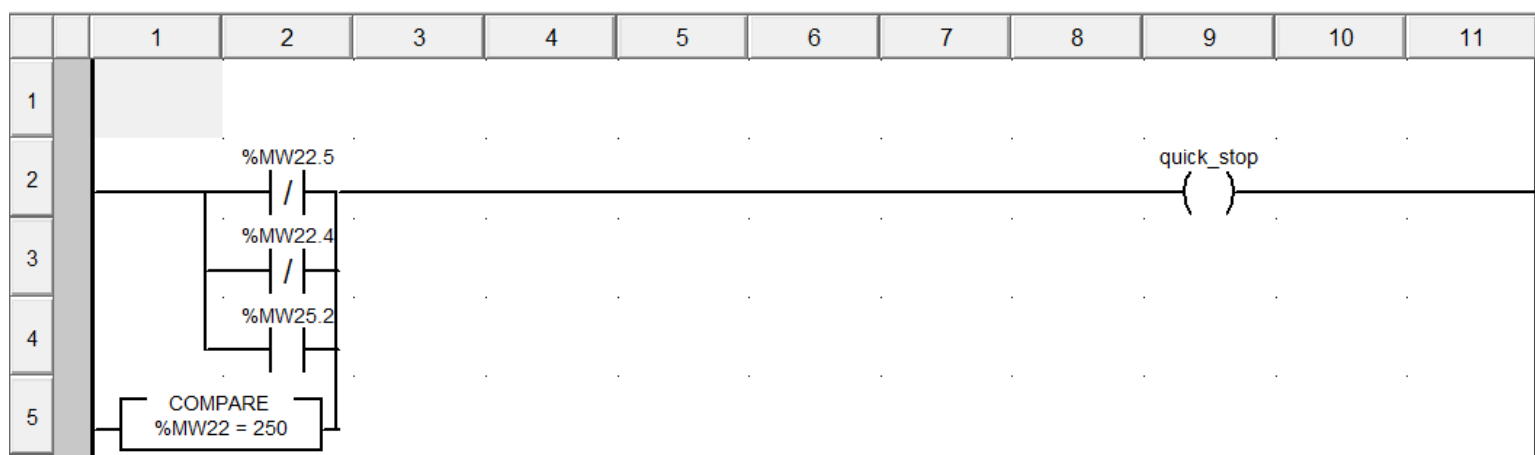
Transition entre S_1_4 et S_1_5



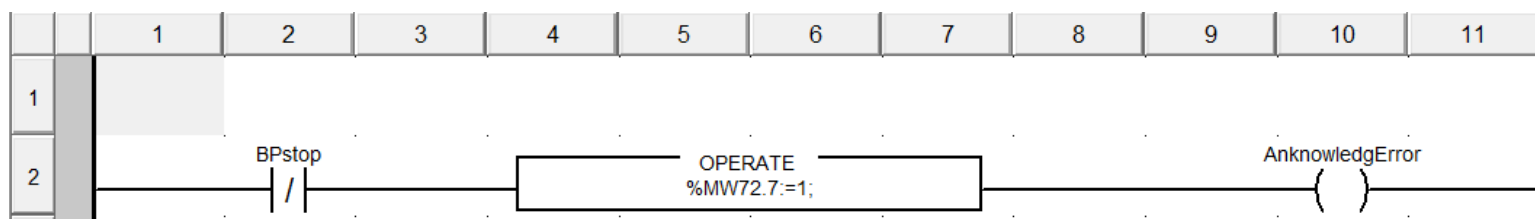
Transition entre S_1_5 et S_1_6



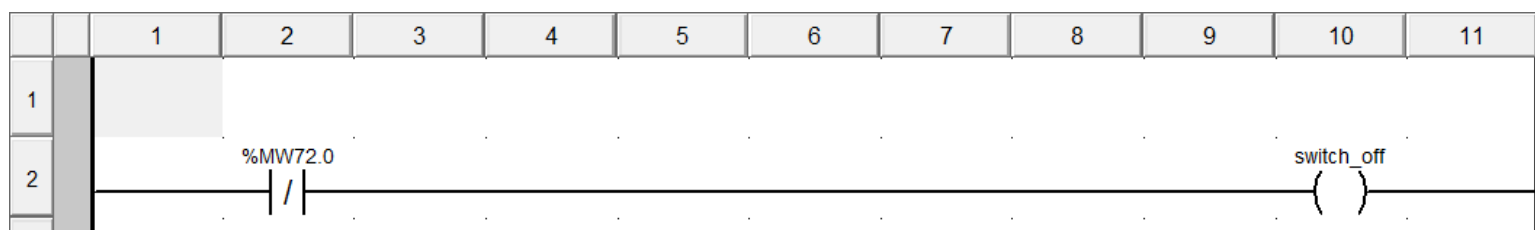
Transition entre S_1_6 et S_1_7



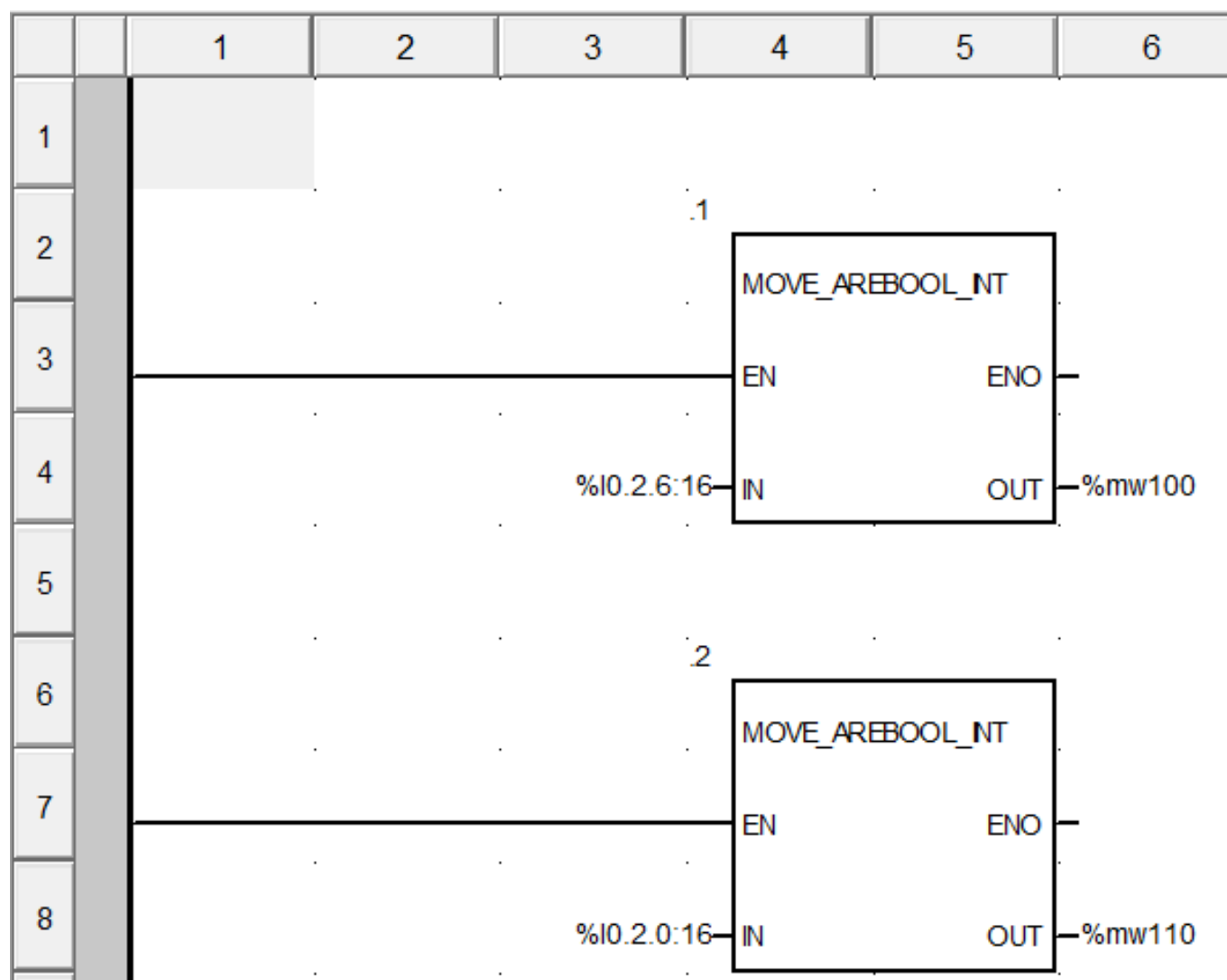
Transition entre S_1_7 et S_1_2



Transition entre S_1_7 et S_1_3



ANNEXE 5



ANNEXE 6

