



الجمهورية الجزائرية الديمقراطية الشعبية

République Algérienne Démocratique et Populaire

وزارة التعليم العالي والبحث العلمي

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

جامعة وهران 2 محمد بن أحمد Université d'Oran II Mohamed Ben Ahmed

معهد الصيانة والأمن الصناعي Institut de Maintenance et de sécurité industrielle

Département de Maintenance en Electromécanique



MÉMOIRE

de fin d'études pour l'obtention du diplôme de Master

Filière : Génie Industriel

Spécialité : Maintenance-Fiabilité-Qualité

Thème :

Etude et Simulation d'une Remplisseuse Automatique de Liquides

Présenté par :

MAMERI Mohamed Larbi
LITIM Mohammed Abdenmour

Devant le jury composé de :

Nom et Prénom	Grade	Etablissement	Qualité
BELARIFI Farid	Pr	IMSI (Université d'Oran 2)	Président
REGUIEG YSSAAD Sadek	MAA	IMSI (Université d'Oran 2)	Encadreur
BELKHODJA Leila	MCB	IMSI (Université d'Oran 2)	Examinatrice

Année universitaire: 2024/2025

Remerciements

Tout d'abord, nous tenons à remercier DIEU clément et miséricordieux de nous avoir donné la force et le courage de mener à bien ce travail.

On tient à remercier très chaleureusement notre promoteur Monsieur Sadek Reguieg Yssaad enseignant à l'IMSI

Université Oran2 d'avoir accepté d'être notre encadreur dans ce travail de fin d'études. Ses conseils et son soutien ont été pour nous un vrai support scientifique et personnel. Qu'il trouve ici l'expression de notre respect et de notre profonde reconnaissance.

Nous adressons nos remerciements aux membres de jury d'avoir accepté de nous faire l'honneur de juger ce travail.

Nous sincères remerciements pour tous les enseignants du département de maintenance en électromécanique.

Enfin, nous tenons à remercier toutes les personnes qui ont contribué de près ou de loin au bon déroulement de nos travaux et études.

MAMERI Mohamed Larbi

LITIM Mohammed Abdennour

Résumé. Les lignes de remplissage automatique de liquides sont essentielles dans les secteurs alimentaires, cosmétiques et des détergents. Ce travail porte sur la modélisation et la simulation d'une ligne de remplissage industrielle, intégrant divers équipements opérationnels tels que capteurs, pré-actionneurs, actionneurs, effecteurs, terminaux IHM et dispositifs de signalisation. La simulation permet d'analyser et d'optimiser le fonctionnement du système avant toute implémentation physique, en assurant un contrôle précis des opérations de remplissage, de convoyage, de capsulage et d'étiquetage.

Mots clés : Remplissage automatique, Simulation industrielle, Capteur, Actionneur, IHM, Signalisation.

ملخص. تُعد خطوط التعبئة الأوتوماتيكية للسوائل ضرورية في الصناعات الغذائية ومستحضرات التجميل والمنظفات. يركز هذا العمل على نمذجة ومحاكاة خط تعبئة صناعي، متضمناً أجهزة تشغيلية متنوعة مثل الحساسات، والمشغلات الأولية، والمشغلات، والمؤثرات الطرفية، وأجهزة الإشارة والتنبيه. تسمح المحاكاة بتحليل وتحسين أداء النظام قبل التنفيذ الفعلي، مع ضمان التحكم الدقيق في عمليات التعبئة والنقل والتغطية ووضع الملصقات.

كلمات مفتاحية : تعبئة أوتوماتيكية، محاكاة صناعية، حساس، مشغل، مؤثر طرفي، إشارة وتنبيه.

Abstract. Automatic liquid filling lines are essential in the food, cosmetic, and detergent industries. This work focuses on modeling and simulating an industrial filling line, integrating various operational devices such as sensors, pre-actuators, actuators, effectors, HMI terminals, and signaling devices. Simulation enables analysis and optimization of the system's performance prior to physical implementation, ensuring precise control of filling, conveying, capping, and labeling operations.

Keywords: Automatic Filling, Industrial Simulation, Sensor, Actuator, HMI, Signaling.

Table des matières

Introduction générale.....	1
Chapitre I : Machines de Remplissage Automatique de Liquides	2
1. Introduction.....	3
2. Système de remplissage	3
2.1. Remplissage à niveau.....	3
2.2. Remplissage débit-métrique	4
2.3. Remplissage pondéral	4
2.4. Remplissage volumétrique	5
3. Les machines de remplissage	5
3.1. Définition	5
3.2. Les types de machine de remplissage	6
3.3. Les avantages et désavantages de chaque type de remplisseuse	7
3.4. Le choix des machines de remplissage.....	8
3.5 Domaine d'utilisation.....	9
4 Conclusion	9
Chapitre II : Partie opérative d'une ligne de remplissage automatique	10
1. Introduction	11
2. Les Capteurs	11
2.1. Définition	11
2.2. Type de détection.....	11
2.3. Type des capteurs	11
2.4. Branchement d'un capteur	21
3. Les Pré-Actionneurs	22
3.1. Les distributeurs pneumatiques	22
3.2. Les pré-actionneurs électriques	24
3.3. Les variateurs de vitesse (VFDs).....	28
4. Les actionneurs.....	30
4.1. Les vérins pneumatiques	30
4.2. Les moteurs électriques.....	32
4.3. Les réducteurs.....	41
5. Les effecteurs.....	42
5.1. Définition.....	42
5.2. Les types d'effecteurs.....	42
6. Les terminaux IHM.....	44
7. Dispositifs de signalisation.....	44

8. Conclusion	45
Chapitre III : Partie Commande d'une ligne de remplissage automatique	46
1. Introduction.....	47
2. Automatisation	47
3. Objectifs de l'automatisation.....	47
4. La structure d'un system automatisé	47
4.1. Partie opératif	48
4.2. Partie commande	48
4.3. Partie IHM.....	48
5. API	48
5.1. Structure d'un API.....	48
5.2. Cycle de balayage	50
6. Langage de programmation selon la norme IEC 61131-3.....	52
6.1. Langage IL	53
6.2. Langage ST	53
6.3. Langage GRAFCET	53
6.4. Langage FBD	54
6.5. Langage LD	54
7. Grafcet dans les systèmes d'automatisation industrielle	55
7.1. Utilisation du GRAFCET dans l'automatisation	55
7.2. Eléments Fondamentaux d'un diagramme GRAFCET	56
7.3. Références industrielle	56
8. Conclusion	57
Chapitre IV : Réalisation et Programmation du Système.....	58
1. Introduction.....	59
2. Logiciel CODESYS	59
3. Simulation et programmation d'une machine de remplissage automatique	59
3.1 Les moteurs utilisés.....	59
3.2 Création du projet.....	61
3.3 Types des variables utilisés	64
3.4 Fonctionnement normal du programme	65
3.4.1 Start	65
3.4.2 TankCheck	65
3.4.3 BottleReady	69
3.4.4 BottleFillingReady	72
3.4.5 BottleFilling	75
3.4.6 CappingReady	78

3.4.7 JawReady	81
3.4.8 ChuckRotation	84
3.4.9 Labeling	87
3.4.10 Quality control	92
3.5 Fonctionnement des erreurs	95
3.5.1 Les erreurs d'étape TankCheck.....	95
3.5.2 Les erreurs d'étape BottleReady	98
3.5.3 Les erreurs d'étape BottleFillingReady	102
3.5.4 Les erreurs d'étape BottleFilling.....	103
3.5.5 Les erreurs d'étape CappingReady.....	108
3.5.6 Les erreurs d'étape JawReady.....	113
3.5.7 Les erreurs d'étape ChuckRotation	115
3.5.8 Les erreurs d'étape Labeling.....	118
4 Conclusion	124
Conclusion générale.....	125
Bibliographie	

Liste des figures

Figure I.1 Processus de remplissage sur une ligne automatisée.....	3
Figure I.2 Principe de remplissage a niveau	3
Figure I.3 Principe du remplissage débit-métrique avec débitmètre.....	4
Figure I.4 Principe du remplissage pondéral avec balance	4
Figure I.5 Principe du remplissage volumétrique par pompe	5
Figure I.6 Machine de remplissage Automatique.....	6
Figure I.7 Machine de remplissage semi-automatique.....	6
Figure I.8 Machine de remplissage manuel	7
Figure II.1 Principe de fonctionnement d'un capteur	11
Figure II.2 Principe de fonctionnement d'un capteur passif	11
Figure II.3 Montage électronique d'un capteur en le cas de changement de résistivité.....	12
Figure II.4 Principe de fonctionnement d'un capteur actif	13
Figure II.5 Principe d'un effet thermoélectrique	13
Figure II.6 Un capteur In-Sight SnAPP qui classe les roulements à billes sur une ligne de production comme OK ou NG.	14
Figure II.7 Les capteurs TOR.....	15
Figure II.8 Principe de fonctionnement d'un capteur TOR.....	16
Figure II.9 Signal d'un capteur analogique de température	17
Figure II.10 Manomètre analogique.....	17
Figure II.11 Principe de conversion d'une force mécanique en signal électrique dans un capteur de force à jauges de contrainte	18
Figure II.12 Schéma d'intégration d'un capteur de couple rotatif entre moteur et système d'actionnement .	19
Figure II.13 Thermomètre digital.....	19
Figure II.14 Signal logique codé.....	19
Figure II.15 Capteur de proximité.....	20
Figure II.16 Principe de la détection pour un capteur photoélectrique	20
Figure II.17 Branchement d'un capteur PNP avec trois fils.....	21
Figure II.18 Branchement d'un capteur NPN avec trois fils.....	21
Figure II.19 Principe du fonctionnement des pré-actionneurs.....	22
Figure II.20 Principe de fonctionnement d'un distributeur a clapets.....	22
Figure II.21 Principe de fonctionnement d'un distributeur à tiroirs.....	23
Figure II.22 Distributeur avec 2 orifices et 2 positions	23
Figure II.23 Distributeur monostable	23
Figure II.24 Distributeur bistable	23
Figure II.25 Relais électromagnétique.....	24
Figure II.26 Principe de fonctionnement d'un relais magnétique.....	25
Figure II.27 Un relais statique	25
Figure II.28 Les composants d'un contacteur.....	27
Figure II.29 Schéma de puissance d'un contacteur	27
Figure II.30 Variateur de vitesse	28
Figure II.31 circuit d'un redresseur.....	28
Figure II.32 Circuit intermédiaire d'un VFD	29
Figure II.33 Circuit de l'onduleur (transistors)	29
Figure II.34 Principe de fonctionnement d'un vérin pneumatique.....	30
Figure II.35 Principe de fonctionnement et la symbolisation d'un VSE avec tige rentrée au repos.....	30
Figure II.36 Principe de fonctionnement et la symbolisation d'un VSE avec tige sortie au repos	30
Figure II.37 Principe de fonctionnement et la symbolisation d'un VDE.....	31
Figure II.38 Constituants d'un vérin VDE amorti	31
Figure II.39 Vérin oscillant	31
Figure II.40 Symbolisation d'un vérin oscillant	31
Figure II.41 La règle de la main droite	32
Figure II.42 Principe de fonctionnement d'un moteur monophasé.....	33

Figure II.43 Principe de fonctionnement d'un BLDC	33
Figure II.44 Principe du fonctionnement d'un moteur pas à pas (excitation de la phase A).....	35
Figure II.45 Principe du fonctionnement d'un moteur pas à pas (excitation de phase B)	35
Figure II.46 Les composants d'un moteur pas à pas (5 phases).....	36
Figure II.47 Onde d'un system triphasé.....	38
Figure II.48 La force électromagnétique.....	38
Figure II.49 Les composants d'un moteur triphasé asynchrone.....	39
Figure II.50 Symbole d'un MAT	40
Figure II.51 Triangle-Δ	40
Figure II.52 Étoiles-Y	40
Figure II.53 Schéma de la commande d'un moteur triphasé.....	40
Figure II.54 Un réducteur	41
Figure II.55 Les buses utilisées dans les machines de remplissage	42
Figure II.56 Les brides d'un système de remplissage des 4 bouteilles	42
Figure II.57 Distributeur de bouchons.....	43
Figure II.58 Assemblage de mandrin à trois mâchoires avec Assemblage de mandrin à trois mâchoires avec limiteur de couple magnétique.....	43
Figure II.59 Panneau d'IHM industrielle.....	44
Figure II.60 Des alarmes dans un site industriel.....	45
Figure III.1 Structure d'un système automatisés	47
Figure III.2 Structure interne d'un API	48
Figure III.3 Module de puissance d'un API	49
Figure III.4 Unité centrale de traitement d'un API	49
Figure III.5 E/S interfaces	50
Figure III.6 Le cycle de balayage	52
Figure III.7 Exemple d'une programmation avec GRAFCET	54
Figure III.8 Exemple d'une programmation avec FBD.....	54
Figure III.9 Exemple d'une programmation avec LD	55
Figure III.10 Les élément principe d'une GRAFCET	56
Figure IV.1 Logo du logiciel CODESYS	59
Figure IV.2 Création d'un nouveau projet	61
Figure IV.3 Nommer le projet	62
Figure IV.4 Choisir le langage de programmation	62
Figure IV.5 La fenêtre principale de notre projet.	63
Figure IV.6 Table des variables.....	64
Figure IV.7 Bloc d'étape TankCheck	65
Figure IV.8 Initialisation et Flags TankCheck.....	66
Figure IV.9 Commande Manuelle et Vitesse Moteur	66
Figure IV.10 Défaut Capteurs et Surcharge Moteur	67
Figure IV.11 Niveau Bas et Remplissage Supplémentaire	67
Figure IV.12 Niveau Haut et Arrêt	68
Figure IV.13 Simulation Courant Moteur et Mapping IHM.....	68
Figure IV.14 Transition entre TankCheck et BottleReady	68
Figure IV.15 Bloc d'étape BottleReady.....	69
Figure IV.16 Initialisation et indicateurs de alimentation.....	69
Figure IV.17 Commande manuelle de la vanne et du convoyeur.	69
Figure IV.18 Filtrage du capteur de bouteille et gestion de la vanne.....	70
Figure IV.19 Forçage manuel du moteur du convoyeur	70
Figure IV.20 Détection des défauts et gestion de la surcharge du convoyeur.	71
Figure IV.21 Simulation du courant du convoyeur et transfert des états vers l'IHM	71
Figure IV.22 Transition entre BottleReady et BottleFillingReady	71
Figure IV.23 Bloc d'étape FillingBottleReady	72
Figure IV.24 Initialisation et indicateurs du remplissage de bouteilles	72
Figure IV.25 Commande manuelle et état moteur du convoyeur	72
Figure IV.26 Commande de la vitesse du convoyeur	73
Figure IV.27 Détection du capteur de position et gestion des électrovannes (SV).....	73
Figure IV.28 Minuteur d'attente et gestion des défauts de position	74

Figure IV.29 Simulation du courant moteur et transfert des états vers l'IHM.....	74
Figure IV.30 Transition entre FillingBottleReady et FillingBottle.....	74
Figure IV.31 Bloc d'étape FillingBottle	75
Figure IV.32 Initialisation et indicateurs du remplissage	75
Figure IV.33 Commande manuelle des actionneurs et du moteur	75
Figure IV.34 Commande de la vitesse du moteur pompe.....	76
Figure IV.35 Filtrage des capteurs et calcul du courant moteur	76
Figure IV.36 Mapping IHM (états actionneurs, moteur, défauts).....	76
Figure IV.37 Détection de surcharge du moteur pompe.....	77
Figure IV.38 Gestion du temps et des défauts de capteurs	77
Figure IV.39 Fin normale du remplissage	78
Figure IV.40 Transition entre FillingBottle et CappingBottleReady.....	78
Figure IV.41 Bloc d'étape CappingBottleReady.	78
Figure IV.42 Initialisation et indicateurs d'état.....	78
Figure IV.43 Commande manuelle et état moteur du convoyeur.....	79
Figure IV.44 Commande manuelle de la vitesse du convoyeur et calcul du courant	79
Figure IV.45 Détection de surcharge moteur du convoyeur et des capsules	79
Figure IV.46 Anti Rebond et Temporisation Capteur Bouteille et Capsule	80
Figure IV.47 Activation des électrovannes de capsulage	80
Figure IV.48 Détection des défauts bouteille et capsule.....	81
Figure IV.49 Affichage des états moteurs sur l'IHM	81
Figure IV.50 Transition entre CappingBottleReady et JawReady.....	81
Figure IV.51 Bloc d'étape JawReady	81
Figure IV.52 Initialisation et indicateurs d'état.....	82
Figure IV.53 Commande manuelle des électrovannes (SV).....	82
Figure IV.54 Filtrage du capteur ChuckDown et Minuteur d'attente	82
Figure IV.55 Atteinte de la position basse du Chuck et activation de la mâchoire.....	83
Figure IV.56 Détection et gestion du défaut ChuckDown.....	83
Figure IV.57 Mappings IHM (états électrovannes et défauts).....	83
Figure IV.58 Transition entre CappingBottleReady et JawReady.....	83
Figure IV.59 Bloc d'étape ChuckRotation	84
Figure IV.60 Initialisation et gestion des modes manuels	84
Figure IV.61 Détection de la prise du bouchon par la mâchoire	84
Figure IV.62 Rotation du moteur pas à pas	85
Figure IV.63 Surveillance du couple et gestion des défauts	85
Figure IV.64 Arrêt le moteur et actionneurs.....	86
Figure IV.65 Gestion du défaut de prise mâchoire	86
Figure IV.66 Mappage IHM et arrêt du convoyeur en cas de défaut.....	86
Figure IV.67 Transition entre ChuckRotation et Labeling	86
Figure IV.68 Bloc d'étape Labeling	87
Figure IV.69 Initialisation et indicateurs d'état.....	87
Figure IV.70 Gestion des modes manuels	88
Figure IV.71 Calcul du courant absorbé.....	88
Figure IV.72 Vérification des surcharges	89
Figure IV.73 Filtrage des capteurs.....	89
Figure IV.74 Temporisation et défaut d'arrivée bouteille	90
Figure IV.75 Temporisation et défaut tip	90
Figure IV.76 Temporisation et défaut gap.....	91
Figure IV.77 Action du couteau et coupe d'étiquette	91
Figure IV.78 Transition entre Labeling et QC.....	91
Figure IV.79 Bloc d'étape QC.....	92
Figure IV.80 Initialisation et indicateurs d'état QC.....	92
Figure IV.81 Commande manuelle et automatique du convoyeur.....	92
Figure IV.82 Calcul du courant moteur et détection de surcharge	93
Figure IV.83 Commande manuelle/automatique de l'électrovanne de rejet.....	93
Figure IV.84 Inspection et détection des défauts QC	93
Figure IV.85 Décision d'acceptation ou de rejet QC.....	94

Figure IV.86 Mapping des états et défauts vers l'IHM.....	94
Figure IV.87 Transition entre QC et saut.....	94
Figure IV.88 Symbol du saut à l'étape Initial Start	94
Figure IV.89 Transition entre l'étape TankCheck et l'étape d'erreur TankFault	95
Figure IV.90 Affichage de l'erreur sur IHM et activation des alarmes	95
Figure IV.91 Forçage manuel du moteur de pompe 1	95
Figure IV.92 Enregistrement et Réinitialisation du défaut par l'opérateur	95
Figure IV.93 Récupération automatique si le capteur détecte la cuve	96
Figure IV.94 Forçage manuel de l'alarme lumineuse et de l'avertisseur sonore	96
Figure IV.95 Les transitions et sautes d'étape TankFault.....	96
Figure IV.96 Transition entre TankCheck et MidSensorError	96
Figure IV.97 Affichage de l'erreur sur IHM et activation des alarmes	96
Figure IV.98 Enregistrement du default et Réinitialisation du défaut par l'opérateur	97
Figure IV.99 Rétablissement automatique si le capteur revient à la normale.....	97
Figure IV.100 Les transitions et sautes d'étape MidSensorError	97
Figure IV.101 Transition entre TankCheck et MotorPump1Overload	97
Figure IV.102 Affichage de l'erreur sur IHM et activation des alarmes	97
Figure IV.103 Enregistrement du default et Réinitialisation du défaut par l'opérateur	98
Figure IV.104 Rétablissement automatique si le courant redevient normal	98
Figure IV.105 Les transitions et sautes d'étape MotorPump1Overload	98
Figure IV.106 Transition entre BottleReady et BottleFault.....	98
Figure IV.107 Affichage de l'erreur sur IHM et activation des alarmes	99
Figure IV.108 Forçage manuel du vérin et du moteur convoyeur	99
Figure IV.109 Enregistrement du default et Réinitialisation du défaut par l'opérateur	99
Figure IV.110 Récupération automatique si la bouteille est détectée	100
Figure IV.111 Les transitions et sautes d'étape BottleReady	100
Figure IV.112 Transition entre BottleReady et ConveyorMotorFaultBottleReady	100
Figure IV.113 Affichage de l'erreur sur IHM et activation des alarmes	100
Figure IV.114 Forçage manuel du moteur convoyeur	101
Figure IV.115 Enregistrement du default et Réinitialisation du défaut par l'opérateur	101
Figure IV.116 Rétablissement automatique si le courant du moteur redevient normal	101
Figure IV.117 Les transitions et sautes d'étape ConveyorMotorFaultBottleReady	101
Figure IV.118 Transition entre BottleFillingReady et BottleFillingPositionFault	102
Figure IV.119 Affichage de l'erreur sur IHM et activation des alarmes	102
Figure IV.120 Enregistrement du default et Réinitialisation du défaut par l'opérateur	102
Figure IV.121 Récupération automatique avec temporisation du capteur	102
Figure IV.122 Les transitions et sautes d'étape BottleFillingPositionFault.....	103
Figure IV.123 Les transitions et sautes d'étape ConveyorMotorFaultBottleFillingReady	103
Figure IV.124 Transition entre BottleFilling et FillingTimeoutFault	103
Figure IV.125 Affichage de l'erreur sur IHM et activation des alarmes	103
Figure IV.126 Forçage manuel et mise en sécurité des actionneurs et synchronisation d'IHM	104
Figure IV.127 Enregistrement du default et Réinitialisation du défaut par l'opérateur	104
Figure IV.128 Récupération automatique si la bouteille est détectée pleine	104
Figure IV.129 Les transitions et sautes d'étape FillingTimeoutFault	105
Figure IV.130 Transition entre BottleFilling et NozzlePressureFault	105
Figure IV.131 Affichage de l'erreur sur IHM et activation des alarmes	105
Figure IV.132 Enregistrement du default et Réinitialisation du défaut par l'opérateur	105
Figure IV.133 Récupération automatique si la pression de buse revient	106
Figure IV.134 Les transition et sautes d'étape NozzlePressureFault.....	106
Figure IV.135 Transition entre BottleFilling et FillingSensorFault.....	106
Figure IV.136 Affichage de l'erreur sur IHM et activation des alarmes	106
Figure IV.137 Enregistrement du default et Réinitialisation du défaut par l'opérateur	106
Figure IV.138 Récupération automatique si le capteur bouteille pleine est déclenché.....	107
Figure IV.139 Les transition et sautes d'étape FillingSensorFault	107
Figure IV.140 Transition entre BottleFilling et FillingMotorOverload.....	107
Figure IV.141 Affichage de l'erreur sur IHM et activation des alarmes	107
Figure IV.142 Enregistrement du default et Réinitialisation du défaut par l'opérateur	108

Figure IV.143	Récupération automatique si la surcharge moteur pompe 2 est levée	108
Figure IV.144	Les transitions et sautes d'étape FillingMotorOverload.....	108
Figure IV.145	Transition entre BottleCappingReady et CappingBottlePositionFault.....	108
Figure IV.146	Affichage de l'erreur sur IHM et activation des alarmes	108
Figure IV.147	Forçage manuel du moteur du convoyeur et les vérins	109
Figure IV.148	Enregistrement du default et Réinitialisation du défaut par l'opérateur	109
Figure IV.149	Récupération automatique si la bouteille est détectée	109
Figure IV.150	Les transitions et sautes d'étape CappingBottlePositionFault.....	110
Figure IV.151	Les transitions et sautes d'étape ConveyorMotorFaultBottleCappingReady	110
Figure IV.152	Transition entre BottleCappingReady et CapFeederSensorFault	110
Figure IV.153	Affichage de l'erreur sur IHM et activation des alarmes	110
Figure IV.154	Forçage manuel des actionneurs et moteur	111
Figure IV.155	Enregistrement du default et Réinitialisation du défaut par l'opérateur	111
Figure IV.156	Récupération automatique si le courant moteur redevient normal	111
Figure IV.157	Les transition et sautes d'étape CapFeederMotorFault	112
Figure IV.158	Transition entre BottleCappingReady et CapFeederSensorFault	112
Figure IV.159	Affichage de l'erreur sur IHM et activation des alarmes	112
Figure IV.160	Enregistrement du default et Réinitialisation du défaut par l'opérateur	112
Figure IV.161	Récupération automatique si le bouchon est détecté	113
Figure IV.162	Les transitions et sautes d'étape CapFeederSensorFault	113
Figure IV.163	Transition entre JawReady et ChuckDownFault	113
Figure IV.164	Affichage de l'erreur sur IHM et activation des alarmes	113
Figure IV.165	Forçage manuel et mise en sécurité des vérins de capsulage	114
Figure IV.166	Enregistrement du default et Réinitialisation du défaut par l'opérateur	114
Figure IV.167	Récupération automatique si le capteur position basse est déclenché	115
Figure IV.168	Les transitions et sautes d'étape ChuckDownFault.....	115
Figure IV.169	Transition entre ChuckRotation et JawClutchFault	115
Figure IV.170	Affichage de l'erreur sur IHM et activation des alarmes	115
Figure IV.171	Forçage manuel et mise en sécurité des actionneurs	116
Figure IV.172	Enregistrement du default et Réinitialisation du défaut par l'opérateur	116
Figure IV.173	Récupération automatique si le capteur mâchoire est déclenché	116
Figure IV.174	Les transitions et sautes d'étape JawClutchFault	117
Figure IV.175	Transition entre ChuckRotation et TorqueFault.....	117
Figure IV.176	Affichage de l'erreur sur IHM et activation des alarmes	117
Figure IV.177	Enregistrement du default et Réinitialisation du défaut par l'opérateur	117
Figure IV.178	Récupération automatique si le couple redevient correct	117
Figure IV.179	Les transitions et sautes d'étape TorqueFault	118
Figure IV.180	Transition entre Labeling et LabelTipNotReadyFault	118
Figure IV.181	Affichage de l'erreur sur IHM et activation des alarmes	118
Figure IV.182	Enregistrement du default et Réinitialisation du défaut par l'opérateur	118
Figure IV.183	Récupération automatique si le couple redevient correct	118
Figure IV.184	Les transitions et sautes d'étape LabelTipNotReady	119
Figure IV.185	Transition entre Labeling et LabelGapNotDetected.....	119
Figure IV.186	Affichage de l'erreur sur IHM et activation des alarmes	119
Figure IV.187	Enregistrement du default et Réinitialisation du défaut par l'opérateur	119
Figure IV.188	Récupération automatique si le couple redevient correct	120
Figure IV.189	Les transitions et sautes d'étape LabelGapNotDetected	120
Figure IV.190	Transition entre Labeling et BottleArrivalFault	120
Figure IV.191	Affichage de l'erreur sur IHM et activation des alarmes	120
Figure IV.192	Enregistrement du default et Réinitialisation du défaut par l'opérateur	120
Figure IV.193	Récupération automatique si le couple redevient correct	121
Figure IV.194	Les transitions et sautes d'étape BottleArrivalFault.....	121
Figure IV.195	Les transitions et sautes d'étape ConveyorMotorFaultLabeling	121
Figure IV.196	Transition entre Labeling et SideLeftConveyorFault/ SideRightConveyorFault	121
Figure IV.197	Affichage de l'erreur sur IHM et activation des alarmes	122
Figure IV.198	Forçage manuel du moteur convoyeur latéral droit/gauche	122
Figure IV.199	Enregistrement du default et Réinitialisation du défaut par l'opérateur	122

Figure IV.200 Récupération automatique si le courant moteur redevient normal	123
Figure IV.201 Les transitions et sautes d'étape SideLeftConveyorFault/SideRightConveyorFault.....	123
Figure IV.202 Transition entre Labeling et LabelTipMotorFault.....	123
Figure IV.203 Affichage de l'erreur sur IHM et activation des alarmes	123
Figure IV.204 Forçage manuel du moteur extrémité d'étiquette	123
Figure IV.205 Enregistrement du default et Réinitialisation du défaut par l'opérateur	124
Figure IV.206 Récupération automatique si le courant moteur redevient normal	124
Figure IV.207 Les transitions et sautes d'étape LabelTipMotorFault	124

Liste des abréviations

AC – Alternating Current

API – Automate Programmable Industriel

BLDC – Brushless Direct Current Motor

CFC – Continuous Function Chart

DC – Direct Current

FBD – Function Block Diagram

GRAFCET – Graphe Fonctionnel de Commande Étape Transition

HMI – Human-Machine Interface

LD – Ladder Diagram

PLC – Programmable Logic Controller

QC – Quality Control

SFC – Sequential Function Chart

ST – Structured Text

SV – Solenoid Valve

VDE – Vérin double effet

VFDs – Variable Frequency Drives

VSE – Vérin simple effet

Introduction générale

L'industrie moderne évolue rapidement vers l'automatisation, poussée par la recherche de performance, de qualité et de flexibilité. L'automatisation ne se limite plus à la simple mécanisation : elle intègre des technologies intelligentes qui rendent les machines plus autonomes et précises, tout en réduisant l'intervention humaine, surtout pour les tâches répétitives ou risquées. Elle aide ainsi les entreprises à rester compétitives, à maîtriser les coûts et à garantir la qualité des produits. Ce progrès repose sur deux éléments essentiels : la partie opérative, qui réalise les actions physiques, et la partie commande, qui gère la logique et la prise de décision.

La partie opérative regroupe les capteurs (yeux et oreilles du système), les pré-actionneurs (qui transmettent les ordres), et les actionneurs (qui réalisent les mouvements). La partie commande, véritable "cerveau", traite les informations et pilote le système, généralement via des automates programmables. Ces deux parties fonctionnent ensemble grâce à un programme qui définit l'enchaînement des actions et la gestion des erreurs.

Notre projet porte sur un processus clé pour de nombreuses industries : le remplissage automatique de bouteilles. La précision, la rapidité et la fiabilité sont essentielles pour éviter les pertes et satisfaire les clients. Ce mémoire vise à concevoir et simuler une machine de remplissage automatisée, à l'aide du logiciel CODESYS et des langages GRAFCET et ST. L'objectif est de réaliser un prototype capable d'assurer un remplissage précis, tout en gérant efficacement les défauts et les alarmes pour garantir la sécurité et la continuité. Le choix du GRAFCET s'explique par sa clarté et sa facilité à modéliser des séquences complexes.

Ce mémoire est structuré en quatre chapitres :

1. Présentation générale du domaine du remplissage industriel et des tendances actuelles.
2. Description de la partie opérative : capteurs, Pré-actionneurs, actionneurs et fonctionnement.
3. Exploration de la partie commande et des principes de l'automatisation, avec un accent sur le langage GRAFCET.
4. Mise en œuvre pratique : programmation du cycle de remplissage et simulation sur CODESYS, avec la gestion des erreurs.

Chapitre I :
Machines de Remplissage
Automatique de Liquides

I.1. Introduction

Dans ce chapitre, nous présentons une description générale des systèmes de remplissage automatisés de liquides, en mettant l'accent sur leurs types, avantages et domaines d'utilisation.

I.2. Systèmes de remplissage [1] [2] [3] [4]

Le terme remplissage désigne l'action de conditionner un produit liquide (visqueux, mousseux, pâteux). Dans cette étude, le processus est modélisé et simulé avant toute implémentation physique. En ce qui concerne les produits liquides, ceux-ci sont transférés d'une cuve de stockage à un contenant final (une bouteille, un seau...) tout en étant dosé. Il ne faut pourtant pas se fier à cette apparente simplicité : il existe une foule de variantes qui déterminent la technique de remplissage la plus adaptée au conditionnement de sa production.

Aussi, chaque solution possède ses avantages et ses limites.

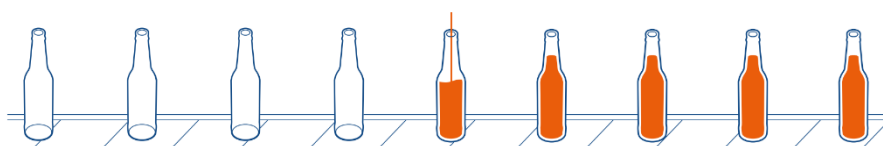


Figure I.1 Processus de remplissage sur une ligne automatisée.

Pour distinguer les différents types de remplissage, il suffit d'identifier la technique employée pour :

- Transférer le liquide
- Doser le liquide

Dès lors, 4 grandes familles sont identifiées :

Remplissage à niveau, Remplissage débit-métrique, Remplissage pondéral, et Remplissage volumétrique

I.2.1. Remplissage à niveau

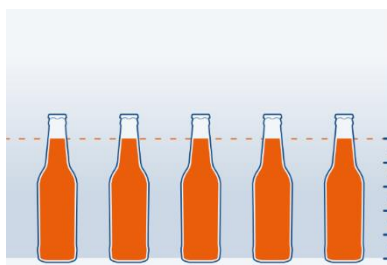


Figure I.2 Principe de remplissage à niveau.

Cette technique de dosage est déterminée par un niveau, déterminé via des capteurs: lorsque le liquide obstrue l'orifice d'échappement de l'air, l'écoulement s'arrête. La quantité dosée (en poids) peut varier à cause des tolérances dimensionnelles de la verrerie

- Précision de dosage +/- en fonction des tolérances dimensionnelles du contenant.
- Hauteur constante de remplissage.
- Dosage de volumes variés.

I.2.2. Remplissage débit-métrique

Cette technique assure un volume/poids constant du produit dosé. Le dosage s'effectue par le biais d'un débitmètre situé sur le bec de remplissage (le capteur est en contact direct avec le produit). Une vanne ferme le circuit lorsque la dose est atteinte.

- Dosage très précis (+/- 0.5% du volume dosé).
- Système hygiénique : pas de contact entre le bec et le contenant (contamination évitée).
- Dosage de volumes variés.

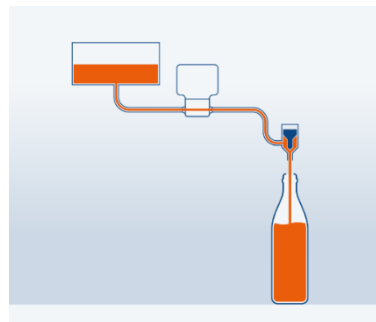


Figure I.3 Principe du remplissage débit-métrique avec débitmètre.

I.2.3. Remplissage pondéral

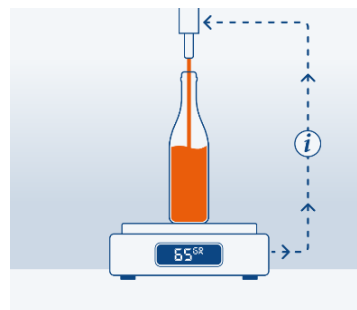


Figure I.4 Principe du remplissage pondéral avec balance.

La remplisseuse pèse le produit dosé à l'aide d'une balance. Le contenant vide est acheminé au niveau du bec de remplissage où une tare est effectuée automatiquement. Le contenant est ensuite rempli jusqu'à ce que le poids demandé soit atteint. La balance (ou cellule de pesage) permet de couper ou d'activer le flux de la pompe et même de changer son débit (petite vitesse en fin de dosage). La balance n'est pas en contact avec le produit, ce qui facilite le nettoyage de la machine.

- Technique hygiénique.
- Précision $<1\%$ du volume dosé → apprécié pour le dosage des produits coûteux.
- Technique semi-automatique : faible cadence.

I.2.4. Remplissage volumétrique

La technique de dosage volumétrique repose sur une pompe effectuant un double rôle : l'alimentation du produit et son dosage. Dans un premier temps, la pompe aspire un volume de produit (situé dans une cuve) pour l'injecter dans un second temps dans le récipient. Il existe plusieurs types de pompes (volumétrique, à lobes, à membranes, péristaltiques), mais nous ne parlerons ici que des techniques les plus souvent utilisées dans l'univers du remplissage industriel.

- Dosage des produits liquides, visqueux, pâteux et même avec morceaux.
- Large choix de pompes.
- Haute précision

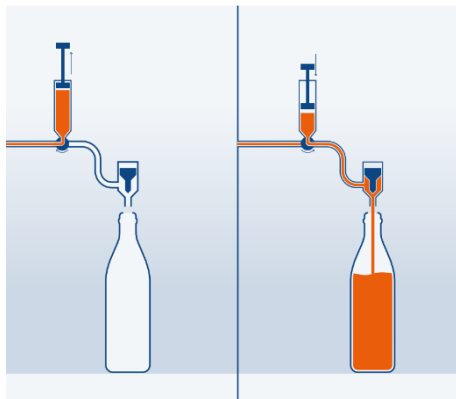


Figure I.5 Principe du remplissage volumétrique par pompe.

I.3. Machines de remplissage [1] [2] [3] [4]

I.3.1. Définition

Les machines de remplissage ou remplisseuses sont conçues pour remplir des matériaux de produits dans des conteneurs. Il peut s'agir de sacs, de pochettes, de sachets, de bouteilles en plastique, de bouteilles en verre, ...etc. cette étude se concentre sur leur modélisation virtuelle pour optimiser leur commande. [5]

L'objet de remplissage peut être de n'importe quelle forme. Comme des granulés, de la poudre ou du liquide. Aujourd'hui utilisées dans de nombreux secteurs, il existe différents types et gammes de machines dans chaque domaine.

I.3.2. Types de machines de remplissage

I.3.2.1 Machines de remplissage automatique

Les machines de remplissage automatique sont des machines qui exécutent les opérations de remplissage de façon automatique sans l'intervention humaine. Ces machines ont pour principal avantage la facilité de travail en cas de cadence élevé donc une productivité élevée.

Leur simulation préalable permet d'identifier les goulots d'étranglement.



Figure I.6 Machine de remplissage Automatique

I.3.2.2 Machines de remplissage semi-automatique

Ces machines sont équipées d'une pompe indépendante. L'opérateur doit placer manuellement le produit sous le bec de remplissage puis actionner la pompe. À la fin du dosage, l'opérateur récupère manuellement le contenant rempli.



Figure I.7 Machine de remplissage semi-automatique

I.3.2.3. Machines de remplissage manuel

Ces machines fonctionnent à l'aide d'une pompe à piston. L'opérateur remplit le produit dans la trémie et actionne un levier. La pompe à piston envoie ensuite le liquide dans le récipient via un bec fin en acier.



Figure I.8 Machine de remplissage manuel

I.3.3 Les avantages et désavantages de chaque type de remplisseuse

I.3.3.1 Remplisseuse automatique

Avantages :

- Elles augmentent la vitesse de conditionnement, ce qui réduit la durée du processus.
- Elles sont intégrées à des systèmes de contrôle API qui permettent aux conditionneurs de configurer la machine en fonction du type de récipient et de la quantité souhaitée par récipient.
- Elles offrent une bonne cadence de travail.
- Elles offrent un dosage très précis.

Désavantage :

Le principal inconvénient de cette machine est le prix, les machines de remplissage automatique sont en effet très chères à comparaison avec les autres deux types.

I.3.3.2 Remplisseuse semi-automatique

Avantage :

- Offre une bonne cadence de travail à un prix plus raisonnable qu'avec une machine de remplissage entièrement automatique.

Désavantage :

- N'est pas équipée avec des convoyeurs électriques.

I.3.3.3 Remplisseuse manuelle

Avantage :

- Moins chères.

Désavantages :

- Productivité faible à comparaison avec les autres types.
- La précision dépend sur l'opérateur.
- L'existence d'un opérateur est essentielle pour le fonctionnement de la machine.

I.3.4. Le choix des machines de remplissage

On a des plusieurs facteurs avant choisir les remplisseurs doivent être pris en compte :

I.3.4.1 Le produit à conditionner

La nature du produit est importante. Est-ce un produit liquide, visqueux, mousseux, pâteux, gazeux ou bien solide ?

I.3.4.2 Le type de contenant

C'est crucial de considérer le type de contenant utilisé comme sa matière, la forme et la taille et également des autres points avec le contenant, comme si le contenant a un bouchon, une machine de remplissage avec boucheuse pourrait être utile, tout ça doit être considéré pour l'investissement d'une machine flexible capable de gérer une variété de formes et de tailles de contenants.

I.3.4.3 La cadence de production et l'espace

La taille de l'entreprise et le budget font un grand effet sur la remplisseuse, les remplisseuses grandes ont des cadences de production plus élevé en nombre de récipients remplis que les petites remplisseuses, donc pour de grandes productions, privilégier des grandes remplisseuses et grand espace.

I.3.4.4 L'entretien

Si le remplisseuse est utilisé pour remplir plusieurs différents liquides, il est nécessaire de la nettoyer chaque fois le liquide est changé pour éviter la contamination entre différents produits, cependant, le nettoyage est un processus compliquer car il nécessite que démonter les partie de la machine qui sont en contact avec le produit, donc pour faciliter ce processus, certaines machines électriques sont équipées d'un système de nettoyage intégrée. Ce système permet de nettoyer la machine automatiquement sans avoir à la démonter, et gagner plus de temps et d'effort.

I.3.5 Domaines d'utilisation

I.3.5.1 Remplissage manuel

Ce type de remplissage est généralement utilisé pour les petites productions, startups, petites entreprises ou pour des produits spécifiques qui nécessitent une attention particulière.

I.3.5.2 Remplissage semi-automatique

Ce type de remplissage est utilisé lorsque la production est plus grande mais pas assez grande pour justifier l'utilisation d'une machine de remplissage entièrement automatique, il est utilisé par les entreprises moyennes.

I.3.5.3 Remplissage automatique

Ce type de remplissage est utilisé par les grandes entreprises ou la production est élevée, Il est couramment utilisé dans diverses industries, notamment l'agroalimentaire, la cosmétique et l'industrie pharmaceutique.

En général utilisation de cette remplisseuse est dans la majorité d'usines comme (production de jus, eau, l'huile, lait...).

I.4 Conclusion

Ce chapitre a présenté les différents types de machines de remplissage. Les principes de fonctionnement et les domaines d'application ont été abordés afin de fournir les bases nécessaires à l'analyse et à l'optimisation virtuelle des procédés de remplissage.

Chapitre II :

Partie Opérative d'une Ligne de Remplissage Automatique

II.1. Introduction

Dans ce chapitre nous allons présenter les outils nécessaires pour la partie opérative de notre ligne de remplissage automatique, expliquant les différents équipements et composantes utilisées, leur description, types et modes de fonctionnement.

II.2. Les Capteurs

II.2.1. Définition

Dispositif qui transforme une grandeur physique en une grandeur (généralement un signal) électrique normée (exemple Norme 4-20 mA), interprétable / exploitable, par un système de détection, d'observation, de diagnostic, ou de contrôle/commande d'un processus industriel. [6] Dans le cadre d'une simulation, la transformation de la grandeur physique en signal électrique est généralement représentée par des valeurs numériques générées ou modifiées par le logiciel.



Figure II.1 Principe de fonctionnement d'un capteur

II.2.2. Type de détection

Dans une simulation, l'événement de détection est déclenché selon la logique du programme, simulant le comportement du capteur dans un environnement réel.

- ❖ **Détection par contact** : le capteur doit entrer en contact physique avec un phénomène pour le détecter
- ❖ **Détection sans contact** : le capteur détecte le phénomène à proximité de celui-ci [7]

II.2.3. Type des capteurs

Selon le scénario simulé, différents types de capteurs peuvent être intégrés virtuellement à chaque poste, comme la détection de présence de bouteille ou le contrôle de position des éléments mobiles.

Il y a des différents types des capteurs, ont à explorer leurs applications et leur importance dans divers domaines.

II.2.3.1. Les Capteur Passif

Le capteur est passif lorsque la grandeur électrique exploitable est une impédance à dominante capacitive, inductive résistive.

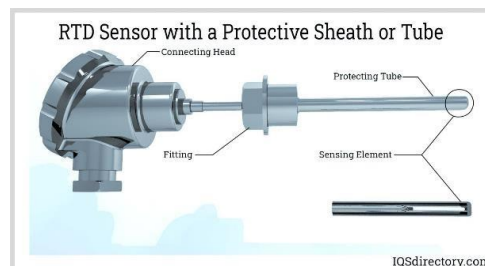


Figure II.2 Principe de fonctionnement d'un capteur passif

Dans la figure **II.2**, une jauge de contrainte résistive (transducteur) est collée sur l'extérieur d'un tube. Sous l'influence d'une variation de pression, la jauge subit une déformation identique au tube (corps d'épreuve), et sa résistance ohmique varie.

II.2.3.1.1. Facteurs d'Influence sur l'Impédance des Capteurs

La géométrie et les dimensions d'un capteur peuvent influencer son impédance en affectant la manière dont il interagit avec le mesurande (la grandeur que le capteur mesure).

Les matériaux utilisés dans la fabrication du capteur ont des propriétés électriques spécifiques qui affectent son impédance.

Le mesurande, ou la grandeur mesurée par le capteur, peut agir directement sur les caractéristiques géométriques ou dimensionnelles du capteur, sur ses propriétés électriques, ou parfois sur les deux simultanément. [8]

II.2.3.1.2. Les circuits conditionnés associé

Les variations d'impédance ne sont mesurables dans un capteur passif qu'en intégrant le capteur dans un circuit électrique alimenté, il faut leur appliquer une tension pour obtenir un signal de sortie.

Exemple :

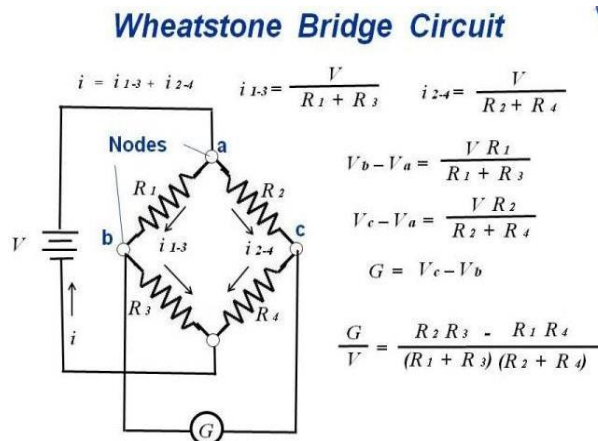


Figure II.3 Montage électronique d'un capteur en le cas de changement de résistivité

Si nous prenons la jauge de la **Figure II.2**, le changement de sa résistance peut être calculé en l'intégrant dans un circuit de Wheatstone.

II.2.3.1.3. Les avantages d'un capteur passif

❖ Avantages :

Les capteurs passifs ne nécessitent pas une source d'énergie élevée pour détecter la cible. Les capteurs passifs peuvent être moins coûteux à exploiter. Les capteurs passifs ne sont pas affectés par les conditions météorologiques ou les interférences atmosphériques.

Les capteurs passifs peuvent capturer des images avec une haute résolution spatiale.

Les capteurs passifs peuvent fournir des informations précises sur l'énergie naturelle émise ou réfléchie par la cible.

❖ **Désavantages :**

Les capteurs passifs ne peuvent pas pénétrer les nuages ou la fumée pour détecter la cible.

Les capteurs passifs ne peuvent pas être utilisés pour détecter des objets cachés derrière d'autres objets.

Les capteurs passifs sont limités à la quantité d'énergie naturelle émise ou réfléchie Par la cible.

[9]

II.2.3.2. Les Capteurs Actifs

Le capteur est actif lorsque, soumis au mesurande, il se comporte en générateur électrique : générateur de charge, de courant, ou de force électromotrice.

Un capteur actif est généralement fondé dans son principe sur un effet physique qui assure la conversion en énergie électrique de la forme d'énergie propre au mesurande.

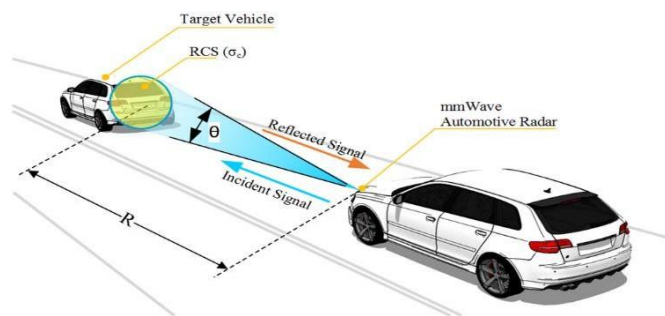


Figure II.4 Principe de fonctionnement d'un capteur actif

II.2.3.2.1. Les types d'énergie en capteur actif par le mesurande

❖ **Effet thermoélectrique :**

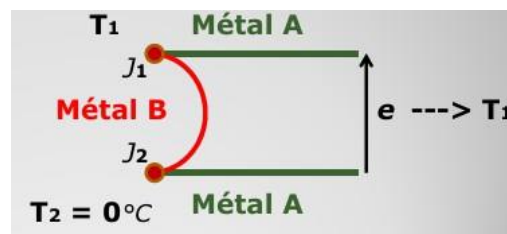


Figure II.5 Principe d'un effet thermoélectrique

Un circuit formé de deux conducteurs de nature chimique différente dont les jonctions J1 et J2 sont à des températures différentes T1 et T2, est le siège d'une force électromotrice : $e(T_1, T_2)$ proportionnelle à la différence de température.

❖ Effet photoélectrique :

En général, le concept de cet effet est dépendu sur la libération de charges électriques dans la matière sous l'influence d'un rayonnement lumineux, ou plus généralement électromagnétique, dont la longueur d'onde est inférieure à une valeur seuil, caractéristique du matériau.

❖ Effet piézoélectrique :

L'application d'une contrainte mécanique à certains matériaux avec des propriétés piézoélectriques comme Quartz entraîne une déformation qui crée des charges électriques égales

II.2.3.2. Les avantages d'un capteur actif

❖ Avantages :

Les capteurs actifs peuvent pénétrer les nuages et la fumée pour détecter la cible.

Les capteurs actifs peuvent être utilisés pour détecter des objets cachés derrière d'autres objets.

Les capteurs actifs peuvent fonctionner à tout moment du jour ou de la nuit. Les capteurs actifs peuvent générer des images 3D de la cible.

❖ Désavantages :

Les capteurs actifs nécessitent une source d'énergie élevée pour générer l'énergie nécessaire à la détection de la cible.

Les capteurs actifs peuvent être affectés par des facteurs externes tels que les conditions météorologiques et les interférences atmosphériques.

Les capteurs actifs peuvent être coûteux à exploiter.

II.2.3.2.3. Caméras intelligentes pour l'automatisation industrielle

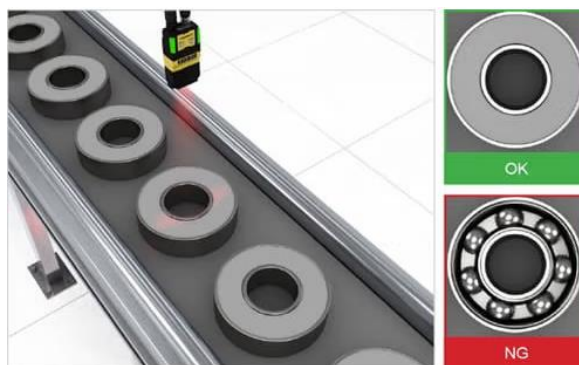


Figure II.6 Un capteur In-Sight SnAPP qui classe les roulements à billes sur une ligne de production comme OK ou NG.

Dans l'automatisation de la fabrication, les caméras intelligentes sont au cœur de systèmes de vision précis et efficaces. Ces caméras sont des appareils compacts et faciles à utiliser qui combinent l'acquisition d'images, le traitement et l'analyse de données en un seul outil.

Les caméras intelligentes de vision industrielle combinent une caméra et un ordinateur en un seul appareil. Contrairement aux caméras traditionnelles qui se contentent de capturer des images, elles traitent et analysent les images sans avoir besoin d'un ordinateur séparé.

Il rendent l'automatisation et le contrôle qualité beaucoup plus rapides, fiables et efficaces – tout en occupant moins d'espace et en étant plus simples à installer et à entretenir que les systèmes de vision industrielle traditionnels avec PC séparés.

Les caméras intelligentes modernes intègrent souvent des fonctionnalités d'intelligence artificielle qui les rendent faciles à installer, flexibles et suffisamment puissantes pour relever des défis industriels complexes. [10]

❖ Applications des systèmes de vision industrielle :

Les systèmes de vision industrielle sont utilisés dans de nombreux secteurs, notamment la fabrication, l'automobile, l'électronique, la pharmacie, et bien d'autres. Ces systèmes permettent de réaliser des tâches telles que :

L'inspection des produits pour détecter les défauts et garantir le respect des normes de qualité.

La mesure précise des dimensions des objets.

L'identification et la lecture des codes-barres, QR codes et autres marquages pour le suivi et le tri.

II.2.3.3 Les Capteur TOR

Un capteur TOR (tout-ou-rien) est un capteur qui transforme un phénomène physique en signal électrique logique qui ne peut prendre que deux valeurs. Ces deux valeurs sont nommées **états haut** et **état bas**. [13]

II.2.3.3.1. Principe de fonctionnement



Figure II.7 Les capteurs TOR

Le capteur TOR possède deux seuils distincts afin d'éviter que la sortie ne devienne instable lorsque l'entrée est très proche du seuil.

Pour que la sortie du capteur passe à l'état haut, il faut que la grandeur d'entrée passe au-dessus du seuil haut.

Pour que la sortie du capteur passe à l'état bas, il faut que la grandeur d'entrée passe en dessous du seuil bas. [13]

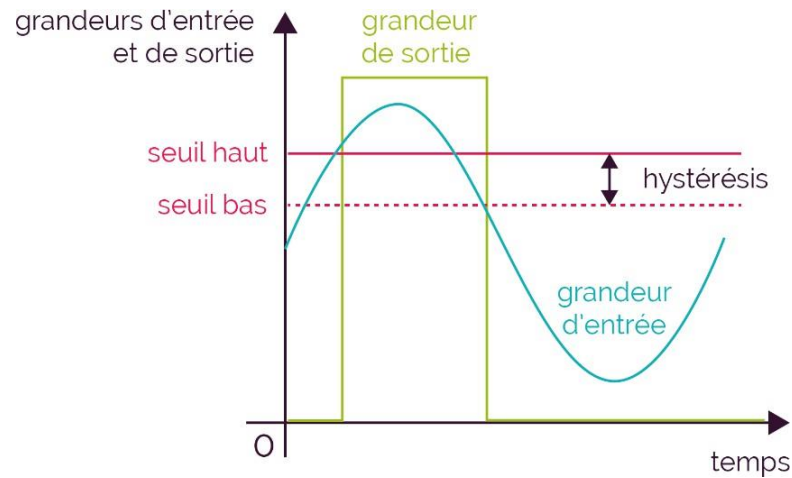


Figure II.8 Principe de fonctionnement d'un capteur TOR

II.2.3.3.2. Domaine d'application

Les capteurs TOR peuvent être utilisés pour détecter la présence d'une bouteille à une certaine position sur la ligne de production

Ils peuvent également être utilisés pour détecter si le niveau d'un liquide dans un réservoir atteint un certain seuil

Dans le cadre de la simulation, ces capteurs peuvent être utilisés pour modéliser la détection de la présence de bouteilles ou le contrôle du niveau de liquide à différentes étapes de la ligne de remplissage.

II.2.3.4. Les Capteurs Analogiques

II.2.3.4.1. Principe de fonctionnement

Un capteur analogique fournit un signal électrique analogique qui peut prendre toutes les valeurs possibles entre une valeur minimale et une valeur maximale proportionnel à la grandeur physique mesurée.

Domaines d'application :

Les capteurs de pression analogiques peuvent être utilisés pour surveiller la pression dans les tuyaux, les réservoirs ou les moteurs.

Les capteurs de température analogiques peuvent être utilisés pour surveiller la température des liquides avant, pendant et après le processus de remplissage.

- Dans ce graphe, on a un capteur de température analogique pourra fournir une tension proportionnelle à la température qu'il mesure. Pour chaque augmentation de 1 °C à partir de 0°C, la tension augmentera de 0,1 V. [13]

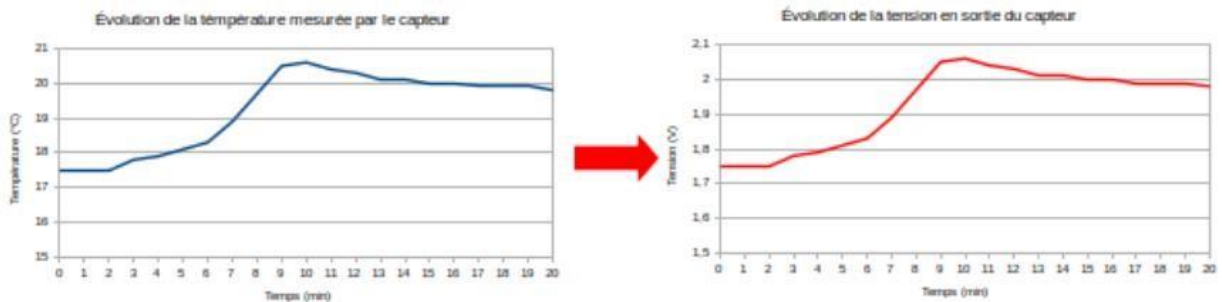


Figure II.9 Signal d'un capteur analogique de température



Figure II.10 Manomètre analogique

- Un capteur de force analogique à jauges de contrainte permet de convertir une force mécanique en un signal électrique analogique. Lorsque la force est appliquée sur le corps du capteur, celui-ci subit une déformation au niveau de la surface du matériau. Cette déformation, appelée allongement, est détectée par des jauges de contrainte fixées sur le corps d'épreuve. Les jauges de contrainte transforment cet allongement mécanique en une variation de résistance électrique.

La force F en Newton [N] est donnée par la formule suivante :

$$F = m \times g$$

Où m est la masse en kilogrammes et g l'accélération de la pesanteur (environ 10 m/s^2).

Lorsque le capteur subit une force, il se produit un allongement ε exprimé comme suit :

$$\varepsilon = \Delta l / l_0$$

Où Δl est la variation de longueur et l_0 la longueur initiale. L'allongement est mesuré en $[\mu\text{m/m}]$.

La sensibilité du capteur dépend aussi du module de Young E du matériau, qui relie la contrainte à l'allongement. Pour l'acier, $E \approx 200\,000 \text{ N/mm}^2$, et pour l'aluminium, $E \approx 70\,000 \text{ N/mm}^2$.

Les jauges de contrainte génèrent une variation de résistance proportionnelle à l'allongement, qui est ensuite convertie en tension analogique à l'aide d'un montage en pont de Wheatstone (**Figure II.3**) et d'un amplificateur en pont. Le signal de sortie est alors analogique, proportionnel à la force exercée sur le capteur.

Les principaux avantages des capteurs de force analogiques à jauges de contrainte sont leur haute précision, leur stabilité à long terme et leur capacité à mesurer aussi bien des charges statiques que dynamiques. Ils sont également robustes face à la fatigue et aux vibrations, et leur conception permet une bonne compensation des effets de température.

Ainsi, le capteur de force analogique est largement utilisé dans des applications industrielles pour mesurer des forces, des charges ou des contraintes mécaniques de manière fiable et précise. [11]

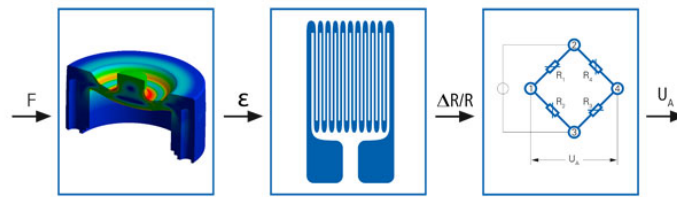


Figure II.11 Principe de conversion d'une force mécanique en signal électrique dans un capteur de force à jauges de contrainte

- Le capteur de couple est un transducteur qui convertit une mesure de couple, qu'elle soit statique ou dynamique, en un signal électrique proportionnel. Il existe deux grandes catégories : le capteur de couple rotatif (dynamique) et le capteur de couple statique de réaction. Le capteur rotatif est utilisé pour mesurer le couple sur des arbres en rotation, comme dans les moteurs ou les bancs d'essai, tandis que le capteur statique est adapté aux mesures sur des systèmes fixes.

Le principe de fonctionnement repose principalement sur l'utilisation de jauges de contrainte collées sur un corps métallique (acier inoxydable ou aluminium). Lorsque le couple est appliqué, le corps du capteur se déforme légèrement, modifiant la résistance des jauges. Cette variation est convertie en une tension via un pont de Wheatstone, offrant une mesure précise et fiable du couple.

Les capteurs rotatifs peuvent transmettre le signal par collecteur tournant, télémètre sans fil ou transformateurs rotatifs. Certains modèles intègrent un codeur pour mesurer l'angle ou la vitesse de rotation. Il existe aussi des capteurs sans contact basés sur la magnétostriction, capables de fonctionner dans des environnements exigeants.

Le choix d'un capteur de couple dépend de l'application : type de montage, sens de rotation, précision, vitesse, robustesse et type de signal de sortie (analogique ou numérique). Une sélection judicieuse permet d'assurer la compatibilité et la précision du système de mesure, éléments essentiels pour toute application de contrôle ou de simulation de serrage, comme cela sera abordé dans les chapitres suivants. [12]

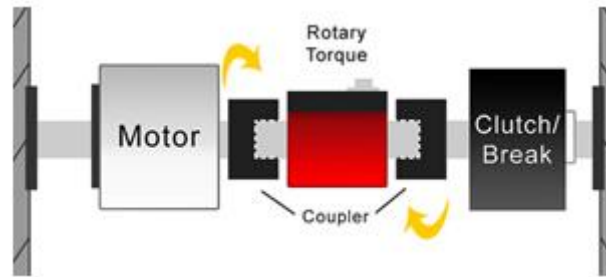


Figure II.12 Schéma d'intégration d'un capteur de couple rotatif entre moteur et système d'actionnement

II.2.3.5 Les Capteurs Numériques

II.2.3.5.1. Principe de fonctionnement



Figure II.13 Thermomètre digital

Un capteur numérique est un capteur qui effectue successivement La transduction d'un phénomène physique en signal électrique analogique et la numérisation du signal analogique en signal logique.

Exemple d'un capteur de température numérique va mesurer puis transmettre la valeur 16,9 °C à un microcontrôleur, en se servant du protocole UART. Voici à quoi ressemble le signal qui permet de transmettre cette valeur : [13]

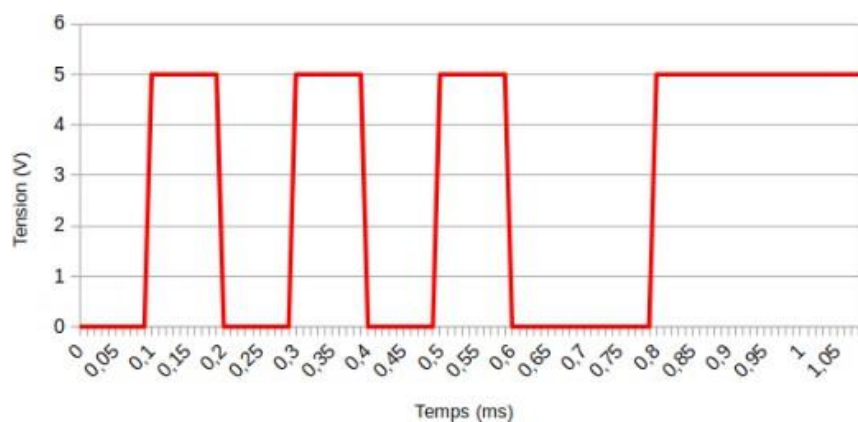


Figure II.14 Signal logique codé

II.2.3.5.2. Domaine d'application

Surveillance environnementale (mesure de paramètres tels que l'humidité, la qualité de l'air et les niveaux de radiation).

Traitement digital du résultat sur les mesurande [14]

II.2.3.6. Les Capteurs de proximité

II.2.3.6.1. Définition

Les capteurs de proximité sont des dispositifs électroniques qui permettent de détecter la présence ou l'absence d'objets sans aucun contact physique.

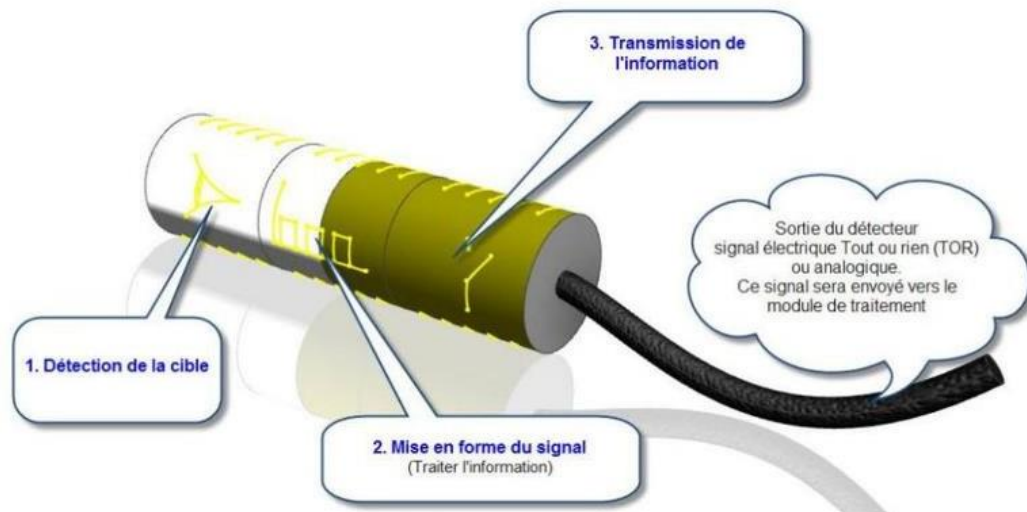


Figure II.15 Capteur de proximité

II.2.3.6.2. Les type des capteurs de proximité

Les capteurs de proximité Capacitifs, Les capteurs de proximité Inductifs, Les capteurs de proximité Photoélectrique, Les capteurs de proximité Magnétique, Les capteurs de proximité Ultrasons

II.2.3.6.3. Capteur de proximité Photoélectrique

II.2.3.6.3.1. Définition

Dans les systèmes automatisés, on peut détecter une présence, sans contact avec un objet ou une personne, voir même en être très éloigné. On utilise entre autres des détecteurs de proximité photo électrique, pour nous projet on va utiliser le mode de proximité

II.2.3.6.3.2. Principe de fonctionnement

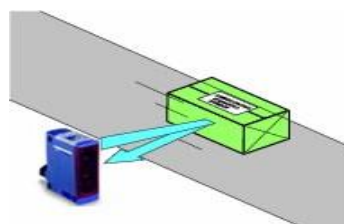


Figure II.16 Principe de la détection pour un capteur photoélectrique

Dans le mode de proximité on a l'émetteur et le récepteur dans le même boîtier, le composant d'émission est une diode DEL qui transforme le signal électrique en signal lumineux monochromatique, et le récepteur est une photodiode ou phototransistor qui génère u courant électrique par l'effet de photoluminescence.

La portée du capteur dépend du pouvoir réfléchissant et de la taille de l'objet. En l'absence de l'objet, le faisceau n'est pas renvoyé, mais en présence de l'objet, il y a détection. Les détecteurs sont équipés d'un dispositif de réglage de la sensibilité.

Avantages :

Un seul boîtier (émetteur + récepteur),

Détection de cible à coefficient de réflexion élevé (de 0,6 à 0,9). Détection pour tous les objets appart les objets opaques. [15]

II.2.4. Branchement d'un capteur

Il y a des plusieurs types pour le branchement d'un capteur, mais nous sommes intéressés par la technique de 3 fils utilisé dans notre projet

II.2.4.1. Par 3 fils

II.2.4.1.1. Définition

Un détecteur de proximité en courant continu dans notre projet est connecté avec trois fils, fil brun d'alimentation de 24 v, fil bleu de la masse (ground) et fil noir du signal de sortie [16]

II.2.4.1.2. Détecteur PNP

Lorsque qu'il y a détection, le transistor est passant (contact fermé). Il va donc imposer le potentiel (+) sur la sortie S. La charge est branchée entre la sortie S et le potentiel (-). Ce type de détecteur est adapté aux unités de traitement qui fonctionnent en logique positive.

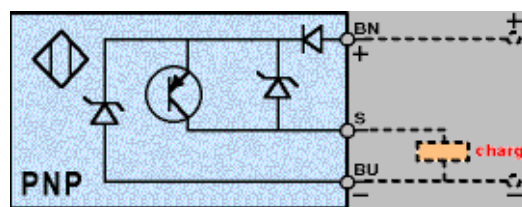


Figure II.17 Branchement d'un capteur PNP avec trois fils

II.2.4.1.3. Détecteur NPN

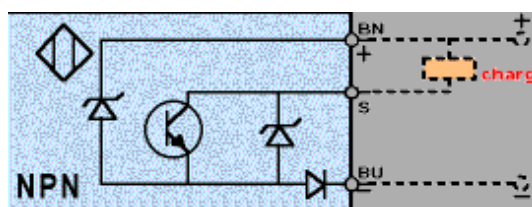


Figure II.18 Branchement d'un capteur NPN avec trois fils

Lorsque qu'il y a détection, le transistor est passant (contact fermé). Il va donc imposer le potentiel (-) sur la sortie S. La charge est branchée entre la sortie S et le potentiel (+). Ce type de détecteur est adapté aux unités de traitement qui fonctionnent en logique négative. On prendra donc soin d'identifier le type de logique utilisée par les unités de traitement (Automate programmable, etc.). [17]

II.3. Les Pré-Actionneurs

Ce sont des éléments de la Partie Opérative qui reçoivent de l'énergie "transportable" et la transforment en énergie "utilisable" pour le système. Ils exécutent les ordres reçus en agissant sur le système ou son environnement.

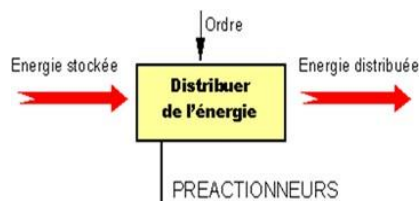


Figure II.19 Principe de fonctionnement des pré-actionneurs

II.3.1. Les distributeurs pneumatiques

II.3.1.1. Définition

Les distributeurs pneumatiques sont des éléments de la chaîne d'énergie. Ils distribuent de l'air comprimé aux actionneurs pneumatiques (vérins, générateurs de vide, moteurs à palettes...) à partir d'un signal de commande (pilotage). [18]

II.3.1.2. Les types des distributeurs pneumatiques

II.3.1.2.1. Les distributeurs à clapets

Ils sont constitués d'équipements mobiles à clapets munis de joints qui, en se déplaçant d'un siège à l'autre, ouvrent ou obturent le passage de l'air comprimé.

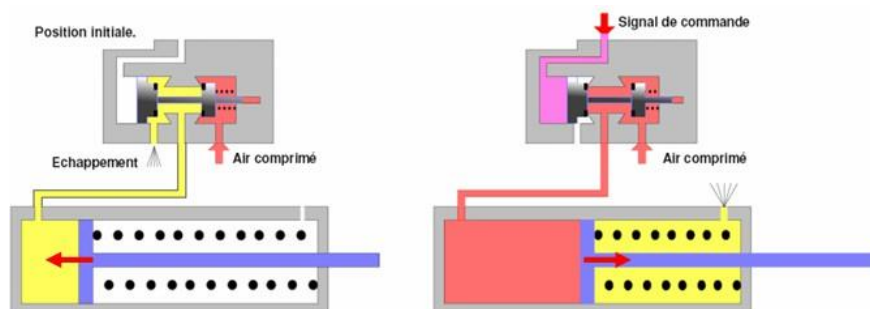


Figure II.20 Principe de fonctionnement d'un distributeur à clapets

II.3.1.2.2. Les distributeurs à tiroirs

L'équipement mobile de ce type d'appareil comprend un axe épaulé, appelé tiroir, qui, en se déplaçant à l'intérieur d'un corps muni de joints d'étanchéité, met les orifices d'utilisation en communication.

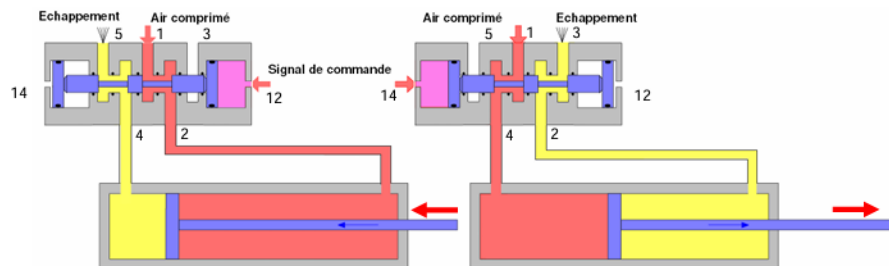


Figure II.21 Principe de fonctionnement d'un distributeur à tiroirs

II.3.1.3. Symbolisation d'un distributeur

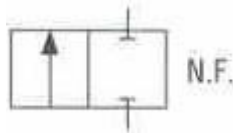


Figure II.22 Distributeur avec 2 orifices et 2 positions

II.3.1.4. Distributeur bistable/monostable

Si le distributeur possède un rappel par ressort on dit qu'il est MONOSTABLE. En présence du signal de pilotage, le tiroir bascule et reste dans sa position si le pilotage est maintenu. En l'absence de signal de pilotage le tiroir retrouve sa position repos sous l'action du ressort.

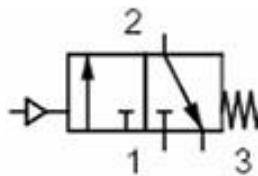


Figure II.23 Distributeur monostable

Si le distributeur possède deux pilotages il est dit BISTABLE. En l'absence de signal de pilotage, le tiroir ne bouge pas et occupe la position qu'il avait précédemment.

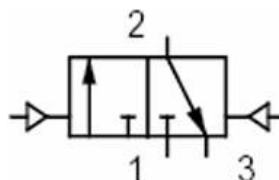


Figure II.24 Distributeur bistable

II.3.1.5. Pilotage d'un distributeur

Pour change la position du distributeur, il y a plusieurs types des commandes :

- Les types des commandes manuelles
- Les types des commandes mécaniques
- Les types des commandes électro-aimant
- Les types des commandes distributeur pilotent

II.3.1.6. Les applications des distributeurs pneumatiques

Les distributeurs pneumatiques servent à répartir et à contrôler les flux d'air comprimé.

Ils sont utilisés dans les systèmes pneumatiques comme es ligne de remplissage automatisés pour acheminer l'air comprimé vers différents consommateurs (ex : vérin) [19]

II.3.2. Les pré-actionneurs électriques

II.3.2.1. Définition

Parmi les pré-actionneurs électriques les plus utilisés on trouve les relais et les contacteurs. Ces dispositifs permettent de commander un circuit de puissance à partir d'un circuit de commande. Les relais sont utilisés avec des circuits intégrés et un petit circuit de commutation (transistor), ils permettent de commander un circuit de puissance (contacteurs, lampes...). Les contacteurs fonctionnent de la même façon que les relais, ils permettent cependant la circulation d'un courant beaucoup plus important. . Les contacteurs sont utilisés pour des très fortes puissances (moteur). [20]

II.3.2.2. Les relais

II.3.2.2.1. Les relais magnétiques

Ils sont faire une transition entre un courant faible et un courant fort. Mais il sert également à commander plusieurs organes simultanément grâce à ses multiples contacts synchronisés.



Figure II.25 Relais électromagnétique

II.3.2.2.1.1 Principe de fonctionnement

Lorsqu'un courant électrique traverse la bobine, elle génère un champ magnétique qui attire l'armature mobile, ce mouvement de l'armature ouvre ou ferme le contact, si l'armature est attiré ; le contact est ferme donc permet le courant de circuler, si le courant de la bobine est coupé, l'armature

Revient à sa position origine et interrompent le contact (contact ouvre).

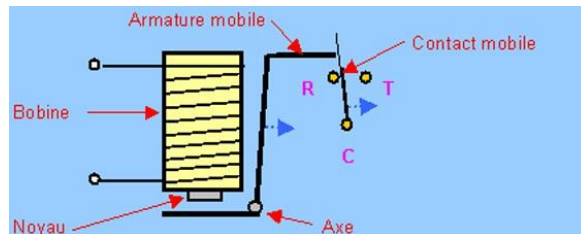


Figure II.26 Principe de fonctionnement d'un relais magnétique

II.3.2.2.1.2 Les type des contacts

Les contacts sont les parties métalliques qui transmettent ou interrompent le courant en fonction de la commande de la bobine

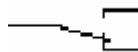
Avec contact NO



Avec contact NF



Avec contact inverseur



II.3.2.2.2. Les relais statiques

Un relais statique est un organe ayant la fonction d'un relais mais réalisé avec des composants électroniques, sans aucune pièce mécanique en mouvement.



Figure II.27 Un relais statique

II.3.2.2.1 Principe de fonctionnement

Un relais statique est constitué de trois circuits :

Circuit d'entrée qui assure l'isolement galvanique entre le circuit de commande et celui de puissance. Cet isolement est assuré par une photo coupleur.

Circuit d'adaptation qui traite le signal d'entrée et assure la commutation du circuit de sortie. En particulier dans le cas de la commutation au zéro de tension, ce circuit assure que la commutation de la sortie a lieu au zéro de tension suivant.

Circuit de sortie qui est composé de l'organe de puissance. Celui-ci peut être soit un triac soit des thyristors antiparallèles. Dans le cas de la commutation de charges continues, l'élément de puissance est soit un transistor soit un MOSFET.

II.3.2.3. Les applications des relais

Les relais statiques sont particulièrement utiles dans les systèmes de contrôle automatisés, tels que les automates programmables, les systèmes de chauffage, de ventilation et de climatisation, ainsi que dans les applications nécessitant une commutation précise et rapide, comme les lignes des remplissages automatique.

Les relais électromagnétiques sont utilisés dans une multitude d'applications pour contrôler efficacement les circuits électriques, Dans les secteurs industriels, ils jouent un rôle crucial dans les systèmes de commande pour machines et processus. [21]

II.3.2.3. Les contacteurs

II.3.2.3.1. Définition

Le contacteur est un appareil mécanique de connexion, capable d'établir, de supporter et d'interrompre des courants dans les conditions normales du circuit, y compris les conditions de surcharge en service.

II.3.2.3.2. Principe de fonctionnement

Le contacteur comporte 4 ensembles fonctionnels :

Le circuit principal

1. support contacts mobiles de pôle
2. contact mobile de pôle « F »
4. boîtier de pôles et chambre de coupure de l'arc
5. connexion de puissance
6. contact fixe de pôle « F »

Le circuit de la commande

3. contact mobile auxiliaire « O »
7. contact fixe auxiliaire « O »

Le circuit auxiliaire

5. Circuit de commande et de signalisation.

Organe moteur d'un contacteur

10. partie fixe de l'électro-aimant

11. bague de déphasage

12. bobine d'attraction

13. ressort de rappel de la partie mobile de l'électro-aimant

14. partie mobile de l'électro-aimant fixation pour bloc auxiliaire

Autres composants :

8. socle

9. amortisseur de choc de l'électro-aimant

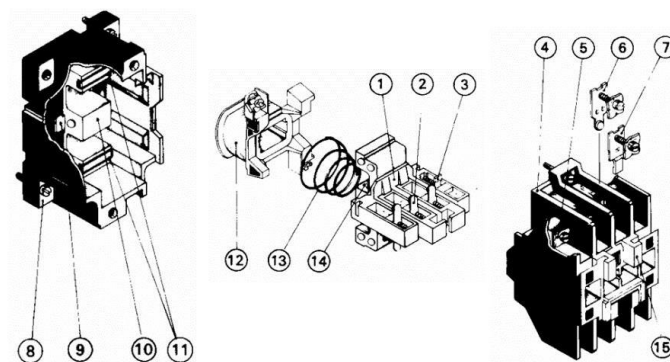


Figure II.28 Les composantes d'un contacteur

II.3.2.3.3. La symbolisation d'un contacteur

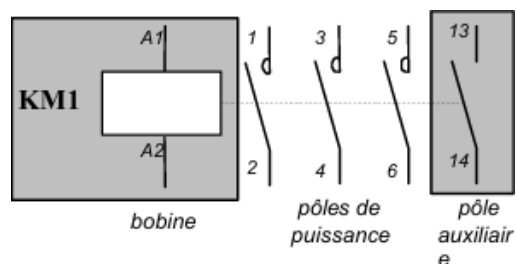


Figure II.29 Schéma de puissance d'un contacteur

II.3.2.3.4. Les applications d'un contacteur

Les contacteurs sont utilisés pour démarrer et arrêter les moteurs des pompes, convoyeurs et autres équipements dans les lignes de remplissage

Les contacteurs sont utilisés pour activer et désactiver les vannes de remplissage et les électrovannes dans les systèmes de dosage et de distribution de liquides

Ils sont associés à des relais de surcharge pour protéger les moteurs contre les courants excessifs dans les lignes de remplissage [22]

II.3.3. Les variateurs de vitesse (VFDs)

II.3.3.1. Définition

Il est important de comprendre le fonctionnement des VFD, ou variateurs de fréquence, car ils sont largement utilisés dans les applications à moteur AC, comme le

Contrôle de moteur par variateur de fréquence, en raison de leurs nombreuses caractéristiques par rapport aux entraînements de moteurs conventionnels. Les VFD offrent une plus grande fonctionnalité et des capacités opérationnelles accrues. En plus du contrôle de vitesse réglable, les variateurs de fréquence offrent des protections telles que la protection contre les phases, la sous-tension et la surtension. Les

Options logicielles et d'interface des VFD permettent à l'utilisateur de contrôler les moteurs à des niveaux souhaités.



Figure II.30 Variateur de vitesse

II.3.3.2. Principe du fonctionnement

Les deux principales caractéristiques des variateurs de fréquence sont les vitesses réglables et les capacités de démarrage et d'arrêt progressifs. Ces deux caractéristiques font des VFD des contrôleurs puissants pour le contrôle des moteurs à courant alternatif. Le VFD se compose principalement de quatre sections : le redresseur, l'intermédiaire, le lien DC, l'onduleur et le circuit de contrôle.

1. Redresseur:

C'est la première étape d'un variateur de fréquence. Elle convertit l'alimentation en courant alternatif fournie par le réseau en courant continu. Cette section peut être unidirectionnelle ou bidirectionnelle selon l'application utilisée. Comme pour le fonctionnement à quatre quadrants du moteur, elle utilise des diodes, des transistors SCR et d'autres dispositifs de commutation électronique. Un minimum de six diodes est nécessaire pour la conversion triphasée, de sorte que l'unité de redressement est considérée comme un convertisseur à six impulsions.

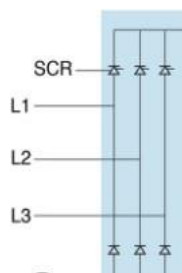


Figure II.31 Circuit d'un redresseur

2. Circuit intermédiaire:

Le courant continu provenant de la section redresseur est acheminé vers le lien DC. Cette section est composée de condensateurs et d'inducteurs pour lisser les ondulations et stocker l'énergie en courant continu. La fonction principale du lien DC est de recevoir, stocker et fournir l'énergie en courant continu.

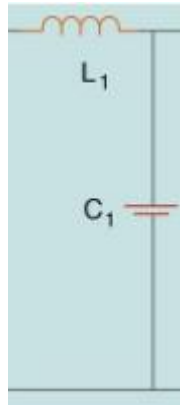


Figure II.32 Circuit intermédiaire d'un VFD

3. L'onduleur

Cette section comprend des interrupteurs électroniques tels que des transistors et des thyristors. Elle reçoit l'énergie en courant continu du lien DC et la convertit en courant alternatif, qui est ensuite délivré au moteur. Elle utilise des techniques de modulation, comme la modulation de largeur d'impulsion, pour varier la fréquence de sortie et ainsi contrôler la vitesse du moteur à induction.

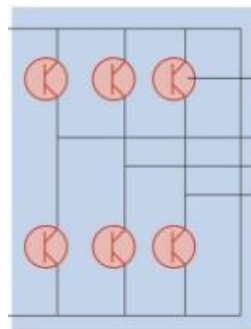


Figure II.33 Circuit de l'onduleur (transistors)

II.3.3.3. Les avantages d'un variateur de la vitesse

Les VFD permettent un positionnement précis de la vitesse du moteur en la comparant continuellement à la vitesse de référence, même en cas de changements des conditions de charge et de perturbations d'entrée telles que les fluctuations de tension, ce qui est parfait pour le contrôle des machines de remplissage.

Ils offrent une opération fluide au démarrage et à l'arrêt, et réduisent également le stress thermique et mécanique sur les moteurs et les entraînements par courroie.

II.4. Les actionneurs

II.4.1. Les vérins pneumatiques

Ils transforment l'énergie d'air comprimé (2 à 10) en énergie mécanique (mouvement avec effort) Ils peuvent soulever, pousser, tirer, tourner, percuter, abloquer, ils sont boucaux utilisant dans les lignes des remplissages. [23]

Dans une simulation de ligne de remplissage, les vérins pneumatiques sont modélisés pour reproduire les mouvements mécaniques tels que le déplacement des bouteilles, le positionnement des éléments mobiles ou l'actionnement des systèmes de fermeture.



Figure II.34 Principe de fonctionnement d'un vérin pneumatique

II.4.1.1. Les vérins simple effet (VSE)

L'ensemble tige-piston se déplace dans un seul sens sous l'action du l'air comprimé. Le retour est effectué par un autre moyen un ressort.



Figure II.35 Principe de fonctionnement et symbolisation d'un VSE avec tige rentrée au repos



Figure II.36 Principe de fonctionnement et symbolisation d'un VSE avec tige sortie au repos

II.4.1.2. Les vérins double effet (VDE)

L'ensemble tige + piston peut se déplacer dans les deux sens sous l'action du fluide sous pression. L'effort en POUSSANT (tige sortante) est légèrement plus grand que l'effort en tirant (entrée de tige) car la pression n'agit pas sur la partie de surface du piston occupée par la tige.

Ils sont boucaux utilisant dans la ligne automatique industrielle tant qu'il Ils permettent un réglage plus aisé de la vitesse de la tige par contrôle du débit à l'échappement et l'amortissement dans les deux sens.

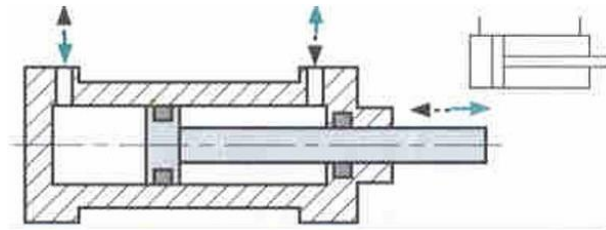


Figure II.37 Principe de fonctionnement et symbolisation d'un VDE

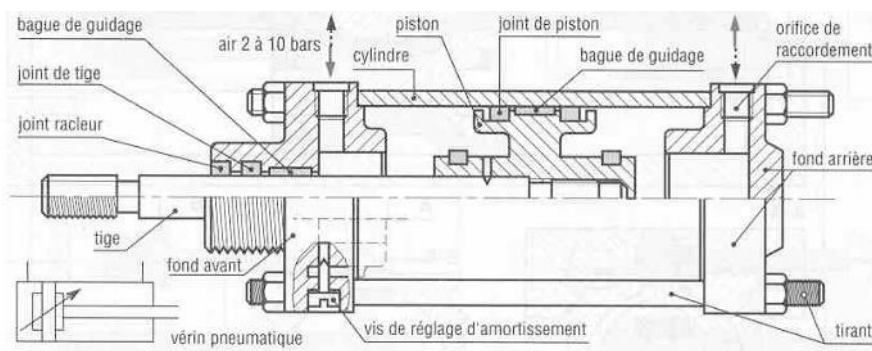


Figure II.38 Constituants d'un vérin VDE amorti

II.4.1.3. Les vérins oscillants

Le vérin oscillant à pignon/crémaillère est une réalisation spéciale du vérin à double effet. Ici, la tige de vérin est réalisée en tant que crémaillère qui entraîne un pignon. De ce fait, le déplacement rectiligne de la tige de vérin est transformé en un déplacement rotatif. Les réalisations standards ont un angle de rotation de 90°, 180°, 270° ou 360°. Sur certains types de vérins oscillants, l'angle de rotation peut être rectifié à quelques degrés près à l'aide d'une vis de réglage.

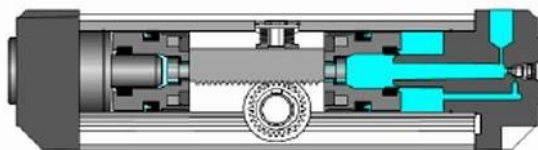


Figure II.39 Vérin oscillant

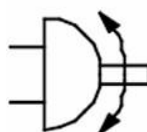


Figure II.40 Symbolisation d'un vérin oscillant

II.4.1.4. Les avantages et désavantages des VSE et VDE

Un problème avec les vérins pneumatiques à simple effet est l'incohérence de la course de sortie. Comme un ressort mécanique est utilisé comme force opposée, il peut finir par s'user. Cela entraîne une longueur de course de sortie incohérente du vérin. Les vérins pneumatiques à simple effet ne seraient pas utilisés dans des applications où la précision est requise à cause de cela.

Contrairement aux vérins à simple effet, les vérins à double effet ont une force de sortie constante sur toute la longueur de leur course.

Un des inconvénients est le besoin d'air comprimé des deux côtés du piston. [24]

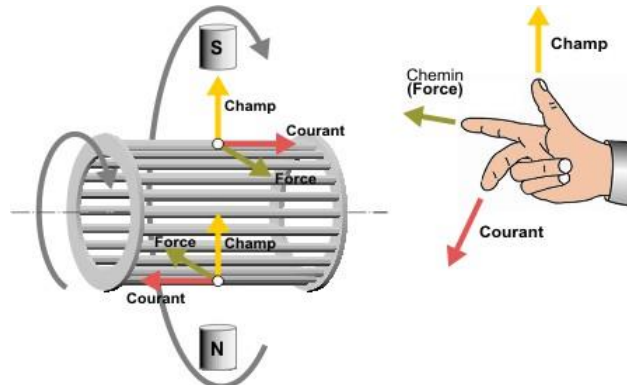


Figure II.41 La règle de la main droite

II.4.2. Les moteurs électriques

II.4.2.1. Définition

Les moteurs électriques sont les équipements plus utilisés dans chaque application qui demande rotation, ils sont transformés l'énergie électrique à énergie mécanique rotatif [25]

Dans le cadre de la simulation, les moteurs électriques sont modélisés pour reproduire les mouvements rotatifs nécessaires à l'automatisation de la ligne de remplissage.

Il y a deux types major des moteurs : **triphasé** et **monophasé**

II.4.2.2. Les moteurs monophasés

II.4.2.2.1. Définition

Le moteur monophasé est idéal pour une utilisation domestique. C'est une alimentation courante chez les particuliers. Cela signifie que le courant électrique comporte qu'une seule phase. Un moteur monophasé est donc idéal pour les appareils nécessitant 230 Volts ou moins.

Il peut utiliser dans les startups industrielles petites qui demandent automation simple comme les moteurs pas à pas.

II.4.2.2.2. Principe de fonctionnement

D'une part sur la création d'un courant électrique induit dans un conducteur placé dans un champ magnétique tournant. Le conducteur en question est un des barreaux de la cage d'écureuil ci-dessous constituant le rotor du moteur. L'induction du courant ne peut se faire que si le conducteur est en court-circuit (c'est le cas puisque les deux bagues latérales relient tous les barreaux).

D'autre part, sur la création d'une force motrice sur le conducteur considéré (parcouru par un courant et placé dans un champ magnétique tournant ou variable) dont le sens est donné par la règle des trois doigts de la main droite.

Les moteurs monophasés à induction ne sont pas auto-démarrant, et il est indésirable de recourir à la rotation mécanique de l'arbre ou à la traction d'une courroie pour les démarrer. Pour rendre un moteur monophasé à induction auto-démarrant, nous devons produire un champ magnétique statique tournant. Cela peut être réalisé en convertissant une alimentation monophasée en une alimentation biphasée à l'aide d'un enroulement supplémentaire. Lorsque le moteur atteint une vitesse suffisante, le dispositif de démarrage (c'est-à-dire l'enroulement supplémentaire) peut être retiré en fonction du type de moteur. [26]

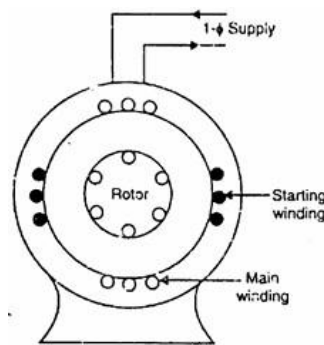


Figure II.42 Principe de fonctionnement d'un moteur monophasé

Il y a plusieurs types des moteurs classifiés par dépendant sur la méthode d'auto-démarrant

- Moteur a phase auxiliaire résistif
- Moteur a phase auxiliaire capacitif
- Moteur a condensateur permanent

II.4.2.2.3. Moteur brushless DC

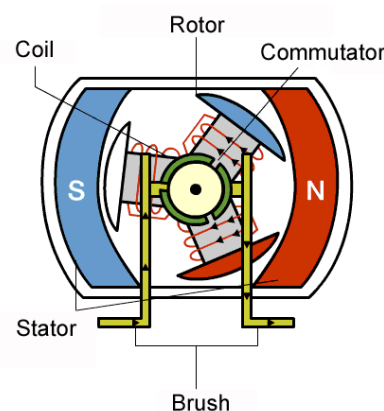


Figure II.43 Principe de fonctionnement d'un BLDC

Les moteurs à courant continu à balais (DC) sont simples et peu coûteux, ce qui en fait un choix courant pour des applications telles que l'ouverture/fermeture de tiroirs ou la motorisation de vitres électriques dans les véhicules. Cependant, leurs balais et leur collecteur s'usent rapidement à cause du contact permanent, nécessitant un entretien régulier.

Les moteurs pas à pas sont pilotés par impulsions ; ils effectuent une rotation précise à chaque impulsion, ce qui les rend adaptés au positionnement dans des équipements comme les imprimantes ou les fax. Les moteurs synchrones tournent en synchronisme avec la fréquence du courant d'alimentation, typiquement utilisés dans les plateaux tournants de micro-ondes. Les moteurs asynchrones (induction) étaient autrefois courants dans les ventilateurs et les lave-linge.

Les moteurs brushless DC (BLDC) fonctionnent sans balais. Leur rotor est un aimant permanent, tandis que les bobines sont fixées sur le stator, supprimant ainsi le besoin de collecteur et de balais. Le courant est directement acheminé vers les bobines fixes, et la rotation est pilotée en modulant l'intensité et la direction du courant. Cette conception permet aux moteurs BLDC de délivrer un couple maximal en continu et d'assurer un contrôle précis de la vitesse et de la force.

❖ **Avantages des moteurs BLDC :**

- Une efficacité élevée, permettant une opération continue au couple maximal.
- Une excellente contrôlabilité, avec ajustement précis du couple et de la vitesse grâce aux systèmes de rétroaction, ce qui réduit la consommation énergétique et la génération de chaleur.
- Une grande durabilité et une faible génération de bruit électrique, en raison de l'absence de balais et de collecteur.

Ces caractéristiques rendent les moteurs BLDC particulièrement adaptés aux équipements nécessitant un fonctionnement continu, comme les lignes de remplissage automatiques dans l'industrie agroalimentaire, les systèmes de convoyage, les machines de dosage, ainsi que les machines de conditionnement et d'emballage. Ils sont également utilisés dans les machines à laver, les climatiseurs, les ventilateurs, les aspirateurs et les disques durs, où leur efficacité, leur fiabilité et leur longévité apportent des bénéfices majeurs. [27]

II.4.2.2.4. Moteur pas à pas

Les moteurs pas à pas sont des moteurs contrôlés numériquement utilisés pour un positionnement précis. Ils permettent un contrôle simple et précis de l'angle de rotation et de la vitesse de rotation, ce qui les rend adaptés à une grande variété d'applications. Moteur oriental a intégré des moteurs pas à pas hybrides dans sa gamme de produits pour offrir des niveaux de précision et de performance supérieurs par rapport aux autres moteurs pas à pas. Ces moteurs ont été utilisés dans de nombreuses applications, allant des équipements industriels à l'automatisation de bureau. [28]

II.4.2.2.4.1 Principe de fonctionnement

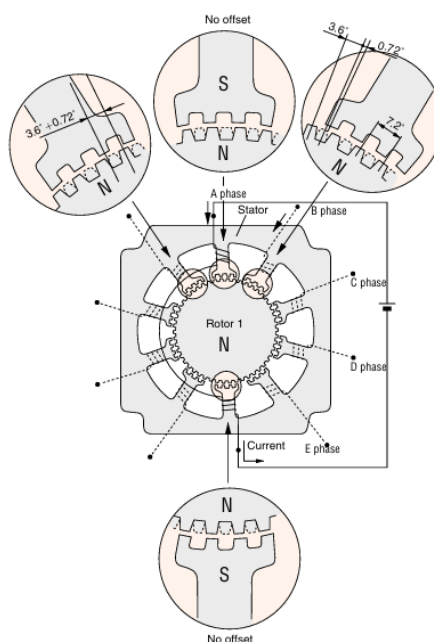


Figure II.44 Principe du fonctionnement d'un moteur pas à pas (excitation de la phase A)

Lorsque la phase A est excitée dans la figure **II.44**, ses pôles sont magnétisés au sud et attirent les dents du rotor 1, qui sont magnétisées au nord, tout en repoussant les dents du rotor 2, qui sont magnétisées au sud, ce qui l'équilibre pour s'arrêter. Les dents des pôles de la phase B, qui ne sont pas excités, sont désalignées par rapport aux dents magnétisées au sud du rotor 2, de sorte qu'elles sont décalées de $0,72^\circ$.

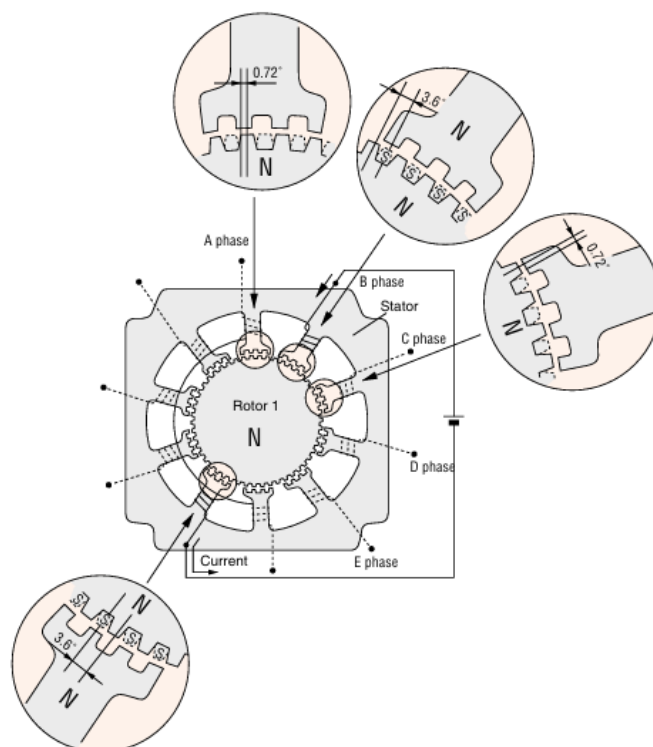


Figure II.45 Principe du fonctionnement d'un moteur pas à pas (excitation de phase B)

Lorsque l'excitation passe de la phase A à la phase B dans la figure II.45, les pôles de la phase B sont magnétisés au nord, attirant la polarité sud du rotor 2 et repoussant la polarité nord du rotor 1. En d'autres termes, lorsque l'excitation passe de la phase A à la phase B, le rotor tourne de $0,72^\circ$. Au fur et à mesure que l'excitation passe de la phase A à la phase B, puis à la phase C, à la phase D, à la phase E et à nouveau à la phase A, le moteur pas à pas effectue des rotations précises de $0,72^\circ$ à chaque étape. Pour le faire tourner en sens inverse, inversez l'ordre d'excitation : phase A, phase E, phase D, phase C, phase B, phase A. La haute résolution ($0,72^\circ$) est obtenue grâce au décalage mécanique entre les structures du stator et du rotor, ce qui permet un positionnement précis sans utiliser d'encodeur ou d'autre capteur. Les seuls facteurs susceptibles de réduire la précision d'arrêt sont les variations de précision de traitement, de précision d'assemblage et de résistance continue de la bobine. Une précision d'arrêt élevée de 3 minutes d'arc (sans charge) est réalisable. Le rôle de commutation des phases est assuré par le driver, dont le timing est fourni par le signal d'impulsion entrant dans le driver. Dans l'exemple ci-dessus, l'excitation se fait une phase à la fois, mais pour une utilisation optimale des bobines, quatre ou cinq phases devraient être excitées simultanément. [28]

II.4.2.2.4.2 La structure d'un moteur pas à pas

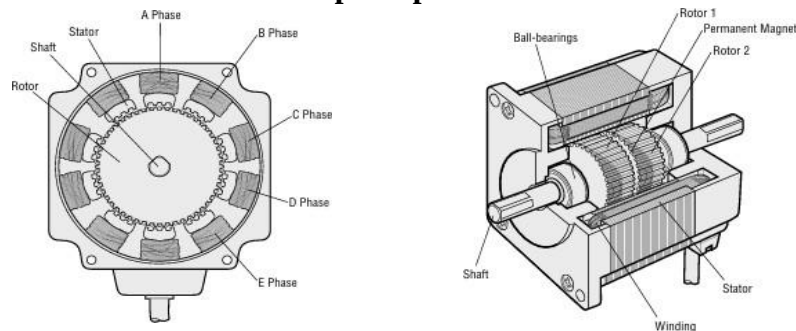


Figure II.46 Les composants d'un moteur pas à pas (5 phases)

La figure II.46 montre deux coupes transversales d'un moteur pas à pas hybride à 5 phases.

Les moteurs pas à pas hybrides sont principalement composés de deux parties : le stator et le rotor. Le rotor, quant à lui, est constitué de trois composants : rotor 1, rotor 2 et l'aimant permanent. Les rotors sont magnétisés dans la direction axiale, avec le rotor 1 polarisé au nord et le rotor 2 polarisé au sud. Le stator contient 10 pôles magnétiques avec de petites dents, chacun d'entre eux étant enroulé de fil pour former une bobine. La bobine est connectée au pôle magnétique opposé et est enroulée de manière à devenir magnétisée vers le même pôle lorsque du courant y circule. (Le passage d'un courant dans une bobine donnée magnétise les pôles opposés avec le même magnétisme, soit un pôle nord, soit un pôle sud.) Les deux pôles opposés forment une seule phase. Étant donné qu'il y a cinq phases (A à E), ce moteur est appelé un moteur pas à pas à 5 phases. Le rotor présente 50 dents à l'extérieur, avec les dents du rotor 1 et du rotor 2 décalées mécaniquement d'un demi-pas.

❖ Caractéristiques :

Les moteurs pas à pas sont compacts mais génèrent un couple élevé. Cela permet une excellente accélération et des mouvements rapides.

Les moteurs pas à pas produisent un couple de maintien élevé même à l'arrêt. La position d'arrêt peut être maintenue sans avoir besoin d'un frein mécanique.

Les moteurs pas à pas à 5 phases effectuent une rotation de $0,72^\circ$ à chaque impulsion et les moteurs pas à pas à 2 phases : Il effectue une rotation de $1,8^\circ$ à chaque impulsion. La distance angulaire parcourue correspond au nombre d'impulsions entrées.

II.4.2.2.5. Les applications des moteurs monophasés

Les moteurs monophasé résistive, capacitive, capacitive permanent, brushless DC ou pas à pas sont parfait pour les applications qui n'est besoin pas des efforts brule, alors, l'application de ces types du moteurs monophasé est une bonne décision pour les machines des remplissages automatique.

II.4.2.3. Les Moteurs triphasés

Un moteur triphasé est un moteur électrique qui fonctionne avec une alimentation électrique triphasée, utilisant trois courants alternatifs pour générer un champ magnétique rotatif. Comparés aux moteurs monophasés, les moteurs triphasés sont plus efficaces et puissants, ce qui les rend idéaux pour les applications industrielles qui demande puissance plus élevé d'un moteur monophasé comme les convoyeurs dans une ligne de remplissage. [29]

II.4.2.3.1. Principe d'une induction magnétique

Les moteurs triphasés fonctionnent selon le principe de l'induction électromagnétique qui stipule que lorsqu'un conducteur tel qu'une bobine ou une boucle de fil est placé dans un champ magnétique variable, une force électromotrice (FEM) induite est générée dans le conducteur. Il a également observé que le courant circulant dans un conducteur tel qu'un fil génère un champ magnétique et que ce champ magnétique variera à mesure que le courant dans le fil change en magnitude ou en direction. Cela s'exprime sous forme mathématique en reliant le rotationnel du champ électrique au taux de variation temporelle du flux magnétique :

$\nabla \times \vec{E}$: Ce symbole est appelé le "rotationnel" du champ électrique $\vec{E}$. Il mesure la rotation ou la

$$\vec{\nabla} \times \vec{E} = - \frac{\partial \vec{B}}{\partial t}$$

force de tourbillon du champ électrique en un point donné.

$\vec{E}$: Il s'agit du vecteur du champ électrique, qui représente la force électrique par unité de charge en tout point de l'espace.

$-\partial \vec{B} / \partial t$: Ce terme représente la dérivée partielle négative du champ magnétique $\vec{B}$ par rapport au temps.

$\vec{B}$: le vecteur du champ magnétique, qui représente l'influence magnétique des courants électriques et des matériaux magnétiques.

$\partial \vec{B} / \partial t$: Ce symbole de dérivée partielle indique le taux de variation du champ magnétique par rapport au temps.

II.4.2.3.2. Principe du fonctionnement d'un moteur asynchrone triphasé

En général, les moteurs asynchrones sont largement utilisés dans les systèmes de convoyage en raison de leur robustesse, de leur rentabilité et de leur capacité à gérer efficacement des charges et des vitesses variables.

❖ Onde sinusoïdale triphasée:

L'énergie électrique fournie aux foyers est monophasée, ce qui signifie qu'il y a un conducteur porteur de courant ainsi qu'une connexion neutre et une connexion de mise à la terre. Dans l'énergie triphasée, utilisée dans les environnements industriels et commerciaux pour faire fonctionner des machines plus grandes ayant des besoins en puissance plus importants, il y a trois conducteurs de courant électrique, chacun fonctionnant avec un déphasage de 120 degrés. Si on le visualise graphiquement, chaque phase apparaîtrait comme une onde sinusoïdale séparée, qui se combine ensuite comme illustré dans l'image ci-dessous

:

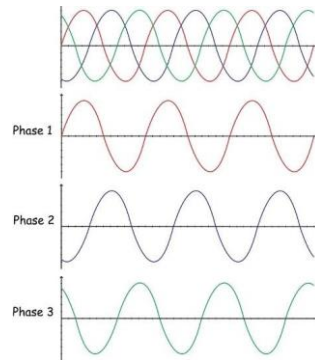


Figure II.47 Onde d'un system triphasé

❖ La force du champ magnétique et le glissement

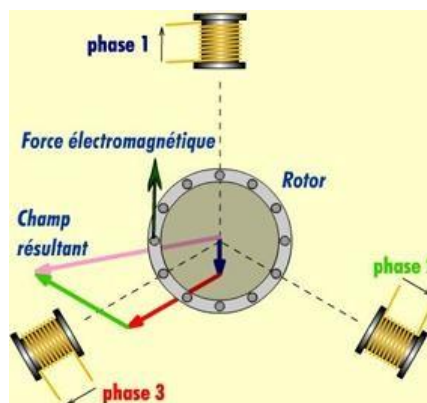


Figure II.48 La force électromagnétique

Pour les moteurs à induction, lorsque le stator est alimenté par une source d'énergie électrique triphasée, chaque bobine génère un champ magnétique dont les pôles (nord ou sud) changent de position lorsque le courant alternatif oscille à travers un cycle complet. Étant donné que chacune des trois phases du courant alternatif est déphasée de 120 degrés, la polarité magnétique des trois bobines n'est pas identique à un instant précis. Cette condition conduit à ce que le stator produise ce qui est appelé un champ magnétique tournant ou RMF (Champ de Rotation Magnétique). Alors que le rotor se trouve au centre des bobines du stator, le champ magnétique changeant du stator induit un courant dans les bobines du rotor, ce qui à son tour

résulte en un champ magnétique opposé généré par le rotor. Le champ du rotor cherche à aligner sa polarité avec celle du champ du stator, ce qui entraîne l'application d'un couple net sur l'arbre du moteur et son début de rotation alors qu'il cherche à aligner son champ. Notez que dans le moteur à induction triphasé, il n'y a pas de connexion électrique directe au rotor ; la rotation du moteur est causée par l'induction magnétique.

Avec les moteurs à induction triphasés, le rotor cherche à maintenir un alignement avec le RMF du stator, mais ne parvient jamais à le réaliser, c'est pourquoi les moteurs à induction sont également appelés moteurs asynchrones. Le phénomène qui cause le retard de vitesse du rotor par rapport à la vitesse du RMF est appelé glissement, et il est exprimé comme suit :

$$\text{Slip} = \frac{N_s - N_r}{N_s}$$

N_s : La vitesse synchrone du champ rotatif. N_r : speed of the rotor

Pour calculé la vitesse synchrone, il faut applique la loi ci-dessous:

$$N_s = \frac{120 * f}{P}$$

N_s : La vitesse du RMF f : La fréquence appliqué P : nombre des pôles

II.4.2.3.3. Constitution d'un moteur asynchrone triphasé

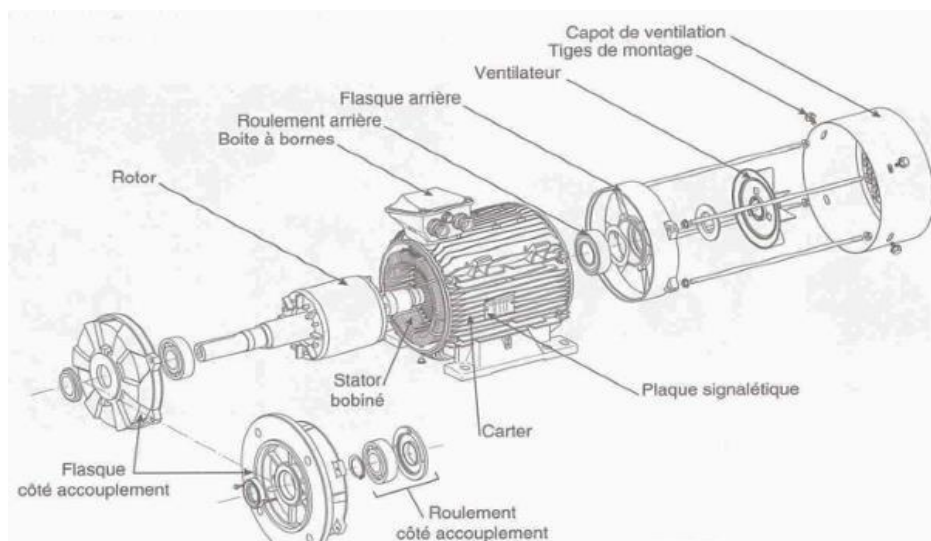


Figure II.49 Les composantes d'un moteur triphasé asynchrone

II.4.2.3.3. Symbole et couplage d'un moteur asynchrone triphasé

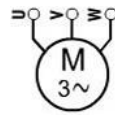


Figure II.50 Symbole d'un MAT

❖ Couplage

Il y a deux types pour coupler un moteur :

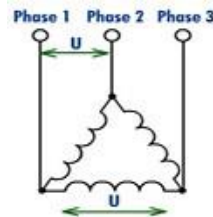


Figure II.51 Triangle- Δ

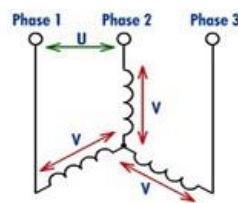


Figure II.52 Étoiles-Y

II.4.2.3.4. Schéma de la commande

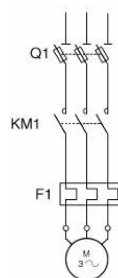


Figure II.53 Schéma de la commande d'un moteur triphasé.

Q1 : disjoncteur

KM1 : relais

F1 : Contacteur

II.4.3. Les réducteurs

II.4.3.1. Définition

Un réducteur est un type de dispositif de transmission de puissance couramment utilisé dans les systèmes mécaniques. Les réducteurs sont utilisés soit pour réduire la vitesse du mouvement de rotation provenant d'un arbre d'entrée à haute vitesse, soit pour le transmettre à un arbre de sortie à vitesse plus basse, ou encore pour générer un couple plus élevé à l'arbre de sortie. Cela est particulièrement utile dans les systèmes de convoyeurs.

Les réducteurs utilisent des roues dentées pour transformer un mouvement de rotation en un autre. Lorsqu'une grande roue dentée sur l'arbre d'entrée interagit avec une petite roue dentée sur l'arbre de sortie, la vitesse diminue tandis que le couple augmente. [30]



Figure II.54 Un réducteur

II.4.3.2. Les avantages et inconvénients d'un réducteur

❖ Avantages

- Une capacité de couple accrue.
- Capacité de fonctionner à des vitesses plus basses. Meilleure capacité à gérer les charges mécaniques. Amélioration de l'efficacité énergétique.
- Ces avantages font des moteurs réducteurs des choix idéaux pour les machines industrielles, les convoyeurs, les chauffages et de nombreuses autres applications.

❖ Désavantages

- Ils peuvent ajouter de la complexité et du poids à un système.
- Ils peuvent nécessiter plus de maintenance.

II.5. Les effecteurs

II.5.1. Définition

En automatisme et en robotique, ce terme désigne généralement un dispositif utilisé pour produire un changement souhaité sur un objet en réponse à certains signaux d'entrée. On l'appelle aussi « actionneur », terme qui décrit plus précisément sa fonctionnalité. Il s'agit d'une fonction fondamentale dans les systèmes artificiels, impliquant la maîtrise du contrôle et du flux d'énergie. [31]

Dans la simulation, cette notion prend tout son sens, puisque la dynamique du système étudié dépend directement de l'action réalisée par l'actionneur en fonction des paramètres d'entrée simulés.

II.5.2. Les types d'effecteurs

Dans notre simulation, on a utilisé un nombre des effecteurs pour simulée une machine de remplissage.

II.5.2.1. Les buses

Les buses sur une machine de remplissage de liquides sont des éléments essentiels qui jouent un rôle déterminant dans l'efficacité et la précision du processus de remplissage. Une conception précise des buses permet d'assurer un dosage exact du produit dans chaque contenant, ce qui contribue à la préservation du produit en évitant les pertes ou les contaminations. Par ailleurs, la polyvalence des buses permet d'adapter la machine à différents types de liquides et à une grande variété de contenants, renforçant ainsi la flexibilité et la performance globale du système de remplissage. [32]



Figure II.55 Les buses utilisées dans les machines de remplissage

II.5.2.2. Les brides

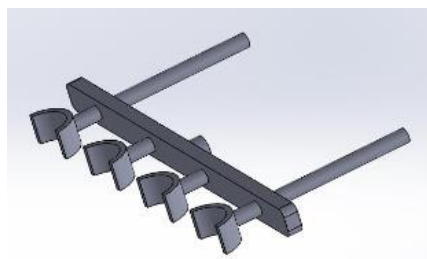


Figure II.56 Les brides d'un système de remplissage des 4 bouteilles

La bouteille est maintenue en position pendant les opérations de remplissage et de bouchage grâce à des brides conçues en forme semi-circulaire, placées face à face. Le mouvement alternatif (va-et-vient) des brides est déclenché par un capteur de proximité capacitif infrarouge. Lorsque le voyant rouge du capteur s'allume (indiquant que la bouteille est en place), les brides sont actionnées pour avancer et fixer la bouteille, permettant ainsi l'opération de remplissage ou de bouchage. [33]

II.5.2.3. Alimenteuse de bouchons

Les systèmes vibrants d'alimentation par bol acheminent progressivement les bouchons du centre vers le bord du bol à l'aide d'une piste hélicoïdale. Seuls les bouchons bien orientés basculent dans le système d'alimentation, tandis que les autres retombent au centre du bol. Les systèmes centrifuges comportent un rotor conique au centre du bol et utilisent la force rotative pour pousser les bouchons vers le bord à grande vitesse. Les bouchons correctement orientés entrent dans la goulotte de distribution, tandis qu'un souffle d'air renvoie les autres au centre. [34]



Figure II.57 Distributeur de bouchons

II.5.2.4. Mandrin à mâchoires



Figure II.58 Assemblage de mandrin à trois mâchoires avec Assemblage de mandrin à trois mâchoires avec limiteur de couple magnétique

L'assemblage mandrin à mâchoires avec limiteur de couple est une pièce de rechange de machine de capsulage, conçue pour garantir une application précise et sécurisée des bouchons. Doté d'un mécanisme de préhension à mâchoires et d'un limiteur de couple, ce mandrin assure un serrage constant, évitant le sur-serrage ou l'endommagement des bouchons. Idéal pour les procédés industriels de remplissage, il améliore l'efficacité et la fiabilité des machines de capsulage automatiques et semi-automatiques. [3]

II.6. Les terminaux IHM

L'IHM (Interface Homme-Machine) permet à l'opérateur d'interagir avec la machine dans l'environnement du procédé de fabrication, en affichant visuellement des données telles que le temps de production et les tendances, offrant ainsi un aperçu de la performance et de l'évolution mécanique. L'IHM facilite également l'interaction directe entre l'humain et la machine, tant pour l'entrée que pour la sortie de données. [36]



Figure II.59 Panneau d'IHM industrielle

II.7. Dispositifs de signalisation

Dans l'automatisation industrielle, les alarmes servent d'indicateurs critiques de la performance du système, des écarts de processus et des dangers potentiels. Un système d'alarme bien conçu aide les opérateurs à identifier et à réagir rapidement aux conditions anormales, évitant ainsi les dommages aux équipements, les pertes de production et les incidents de sécurité. Cependant, une mauvaise gestion des alarmes peut entraîner une surcharge des opérateurs, où un excès d'alarmes ou des alarmes non pertinentes provoquent confusion, avertissements manqués et temps de réaction ralentis.

Pour garantir une gestion efficace des alarmes, les installations industrielles doivent adopter une approche structurée de la conception, de la hiérarchisation et de la réponse aux alarmes. Cet article explore les principes clés de la gestion des alarmes, les pièges courants et les meilleures pratiques pour concevoir un système qui améliore l'efficacité sans submerger les opérateurs.

La fonction principale d'un système d'alarme est d'alerter les opérateurs des conditions anormales nécessitant une attention particulière. Sans une gestion appropriée, les alarmes peuvent devenir envahissantes, menant à des « inondations d'alarmes » où trop d'alertes surviennent simultanément. Dans de telles situations, les opérateurs peuvent avoir du mal à distinguer les alarmes critiques des notifications mineures, ce qui augmente le risque d'avertissements manqués et de réponses retardées.

Une mauvaise gestion des alarmes peut entraîner de graves conséquences opérationnelles et de sécurité, notamment :

- Stress et fatigue accrus chez les opérateurs
- Dommages aux équipements dus à des réponses ignorées ou retardées
- Fiabilité et performance du système réduites
- Risque élevé d'incidents de sécurité et de non-conformité réglementaire

Un système d'alarme bien structuré permet aux opérateurs de se concentrer sur les problèmes les plus critiques tout en minimisant les distractions causées par des alarmes inutiles ou répétitives. [37]



Figure II.60 Des alarmes dans un site industriel

II.8. Conclusion

Dans ce chapitre, nous avons présenté l'ensemble des équipements de la partie opérative d'une ligne de remplissage automatique et leur principe de fonctionnement, du capteur jusqu'aux pré-actionneurs, actionneurs et effecteurs. Nous avons également intégré les terminaux IHM et les dispositifs de signalisation, afin d'illustrer leur rôle dans la supervision et la sécurité du système simulé, comme nous le verrons au niveau du prochain chapitre.

Chapitre III

Partie Commande d'une Ligne de Remplissage Automatique

III.1. Introduction

Dans ce chapitre, nous examinerons la section de commande d'une ligne de remplissage automatisée, son rôle dans le processus, ainsi que les programmes utilisés.

III.2. Automatisation

L'automatisation consiste à rendre automatiques des opérations qui nécessitaient auparavant une intervention humaine. Dans l'industrie, l'automatisation est devenue indispensable car elle permet l'exécution quotidienne des tâches les plus fastidieuses, répétitives et dangereuses. Elle se concentre sur l'étude du contrôle des systèmes industriels. Les techniques et méthodes d'automatisation évoluent continuellement, intégrant des technologies telles que l'électromécanique, l'électronique, la pneumatique et l'hydraulique. L'automatisation est présente dans tous les secteurs d'activité. [39]

III.3. Objectifs

- Augmenter la productivité du système (quantité de produits fabriqués dans un temps donné).
- Améliorer la flexibilité de la production.
- Améliorer la qualité des produits.
- S'adapter à des contextes spécifiques :
 - Environnements hostiles pour les humains.
 - Tâches difficiles nécessitant un effort physique ou intellectuel pour les humains.
- Renforcer la sécurité.

III.4. Structure d'un système automatisé [39]

Les systèmes automatisés utilisés dans les lignes de remplissage industrielles lourdes se composent de trois sections :

- Partie opérative.
- Partie commande (contrôle).
- Partie supervision.

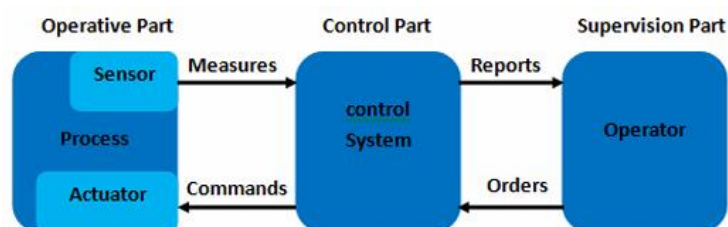


Figure III.1 Structure d'un système automatisé.

III.4.1. Partie opérative

Il s'agit de l'aspect mécanique et électromécanique de la ligne de remplissage, qui se compose principalement de deux types de machines qui ont été discutés et expliqués en détail dans le chapitre précédent :

- Pré-actionneurs.
- Actionneurs.

III.4.2. Partie commande (contrôle)

Elle donne des ordres de fonctionnement à la partie opérative. Elle est considérée comme le "cerveau" du système. La partie commande remplace l'opérateur en fournissant des ordres à la partie opérative en fonction du :

- Code de programme.
- Informations reçues par les capteurs.
- Instructions données par l'utilisateur.

III.4.3. Partie IHM

Composée de panneaux de commande et de consoles de signalisation, elle permet à l'opérateur de contrôler le système (démarrage, arrêt, lancement de cycle, ...etc.). Elle offre également une visualisation des différents états du système via des indicateurs, des terminaux de dialogue ou des interfaces homme-machine (IHM).

III.5. API

Un automate programmable industriel (API), en anglais, Programmable Logic Controller (PLC) est un dispositif électronique programmable conçu pour contrôler des processus industriels par un traitement séquentiel. Il envoie des ordres aux pré-actionneurs (partie opérationnelle du côté des actionneurs) en fonction des données d'entrée provenant des capteurs (partie commande du côté des capteurs). [39]

III.5.1. Structure d'un API

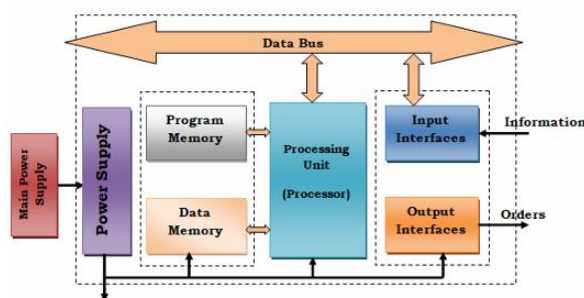


Figure III.2 Structure interne d'un API.

1. Module d'alimentation : Il assure la distribution de l'alimentation aux différents modules. Il est essentiel car il convertit un courant alternatif (AC) en une basse tension continue (24V) nécessaire aux processeurs et aux autres modules d'entrée-sortie.



Figure III.3 Module de puissance d'un API.

2. Unité Centrale de Traitement (CPU) : Il s'agit de l'assemblage électrique qui exécute les tâches dans la mémoire du programme. Elle contient le microprocesseur et réalise toutes les fonctions de traitement logiques, arithmétiques, et numériques (transfert, comptage, temporisation, etc.).

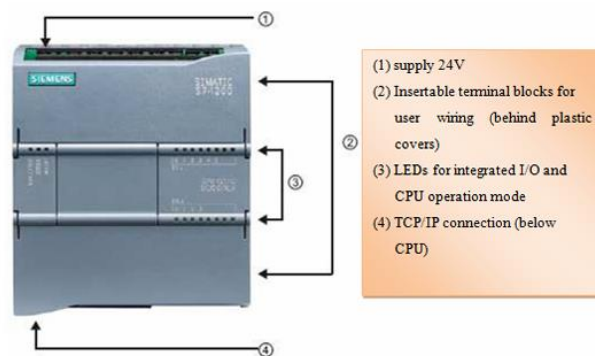


Figure III.4 Unité centrale de traitement d'un API.

3. Bus Interne : Il s'agit d'un ensemble de conducteurs qui établissent la connexion entre les divers éléments du système d'API et les extensions possibles. Le bus est organisé en plusieurs sous-ensembles, chacun destiné à transporter un type d'information spécifique.
4. Zone de Mémoire : Elle est responsable des fonctions suivantes :
 - Recevoir des informations des capteurs d'entrée.
 - Recevoir des informations générées par le processeur et destinées à contrôler les sorties (valeurs de comptage, valeurs de temporisation, etc.).
 - Recevoir et stocker le programme de processus.

5. Interfaces d'Entrée/Sortie : Ces interfaces permettent au processeur de recevoir et d'envoyer des informations aux dispositifs externes.
 - Interface d'Entrée : Elle permet de recevoir des informations logiques provenant des capteurs et du bouton-poussoir de la console connectés à l'Automate Programmable Industriel (API).
 - Interface de Sortie : Elle permet à l'unité centrale de traitement de transmettre des ordres aux divers pré-actionneurs (relais, électrovannes, etc.) et aux éléments de signalisation (indicateurs) sur le panneau tout en assurant une isolation électrique.
 - En général, il existe deux types de modules d'E/S : les E/S numériques (DI/DO) et les E/S analogiques (AI/AO).

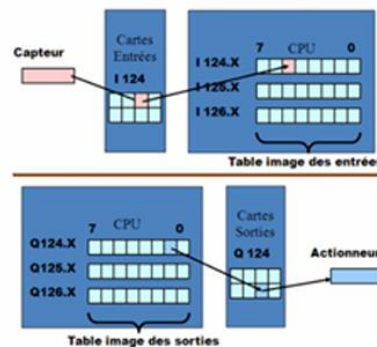


Figure III.5 E/S interfaces.

III.5.2. Cycle de balayage [40]

L'API adopte un mode de traitement cyclique impliquant une centralisation des entrées et sorties, et une exécution périodique et répétée du programme utilisateur. Le temps nécessaire pour accomplir un cycle complet est appelé cycle de balayage.

À chaque cycle, le processeur central (CPU) exécute le programme à partir de la première instruction, et continue séquentiellement jusqu'à la dernière. En l'absence d'instruction de saut, ce processus est continu : le programme revient automatiquement au début pour un nouveau cycle.

Outre l'exécution du programme utilisateur, le CPU réalise, à chaque cycle :

- Le traitement interne.
- L'échantillonnage des entrées.
- La gestion des communications.
- L'autodiagnostic.
- Le rafraîchissement des sorties.

III.5.2.1. Initialisation et autodiagnostic

Le processus global de fonctionnement d'un API comprend trois phases principales :

- Traitement à la mise sous tension.

- Processus de balayage.
- Traitement des erreurs.

À la mise sous tension, le CPU exécute une série de traitements internes, sous la direction du programme système, incluant :

- L'initialisation du matériel.
- La vérification de la configuration des modules E/S.
- La configuration de la mémoire de sauvegarde en cas de panne.
- et d'autres tâches d'initialisation.

L'API dispose d'un puissant mécanisme d'autodiagnostic. À chaque cycle, des vérifications sont réalisées pour s'assurer du bon fonctionnement :

- de l'alimentation.
- des composants matériels internes.
- et de la validité syntaxique du programme.

En cas d'anomalie, une LED s'allume sur le panneau de commande, un relais d'erreur est activé, et un code d'erreur est enregistré dans un registre spécial. Selon le type d'erreur, l'API peut suspendre l'exécution du programme et passer en mode STOP.

III.5.2.2. Durée du cycle de balayage et performance

Lorsque l'API fonctionne normalement, la durée du cycle dépend de :

- La complexité du programme utilisateur.
- La vitesse de calcul du CPU.
- et l'état des points d'E/S.

En général, l'exécution de 1 Ko d'instructions nécessite 1 à 10 ms. Toutefois, le temps d'exécution varie selon l'instruction, ce qui rend la durée du cycle non uniforme. Dans les systèmes à haute vitesse, il est nécessaire d'optimiser le matériel et le logiciel afin de réduire cette durée.

III.5.2.3. Étapes du cycle de balayage

Le cycle de balayage d'un API est divisé en trois phases fondamentales :

1. Phase d'échantillonnage des entrées :

L'API lit les signaux de toutes les bornes d'entrée et stocke leur état dans une mémoire image d'entrée (input image register).

Pendant les phases suivantes, cette mémoire reste isolée : son contenu est inchangé jusqu'au prochain cycle.

2. Phase d'exécution du programme :

Pendant cette phase, le programme utilisateur est exécuté selon une logique séquentielle (de haut en bas, de gauche à droite).

En présence d'une instruction de saut, l'exécution suit l'adresse conditionnelle.

L'automate lit les états :

- Des entrées depuis la mémoire image d'entrée,
- Des relais internes depuis la mémoire image de sortie,

Les résultats sont ensuite stockés dans cette mémoire de sortie.

3. Phase de rafraîchissement des sorties :

Les résultats des calculs sont transmis aux verrous de sortie (*output latches*) qui activent les dispositifs connectés aux sorties. Par exemple, si un relais interne est à l'état logique 1, le contact est fermé et la charge externe est alimentée. L'état de sortie est conservé pendant tout le cycle.

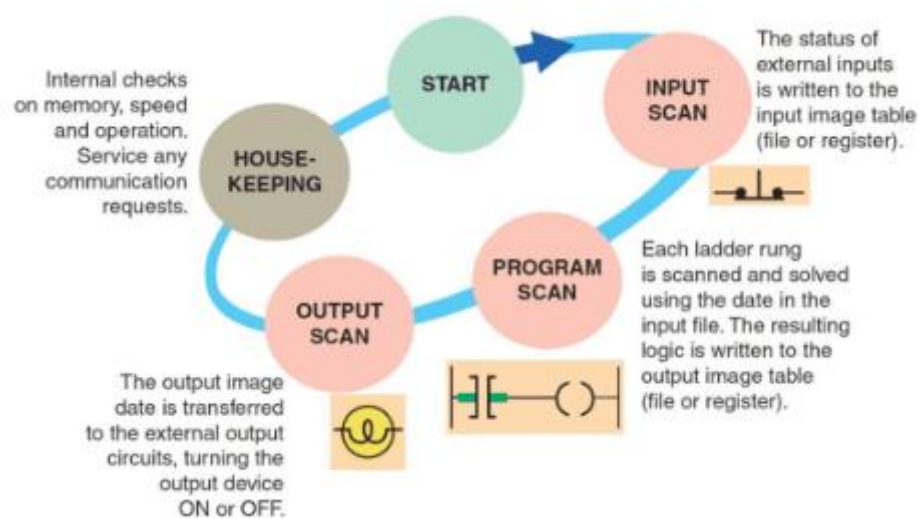


Figure III.6 Etapes du cycle de balayage.

III 6. Langages de programmation selon la norme IEC 61131-3 [41]

Cette section présente les cinq langages de programmation définis par la norme IEC 61131-3, largement utilisés dans l'automatisation industrielle. Il est une norme internationale relative aux langages de programmation des automates programmables industriels. Selon cette norme, un programme comprend les éléments suivants : structures, unités organisationnelles de programme (POUs), variables globales.

III.6.1. Langage IL

Le langage IL (Instruction List) dans CODESYS se compose d'une série d'instructions textuelles, chaque ligne représentant une instruction distincte. Chaque instruction commence par un opérateur et peut être suivie d'un ou plusieurs opérandes, séparés par des virgules, selon le type d'opération. Une étiquette (label) facultative peut précéder l'instruction, suivie d'un deux-points (:).

Un commentaire peut être ajouté à la fin de chaque ligne à l'aide de (* commentaire *). Des lignes vides peuvent être insérées pour améliorer la lisibilité du code.

Exemple:

```
LD 17
ST lint (* comment *)
GE 5
JMPC next
LD idword
EQ istruct.sdword
STN test next
```

III.6.2. Langage ST

Le langage Structured Text (ST) dans CODESYS est un langage de haut niveau semblable au Pascal, permettant d'écrire des blocs d'instructions structurées. Il prend en charge des structures classiques comme IF...THEN...ELSE, des boucles (WHILE...DO, FOR...DO), et des appels de fonctions ou blocs fonctionnels. C'est un langage très adapté pour le traitement logique complexe, les calculs mathématiques, et les structures conditionnelles ou répétitives.

Exemple:

```
IF value < 7 THEN
  WHILE value < 8 DO
    value := value + 1;
  END_WHILE;
END_IF;
```

III.6.3. Langage GRAFCET

Le GRAFCET est un langage de programmation graphique proposé dans CODESYS. Il permet de décrire de façon claire et structurée l'enchaînement chronologique d'actions dans un programme automatisé. Le GRAFCET repose sur des étapes (Steps) et des transitions (Transitions) pour définir la logique séquentielle, ce qui le rend idéal pour modéliser des processus industriels comme les lignes de remplissage, les cycles de production, ou les séquences de démarrage.

Exemple :

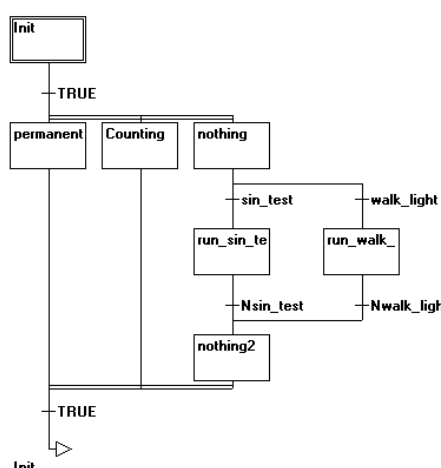


Figure III.7 Exemple d'une programmation avec GRAFCET.

III.6.4. Langage FBD

Normes internationales normalisées par la norme IEC 61131-3 : Chaque API peut être programmé en utilisant une console de programmation propriétaire ou un ordinateur équipé du logiciel spécifique fourni par le fabricant.

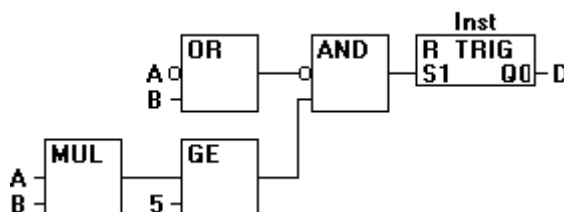


Figure III.8 Exemple d'une programmation avec FBD.

III.6.5. Langage LD

Le Ladder Diagram (LD), ou langage à contacts, est un langage de programmation graphique inspiré directement des schémas électriques de relais. Dans CODESYS, il est largement utilisé pour représenter des opérations logiques de manière visuelle et intuitive, en particulier dans les systèmes automatisés industriels.

D'une part, le LD est particulièrement adapté à la construction de logiques de commutation, comme des conditions de démarrage ou d'arrêt. D'autre part, il permet également de créer des réseaux similaires à ceux du Function Block Diagram (FBD), facilitant ainsi le contrôle des appels d'autres blocs fonctionnels (POU).

Un programme en Ladder Diagram est constitué d'une série de réseaux (networks). Chaque réseau est délimité verticalement par une ligne de courant gauche et droite, entre lesquelles sont placés un circuit logique composé de contacts (normally open/closed), de bobines (coils), et de lignes de connexion.

Le principe de fonctionnement repose sur la propagation de la condition logique de gauche à droite. Sur le côté gauche, une série de contacts transmettent une condition booléenne :

Si la variable associée à un contact est à **TRUE (ON)**, le courant logique passe à droite.

Si elle est à **FALSE (OFF)**, le flux logique est interrompu.

Ainsi, la sortie de chaque réseau (typiquement représentée par une bobine) reçoit la valeur logique en fonction des états des contacts précédents. Le LD est donc un excellent choix pour la visualisation directe de la logique de contrôle.

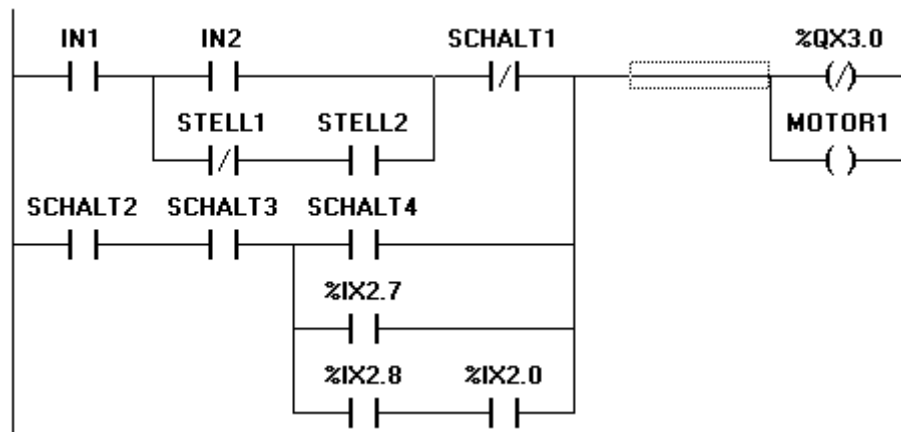


Figure III.9 Exemple d'une programmation avec LD.

III.7. GRAFCET dans les systèmes d'automatisation industrielle

III.7.1. Utilisation du GRAFCET dans l'automatisation

Les GRAFCET sont des langages de programmation visuels, proches des organigrammes utilisés en algorithmique. Ils sont largement employés dans les domaines de l'automatisation industrielle et de la simulation pour modéliser des systèmes comportant plusieurs états ou séquences d'opérations.

➤ Pourquoi il est utilisé :

- Visualisation intuitive : la représentation graphique facilite la compréhension des processus complexes, même pour des non-programmeurs.
- Décomposition des processus : permet de diviser un système vaste en étapes simples et claires, reliées par des transitions qui définissent le passage d'une étape à une autre.
- Optimisation de la simulation : dans notre projet, l'utilisation du GRAFCET simplifie la conception et l'analyse de scénarios séquentiels ou parallèles, rendant le comportement du système plus accessible.
- Débogage et maintenance facilités : il est aisé d'identifier et de suivre l'étape active lors de l'exécution, ce qui accélère la résolution des problèmes.

- Modularité : les séquences peuvent être réutilisées sous forme de blocs ou sous-routines, favorisant la flexibilité et l'évolutivité du système.
- Fiabilité accrue : la structure séquentielle et visuelle réduit les erreurs de logique, ce qui est primordial dans les applications critiques du secteur industriel.

En résumé, l'approche GRAFCET offre un cadre adapté à la simulation et au contrôle de processus industriels complexes, et constitue une solution privilégiée dans notre domaine de recherche et de développement. [42]

III.7.2. Éléments fondamentaux d'un diagramme GRAFCET

Chaque programme GRAFCET est composé de plusieurs blocs fonctionnels principaux :

➤ Éléments :

- Étapes (Steps) : Représentent des états stables du système (exemples: pompe activée, vanne fermée, ...etc.).
- Actions : Instructions associées à chaque étape (exemple: 'MOTOR := TRUE').
- Transitions : Conditions logiques qui déclenchent le passage à l'étape suivante.
- Sauts (Jumps) : Permettent de retourner à une étape antérieure (boucles).
- Quantificateurs d'action : Contrôlent la durée d'exécution d'une action (N, P, S, R, ...etc.).

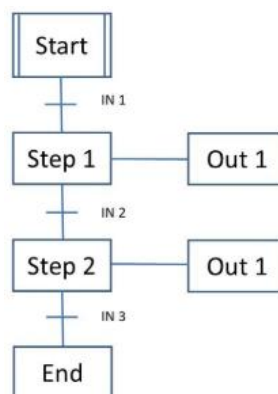


Figure III.10 Les éléments principaux d'une GRAFCET.

III.7.3. Références industrielles

De nombreux logiciels industriels, tels que CODESYS, Siemens TIA Portal, ou encore Schneider Electric EcoStruxure, implémentent pleinement le GRAFCET dans leur environnement de développement. Cela confirme son importance dans les pratiques professionnelles réelles.

Dans notre projet, le choix du GRAFCET est justifié par :

- La séquentialité du processus de remplissage.
- La nécessité de visualiser chaque phase du système.
- La possibilité d'implémentation fluide dans CODESYS.

III.8. Conclusion

Dans ce chapitre, nous avons présenté les principaux langages de programmation utilisés dans l'automatisation, avec un accent particulier sur le GRAFCET, choisi pour la simulation à venir. Nous avons justifié son utilisation, expliqué son intégration avec d'autres langages comme LD et ST, et détaillé le fonctionnement interne des automates. Ces éléments serviront de base technique pour la simulation de notre machine de remplissage.

Chapitre IV

Programmation et Simulation de la Remplisseuse Automatique

IV.1. Introduction

Nous proposons dans ce chapitre de faire la programmation et la simulation d'une machine de remplissage automatique de liquides à l'aide du logiciel CODESYS. Les langages de programmation utilisés sont GRAFCET (SFC) et ST.

IV.2. Logiciel CODESYS

CODESYS est un environnement de développement intégré (IDE) destiné à la programmation d'applications pour automates conformément à la norme industrielle internationale IEC 61131-3. [43]

Il prend en charge cinq langages de programmation principaux, normalisés et utilisés dans l'automatisation industrielle :

- ST
- LD
- FBD
- SFC
- CFC



Figure IV.1 Logo du logiciel CODESYS

IV.3 Simulation et programmation d'une machine de remplissage automatique

IV.3.1. Les Moteurs Utilisés

1. PumpMotor1 (Pompe de remplissage cuve)
 - Type : Moteur asynchrone triphasé IEC (TEFC, 4 pôles)
 - Tension nominale : 400 V AC (triphasé), 50 Hz
 - Puissance nominale : 0,55 kW
 - Courant nominal : 1,35 A
 - Charge maximale en service : 1,50 A
 - Seuil de surcharge : 1,60 A
 - Vitesse nominale : 1 450 tr/min
 - Couple nominal : $\approx 3,6$ Nm

2. PumpMotor2 (Pompe de remplissage bouteille)

- Type : Moteur asynchrone triphasé IEC (commandé par variateur de vitesse)
- Tension nominale : 400 V AC (triphasé), 50 Hz
- Puissance nominale : 0,37 kW
- Courant nominal : 1,25 A
- Charge maximale en service : 1,50 A
- Seuil de surcharge : 1,60 A
- Vitesse nominale : 1 450 tr/min (utilisation typique à 75 %)
- Couple nominal : $\approx 2,4$ Nm

3. ConveyorMotor (Convoyeur principal)

- Type : Moteur asynchrone triphasé avec réducteur
- Tension nominale : 400 V AC (triphasé), 50 Hz
- Puissance nominale : 0,25 kW
- Courant nominal : 1,05 A
- Charge maximale en service : 1,20 A
- Seuil de surcharge : 1,30 A
- Vitesse de sortie : 70–80 tr/min (après réducteur)
- Couple nominal (sortie) : ≈ 30 Nm

4. StepperMotor (Mandrin de serrage du bouchon)

- Type : Moteur pas à pas bipolaire avec driver numérique
- Tension nominale : 48 V DC
- Courant nominal : 4,2 A
- Puissance nominale : 200 W
- Nombre de pas par tour : 200 pas (angle de pas : $1,8^\circ$)
- Couple nominal : 2,2 Nm
- Couple maximal (à basse vitesse) : 2,5 Nm
- Vitesse de rotation typique : 0–300 tr/min

5. SideConveyor_Left / SideConveyor_Right (Convoyeurs latéraux étiquetage)

- Type : Moteur asynchrone triphasé compact (ou option moteur DC)
- Tension nominale : 400 V AC (triphasé), 50 Hz
- Puissance nominale : 0,12 kW
- Courant nominal : 0,70 A
- Charge maximale en service : 0,80 A
- Seuil de surcharge : 0,90 A
- Vitesse nominale : 1 450 tr/min (réduite)
- Couple utile : ≈ 2 –3 Nm

6. LabelTipMotor (Moteur de présentation du bord d'étiquette)

- Type : Moteur brushless DC (BLDC)
- Tension nominale : 24 V DC
- Puissance nominale : 40 W
- Courant nominal : 0,45 A
- Charge maximale en service : 0,50 A
- Seuil de surcharge : 0,60 A
- Vitesse nominale : 3 000 tr/min
- Couple nominal : $\approx 0,13$ Nm

7. CapFeederMotor (Moteur d'alimentation des bouchons)

- Type : Moteur brushless DC (BLDC) avec réducteur
- Tension nominale : 24 V DC
- Puissance nominale : 60 W
- Courant nominal : 0,80 A
- Charge maximale en service : 0,90 A
- Seuil de surcharge : 1,00 A
- Vitesse de sortie : 100 tr/min (après réducteur)
- Couple de sortie : $\approx 4\text{--}5$ Nm

IV.3.2. Création du projet

Afin de créer un nouveau projet, nous devons suivre ces étapes:

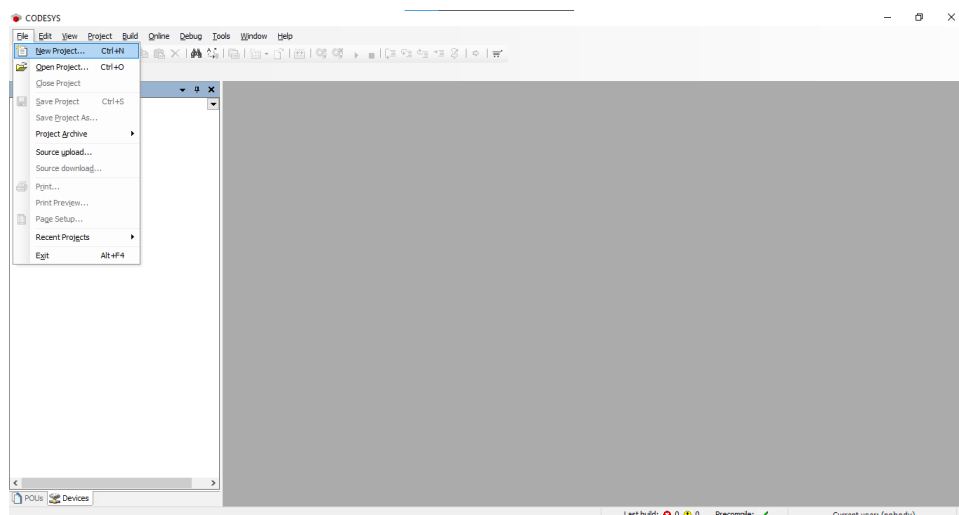


Figure IV.2 Création d'un nouveau projet

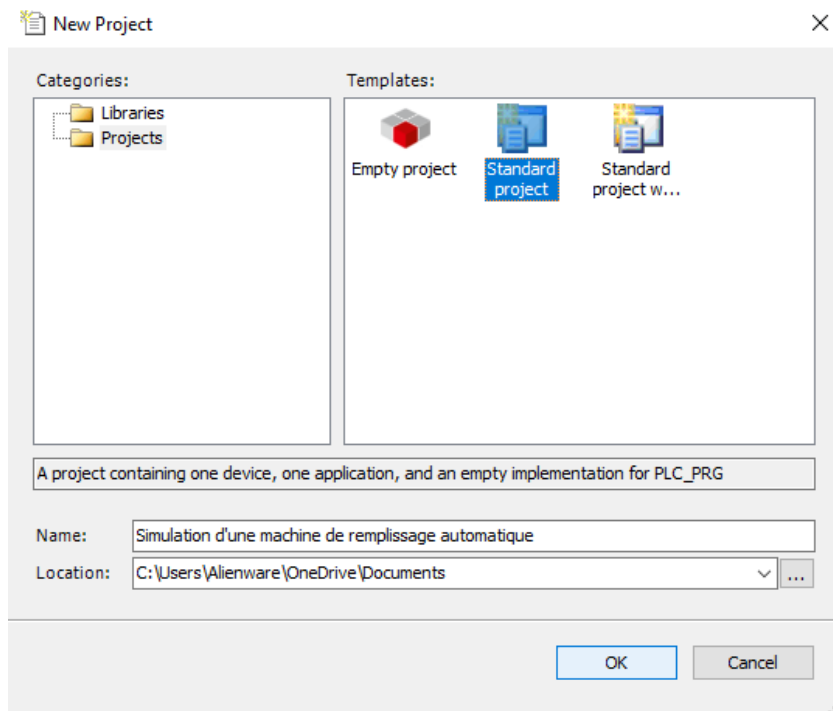


Figure IV.3 Nommer le projet

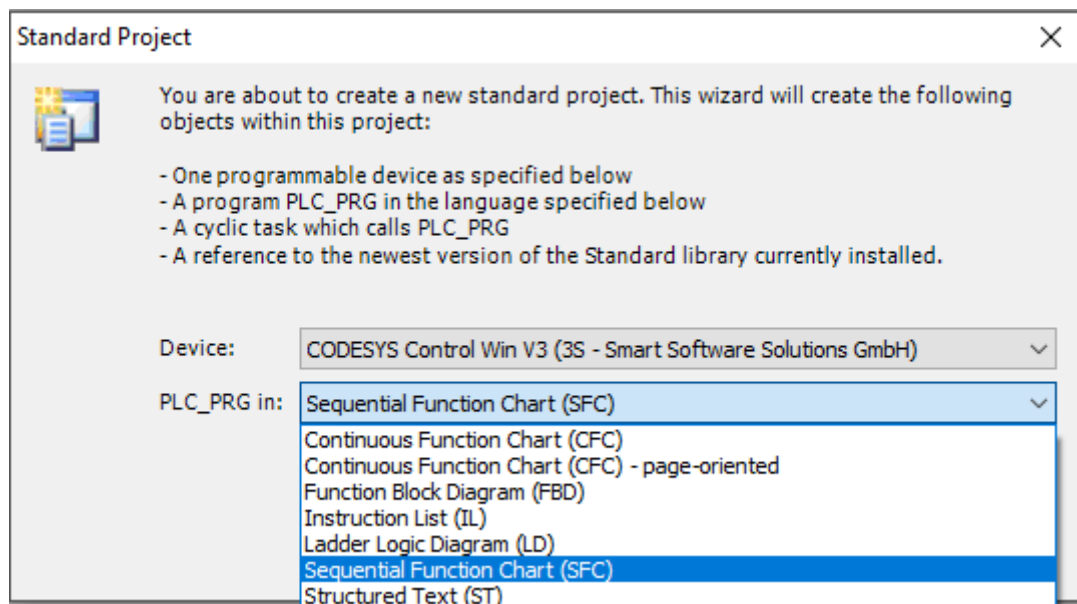


Figure IV.4 Choisir le langage de programmation

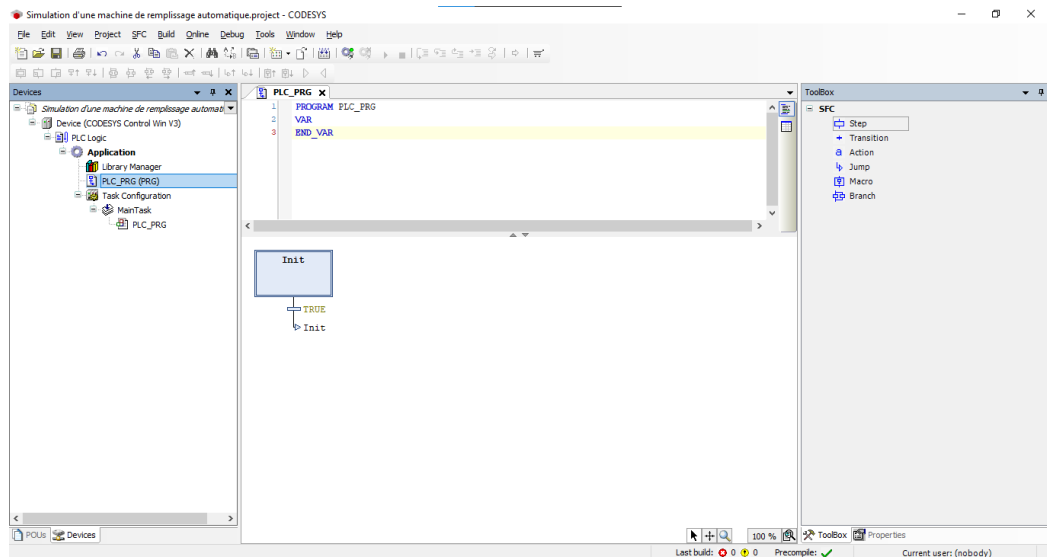


Figure IV.5 La fenêtre principale de notre projet.

Dans notre programme, nous aurons de nombreuses étapes différentes à travers la machine :

Étape	Nom de l'Étape	Fonction principale	Fonctionnalités clés
1	Start	Lancer le cycle et initialiser les composants	Vérification des capteurs initiaux, remise à zéro des alarmes, mise en attente de la commande de démarrage.
2	TankCheck	Surveillance du niveau de cuve	Lecture capteurs, contrôle pompe/voyants, gestion temporisations et défauts, affichage IHM
3	BottleReady	Détection et gestion de l'arrivée bouteille	Lecture capteur avec debouncing, contrôle porte/convoyeur, Minuteur s de porte et de défaut, gestion d'alarme/IHM
4	BottleFillingReady	Préparer les bouteilles pour le remplissage.	Positionnement des bouteilles, vérification de la présence, activation du convoyeur ou des vérins de maintien si nécessaire.
5	BottleFilling	Assurer le remplissage précis des bouteilles.	Contrôle du débit de la pompe, surveillance du niveau de remplissage, arrêt automatique à la fin, gestion des débordements.

6	CappingReady	Placer les bouteilles sous la tête de capsulage et préparer le bouchon.	Positionnement sous le système de capsulage, vérification de la présence du bouchon, mise en sécurité si problème détecté.
7	JawReady	Préparer et engager la mâchoire pour saisir le bouchon.	Activation de l'air comprimé pour la mâchoire, détection de la prise du bouchon, gestion des défauts de mécanisme.
8	ChuckRotation	Visser le bouchon sur la bouteille avec précision.	Contrôle du moteur pas-à-pas pour la rotation, surveillance de la complétion du vissage, gestion des erreurs mécaniques.
9	Labeling	Appliquer l'étiquette sur la bouteille de façon correcte.	Positionnement de la bouteille, gestion du déroulement de l'étiquette, contrôle de la coupe, vérification de l'application.
10	QualityControl	Contrôler la qualité finale du produit avant la sortie.	Vérification de la présence du bouchon et de l'étiquette, détection de défauts (poids, forme), activation des alarmes QC.

IV.3.3. Types des variables utilisés

```

1  PROGRAM PLC_FRG
2  VAR_INPUT
3      Sensor_Tank_Low      : BOOL;
4      Sensor_Tank_Mid      : BOOL;
5      Sensor_Tank_High     : BOOL;
6      FeederBottle_Sensor  : BOOL;
7      Pump_Speed_Setpoint  : REAL := 75.0;
8      SimulateFeedback     : BOOL := FALSE;
9      SupervisorResetRequested : BOOL := FALSE;
10     FillingConveyorSensor : BOOL;
11     AlarmLight_ManualOff  : BOOL;
12     Buzzer_ManualOff      : BOOL;
13     Operator_Resume_Button : BOOL;
14     FullBottleSensor      : BOOL;
15     NozzlePressureSensor  : BOOL;
16     OperatorResetRequested : BOOL;
17     CappingConveyorSensor : BOOL;
18     ChuckDownSensor       : BOOL;
19     JawClutchSensor       : BOOL;
20     StepperCappingDoneSensor : BOOL;
21     CappingRotationAmount : REAL;
22     BottleLabellingPositionSensor : BOOL := FALSE;
23     LabelPositionSensor   : BOOL := FALSE;
24     LabelGapSensor        : BOOL := FALSE;
25     VisionLabelPresent    : BOOL := FALSE;
26     VisionCapPresent      : BOOL := FALSE;
27     WeightSensorValue     : REAL := 0.0;
28     BottleShapeOK         : BOOL := FALSE;
29     BottleArrivedQC       : BOOL := FALSE;
30     LabelTipSensor        : BOOL := FALSE;
31     BottleQCSensor        : BOOL := FALSE;
32 END_VAR
33 VAR_OUTPUT

```

Figure IV.6 Table des variables

1. BOOL

Description : Variable logique, ne prend que deux états : TRUE (vrai) ou FALSE (faux).

Utilisation : Pour les états, commandes, capteurs, drapeaux (ex : capteur activé, défaut, marche/arrêt).

2. REAL

Description : Nombre à virgule flottante, permet de représenter des valeurs réelles (décimales).

Utilisation : Pour des mesures physiques (vitesse, poids, durée, rotation, consigne...).

3. INT

Description : Nombre entier, sans décimale (ex : -32768 à 32767).

Utilisation : Pour des compteurs, index d'étapes, gestion de boucles simples.

4. STRING[size]

Description : Chaîne de caractères, utilisée pour stocker du texte.

Utilisation : Pour afficher des messages sur l'IHM (ex : M_IHM_StatusText), logs ou diagnostics.

5. TON

Description : Bloc fonction temporisateur (Minuteur ON Delay, selon la norme IEC 61131-3).

Utilisation : Pour la gestion des délais, temporisations, timeouts, séquences temporisées.

Principaux attributs :

- IN : Entrée logique pour démarrer le Minuteur.
- PT : Durée programmée du Minuteur.
- Q : Sortie logique (TRUE lorsque le temps est écoulé).
- ET : Valeur du temps écoulé.

IV.3.4. Fonctionnement normal du programme

IV.3.4.1. Start

Cette étape est nécessaire pour réinitialiser toutes les variables utilisées dans la simulation.

IV.3.4.2. TankCheck

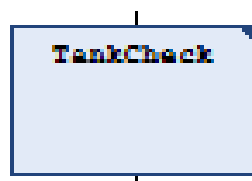


Figure IV.7 Bloc d'étape TankCheck

Ce bloc de code gère la surveillance et le contrôle du niveau d'une cuve à l'aide de trois capteurs (niveau bas, milieu, haut). Il utilise des temporisateurs pour détecter les défauts de capteurs ou de niveau, et commande la pompe ainsi que les voyants lumineux en fonction des états détectés. En cas d'anomalie, le système affiche un message d'erreur sur l'IHM et arrête la pompe pour garantir la sécurité.

1. Code d'action actif :

- Cette section initialise tous les indicateurs et états du processus de remplissage du réservoir. Elle garantit que chaque cycle commence avec des valeurs propres : état des feux de niveau, défauts, états du moteur et des bouteilles, pour une gestion fiable du système.

```
Light_TankLow := FALSE;
Light_TankMid := FALSE;
Light_TankHigh := FALSE;
TankReady := FALSE;
TankFault := FALSE;
MidSensorError := FALSE;
PumpMotor1_OverloadFault := FALSE;
BottleAccepted := FALSE;
BottleRejected := FALSE;
```

Figure IV.8 Initialisation et Flags TankCheck

- Ici, la logique de commande du moteur de la pompe est présentée. L'opérateur peut activer ou désactiver le moteur et ajuster sa vitesse en mode manuel, alors qu'en mode automatique le moteur reste toujours actif et la vitesse est gérée par le programme.

```
IF ManualOverride_pumpmotor1 THEN
    PumpMotor1_State := M_HMI_PumpMotor1_Control;
    IF PumpMotor1_State THEN
        PumpMotor1_SpeedSetpoint := M_HMI_PumpMotor1_Speed;
    ELSE
        PumpMotor1_SpeedSetpoint := 0.0;
    END_IF
ELSE
    PumpMotor1_State := TRUE;
```

Figure IV.9 Commande Manuelle et Vitesse Moteur

- Ce bloc traite les défauts critiques : absence de tous les capteurs de niveau et surcharge du moteur. Des temporisations permettent de confirmer la persistance du défaut avant d'arrêter le remplissage, et un message d'erreur s'affiche pour l'opérateur.

```

IF NOT Sensor_Tank_Low AND NOT Sensor_Tank_Mid AND NOT Sensor_Tank_High THEN
    TankFaultConfirmTimer(IN := TRUE, PT := T#15S);
    IF TankFaultConfirmTimer.Q THEN
        TankFault := TRUE;
        PumpMotor1_SpeedSetpoint := 0.0;
        M_HMI_StatusText := 'ERROR: Tank level sensors not detected!';
        RETURN;
    ELSE
        PumpMotor1_SpeedSetpoint := 0.0;
        M_HMI_StatusText := 'Waiting for tank level sensors...';
        RETURN;
    END_IF
ELSE
    TankFaultConfirmTimer(IN := FALSE);
END_IF

IF PumpMotor1_Absorp_mA > 2000.0 THEN
    PumpMotor1_OverloadFault := TRUE;
    PumpMotor1_SpeedSetpoint := 0.0;
    M_HMI_StatusText := 'ERROR: Pump motor overload detected!';
    RETURN;
END_IF

```

Figure IV.10 Défaut Capteurs et Surcharge Moteur

- Cette partie gère le remplissage initial lorsque le niveau est bas, et l'attente du capteur de niveau intermédiaire. Elle inclut aussi le temps de remplissage supplémentaire lorsque le niveau moyen est atteint, pour garantir que le réservoir soit suffisamment rempli avant de passer à l'étape suivante.

```

IF Sensor_Tank_Low AND NOT Sensor_Tank_Mid AND Sensor_Tank_High THEN
    PumpMotor1_SpeedSetpoint := 100.0;
    Light_TankLow := TRUE;
    Pump_ExtraRun_Timer(IN := FALSE);
    MidSensorWaitTimer(IN := TRUE, PT := T#5M);
    IF MidSensorWaitTimer.Q AND NOT Sensor_Tank_Mid THEN
        PumpMotor1_SpeedSetpoint := 0.0;
        M_HMI_StatusText := 'ERROR: Mid tank sensor did not activate!';
        MidSensorError := TRUE;
        RETURN;
    ELSE
        M_HMI_StatusText := 'Filling the tank...';
        RETURN;
    END_IF
ELSE
    MidSensorWaitTimer(IN := FALSE);
END_IF

IF Sensor_Tank_Low AND Sensor_Tank_Mid AND NOT Sensor_Tank_High THEN
    Light_TankMid := TRUE;
    Pump_ExtraRun_Timer(IN := TRUE, PT := T#5M);
    IF NOT Pump_ExtraRun_Timer.Q THEN
        PumpMotor1_SpeedSetpoint := 100.0;
        M_HMI_StatusText := 'Running extra fill time...';
    ELSE
        PumpMotor1_SpeedSetpoint := 0.0;
        M_HMI_StatusText := 'Waiting for tank high level...';
    END_IF
ELSE
    Pump_ExtraRun_Timer(IN := FALSE);
END_IF

```

Figure IV.11 Niveau Bas et Remplissage Supplémentaire

- Lorsque le capteur de niveau haut est activé, le moteur de la pompe ralentit jusqu'à l'arrêt, le témoin de niveau haut s'allume et le système indique que le réservoir est prêt pour le reste du processus.

```

IF Sensor_Tank_Low AND Sensor_Tank_Mid AND Sensor_Tank_High THEN
    PumpMotor1_SpeedSetpoint := 0.0;
    Pump_ExtraRun_Timer(IN := FALSE);
    Light_TankHigh := TRUE;
    TankReady := TRUE;
    M_HMI_StatusText := 'Tank ready for process!';
ELSE
    TankFault := TRUE;
END_IF
END_IF

```

Figure IV.12 Niveau Haut et Arrêt

- Cette section simule le courant absorbé par le moteur en fonction de la vitesse et transmet toutes les informations essentielles à l'IHM : états des feux, défauts, erreurs et états du moteur. Cela permet à l'opérateur de suivre en temps réel la situation du remplissage.

```

IF PumpMotor1_SpeedSetpoint > 0.0 THEN
    PumpMotor1_Absorp_mA := 1500.0 * (PumpMotor1_SpeedSetpoint / 100.0);
ELSE
    PumpMotor1_Absorp_mA := 0.0;
END_IF

M_HMI_PumpMotor1_Speed := PumpMotor1_SpeedSetpoint;
M_HMI_PumpMotor1_On := PumpMotor1_State;
M_HMI_PumpMotor1_Absorp_mA := PumpMotor1_Absorp_mA;
M_HMI_Light_TankLow := Light_TankLow;
M_HMI_Light_TankMid := Light_TankMid;
M_HMI_Light_TankHigh := Light_TankHigh;
M_HMI_TankReady := TankReady;
M_HMI_TankFault := TankFault;
M_HMI_MidSensorError := MidSensorError;
M_HMI_PumpMotor1_OverloadFault := PumpMotor1_OverloadFault;

```

Figure IV.13 Simulation Courant Moteur et Mapping IHM

2. Transition à la prochaine étape :

```

| TankReady = TRUE AND NOT PumpMotor1_OverloadFault AND NOT TankFault AND NOT MidSensorError

```

Figure IV.14 Transition entre TankCheck et BottleReady

IV.3.4.3. BottleReady

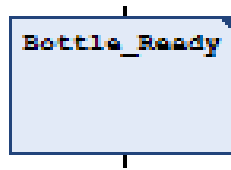


Figure IV.15 Bloc d'étape BottleReady.

Ce bloc de code gère la détection de la présence d'une bouteille à l'entrée du système, à l'aide d'un capteur. Il commande l'ouverture de la porte d'alimentation ainsi que le démarrage du convoyeur lorsque la présence de la bouteille est confirmée. Il utilise des temporisations pour garantir la fiabilité de la détection (debouncing) et pour refermer la porte après un temps donné. En cas d'absence prolongée de la bouteille, il signale un défaut et affiche un message d'erreur sur l'IHM.

1. Code d'action active:

- Cette section initialise toutes les variables principales et les indicateurs d'état pour le processus d'alimentation des bouteilles. Elle permet d'assurer que le système démarre dans une situation sûre et contrôlée.

```
SV_BottleFeederGate := FALSE;  
Conveyor_Motor_SpeedSetpoint := 0.0;  
BottleReady := FALSE;  
BottleFault := FALSE;  
ConveyorMotor_OverloadFault := FALSE;
```

Figure IV.16 Initialisation et indicateurs de alimentation

- Ce bloc permet à l'opérateur de prendre la main sur la vanne d'alimentation et le moteur du convoyeur. En mode manuel, les commandes de l'IHM déterminent directement l'ouverture de la vanne et le fonctionnement du moteur.

```
IF ManualOverride_SV_BottleFeederGate THEN  
    SV_BottleFeederGate := M_HMI_SV_BottleFeederGate_Control;  
ELSE
```

Figure IV.17 Commande manuelle de la vanne et du convoyeur.

- Ici, le signal du capteur de bouteille est filtré pour éviter les fausses détections. Si la présence de la bouteille est confirmée, la vanne s'ouvre pour permettre l'entrée de la bouteille. Des temporisations assurent le bon déroulement de l'action. Et la vitesse du convoyeur est contrôlée automatiquement par la présence de la bouteille.

```

IF FeederBottle_Sensor THEN
  BottleSensorDebounceTimer(IN := TRUE, PT := T#500MS);
  IF BottleSensorDebounceTimer.Q THEN
    BottleReadyDebounced := TRUE;
  END_IF
ELSE
  BottleSensorDebounceTimer(IN := FALSE);
  BottleReadyDebounced := FALSE;
END_IF

IF BottleReadyDebounced THEN
  Conveyor_MotorVFD_Speed := 75.0;
  SV_BottleFeederGate := TRUE;
  BottleFeederGateTimer(IN := TRUE, PT := T#1S);
ELSE
  SV_BottleFeederGate := FALSE;
  BottleFeederGateTimer(IN := FALSE);
END_IF
IF BottleFeederGateTimer.Q THEN
  BottleReady := TRUE;
  SV_BottleFeederGate := FALSE;
END_IF
IF ManualOverride_ConveyorMotor THEN
  ConveyorMotor_State := M_HMI_ConveyorMotor_Control;
ELSE
  ConveyorMotor_State := TRUE;
END_IF
END_IF

```

Figure IV.18 Filtrage du capteur de bouteille et gestion de la vanne.

- La vitesse du convoyeur est peut déterminée par l’opérateur en mode manuel, soit automatiquement selon la présence de la bouteille. Cela garantit un transfert fluide et sécurisé des bouteilles vers la suite du processus.

```

IF ManualOverride_ConveyorMotor THEN
  IF ConveyorMotor_State THEN
    Conveyor_MotorVFD_Speed := M_HMI_ConveyorMotor_Speed;
  ELSE
    Conveyor_MotorVFD_Speed := 75.0;
  END_IF
END_IF

```

Figure IV.19 Forçage manuel du moteur du convoyeur

- Cette section surveille la surcharge du moteur du convoyeur ainsi que l’absence prolongée de bouteille. Si un défaut est détecté, le système arrête le convoyeur et informe l’opérateur via l’IHM.

```

IF NOT ManualOverride_ConveyorMotor THEN
  IF Conveyor_Motor_Absorp_mA > 2000.0 THEN
    ConveyorMotor_OverloadFault := TRUE;
    Conveyor_MotorVFD_Speed := 0.0;
    Conveyor_Motor_State := FALSE;
    M_HMI_StatusText := 'ERROR: Conveyor motor overload detected!';
    RETURN;
  END_IF
  IF NOT BottleReady THEN
    BottleFaultConfirmTimer(IN := TRUE, PT := T#5S);
    IF BottleFaultConfirmTimer.Q THEN
      BottleFault := TRUE;
      Conveyor_MotorVFD_Speed := 0.0;
      Conveyor_Motor_State := FALSE;
      M_HMI_StatusText := 'ERROR: Check bottle infeed sensor!';
      RETURN;
    ELSE
      BottleFault := FALSE;
      M_HMI_StatusText := 'Waiting for bottle infeed sensor...';
    END_IF
  ELSE
    BottleFaultConfirmTimer(IN := FALSE);
  END_IF
END_IF

```

Figure IV.20 Détection des défauts et gestion de la surcharge du convoyeur.

Le courant absorbé par le moteur du convoyeur est simulé en fonction de sa vitesse. Toutes les informations importantes (état, vitesse, défauts, présence de bouteille) sont transmises à l'IHM pour permettre à l'opérateur de superviser et d'ajuster le processus en temps réel.

```

IF Conveyor_MotorVFD_Speed > 0.0 THEN
  Conveyor_Motor_Absorp_mA := 1200.0 * (Conveyor_Motor_SpeedSetpoint / 100.0);
ELSE
  Conveyor_Motor_Absorp_mA := 0.0;
END_IF

M_HMI_SV_BottleFeederGate := SV_BottleFeederGate;
M_HMI_ConveyorMotor_Speed := Conveyor_MotorVFD_Speed;
M_HMI_ConveyorMotor_Absorp_mA := Conveyor_Motor_Absorp_mA;
M_HMI_ConveyorMotor_On := Conveyor_Motor_State;
M_HMI_ConveyorMotor_OverloadFault := ConveyorMotor_OverloadFault;
M_HMI_BottleReady := BottleReady;
M_HMI_BottleFault := BottleFault;

```

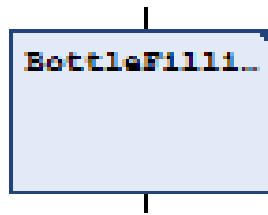
Figure IV.21 Simulation du courant du convoyeur et transfert des états vers l'IHM

3. Transition a prochaine étape :



Figure IV.22 Transition entre BottleReady et BottleFillingReady

IV.3.4.4. BottleFillingReady



IV.23 Bloc d'étape FillingBottleReady

Ce bloc de code gère l'arrivée de la bouteille à la position de remplissage. Il utilise une temporisation de débouncing pour garantir la fiabilité du capteur de position. Lorsque la bouteille est détectée, le convoyeur s'arrête et le système signale que le poste de remplissage est prêt. En cas d'absence prolongée de la bouteille, un défaut est généré et le convoyeur est arrêté pour sécurité.

1. Code d'action active :

- Cette section prépare tous les états et indicateurs nécessaires pour la gestion de l'arrivée de la bouteille en position de remplissage, assurant le démarrage du cycle dans un état sécurisé.

```
M_HMI_StatusText := 'Moving to Filling...';  
FillingReady := FALSE;  
FillingPositionFault := FALSE;  
ConveyorMotor_OverloadFault := FALSE;
```

Figure IV.24 Initialisation et indicateurs du remplissage de bouteilles

- L'opérateur peut contrôler le démarrage ou l'arrêt du moteur du convoyeur en mode manuel, sinon le système le gère automatiquement pour garantir la continuité du processus.

```
IF ManualOverride_ConveyorMotor THEN  
    ConveyorMotor_State := M_HMI_ConveyorMotor_Control;  
ELSE  
    ConveyorMotor_State := TRUE;  
END_IF
```

Figure IV.25 Commande manuelle et état moteur du convoyeur

- La vitesse du convoyeur est fixée soit par l'opérateur, soit automatiquement à une valeur prédéfinie, assurant ainsi le déplacement optimal de la bouteille vers la position de remplissage.

```

IF ManualOverride_ConveyorMotor THEN
  IF ConveyorMotor_State THEN
    Conveyor_MotorVFD_Speed := M_HMI_ConveyorMotor_Speed;
  ELSE
    Conveyor_MotorVFD_Speed := 0.0;
  END_IF
ELSE
  Conveyor_MotorVFD_Speed := 75.0;

```

Figure IV.26 Commande de la vitesse du convoyeur

- Le capteur de position est filtré (Filtrage anti-rebond) pour garantir la présence réelle de la bouteille. Lorsque la bouteille est détectée, le système active les électrovannes de remplissage et de maintien après un délai sécurisé.

```

IF FillingConveyorSensor THEN
  FillingConveyorDebounceTimer(IN := TRUE, PT := T#500MS);
  IF FillingConveyorDebounceTimer.Q THEN
    FillingConveyorSensorDebounced := TRUE;
  ELSE
    FillingConveyorSensorDebounced := FALSE;
  END_IF
ELSE
  FillingConveyorDebounceTimer(IN := FALSE);
  FillingConveyorSensorDebounced := FALSE;
END_IF
IF Conveyor_Motor_Absorp_mA > 2000.0 THEN
  ConveyorMotor_OverloadFault := TRUE;
  Conveyor_MotorVFD_Speed := 0.0;
  Conveyor_Motor_State := FALSE;
  M_HMI_StatusText := 'ERROR: Conveyor motor overload detected!';
  RETURN;
END_IF
IF FillingConveyorSensorDebounced THEN
  Conveyor_MotorVFD_Speed := 0.0;
  BottleWaitTimerFilling(IN := FALSE);
  FillingReady := TRUE;
  FillingPositionFault := FALSE;
  M_HMI_StatusText := 'Bottle is ready at filling position.';

  SV_Air_NozzleFillingTimer(IN := TRUE, PT := T#400MS);
  SV_Air_FillingClampsTimer(IN := TRUE, PT := T#400MS);

  IF SV_Air_NozzleFillingTimer.Q THEN
    SV_Air_NozzleFilling := TRUE;
  ELSE
    SV_Air_NozzleFilling := FALSE;
  END_IF

  IF SV_Air_FillingClampsTimer.Q THEN
    SV_Air_FillingClamps := TRUE;
  ELSE
    SV_Air_FillingClamps := FALSE;
  END_IF
ELSE
  SV_Air_NozzleFillingTimer(IN := FALSE);
  SV_Air_FillingClampsTimer(IN := FALSE);
  SV_Air_NozzleFilling := FALSE;
  SV_Air_FillingClamps := FALSE;
END_IF

```

Figure IV.27 Détection du capteur de position et gestion des électrovannes (SV)

- Un Minuteur surveille l'arrivée de la bouteille en position de remplissage. Si la bouteille n'arrive pas dans le délai imparti, le système signale un défaut et arrête le convoyeur pour éviter les erreurs de processus.

```

BottleWaitTimerFilling(IN := TRUE, PT := T#6s);

IF BottleWaitTimerFilling.Q AND NOT FillingConveyorSensorDebounced THEN
    Conveyor_MotorVFD_Speed := 0.0;
    Conveyor_Motor_State := FALSE;
    FillingPositionFault := TRUE;
    FillingReady := FALSE;
    M_HMI_StatusText := 'ERROR: Bottle did not arrive at filling position!';
    RETURN;
ELSE
    FillingPositionFault := FALSE;
    M_HMI_StatusText := 'Waiting for bottle at filling position...';
END_IF
END_IF

```

Figure IV.28 Minuteur d'attente et gestion des défauts de position

- Le courant moteur est simulé selon la vitesse, et toutes les informations clés (états, défauts, préparation) sont transférées à l'IHM pour permettre le suivi et la gestion du processus par l'opérateur.

```

IF Conveyor_MotorVFD_Speed > 0.0 THEN
    Conveyor_Motor_Absorp_mA := 1200.0 * (Conveyor_Motor_SpeedSetpoint / 100.0);
ELSE
    Conveyor_Motor_Absorp_mA := 0.0;
END_IF

M_HMI_ConveyorMotor_Speed := Conveyor_MotorVFD_Speed;
M_HMI_ConveyorMotor_Absorp_mA := Conveyor_Motor_Absorp_mA;
M_HMI_ConveyorMotor_On := ConveyorMotor_State;
M_HMI_ConveyorMotor_OverloadFault := ConveyorMotor_OverloadFault;
M_HMI_FillingReady := FillingReady;
M_HMI_FillingPositionFault := FillingPositionFault;

```

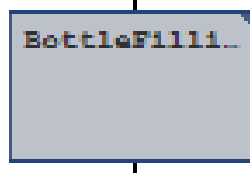
Figure IV.29 Simulation du courant moteur et transfert des états vers l'IHM

2. Transition a prochaine étape :


FillingReady

IV.30 Transition entre FillingBottleReady et FillingBottle

IV.3.4.5. FillingBottle



IV.31 Bloc d'étape FillingBottle

Ce bloc de code gère l'étape de remplissage de la bouteille. Il active la pompe et les actionneurs nécessaires, surveille la pression du bec et la détection de la bouteille pleine à l'aide de capteurs et de temporisations, et traite les éventuelles erreurs (temps dépassé, problème de pression, défaut capteur). À la fin du remplissage ou en cas de défaut, le système arrête la pompe et informe l'opérateur via l'IHM.

1. Code d'action active :

- Initialisation des états et des indicateurs pour la séquence de remplissage, garantissant le démarrage dans un état sûr et prêt à traiter la bouteille.

```
FillingDone := FALSE;
FillingTimeoutFault := FALSE;
FillingNozzlePressureFault := FALSE;
FullBottleSensorFault := FALSE;
PumpMotor2VFD_OverloadFault := FALSE;
SV_Air_NozzleFilling := TRUE;
SV_Air_FillingClamps := TRUE;
M_HMI_StatusText := 'Bottle detected. Filling in process...';
```

Figure IV.32 Initialisation et indicateurs du remplissage

- Permet à l'opérateur de contrôler manuellement les actionneurs (serrage, buse) et le moteur de la pompe, sinon le programme gère tout en mode automatique.

```
IF ManualOverride_SV_Air_FillingClamps THEN
    SV_Air_FillingClamps := M_HMI_SV_Air_FillingClamps_Control;
ELSE
    SV_Air_FillingClamps := TRUE;
END_IF

IF ManualOverride_SV_Air_NozzleFilling THEN
    SV_Air_NozzleFilling := M_HMI_SV_Air_NozzleFilling_Control;
ELSE
    SV_Air_NozzleFilling := TRUE;
END_IF

IF ManualOverride_PumpMotor2VFD THEN
    PumpMotor2VFD_State := M_HMI_PumpMotor2VFD_Control;
ELSE
    PumpMotor2VFD_State := TRUE;
END_IF
```

Figure IV.33 Commande manuelle des actionneurs et du moteur

- Détermine la vitesse du moteur de la pompe selon le mode (manuel ou automatique), assurant un remplissage adapté à chaque situation.

```

IF ManualOverride_PumpMotor2VFD THEN
  IF PumpMotor2VFD_State THEN
    PumpMotor2VFD_Speed := M_HMI_PumpMotor2VFD_Speed;
  ELSE
    PumpMotor2VFD_Speed := 0.0;
  END_IF
ELSE
  PumpMotor2VFD_Speed := 75.0;
END_IF

```

Figure IV.34 Commande de la vitesse du moteur pompe

- Filtre les signaux des capteurs pour éviter les fausses détections et calcule le courant absorbé par le moteur selon la vitesse, permettant une surveillance fiable du processus.

```

IF NozzlePressureSensor THEN
  NozzlePressureDebounceTimer(IN := TRUE, PT := T#500MS);
  IF NozzlePressureDebounceTimer.Q THEN
    NozzlePressureSensorDebounced := TRUE;
  ELSE
    NozzlePressureSensorDebounced := FALSE;
  END_IF
ELSE
  NozzlePressureDebounceTimer(IN := FALSE);
  NozzlePressureSensorDebounced := FALSE;
END_IF

IF FullBottleSensor THEN
  FullBottleDebounceTimer(IN := TRUE, PT := T#500MS);
  IF FullBottleDebounceTimer.Q THEN
    FullBottleSensorDebounced := TRUE;
  ELSE
    FullBottleSensorDebounced := FALSE;
  END_IF
ELSE
  FullBottleDebounceTimer(IN := FALSE);
  FullBottleSensorDebounced := FALSE;
END_IF

IF PumpMotor2VFD_Speed > 0.0 THEN
  PumpMotor2VFD_Absorp_mA := 1500.0 * (PumpMotor2VFD_Speed / 100.0);
ELSE
  PumpMotor2VFD_Absorp_mA := 0.0;
END_IF

```

Figure IV.35 Filtrage des capteurs et calcul du courant moteur

- Transfère tous les états importants (actionneurs, moteur, défauts) vers l'IHM pour assurer le suivi et l'intervention rapide par l'opérateur en cas de problème.

```

M_HMI_PumpMotor2_Speed := PumpMotor2VFD_Speed;
M_HMI_PumpMotor2_Absorp_mA := PumpMotor2VFD_Absorp_mA;
M_HMI_PumpMotor2VFD_On := PumpMotor2VFD_State;
M_HMI_PumpMotor2VFD_OverloadFault := PumpMotor2VFD_OverloadFault;
M_HMI_SV_Air_FillingClamps := SV_Air_FillingClamps;
M_HMI_SV_Air_NozzleFilling := SV_Air_NozzleFilling;

```

Figure IV.36 Mapping IHM (états actionneurs, moteur, défauts)

- Cette section surveille le courant absorbé par le moteur de la pompe. Si la valeur dépasse le seuil de sécurité, le système déclenche un défaut, arrête le moteur et les actionneurs, et signale l'erreur à l'opérateur.

```

IF PumpMotor2VFD_Absorp_mA > 1600.0 THEN
    PumpMotor2VFD_OverloadFault := TRUE;
    PumpMotor2VFD_Speed := 0.0;
    PumpMotor2VFD_State := FALSE;
    SV_Air_NozzleFilling := FALSE;
    SV_Air_FillingClamps := FALSE;
    M_HMI_StatusText := 'ERROR: Pump motor overload!';
    RETURN;
END_IF

```

Figure IV.37 Détection de surcharge du moteur pompe

- Cette partie gère les défauts liés au dépassement du temps de remplissage et à l'absence de pression ou de détection de la bouteille. En cas d'anomalie, le remplissage est stoppé et une alerte est affichée.

```

FillingTimer(IN := TRUE, PT := T#12S);
NozzlePressureFaultTimer(IN := TRUE, PT := T#3S);
IF NOT NozzlePressureSensorDebounced AND NozzlePressureFaultTimer.Q THEN
    PumpMotor2VFD_Speed := 0.0;
    PumpMotor2VFD_State := FALSE;
    SV_Air_NozzleFilling := FALSE;
    SV_Air_FillingClamps := FALSE;
    FillingNozzlePressureFault := TRUE;
    M_HMI_StatusText := 'ERROR: Nozzle pressure fault! Check air supply.';
    RETURN;
END_IF

IF NOT FullBottleSensorDebounced AND FillingTimer.Q THEN
    PumpMotor2VFD_Speed := 0.0;
    PumpMotor2VFD_State := FALSE;
    SV_Air_NozzleFilling := FALSE;
    SV_Air_FillingClamps := FALSE;
    FullBottleSensorFault := TRUE;
    M_HMI_StatusText := 'ERROR: Filling sensor fault! Check bottle and sensor.';
    RETURN;
END_IF

```

Figure IV.38 Gestion du temps et des défauts de capteurs

- Lorsque la bouteille est détectée comme pleine, le système arrête le moteur, rétracte la buse, relâche les pinces après temporisation et signale la fin du cycle de remplissage, prêt à passer à l'étape suivante.

```

IF FullBottleSensorDebounced THEN
    PumpMotor2VFD_Speed := 0.0;
    SV_Air_NozzleFilling := FALSE;
    IF NOT SV_Air_NozzleFilling THEN
        SV_Air_FillingClampsTimer(IN := TRUE, PT := T#1S);
    ELSE
        SV_Air_FillingClampsTimer(IN := FALSE);
    END_IF
    IF SV_Air_FillingClampsTimer.Q THEN
        SV_Air_FillingClamps := FALSE;
    END_IF
    FillingDone := TRUE;
    FillingTimer(IN := FALSE);
    M_HMI_StatusText := 'Filling complete. Proceeding to next step...';
END_IF

```

Figure IV.39 Fin normale du remplissage

2. Transition a prochaine étape :



Figure IV.40 Transition entre FillingBottle et CappingBottleReady

IV.3.4.6. CappingBottleReady

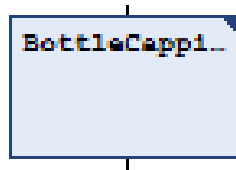


Figure IV. 41 Bloc d'étape CappingBottleReady.

Ce bloc de code gère l'arrivée de la bouteille au poste de capsulage. Il utilise une temporisation de débouncing pour garantir la fiabilité du capteur de position. Lorsque la bouteille est détectée, le convoyeur s'arrête et le système signale que le poste de capsulage est prêt. Si la bouteille n'est pas détectée, un Minuteur d'attente démarre pour surveiller le délai d'arrivée.

1. Code d'action active :
 - Cette section prépare l'étape de bouchage en réinitialisant tous les indicateurs et états, pour garantir la sécurité et la fiabilité du démarrage du cycle.

```

M_HMI_StatusText := 'Filling finished. Moving to Capping...';
CappingReady := FALSE;
BottleCappingPositionFault := FALSE;
ConveyorMotor_OverloadFault := FALSE;
CapFeederMotor_OverloadFault := FALSE;
CapFeederSensorFault := FALSE;

```

Figure IV.42 Initialisation et indicateurs d'état

- Permet à l'opérateur de piloter le moteur du convoyeur en mode manuel, ou laisse le système gérer automatiquement en mode standard.

```

IF ManualOverride_ConveyorMotor THEN
    ConveyorMotor_State := M_HMI_ConveyorMotor_Control;
ELSE
    ConveyorMotor_State := TRUE;
END_IF

```

Figure IV.43 Commande manuelle et état moteur du convoyeur

- La vitesse du convoyeur et l'acheminement des capsules est déterminée selon le mode sélectionné, et le courant des moteurs est calculé pour permettre la surveillance des surcharges.

```

IF ManualOverride_ConveyorMotor THEN
    IF ConveyorMotor_State THEN
        Conveyor_MotorVFD_Speed := M_HMI_ConveyorMotor_Speed;
    ELSE
        Conveyor_MotorVFD_Speed := 0.0;
    END_IF
ELSE
    Conveyor_MotorVFD_Speed := 75.0;
END_IF

IF Conveyor_MotorVFD_Speed > 0.0 THEN
    Conveyor_Motor_Absorp_mA := 1200.0 * (Conveyor_MotorVFD_Speed / 100.0);
ELSE
    Conveyor_Motor_Absorp_mA := 0.0;
END_IF

IF ManualOverride_CapFeederMotor THEN
    IF CapFeederMotor_State THEN
        CapFeederMotor_Speed := M_HMI_CapFeederMotor_Speed;
    ELSE
        CapFeederMotor_Speed := 0.0;
    END_IF
ELSE
    CapFeederMotor_Speed := 100.0;
END_IF

IF CapFeederMotor_Speed > 0.0 THEN
    CapFeederMotor_Absorp_mA := 900.0 * (CapFeederMotor_Speed / 100.0);
ELSE
    CapFeederMotor_Absorp_mA := 0.0;
END_IF

```

Figure IV.44 Commande manuelle de la vitesse du convoyeur et calcul du courant

- Surveille le courant absorbé par le moteur ; en cas de dépassement de seuil, un défaut est déclenché, le cycle interrompu et une alerte affichée à l'opérateur.

```

IF Conveyor_Motor_Absorp_mA > 1300.0 THEN
    ConveyorMotor_OverloadFault := TRUE;
    Conveyor_MotorVFD_Speed := 0.0;
    Conveyor_Motor_State := FALSE;
    BottleWaitTimerCapping(IN := FALSE);
    SV_Air_Chuck := FALSE;
    SV_Air_CappingClamps := FALSE;
    CappingReady := FALSE;
    M_HMI_StatusText := 'ERROR: Conveyor motor overload at capping! Check motor and wiring.';
    RETURN;
END_IF

IF CapFeederMotor_Absorp_mA > 1000.0 THEN
    CapFeederMotor_OverloadFault := TRUE;
    CapFeederMotor_Speed := 0.0;
    CapFeederMotor_State := FALSE;
    M_HMI_StatusText := 'ERROR: Cap feeder motor overload! Check motor and wiring.';
    RETURN;
END_IF

```

Figure IV.45 Détection de surcharge moteur du convoyeur et des capsules

- Ce code filtre les signaux des capteurs bouteille et capsule pour éviter les erreurs dues aux rebonds, puis lance un Minuteur d'attente si la détection n'est pas confirmée.

```

IF CappingConveyorSensor THEN
    CappingConveyorDebounceTimer(IN := TRUE, PT := T#300MS);
    IF CappingConveyorDebounceTimer.Q THEN
        CappingConveyorSensorDebounced := TRUE;
    ELSE
        CappingConveyorSensorDebounced := FALSE;
    END_IF
ELSE
    CappingConveyorDebounceTimer(IN := FALSE);
    CappingConveyorSensorDebounced := FALSE;
END_IF
IF CapFeederSensor THEN
    CapFeederSensorDebounceTimer(IN := TRUE, PT := T#300MS);
    IF CapFeederSensorDebounceTimer.Q THEN
        CapFeederSensorDebounced := TRUE;
    ELSE
        CapFeederSensorDebounced := FALSE;
    END_IF
ELSE
    CapFeederSensorDebounceTimer(IN := FALSE);
    CapFeederSensorDebounced := FALSE;
END_IF
IF CappingConveyorSensorDebounced THEN
    BottleWaitTimerCapping(IN := FALSE);
ELSE
    BottleWaitTimerCapping(IN := TRUE, PT:=T#3S);
END_IF

IF CapFeederSensorDebounced THEN
    CapPresentWaitTimer(IN := FALSE);
ELSE
    CapPresentWaitTimer(IN := TRUE, PT := T#3S);
END_IF

```

Figure IV.46 Anti Rebond et Temporisation Capteur Bouteille et Capsule

- Ce code active les électrovannes de capsulage uniquement lorsque la bouteille et la capsule sont correctement détectées. Le reste du temps, les électrovannes restent désactivées.

```

IF CappingConveyorSensorDebounced AND CapFeederSensorDebounced THEN
    Conveyor_MotorVFD_Speed := 0.0;
    BottleWaitTimerCapping(IN := FALSE);
    CapPresentWaitTimer(IN := FALSE);
    BottleCappingPositionFault := FALSE;
    CapFeederSensorFault := FALSE;
    CappingReady := TRUE;
    SV_Air_Chuck_Timer(IN := TRUE, PT := T#500MS);
    SV_Air_CappingClamps_Timer(IN := TRUE, PT := T#500MS);

    IF SV_Air_Chuck_Timer.Q THEN
        SV_Air_Chuck := TRUE;
    ELSE
        SV_Air_Chuck := FALSE;
    END_IF

    IF SV_Air_CappingClamps_Timer.Q THEN
        SV_Air_CappingClamps := TRUE;
    ELSE
        SV_Air_CappingClamps := FALSE;
    END_IF
ELSE
    SV_Air_Chuck_Timer(IN := FALSE);
    SV_Air_CappingClamps_Timer(IN := FALSE);
    SV_Air_Chuck := FALSE;
    SV_Air_CappingClamps := FALSE;
END_IF

```

Figure IV.47 Activation des électrovannes de capsulage

- Ce code signale un défaut si la bouteille ou la capsule n'est pas détectée à temps, puis arrête les moteurs et désactive les électrovannes.

```

IF BottleWaitTimerCapping.Q AND NOT CappingConveyorSensorDebounce THEN
    BottleCappingPositionFault := TRUE;
    ConveyorMotor_State := FALSE;
    Conveyor_MotorVFD_Speed := 0.0;
    SV_Air_Chuck := FALSE;
    SV_Air_CappingClamps := FALSE;
END_IF

IF CapPresentWaitTimer.Q AND NOT CapFeederSensorDebounce THEN
    CapFeederSensorFault := TRUE;
    CapFeederMotor_State := FALSE;
    CapFeederMotor_Speed := 0.0;
    M_HMI_StatusText := 'ERROR: No cap detected by feeder sensor!';
END_IF

```

Figure IV.48 Détection des défauts bouteille et capsule

- Ce code met à jour l'IHM avec la vitesse, le courant absorbé, l'état de marche et les défauts des moteurs du convoyeur et du cap feeder.

```

M_HMI_CapFeederMotor_Speed := CapFeederMotor_Speed;
M_HMI_CapFeederMotor_Absorp_mA := CapFeederMotor_Absorp_mA;
M_HMI_CapFeederMotor_On := CapFeederMotor_State;
M_HMI_ConveyorMotor_Speed := Conveyor_MotorVFD_Speed;
M_HMI_ConveyorMotor_Absorp_mA := Conveyor_Motor_Absorp_mA;
M_HMI_ConveyorMotor_On := ConveyorMotor_State;
M_HMI_ConveyorMotor_OverloadFault := ConveyorMotor_OverloadFault;
M_HMI_CapFeederMotor_OverloadFault := CapFeederSensorFault;

```

Figure IV.49 Affichage des états moteurs sur l'IHM

2. Transition a prochaine étape :

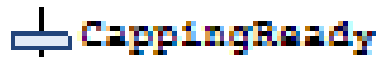


Figure IV.50 Transition entre CappingBottleReady et JawReady

IV.3.4.7. JawReady

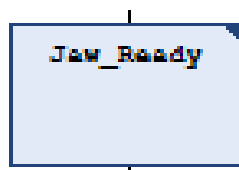


Figure IV.51 Bloc d'étape JawReady

Ce bloc de code gère l'étape de capsulage de la bouteille. Il active les actionneurs nécessaires, surveille la position du mandrin avec un capteur et une temporisation, et engage la mâchoire pour le capsulage lorsque la position est atteinte. En cas de défaut (mandrin non descendu dans le temps imparti), le système signale une erreur et bloque la suite du cycle.

1. Code d'action actif :

- Initialise les variables d'état et les indicateurs pour l'étape de bouchage, assurant que le système démarre dans un état sécurisé.

```
M_HMI_StatusText := 'Bottle detected. Capping in process...';  
  
JawReady := FALSE;  
ChuckDownFault := FALSE;
```

Figure IV.52 Initialisation et indicateurs d'état

- Permet à l'opérateur de prendre le contrôle direct des électrovannes pour le serrage, le mandrin et la mâchoire, sinon le programme gère automatiquement ces actionneurs.

```
IF ManualOverride_SV_Air_CappingClamps THEN  
    SV_Air_CappingClamps := M_HMI_SV_Air_CappingClamps_Control;  
ELSE  
    SV_Air_CappingClamps := TRUE;  
END_IF  
  
IF ManualOverride_SV_Air_Chuck THEN  
    SV_Air_Chuck := M_HMI_SV_Air_Chuck_Control;  
ELSE  
    SV_Air_Chuck := TRUE;  
END_IF  
  
IF ManualOverride_SV_Air_Jaw THEN  
    SV_Air_Jaw := M_HMI_SV_Air_Jaw_Control;  
ELSE  
    SV_Air_Jaw := FALSE;  
END_IF
```

Figure IV.53 Commande manuelle des électrovannes (SV)

- Filtre le signal du capteur de position basse du mandrin et gère un Minuteur qui surveille le délai d'atteinte de cette position.

```
IF ChuckDownSensor THEN  
    ChuckDownDebounceTimer(IN := TRUE, PT := T#300MS);  
    IF ChuckDownDebounceTimer.Q THEN  
        ChuckDownSensorDebounced := TRUE;  
    ELSE  
        ChuckDownSensorDebounced := FALSE;  
    END_IF  
ELSE  
    ChuckDownDebounceTimer(IN := FALSE);  
    ChuckDownSensorDebounced := FALSE;  
END_IF  
  
ChuckDownWaitTimer(IN := NOT ChuckDownSensorDebounced, PT := T#4S);
```

Figure IV.54 Filtrage du capteur ChuckDown et Minuteur d'attente

- Si le mandrin est détecté en position basse, la mâchoire est activée automatiquement (sauf en mode manuel) et le système signale qu'il est prêt pour l'étape suivante.

```

IF ChuckDownSensorDebounced THEN
  IF NOT ManualOverride_SV_Air_Jaw THEN
    SV_Air_Jaw := TRUE;
  END_IF
  JawReady := TRUE;
  ChuckDownWaitTimer(IN := FALSE);
  ChuckDownFault := FALSE;
ELSE
  IF NOT ManualOverride_SV_Air_Jaw THEN
    SV_Air_Jaw := FALSE;
  END_IF
  JawReady := FALSE;
END_IF

```

Figure IV.55 Atteinte de la position basse du Mandrin et activation de la mâchoire

- Si le mandrin n'atteint pas sa position dans le temps imparti, le système signale un défaut, désactive la mâchoire et met à jour l'état pour l'opérateur.

```

IF NOT ChuckDownSensorDebounced AND ChuckDownWaitTimer.Q THEN
  IF NOT ManualOverride_SV_Air_Jaw THEN
    SV_Air_Jaw := FALSE;
  END_IF
  JawReady := FALSE;
  ChuckDownFault := TRUE;
  ChuckDownWaitTimer(IN := FALSE);
  M_HMI_StatusText := 'ERROR: Chuck did not reach down position! Check mechanism/sensor.';
END_IF

```

Figure IV.56 Détection et gestion du défaut ChuckDown

- Transfère tous les états des électrovannes et les défauts vers l'IHM pour permettre le suivi et la supervision par l'opérateur.

```

M_HMI_SV_Air_CappingClamps := SV_Air_CappingClamps;
M_HMI_SV_Air_Chuck := SV_Air_Chuck;
M_HMI_SV_Air_Jaw := SV_Air_Jaw;
M_HMI_JawReady := JawReady;
M_HMI_ChuckDownFault := ChuckDownFault;

```

Figure IV.57 Mappings IHM (états électrovannes et défauts)

2. Transition a prochaine étape :

 JawReady = TRUE

Figure IV.58 Transition entre CappingBottleReady et JawReady

IV.3.4.8. ChuckRotation

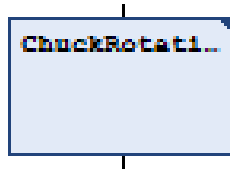


Figure IV.59 Bloc d'étape ChuckRotation

Ce bloc de code gère la phase finale du capsulage : la mâchoire saisit le bouchon, puis le moteur pas à pas effectue la rotation pour visser le bouchon. Chaque sous-action est surveillée par un capteur et une temporisation pour détecter les défauts. En cas de réussite, le système passe à l'étape suivante, sinon il signale une erreur à l'opérateur via l'IHM.

1. Code d'action active :

- Cette partie du code initialise les variables principales du processus de capsulage et permet à l'opérateur d'activer les actionneurs en mode manuel via l'interface IHM.

```
M_HMI_StatusText := 'Capping: Jaw clutching cap...';
CappingDone := FALSE;
JawClutchFaultActive := FALSE;
TorqueFaultActive := FALSE;

IF ManualOverride_SV_Air_Jaw THEN
  SV_Air_Jaw := M_HMI_SV_Air_Jaw_Control;
ELSE
  SV_Air_Jaw := TRUE;
END_IF

IF ManualOverride_SV_Air_CappingClamps THEN
  SV_Air_CappingClamps := M_HMI_SV_Air_CappingClamps_Control;
ELSE
  SV_Air_CappingClamps := TRUE;
END_IF

IF ManualOverride_SV_Air_Chuck THEN
  SV_Air_Chuck := M_HMI_SV_Air_Chuck_Control;
ELSE
  SV_Air_Chuck := TRUE;
END_IF
```

Figure IV.60 Initialisation et gestion des modes manuels

- vérifie que la mâchoire a bien saisi le bouchon à l'aide d'un capteur. La temporisation anti-rebond assure une détection fiable.

```
IF JawClutchSensor THEN
  JawClutchDebounceTimer(IN := TRUE, PT := T#500MS);
  IF JawClutchDebounceTimer.Q THEN
    JawClutchSensorDebounced := TRUE;
  ELSE
    JawClutchSensorDebounced := FALSE;
  END_IF
ELSE
  JawClutchDebounceTimer(IN := FALSE);
  JawClutchSensorDebounced := FALSE;
END_IF

IF NOT ManualOverride_SV_Air_Jaw THEN
  SV_Air_Jaw := JawClutchSensorDebounced;
END_IF

JawClutchWaitTimer(IN := NOT JawClutchSensorDebounced, PT := T#2S);
```

Figure IV.61 Détection de la prise du bouchon par la mâchoire

- Quand la prise du bouchon est validée, la rotation du mandrin commence grâce au moteur pas à pas. Ce moteur offre une grande précision pour le visage du bouchon, ce qui évite d'endommager le flacon ou la capsule. Sa puissance et son couple sont adaptés pour les lignes de conditionnement industrielles.

```

IF JawClutchSensorDebounced THEN
    JawClutchWaitTimer(IN := FALSE);
    JawClutchFaultActive := FALSE;

    IF NOT ManualOverride_StepperChuck THEN
        Stepper_Chuck_Speed := 100.0;
        Stepper_Chuck_Rotation := 720.0;
        Stepper_Chuck_State := TRUE;
    END_IF

    M_HMI_StatusText := 'Capping: Chuck rotating to tighten cap...';

    ChuckRotationTimer (IN := TRUE, PT := T#3S);

```

Figure IV.62 Rotation du moteur pas à pas

- Pendant la rotation, un capteur de couple mesure en continu la force appliquée sur le bouchon. Si la valeur du couple mesurée reste dans la plage [0,8 ; 1,2 Nm], le cycle est validé et le moteur s'arrête.

Si le couple n'est jamais atteint ou si la rotation dépasse le temps maximal (timeout), le cycle s'arrête en défaut : cela protège l'équipement et garantit la qualité du vissage. Détails du capteur de couple :

- Type : Jauge de contrainte ou capteur rotatif
- Plage : 0–2 Nm
- Précision : $\pm 0,05$ Nm
- Montage : Entre le moteur pas à pas et le mandrin

Ce capteur permet d'assurer l'intégrité du bouchon et du flacon, et garantit la bonne fermeture selon les spécifications.

```

IF MeasuredTorque >= TorqueMin AND MeasuredTorque <= TorqueMax AND NOT ChuckRotationTimer.Q THEN
    TorqueFaultActive := FALSE;
    CappingDone := TRUE;
    M_HMI_StatusText := CONCAT(CONCAT('Capping: Finished. Torque ', REAL_TO_STRING(MeasuredTorque)), ' Nm.');
```

```

    IF NOT ManualOverride_StepperChuck THEN
        Stepper_Chuck_Speed := 0.0;
        Stepper_Chuck_Rotation := 0.0;
    END_IF
    IF NOT ManualOverride_SV_Air_Jaw THEN
        SV_Air_Jaw := FALSE;
    END_IF
    IF NOT ManualOverride_SV_Air_CappingClamps THEN
        SV_Air_CappingClamps := FALSE;
    END_IF
    IF NOT ManualOverride_SV_Air_Chuck THEN
        SV_Air_Chuck := FALSE;
    END_IF
ELSEIF ChuckRotationTimer.Q THEN
    IF MeasuredTorque < TorqueMin OR MeasuredTorque > TorqueMax THEN
        TorqueFaultActive := TRUE;
        CappingDone := FALSE;
        M_HMI_StatusText := 'ERROR: Torque out of range after capping! Check chuck/cap/bottle.';
    ELSE
        TorqueFaultActive := TRUE;
        CappingDone := FALSE;
        M_HMI_StatusText := 'ERROR: Chuck timeout, no valid torque!';
    END_IF

```

Figure IV.63 Surveillance du couple et gestion des défauts

- Arrête le moteur et désengager les actionneurs pour finaliser l'étape du bouchage

```

IF NOT ManualOverride_StepperChuck THEN
    Stepper_Chuck_Speed := 0.0;
    Stepper_Chuck_Rotation := 0.0;
    ChuckRotationTimer(IN := FALSE);
END_IF
IF NOT ManualOverride_SV_Air_Jaw THEN
    SV_Air_Jaw := FALSE;
END_IF
IF NOT ManualOverride_SV_Air_CappingClamps THEN
    SV_Air_CappingClamps := FALSE;
END_IF
IF NOT ManualOverride_SV_Air_Chuck THEN
    SV_Air_Chuck := FALSE;
END_IF

```

Figure IV.64 Arrêt le moteur et actionneurs

- Si la mâchoire ne saisit pas le bouchon dans le temps imparti, le système met tout en sécurité et affiche un message d'erreur à l'opérateur.

```

IF JawClutchWaitTimer.Q THEN
    Stepper_Chuck_Speed := 0.0;
    Stepper_Chuck_Rotation := 0.0;
    IF NOT ManualOverride_SV_Air_Jaw THEN
        SV_Air_Jaw := FALSE;
    END_IF
    IF NOT ManualOverride_SV_Air_CappingClamps THEN
        SV_Air_CappingClamps := FALSE;
    END_IF
    IF NOT ManualOverride_SV_Air_Chuck THEN
        SV_Air_Chuck := FALSE;
    END_IF
    JawClutchFaultActive := TRUE;
    CappingDone := FALSE;
    M_HMI_StatusText := 'ERROR: Jaw failed to clutch cap!';
    JawClutchWaitTimer(IN := FALSE);
END_IF

```

Figure IV.65 Gestion du défaut de prise mâchoire

- Toutes les variables de statut (couple, position, défauts, état des actionneurs) sont transmises à l'interface IHM pour permettre à l'opérateur de suivre le processus en temps réel.

```

M_HMI_StepperChuck_Speed := Stepper_Chuck_Speed;
M_HMI_StepperChuck_Rotation := Stepper_Chuck_Rotation;
M_HMI_StepperChuck_On := Stepper_Chuck_State;
M_HMI_JawClutchFault := JawClutchFaultActive;
M_HMI_TorqueFault := TorqueFaultActive;
M_HMI_CappingDone := CappingDone;
M_HMI_MeasuredTorque := MeasuredTorque;

```

Figure IV.66 Mappage IHM et arrêt du convoyeur en cas de défaut

2. Transition a prochaine étape :

 CappingDone - TRUE

Figure IV.67 Transition entre ChuckRotation et Labeling

IV.3.4.9. Labeling

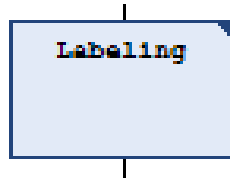


Figure IV.68 Bloc d'étape Labeling

Ce bloc de code gère l'étape d'étiquetage de la bouteille. Il démarre les convoyeurs, utilise plusieurs capteurs avec débouncing pour fiabiliser la détection des états nécessaires (présence bouteille, position du la pointe, détection du gap). Des Minuteur s assurent la sécurité et la robustesse du cycle. En cas d'anomalie (bouteille absente, tip non prêt, gap non détecté), une alarme est générée et le cycle est stoppé. À la fin du cycle, la coupe de l'étiquette est réalisée et l'avancement du produit est signalé à l'opérateur.

1. Code d'action active :

- Prépare tous les indicateurs, états de défaut et variables pour démarrer l'étape de l'étiquetage dans des conditions sûres.

```
M_HMI_StatusText := 'Labeling in progress...';  
BottleLabelingReady := FALSE;  
LabelTipReady := FALSE;  
LabelCutDone := FALSE;  
BottleLabelingDone := FALSE;  
ConveyorMotor_OverloadFault := FALSE;  
SideConveyorLeft_OverloadFault := FALSE;  
SideConveyorRight_OverloadFault := FALSE;  
LabelTipMotor_OverloadFault := FALSE;  
LabelTipNotReady := FALSE;  
LabelGapNotDetected := FALSE;  
BottleArrivalFault := FALSE;
```

Figure IV.69 Initialisation et indicateurs d'état

- Cette section gère le mode manuel/automatique pour chaque actionneur, permettant à l'opérateur d'agir directement sur les moteurs ou le vérin du couteau via l'IHM.

```

IF ManualOverride_ConveyorMotor THEN
    Conveyor_Motor_SpeedSetpoint := M_HMI_ConveyorMotor_Speed;
    Conveyor_Motor_State := M_HMI_ConveyorMotor_Control;
ELSE
    Conveyor_Motor_SpeedSetpoint := 75.0;
    Conveyor_Motor_State := TRUE;
END_IF
IF ManualOverride_SideConveyorLeft THEN
    SideConveyor_Left_SpeedSetpoint := M_HMI_SideConveyorLeft_Speed;
    SideConveyor_Left_State := M_HMI_SideConveyorLeft_Control;
ELSE
    SideConveyor_Left_SpeedSetpoint := 75.0;
    SideConveyor_Left_State := TRUE;
END_IF
IF ManualOverride_SideConveyorRight THEN
    SideConveyor_Right_SpeedSetpoint := M_HMI_SideConveyorRight_Speed;
    SideConveyor_Right_State := M_HMI_SideConveyorRight_Control;
ELSE
    SideConveyor_Right_SpeedSetpoint := 75.0;
    SideConveyor_Right_State := TRUE;
END_IF
IF ManualOverride_LabelTipMotor THEN
    Label_Tip_Motor_SpeedSetpoint := M_HMI_LabelTipMotor_Speed;
    Label_Tip_Motor_State := M_HMI_LabelTipMotor_Control;
ELSE
    IF NOT LabelPositionSensorDebounced THEN
        Label_Tip_Motor_SpeedSetpoint := 80.0;
        Label_Tip_Motor_State := TRUE;
    ELSE
        Label_Tip_Motor_SpeedSetpoint := 0.0;
        Label_Tip_Motor_State := TRUE;
    END_IF
END_IF
IF ManualOverride_SV_Knife THEN
    SV_Knife := M_HMI_SV_Knife_Control;
ELSE
    SV_Knife := FALSE;
END_IF

```

Figure IV.70 Gestion des modes manuels

- Calcule le courant pour chaque moteur

```

Conveyor_Motor_Absorp_mA := 1200.0 * (Conveyor_Motor_SpeedSetpoint / 100.0);
SideConveyor_Left_Absorp_mA := 800.0 * (SideConveyor_Left_SpeedSetpoint / 100.0);
SideConveyor_Right_Absorp_mA := 800.0 * (SideConveyor_Right_SpeedSetpoint / 100.0);
Label_Tip_Motor_Absorp_mA := 500.0 * (Label_Tip_Motor_SpeedSetpoint / 100.0);

```

Figure IV.71 Calcul du courant absorbé

- Vérifie les surcharges. En cas de défaut, le système arrête le cycle et affiche l'alarme correspondante.

```

IF Conveyor_Motor_Absorp_mA > 2000.0 THEN
    ConveyorMotor_OverloadFault := TRUE;
    Conveyor_Motor_SpeedSetpoint := 0.0;
    Conveyor_Motor_State := FALSE;
    M_HMI_StatusText := 'ERROR: Conveyor motor overload detected!';
    RETURN;
END_IF

IF SideConveyor_Left_Absorp_mA > 1000.0 THEN
    SideConveyorLeft_OverloadFault := TRUE;
    SideConveyor_Left_SpeedSetpoint := 0.0;
    SideConveyor_Left_State := FALSE;
    M_HMI_StatusText := 'ERROR: Side conveyor left motor overload detected!';
    RETURN;
END_IF

IF SideConveyor_Right_Absorp_mA > 1000.0 THEN
    SideConveyorRight_OverloadFault := TRUE;
    SideConveyor_Right_SpeedSetpoint := 0.0;
    SideConveyor_Right_State := FALSE;
    M_HMI_StatusText := 'ERROR: Side conveyor right motor overload detected!';
    RETURN;
END_IF

IF Label_Tip_Motor_Absorp_mA > 700.0 THEN
    LabelTipMotor_OverloadFault := TRUE;
    Label_Tip_Motor_SpeedSetpoint := 0.0;
    Label_Tip_Motor_State := FALSE;
    M_HMI_StatusText := 'ERROR: Label tip motor overload detected!';
    RETURN;
END_IF

```

Figure IV.72 Vérification des surcharges

- Ce bloc gère la temporisation anti-rebond pour les capteurs de position bouteille, tip et gap, assurant une fiabilité de la détection et évitant les fausses lectures.

```

IF BottleLabellingPositionSensor THEN
    BottleLabellingPositionDebounceTimer(IN := TRUE, PT := T#300MS);
    IF BottleLabellingPositionDebounceTimer.Q THEN
        BottleLabellingPositionSensorDebounced := TRUE;
    ELSE
        BottleLabellingPositionSensorDebounced := FALSE;
    END_IF
ELSE
    BottleLabellingPositionDebounceTimer(IN := FALSE);
    BottleLabellingPositionSensorDebounced := FALSE;
END_IF

IF LabelTipSensor THEN
    LabelTipDebounceTimer(IN := TRUE, PT := T#400MS);
    IF LabelTipDebounceTimer.Q THEN
        LabelTipSensorDebounced := TRUE;
    ELSE
        LabelTipSensorDebounced := FALSE;
    END_IF
ELSE
    LabelTipDebounceTimer(IN := FALSE);
    LabelTipSensorDebounced := FALSE;
END_IF

IF LabelGapSensor THEN
    LabelGapDebounceTimer(IN := TRUE, PT := T#300MS);
    IF LabelGapDebounceTimer.Q THEN
        LabelGapSensorDebounced := TRUE;
    ELSE
        LabelGapSensorDebounced := FALSE;
    END_IF
ELSE
    LabelGapDebounceTimer(IN := FALSE);
    LabelGapSensorDebounced := FALSE;
END_IF

```

Figure IV.73 Filtrage des capteurs

- Si aucune bouteille n'est détectée à la position d'étiquetage après une temporisation, le système déclenche un défaut d'arrivée, arrête les moteurs et affiche une alerte à l'opérateur.

```

IF NOT BottleLabellingPositionSensorDebounced THEN
  BottleArrivalWaitTimer(IN := TRUE, PT := T#6S);
  IF BottleArrivalWaitTimer.Q THEN
    BottleArrivalFault := TRUE;
    Alarm := TRUE;
    M_HMI_StatusText := 'ERROR: Bottle not detected at labeling position!';
    Conveyor_Motor_SpeedSetpoint := 0.0;
    Conveyor_Motor_State := FALSE;
    SideConveyor_Left_SpeedSetpoint := 0.0;
    SideConveyor_Left_State := FALSE;
    SideConveyor_Right_SpeedSetpoint := 0.0;
    SideConveyor_Right_State := FALSE;
    Label_Tip_Motor_SpeedSetpoint := 0.0;
    Label_Tip_Motor_State := FALSE;
    SV_Knife := FALSE;
    RETURN;
  END_IF
ELSE
  BottleLabellingReady := TRUE;
  BottleArrivalWaitTimer(IN := FALSE);
END_IF

```

Figure IV.74 Temporisation et défaut d'arrivée bouteille

- Si le tip n'est pas détecté dans le temps imparti, le moteur tip est relancé. Si le défaut persiste, les moteurs sont arrêtés et un message d'erreur s'affiche.

```

IF NOT LabelTipSensorDebounced THEN
  IF NOT ManualOverride_LabelTipMotor THEN
    Label_Tip_Motor_SpeedSetpoint := 80.0;
    Label_Tip_Motor_State := TRUE;
  END_IF
  LabelTipWaitTimer(IN := TRUE, PT := T#5S);
  IF LabelTipWaitTimer.Q THEN
    LabelTipNotReady := TRUE;
    M_HMI_StatusText := 'ERROR: Label tip not ready!';
    Conveyor_Motor_SpeedSetpoint := 0.0;
    Conveyor_Motor_State := FALSE;
    SideConveyor_Left_SpeedSetpoint := 0.0;
    SideConveyor_Left_State := FALSE;
    SideConveyor_Right_SpeedSetpoint := 0.0;
    SideConveyor_Right_State := FALSE;
    Label_Tip_Motor_SpeedSetpoint := 0.0;
    Label_Tip_Motor_State := FALSE;
    SV_Knife := FALSE;
    RETURN;
  END_IF
ELSE
  LabelTipReady := TRUE;
  LabelTipWaitTimer(IN := FALSE);
END_IF

```

Figure IV.75 Temporisation et défaut tip

- En cas d'absence de détection du gap après temporisation, le système arrête les moteurs et affiche le défaut à l'opérateur pour intervention.

```

IF LabelTipReady AND BottleLabelingReady AND NOT LabelGapSensorDebounced THEN
  LabelGapWaitTimer(IN := TRUE, PT := T#3S);
  IF LabelGapWaitTimer.Q THEN
    LabelGapNotDetected := TRUE;
    M_HMI_StatusText := 'ERROR: Label gap not detected!';
    Conveyor_Motor_SpeedSetpoint := 0.0;
    Conveyor_Motor_State := FALSE;
    SideConveyor_Left_SpeedSetpoint := 0.0;
    SideConveyor_Left_State := FALSE;
    SideConveyor_Right_SpeedSetpoint := 0.0;
    SideConveyor_Right_State := FALSE;
    Label_Tip_Motor_SpeedSetpoint := 0.0;
    Label_Tip_Motor_State := FALSE;
    SV_Knife := FALSE;
    RETURN;
  END_IF
ELSE
  LabelGapWaitTimer(IN := FALSE);
END_IF

```

Figure IV.76 Temporisation et défaut gap

- Lorsque la pointe et le gap sont détectés, le vérin du couteau est activé pour couper l'étiquette. Une temporisation assure le bon déroulement de la coupe. Après la coupe, le cycle d'étiquetage est marqué comme terminé.

```

IF LabelTipReady AND BottleLabelingReady AND LabelGapSensorDebounced THEN
  IF NOT ManualOverride_SV_Knife THEN
    SV_Knife := TRUE;
  END_IF
  LabelCutTimer(IN := TRUE, PT := T#1S);
  IF LabelCutTimer.Q THEN
    IF NOT ManualOverride_SV_Knife THEN
      SV_Knife := FALSE;
    END_IF
    LabelCutDone := TRUE;
    LabelCutTimer(IN := FALSE);
  END_IF
END_IF

IF LabelCutDone THEN
  BottleLabelingDone := TRUE;
  M_HMI_StatusText := 'Labeling complete. Bottle advancing to next stage.';
END_IF

```

Figure IV.77 Action du couteau et coupe d'étiquette

2. Transition a prochaine étape :

 **BottleLabelingDone - TRUE**

Figure IV.78 Transition entre Labeling et QC

IV.3.4.10. QualityControl

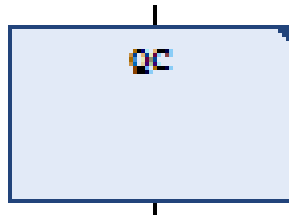


Figure IV.79 Bloc d'étape QC

Ce bloc gère le contrôle qualité en bout de ligne. Il vérifie la présence de l'étiquette et du bouchon, le poids et la forme de la bouteille à l'aide de capteurs et vision. En cas de défaut, la bouteille est rejetée par une électrovanne dédiée ; sinon, elle continue sur le convoyeur. Tous les états et alarmes sont remontés à l'IHM pour l'opérateur.

1. Code d'action active :

- Prépare les variables et états pour commencer l'étape de contrôle qualité.

```
M_HMI_StatusText := 'Quality control in progress...';  
BottleArrivedQC := FALSE;  
LabelMissing := FALSE;  
CapMissing := FALSE;  
BottleTooLight := FALSE;  
BottleDeformed := FALSE;  
BottleAccepted := FALSE;  
BottleRejected := FALSE;
```

Figure IV.80 Initialisation et indicateurs d'état QC

- Gère le mode manuel/automatique pour le moteur convoyeur selon la sélection opérateur.

```
IF ManualOverride_ConveyorMotor THEN  
    Conveyor_Motor_SpeedSetpoint := M_HMI_ConveyorMotor_Speed;  
    Conveyor_Motor_State := M_HMI_ConveyorMotor_Control;  
ELSE  
    Conveyor_Motor_SpeedSetpoint := 75.0;  
    Conveyor_Motor_State := TRUE;  
END_IF
```

Figure IV.81 Commande manuelle et automatique du convoyeur

- Surveille le courant absorber et arrête la ligne en cas de surcharge moteur.

```

Conveyor_Motor_Absorp_mA := 1200.0 * (Conveyor_Motor_SpeedSetpoint / 100.0);

ConveyorMotor_OverloadFault := FALSE;
IF Conveyor_Motor_Absorp_mA > 1300.0 THEN
    ConveyorMotor_OverloadFault := TRUE;
    Conveyor_Motor_SpeedSetpoint := 0.0;
    Conveyor_Motor_State := FALSE;
    SV_RejectGate := FALSE;
    M_HMI_StatusText := 'ERROR: Conveyor motor overload at QC! Check motor and wiring.';
    Alarm := TRUE;
    RETURN;
END_IF

```

Figure IV.82 Calcul du courant moteur et détection de surcharge

- Permet à l'opérateur de piloter la trappe de rejet ou laisse le programme agir automatiquement.

```

IF ManualOverride_SV_RejectGate THEN
    SV_RejectGate := M_HMI_SV_RejectGate_Control;
ELSE
    SV_RejectGate := FALSE;
END_IF

```

Figure IV.83 Commande manuelle/automatique de l'électrovanne de rejet

- Ce bloc vérifie la présence de l'étiquette et du bouchon, le poids et la forme de la bouteille. Chaque défaut détecté active un drapeau d'anomalie.

```

IF BottleQCSensor THEN
    BottleArrivedQC := TRUE;
    Conveyor_Motor_SpeedSetpoint := 0.0;
    Conveyor_Motor_State := FALSE;

    IF NOT VisionLabelPresent THEN
        LabelMissing := TRUE;
    END_IF

    IF NOT VisionCapPresent THEN
        CapMissing := TRUE;
    END_IF

    IF WeightSensorValue < BottleMinWeight THEN
        BottleTooLight := TRUE;
    END_IF

    IF NOT BottleShapeOK THEN
        BottleDeformed := TRUE;
    END_IF

```

Figure IV.84 Inspection et détection des défauts QC

- Ce bloc décide si la bouteille est acceptée (elle poursuit sa route) ou rejetée (dirigée vers la trappe de rebut), selon les résultats des vérifications précédentes.

```

IF VisionLabelPresent AND VisionCapPresent AND (WeightSensorValue >= BottleMinWeight) AND BottleShapeOK THEN
    BottleAccepted := TRUE;
    Conveyor_MotorVFD_Speed := 75.0;
    IF NOT ManualOverride_SV_RejectGate THEN
        SV_RejectGate := FALSE;
    END_IF
    M_HMI_StatusText := 'Bottle passed quality control. Advancing to next station.';
END_IF

IF LabelMissing OR CapMissing OR BottleTooLight OR BottleDeformed THEN
    BottleRejected := TRUE;
    Conveyor_MotorVFD_Speed := 75.0;
    IF SV_RejectGateSensor AND NOT ManualOverride_SV_RejectGate THEN
        SV_RejectGate := TRUE;
    END_IF
    M_HMI_StatusText := 'Bottle failed QC. Diverting to reject bin.';
END_IF
END_IF

```

Figure IV.85 Décision d'acceptation ou de rejet QC

- Transfert tous les états, défauts et résultats QC vers l'IHM pour suivi opérateur.

```

M_HMI_ConveyorMotor_Speed := Conveyor_Motor_SpeedSetpoint;
M_HMI_ConveyorMotor_Absorp_mA := Conveyor_Motor_Absorp_mA;
M_HMI_ConveyorMotor_On := Conveyor_Motor_State;
M_HMI_ConveyorMotor_Overload := ConveyorMotor_OverloadFault;
M_HMI_SV_RejectGate := SV_RejectGate;
M_HMI_BottleArrivedQC := BottleArrivedQC;
M_HMI_LabelMissing := LabelMissing;
M_HMI_CapMissing := CapMissing;
M_HMI_BottleTooLight := BottleTooLight;
M_HMI_BottleDeformed := BottleDeformed;
M_HMI_BottleAccepted := BottleAccepted;
M_HMI_BottleRejected := BottleRejected;

```

Figure IV.86 Mapping des états et défauts vers l'IHM

2. Transition a prochaine étape :

 BottleRejected OR BottleAccepted

Figure IV.87 Transition entre QC et saut

IV.3.4.11. Saut automatique au Start

 Start

Figure IV.88 Symbol du saut à l'étape Initial Start

Ce bloc assure la transition après le contrôle qualité. Il ferme la boucle du cycle de production en repositionnant le système au point de départ et relance la simulation pour traiter une nouvelle bouteille, garantissant ainsi l'enchaînement continu du processus automatisé.

IV.3.5. Fonctionnement des erreurs

IV.3.5.1. Les erreurs d'étape TankCheck

IV.3.5.1.1. L'erreur TankFault

1. Transition :



Figure IV.89 Transition entre l'étape TankCheck et l'étape d'erreur TankFault

2. Ce bloc affiche le message d'erreur sur l'interface opérateur et active l'alarme visuelle et sonore dès que le défaut de cuve est détecté.

```
M_HMI_StatusText := 'ERROR: Tank not detected at tank check station.';
AlarmLight := TRUE;
Buzzer := TRUE;
```

Figure IV.90 Affichage de l'erreur sur IHM et activation des alarmes

3. Ce bloc gère l'activation ou l'arrêt du moteur PumpMotor1 selon le mode manuel choisi par l'opérateur, permettant d'intervenir même en cas de défaut.

```
IF ManualOverride_pumpmotor1 THEN
  PumpMotor1_State := M_HMI_PumpMotor1_Control;
  IF PumpMotor1_State THEN
    PumpMotor1_SpeedSetpoint := M_HMI_PumpMotor1_Speed;
  ELSE
    PumpMotor1_SpeedSetpoint := 0.0;
  END_IF
ELSE
  PumpMotor1_State := FALSE;
  PumpMotor1_SpeedSetpoint := 0.0;
END_IF
```

Figure IV.91 Forçage manuel du moteur de pompe 1

4. Ce morceau de code permet d'enregistrer le défaut TankFault une seule fois lors de son apparition, et autorise l'opérateur à réinitialiser le défaut et les alarmes via une action manuelle, pour reprendre le cycle normal.

```
IF NOT TankFaultLogged THEN
  TankFaultLogged := TRUE;
END_IF

IF OperatorResetRequested THEN
  TankFault := FALSE;
  TankFaultLogged := FALSE;
  AlarmLight := FALSE;
  Buzzer := FALSE;
END_IF
```

Figure IV.92 Enregistrement et Réinitialisation du défaut par l'opérateur

5. Ce bloc assure la remise à zéro automatique du défaut TankFault et des alarmes si le capteur détecte de nouveau la cuve en position.

```

IF Sensor_Tank_Low THEN
    TankFault := FALSE;
    TankFaultLogged := FALSE;
    AlarmLight := FALSE;
    Buzzer := FALSE;
    M_HMI_StatusText := 'TankFault problem solved!';
END_IF

```

Figure IV.93 Récupération automatique si le capteur détecte la cuve

6. Ce code permet à l'opérateur de désactiver manuellement l'alarme et l'avertisseur sonore, même si le défaut persiste.

```

IF NOT AlarmLight_ManualOff THEN
    AlarmLight := TRUE;
ELSE
    AlarmLight := FALSE;
END_IF

IF NOT Buzzer_ManualOff THEN
    Buzzer := TRUE;
ELSE
    Buzzer := FALSE;
END_IF

```

Figure IV.94 Forçage manuel de l'alarme lumineuse et de l'avertisseur sonore

7. Les transitions et les sautes

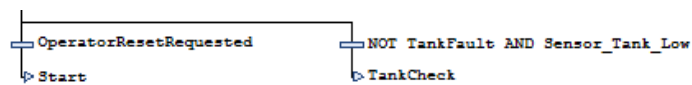


Figure IV.95 Les transitions et sautes d'étape TankFault

IV.3.5.1.2. L'erreur MidSensorError

1. Transition :

```

MidSensorError = TRUE

```

Figure IV.96 Transition entre TankCheck et MidSensorError

2. Ce bloc affiche le message d'erreur sur l'interface opérateur et active l'alarme visuelle et sonore dès que le défaut de cuve est détecté

```

M_HMI_StatusText := 'ERROR: Mid tank level sensor activation timeout!';
AlarmLight := TRUE;
Buzzer := TRUE;

```

Figure IV.97 Affichage de l'erreur sur IHM et activation des alarmes

3. Forçage manuel du moteur de la pompe 1 comme dans l'étape TankFault (Figure IV.91)
4. Ce morceau de code permet d'enregistrer le défaut MidSensorFault une seule fois lors de son apparition, et autorise l'opérateur à réinitialiser le défaut et les alarmes via une action manuelle, pour reprendre le cycle normal.

```

IF NOT MidSensorErrorLogged THEN
    MidSensorErrorLogged := TRUE;
END_IF

IF OperatorResetRequested THEN
    MidSensorError := FALSE;
    MidSensorErrorLogged := FALSE;
    AlarmLight := FALSE;
    Buzzer := FALSE;
END_IF

```

Figure IV.98 Enregistrement du default et Réinitialisation du défaut par l'opérateur

5. Ce bloc restaure l'état normal du système si le capteur intermédiaire détecte à nouveau le niveau, désactive les alarmes et met à jour l'IHM.

```

IF Sensor_Tank_Low AND Sensor_Tank_Mid THEN
    MidSensorError := FALSE;
    MidSensorErrorLogged := FALSE;
    TankReady := TRUE;
    AlarmLight := FALSE;
    Buzzer := FALSE;
    M_HMI_StatusText := 'TankFault problem solved!';
END_IF

```

Figure IV.99 Rétablissement automatique si le capteur revient à la normale

6. Forçage manuel de l'alarme lumineuse et de l'avertisseur sonore (Figure IV.94)
7. Les transitions et les sautes

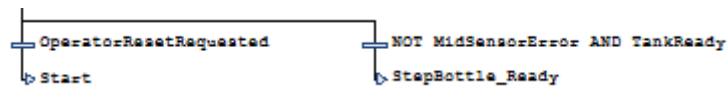


Figure IV.100 Les transitions et sautes d'étape MidSensorError

IV.3.5.1.3. L'erreur MotorPump1Overload

1. Transition

```

PumpMotor1_OverloadFault

```

Figure IV.101 Transition entre TankCheck et MotorPump1Overload

2. Message IHM et activation d'alarme et sonore

```

M_HMI_StatusText := 'ERROR: Pump motor overload! Check pump, drive, or wiring.';
AlarmLight := TRUE;
Buzzer := TRUE;

```

Figure IV.102 Affichage de l'erreur sur IHM et activation des alarmes

3. Forçage manuel du moteur de la pompe 1 comme dans l'étape TankFault (Figure IV.89)
4. Ce morceau de code permet d'enregistrer le défaut MotorPump1Overload une seule fois lors de son apparition, et autorise l'opérateur à réinitialiser le défaut et les alarmes via une action manuelle, pour reprendre le cycle normal.

```

IF NOT PumpMotor1_OverloadFaultLogged THEN
    PumpMotor1_OverloadFaultLogged := TRUE;
END_IF

IF OperatorResetRequested THEN
    PumpMotor1_OverloadFault := FALSE;
    PumpMotor1_OverloadFaultLogged := FALSE;
    AlarmLight := FALSE;
    Buzzer := FALSE;
END_IF

```

Figure IV.103 Enregistrement du défaut et Réinitialisation du défaut par l'opérateur

5. Ce code restaure l'état normal du système et désactive les alarmes si le courant absorbé par le moteur de pompe revient sous le seuil critique.

```

IF PumpMotor1_Absorp_mA < 1900.0 THEN
    PumpMotor1_OverloadFault := FALSE;
    PumpMotor1_OverloadFaultLogged := FALSE;
    AlarmLight := FALSE;
    Buzzer := FALSE;
    M_HMI_StatusText := 'Pump motor overload resolved!';
END_IF

```

Figure IV.104 Rétablissement automatique si le courant redevient normal

6. Forçage manuel de l'alarme lumineuse et de l'avertisseur sonore (Figure IV.94)
7. Les transitions et sautes

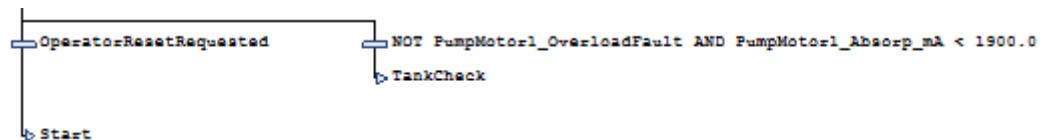


Figure IV.105 Les transitions et sautes d'étape MotorPump1Overload

IV.3.5.2. Les erreurs d'étape BottleReady

IV.3.5.2.1 L'erreur BottleFault

1. Transition

```

BottleFault = TRUE

```

Figure IV.106 Transition entre BottleReady et BottleFault

2. Message IHM et activation d'alarme et sonore

```
M_HMI_StatusText := 'ERROR: Bottle not detected at .!';  
AlarmLight := TRUE;  
Buzzer := TRUE;
```

Figure IV.107 Affichage de l'erreur sur IHM et activation des alarmes

3. Ce code permet à l'opérateur de contrôler manuellement le vérin de l'alimenteur de bouteilles (SV) et le moteur du convoyeur, ou de laisser le système les désactiver automatiquement en cas de défaut.

```
IF ManualOverride_SV_BottleFeederGate THEN  
    SV_BottleFeederGate := M_HMI_SV_BottleFeederGate_Control;  
ELSE  
    SV_BottleFeederGate := FALSE;  
END_IF  
  
IF ManualOverride_ConveyorMotor THEN  
    ConveyorMotor_State := M_HMI_ConveyorMotor_Control;  
    IF ConveyorMotor_State THEN  
        Conveyor_MotorVFD_Speed := M_HMI_ConveyorMotor_Speed;  
    ELSE  
        Conveyor_MotorVFD_Speed := 0.0;  
    END_IF  
ELSE  
    ConveyorMotor_State := FALSE;  
    Conveyor_MotorVFD_Speed := 0.0;  
END_IF
```

Figure IV.108 Forçage manuel du vérin et du moteur convoyeur

4. Ce morceau de code enregistre le défaut BottleFault, et autorise l'opérateur à réinitialiser le défaut et les alarmes via une action manuelle, pour reprendre le cycle normal.

```
IF NOT BottleFaultLogged THEN  
    BottleFaultLogged := TRUE;  
END_IF  
  
IF OperatorResetRequested THEN  
    BottleFault := FALSE;  
    BottleFaultLogged := FALSE;  
    AlarmLight := FALSE;  
    Buzzer := FALSE;  
END_IF
```

Figure IV.109 Enregistrement du défaut et Réinitialisation du défaut par l'opérateur

5. Ce code utilise une temporisation pour vérifier que la bouteille est bien détectée pendant 500 ms avant de relancer le cycle normal, réinitialiser le défaut et désactiver les alarmes.

```

IF FeederBottle_Sensor THEN
  BottleSensorDebounceTimer(IN := TRUE, PT := T#500ms);
  IF BottleSensorDebounceTimer.Q THEN
    BottleReadyDebounced := TRUE;
  ELSE
    BottleReadyDebounced := FALSE;
  END_IF
ELSE
  BottleSensorDebounceTimer(IN := FALSE);
  BottleReadyDebounced := FALSE;
END_IF

IF BottleReadyDebounced THEN
  SV_BottleFeederGate := TRUE;
  Conveyor_MotorVFD_Speed := 75.0;
  BottleFeederGateTimer(IN := TRUE, PT := T#1S);
  IF BottleFeederGateTimer.Q THEN
    BottleReady := TRUE;
    BottleFault := FALSE;
    AlarmLight := FALSE;
    Buzzer := FALSE;
    BottleFaultLogged := FALSE;
    M_HMI_StatusText := 'Bottle fault resolved!';
  ELSE
    SV_BottleFeederGate := FALSE;
    Conveyor_MotorVFD_Speed := 0.0;
    BottleFeederGateTimer(IN := FALSE);
  END_IF
END_IF

```

Figure IV.110 Récupération automatique si la bouteille est détectée

6. Forçage manuel de l'alarme lumineuse et de l'avertisseur sonore (Figure IV.94)
7. Les transitions et sautes

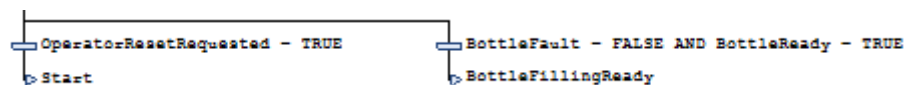


Figure IV.111 Les transitions et sautes d'étape BottleReady

IV.3.5.2.2. L'erreur ConveyorMotorFaultBottleReady

1. Transition

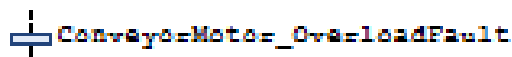


Figure IV.112 Transition entre BottleReady et ConveyorMotorFaultBottleReady

2. Message IHM et activation d'alarme et sonore

```

M_HMI_StatusText := 'ERROR: Conveyor motor overload! Inspect conveyor, drive, or wiring.';
AlarmLight := TRUE;
Buzzer := TRUE;

```

Figure IV.113 Affichage de l'erreur sur IHM et activation des alarmes

3. Ce code permet à l'opérateur de contrôler manuellement le moteur convoyeur ou de laisser le système le désactiver automatiquement en cas de surcharge.

```

IF ManualOverride_ConveyorMotor THEN
    ConveyorMotor_State := M_HMI_ConveyorMotor_Control;
    IF ConveyorMotor_State THEN
        Conveyor_MotorVFD_Speed := M_HMI_ConveyorMotor_Speed;
    ELSE
        Conveyor_MotorVFD_Speed := 0.0;
    END_IF
ELSE
    ConveyorMotor_State := FALSE;
    Conveyor_MotorVFD_Speed := 0.0;
END_IF

```

Figure IV.114 Forçage manuel du moteur convoyeur

4. Ce morceau de code enregistre le défaut ConveyorMotorFaultBottleReady, et autorise l'opérateur à réinitialiser le défaut et les alarmes via une action manuelle, pour reprendre le cycle normal.

```

IF NOT ConveyorMotor_OverloadFaultLogged THEN
    ConveyorMotor_OverloadFaultLogged := TRUE;
END_IF

IF OperatorResetRequested THEN
    ConveyorMotor_OverloadFault := FALSE;
    ConveyorMotor_OverloadFaultLogged := FALSE;
    AlarmLight := FALSE;
    Buzzer := FALSE;
END_IF

```

Figure IV.115 Enregistrement du défaut et Réinitialisation du défaut par l'opérateur

5. Ce code restaure l'état normal du système et désactive les alarmes si le courant absorbé par le moteur du convoyeur repasse sous le seuil critique.

```

IF Conveyor_Motor_Absorp_mA < 1900.0 THEN
    AlarmLight := FALSE;
    Buzzer := FALSE;
    ConveyorMotor_OverloadFaultLogged := FALSE;
    ConveyorMotor_OverloadFault := FALSE;
    M_HMI_StatusText := 'Conveyor motor overload resolved!';
END_IF

```

Figure IV.116 Rétablissement automatique si le courant du moteur redevient normal

6. Forçage manuel de l'alarme lumineuse et de l'avertisseur sonore (Figure IV.94)
7. Les transitions et sautes

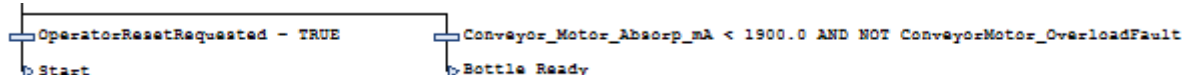


Figure IV.117 Les transitions et sautes d'étape ConveyorMotorFaultBottleReady

IV.3.5.3. Les erreurs d'étape BottleFillingReady

IV.3.5.3.1. L'erreur BottleFillingPositionFault

1. Transition



Figure IV.118 Transition entre BottleFillingReady et BottleFillingPositionFault

2. Message IHM et activation d'alarme et sonore

```
M_HMI_StatusText := 'ERROR: No bottle at filling station. Please check infeed!';  
AlarmLight := TRUE;  
Buzzer := TRUE;
```

Figure IV.119 Affichage de l'erreur sur IHM et activation des alarmes

3. Le control manuelle du moteur du convoyeur (Figure IV.112)
4. Ce morceau de code enregistrer le défaut BottleFillingPositionFault, et autorise l'opérateur à réinitialiser le défaut et les alarmes via une action manuelle, pour reprendre le cycle normal.

```
IF NOT FillingPositionFaultLogged THEN  
    FillingPositionFaultLogged := TRUE;  
END_IF  
  
IF OperatorResetRequested THEN  
    AlarmLight := FALSE;  
    Buzzer := FALSE;  
    FillingPositionFaultLogged := FALSE;  
    FillingPositionFault := FALSE;  
END_IF
```

Figure IV.120 Enregistrement du default et Réinitialisation du défaut par l'opérateur

5. Ce code utilise une temporisation (Filtrage anti-rebond 500ms) pour vérifier que la bouteille est bien présente à la station de remplissage avant de réinitialiser le défaut et les alarmes.

```
IF FillingConveyorSensor THEN  
    FillingConveyorDebounceTimer(IN := TRUE, PT := T#500MS);  
    IF FillingConveyorDebounceTimer.Q THEN  
        FillingConveyorSensorDebounced := TRUE;  
    ELSE  
        FillingConveyorSensorDebounced := FALSE;  
    END_IF  
ELSE  
    FillingConveyorDebounceTimer(IN := FALSE);  
    FillingConveyorSensorDebounced := FALSE;  
END_IF  
  
IF FillingConveyorSensorDebounced THEN  
    AlarmLight := FALSE;  
    Buzzer := FALSE;  
    Conveyor_MotorVFD_Speed := 0.0;  
    BottleWaitTimerFilling(IN := FALSE);  
    FillingReady := TRUE;  
    FillingPositionFault := FALSE;  
    FillingPositionFaultLogged := FALSE;  
    M_HMI_StatusText := 'Bottle is ready at filling position.';  
END_IF
```

Figure IV.121 Récupération automatique avec temporisation du capteur

6. Forçage manuel de l'alarme lumineuse et de l'avertisseur sonore (Figure IV.94)
7. Les transitions et sautes

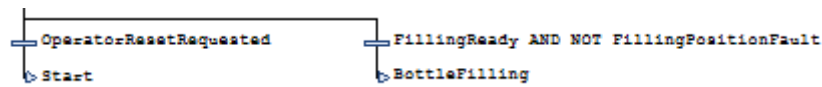


Figure IV.122 Les transitions et sautes d'étape BottleFillingPositionFault

IV.3.5.3.2 L'erreur ConveyorMotorFaultBottleFillingReady

Cette étape d'erreur est une précaution pour la surcharge du moteur du convoyeur pendant que la bouteille est prête pour le remplissage, ce qui signifie qu'il aura le même code du titre (IV.3.5.5)

- Les transitions et sautes

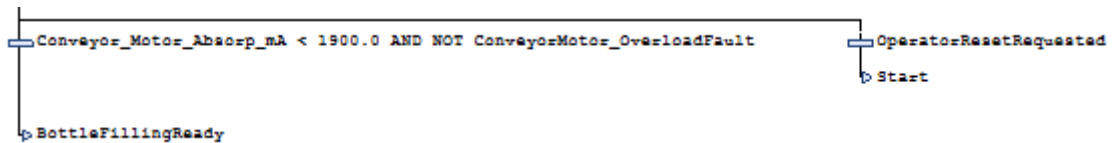


Figure IV.123 Les transitions et sautes d'étape ConveyorMotorFaultBottleFillingReady

IV.3.5.4. Les erreurs d'étape BottleFilling

IV.3.5.4.1. L'erreur FillingTimeoutFault

1. Transition



Figure IV.124 Transition entre BottleFilling et FillingTimeoutFault

2. Message IHM et activation d'alarme et sonore

```
M_HMI_StatusText := 'ERROR: Filling exceeded allowed time! Check sensor/nozzle or bottle.';
AlarmLight := TRUE;
Buzzzer := TRUE;
```

Figure IV.125 Affichage de l'erreur sur IHM et activation des alarmes

3. Ce code permet à l'opérateur d'activer ou désactiver manuellement la pompe et les vérins d'air, ou de les placer automatiquement en position de sécurité en cas d'erreur et met à jour les valeurs de la vitesse et du courant moteur sur l'IHM.

```

IF ManualOverride_PumpMotor2VFD THEN
    PumpMotor2VFD_State := M_HMI_PumpMotor2VFD_Control;
    IF PumpMotor2VFD_State THEN
        PumpMotor2VFD_Speed := M_HMI_PumpMotor2VFD_Speed;
    ELSE
        PumpMotor2VFD_Speed := 0.0;
    END_IF
ELSE
    PumpMotor2VFD_State := FALSE;
    PumpMotor2VFD_Speed := 0.0;
END_IF

IF ManualOverride_SV_Air_NozzleFilling THEN
    SV_Air_NozzleFilling := M_HMI_SV_Air_NozzleFilling_Control;
ELSE
    SV_Air_NozzleFilling := TRUE;
END_IF

IF ManualOverride_SV_Air_FillingClamps THEN
    SV_Air_FillingClamps := M_HMI_SV_Air_FillingClamps_Control;
ELSE
    SV_Air_FillingClamps := TRUE;
END_IF

M_HMI_PumpMotor2_Speed := PumpMotor2VFD_Speed;
M_HMI_PumpMotor2_Absorp_mA := PumpMotor2VFD_Absorp_mA;

```

Figure IV.126 Forçage manuel et mise en sécurité des actionneurs et synchronisation d'IHM

4. Le code pour réinitialiser le défaut, le journal et les alarmes, et de remettre les actionneurs en position normale via l'IHM et enregistre l'erreur une seule fois lors de sa détection.

```

IF NOT FillingTimeoutFaultLogged THEN
    FillingTimeoutFaultLogged := TRUE;
END_IF

IF OperatorResetRequested THEN
    Buzzer := FALSE;
    AlarmLight := FALSE;
    FillingTimeoutFaultLogged := FALSE;
    FillingTimeoutFault := FALSE;
    SV_Air_NozzleFilling := FALSE;
    SV_Air_FillingClamps := FALSE;
END_IF

```

Figure IV.127 Enregistrement du défaut et Réinitialisation du défaut par l'opérateur

5. Ce code utilise une temporisation (Filtrage anti-rebond 500 ms) pour vérifier que la bouteille est bien remplie avant de réinitialiser le défaut et les alarmes.

```

IF FullBottleSensor THEN
    FullBottleDebounceTimer(IN := TRUE, PT := T#500MS);
    IF FullBottleDebounceTimer.Q THEN
        FullBottleSensorDebounced := TRUE;
    ELSE
        FullBottleSensorDebounced := FALSE;
    END_IF
ELSE
    FullBottleDebounceTimer(IN := FALSE);
    FullBottleSensorDebounced := FALSE;
END_IF

IF FullBottleSensorDebounced THEN
    PumpMotor2VFD_Speed := 0.0;
    SV_Air_NozzleFilling := FALSE;
    SV_Air_FillingClamps := FALSE;
    FillingDone := TRUE;
    FillingTimeoutFault := FALSE;
    AlarmLight := FALSE;
    Buzzer := FALSE;
    FillingTimeoutFaultLogged := FALSE;
    M_HMI_StatusText := 'Filling timeout fault resolved!';
END_IF

```

Figure IV.128 Récupération automatique si la bouteille est détectée pleine

6. Forçage manuel de l'alarme lumineuse et de l'avertisseur sonore (Figure IV.94)

7. Les transitions et sautes

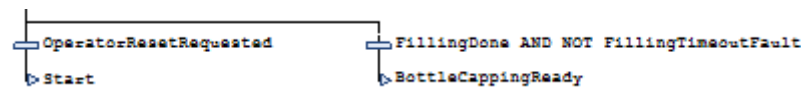


Figure IV.129 Les transitions et sautes d'étape FillingTimeoutFault

IV.3.5.4.2. L'erreur NozzlePressureFault

1. Transition



Figure IV.130 Transition entre BottleFilling et NozzlePressureFault

2. Message IHM et activation d'alarme et sonore

```

M_HMI_StatusText := 'ERROR: Nozzle pressure lost! Check air supply and nozzle.';
AlarmLight := TRUE;
Buzzer := TRUE;
  
```

Figure IV.131 Affichage de l'erreur sur IHM et activation des alarmes

- Forçage manuel et mise en sécurité des actionneurs et synchronisation d'IHM (Figure IV.126)
- Réinitialiser le défaut, le journal et les alarmes, et de remettre les actionneurs en position normale via l'IHM et enregistre l'erreur une seule fois lors de sa détection.

```

IF NOT NozzlePressureFaultLogged THEN
  NozzlePressureFaultLogged := TRUE;
END_IF

IF OperatorResetRequested THEN
  Buzzer := FALSE;
  AlarmLight := FALSE;
  NozzlePressureFaultLogged := FALSE;
  FillingNozzlePressureFault := FALSE;
  SV_Air_NozzleFilling := FALSE;
  SV_Air_FillingClamps := FALSE;
END_IF
  
```

Figure IV.132 Enregistrement du défaut et Réinitialisation du défaut par l'opérateur

- Ce code utilise une temporisation (Filtrage anti-rebond 500 ms) pour vérifier que la pression de la buse est rétablie avant de réinitialiser le défaut et les alarmes.

```

IF NozzlePressureSensor THEN
  NozzlePressureDebounceTimer(IN := TRUE, PT := T#500MS);
  IF NozzlePressureDebounceTimer.Q THEN
    NozzlePressureSensorDebounced := TRUE;
  ELSE
    NozzlePressureSensorDebounced := FALSE;
  END_IF
ELSE
  NozzlePressureDebounceTimer(IN := FALSE);
  NozzlePressureSensorDebounced := FALSE;
END_IF
IF NozzlePressureSensorDebounced THEN
  FillingNozzlePressureFault := FALSE;
  AlarmLight := FALSE;
  Buzzer := FALSE;
  NozzlePressureFaultLogged := FALSE;
  M_HMI_StatusText := 'Nozzle pressure fault resolved!';
END_IF

```

Figure IV.133 Récupération automatique si la pression de buse revient

6. Forçage manuel de l'alarme lumineuse et de l'avertisseur sonore (Figure IV.94)
7. Les transitions et sautes

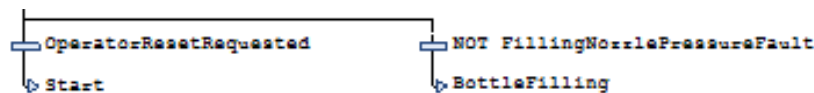


Figure IV.134 Les transition et sautes d'étape NozzlePressureFault

IV.3.5.4.3. L'erreur FillingSensorFault d'étape BottleFilling

1. Transition



Figure IV.135 Transition entre BottleFilling et FillingSensorFault

2. Message IHM et activation d'alarme et sonore

```

M_HMI_StatusText := 'ERROR: Bottle full sensor not triggered! Check bottle and sensor.';
AlarmLight := TRUE;
Buzzer := TRUE;

```

Figure IV.136 Affichage de l'erreur sur IHM et activation des alarmes

3. Forçage manuel et mise en sécurité des actionneurs et synchronisation d'IHM (Figure IV.126)
4. Réinitialiser le défaut, le journal et les alarmes, et de remettre les actionneurs en position normale via l'IHM et enregistre l'erreur une seule fois lors de sa détection.

```

IF NOT FullBottleSensorFaultLogged THEN
  FullBottleSensorFaultLogged := TRUE;
END_IF

IF OperatorResetRequested THEN
  Buzzer := FALSE;
  AlarmLight := FALSE;
  FullBottleSensorFaultLogged := FALSE;
  FullBottleSensorFault := FALSE;
END_IF

```

Figure IV.137 Enregistrement du default et Réinitialisation du défaut par l'opérateur

- Utilisation d'une temporisation (Filtrage anti-rebond 500 ms) pour vérifier que le capteur est bien déclenché avant de réinitialiser le défaut et les alarmes.

```

IF FullBottleSensor THEN
  FullBottleDebounceTimer(IN := TRUE, PT := T#500MS);
  IF FullBottleDebounceTimer.Q THEN
    FullBottleSensorDebounced := TRUE;
  ELSE
    FullBottleSensorDebounced := FALSE;
  END_IF
ELSE
  FullBottleDebounceTimer(IN := FALSE);
  FullBottleSensorDebounced := FALSE;
END_IF

IF FullBottleSensorDebounced THEN
  PumpMotor2VFD_Speed := 0.0;
  SV_Air_NozzleFilling := FALSE;
  SV_Air_FillingClamps := TRUE;
  FillingDone := TRUE;
  FullBottleSensorFault := FALSE;
  AlarmLight := FALSE;
  Buzzer := FALSE;
  FullBottleSensorFaultLogged := FALSE;
  M_HMI_StatusText := 'Full bottle sensor fault resolved!';
END_IF

```

Figure IV.138 Récupération automatique si le capteur bouteille pleine est déclenché

- Forçage manuel de l'alarme lumineuse et de l'avertisseur sonore (Figure IV.94)
- Les transitions et sautes

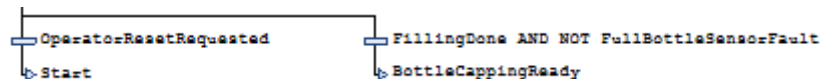


Figure IV.139 Les transition et sautes d'étape FillingSensorFault

IV.3.5.4.4. L'erreur FillingMotorOverload

- Transition

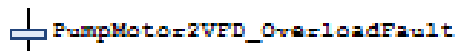


Figure IV.140 Transition entre BottleFilling et FillingMotorOverload

- Message IHM et activation d'alarme et sonore

```

M_HMI_StatusText := 'ERROR: Pump motor overload detected! Check motor and wiring.';
AlarmLight := TRUE;
Buzzer := TRUE;

```

Figure IV.141 Affichage de l'erreur sur IHM et activation des alarmes

- Forçage manuel et mise en sécurité des actionneurs et synchronisation d'IHM (Figure IV.126)
- Réinitialiser le défaut, le journal et les alarmes, et de remettre les actionneurs en position normale via l'IHM et enregistre l'erreur une seule fois lors de sa détection.

```

IF NOT PumpMotor2VFD_OverloadFaultLogged THEN
    PumpMotor2VFD_OverloadFaultLogged := TRUE;
END_IF

IF OperatorResetRequested THEN
    Buzzer := FALSE;
    AlarmLight := FALSE;
    PumpMotor2VFD_OverloadFaultLogged := FALSE;
    PumpMotor2VFD_OverloadFault := FALSE;
END_IF

```

Figure IV.142 Enregistrement du default et Réinitialisation du défaut par l'opérateur

5. Ce code restaure l'état normal des actionneurs et réinitialise le défaut et les alarmes si le courant absorbé par le moteur pompe 2 repasse sous le seuil critique.

```

IF PumpMotor2VFD_Absorp_mA < 1600.0 THEN
    PumpMotor2VFD_Speed := 75.0;
    SV_Air_NozzleFilling := TRUE;
    SV_Air_FillingClamps := TRUE;
    PumpMotor2VFD_OverloadFault := FALSE;
    AlarmLight := FALSE;
    Buzzer := FALSE;
    PumpMotor2VFD_OverloadFaultLogged := FALSE;
    M_HMI_StatusText := 'Pump overload fault resolved!';
END_IF

```

Figure IV.143 Récupération automatique si la surcharge moteur pompe 2 est levée

6. Forçage manuel de l'alarme lumineuse et de l'avertisseur sonore (Figure IV.94)
7. Les transitions et sautes

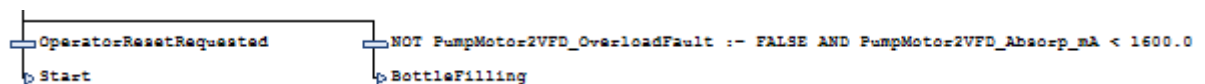


Figure IV.144 Les transitions et sautes d'étape FillingMotorOverload

IV.3.5.5. Les erreurs d'étape BottleCappingReady

IV.3.5.5.1. L'erreur CappingBottlePositionFault

1. Transition

```

| BottleCappingPositionFault

```

Figure IV.145 Transition entre BottleCappingReady et CappingBottlePositionFault

2. Message IHM et activation d'alarme et sonore

```

M_HMI_StatusText := 'ERROR: Bottle not detected at capping station. Please check bottle position!';
AlarmLight := TRUE;
Buzzer := TRUE;

```

Figure IV.146 Affichage de l'erreur sur IHM et activation des alarmes

3. Le control manuelle du moteur du convoyeur et les vérins d'air pour le mandrin (mandrin) et les pinces de capsulage

```

IF ManualOverride_ConveyorMotor THEN
    ConveyorMotor_State := M_HMI_ConveyorMotor_Control;
    IF ConveyorMotor_State THEN
        Conveyor_MotorVFD_Speed := M_HMI_ConveyorMotor_Speed;
    ELSE
        Conveyor_MotorVFD_Speed := 0.0;
    END_IF
ELSE
    ConveyorMotor_State := FALSE;
    Conveyor_MotorVFD_Speed := 0.0;
END_IF

IF ManualOverride_SV_Air_Chuck THEN
    SV_Air_Chuck := M_HMI_SV_Air_Chuck_Control;
ELSE
    SV_Air_Chuck := FALSE;
END_IF

IF ManualOverride_SV_Air_CappingClamps THEN
    SV_Air_CappingClamps := M_HMI_SV_Air_CappingClamps_Control;
ELSE
    SV_Air_CappingClamps := FALSE;
END_IF

```

Figure IV.147 Forçage manuel du moteur du convoyeur et les vérins

4. Réinitialiser le défaut, le journal et les alarmes, et de remettre les actionneurs en position normale via l'IHM et enregistre l'erreur une seule fois lors de sa détection.

```

IF NOT BottleCappingPositionFaultLogged THEN
    BottleCappingPositionFaultLogged := TRUE;
END_IF

IF OperatorResetRequested THEN
    Buzzer := FALSE;
    AlarmLight := FALSE;
    SV_Air_Chuck := FALSE;
    SV_Air_CappingClamps := FALSE;
    BottleCappingPositionFaultLogged := FALSE;
    BottleCappingPositionFault := FALSE;
END_IF

```

Figure IV.148 Enregistrement du default et Réinitialisation du défaut par l'opérateur

5. Utilisation d'une temporisation (Filtrage anti-rebond 500 ms) pour vérifier que la bouteille est bien présente avant de réinitialiser le défaut et les alarmes, puis active les vérins pour le capsulage.

```

IF CappingConveyorSensor THEN
    BottleCappingSensorDebounceTimer(IN := TRUE, PT := T#500ms);
    IF BottleCappingSensorDebounceTimer.Q THEN
        CappingConveyorSensorDebounced := TRUE;
    ELSE
        CappingConveyorSensorDebounced := FALSE;
    END_IF
ELSE
    BottleCappingSensorDebounceTimer(IN := FALSE);
    CappingConveyorSensorDebounced := FALSE;
END_IF

IF CappingConveyorSensorDebounced THEN
    Conveyor_MotorVFD_Speed := 0.0;
    SV_Air_Chuck := TRUE;
    SV_Air_CappingClamps := TRUE;
    CappingReady := TRUE;
    BottleCappingPositionFault := FALSE;
    AlarmLight := FALSE;
    Buzzer := FALSE;
    BottleCappingPositionFaultLogged := FALSE;
    M_HMI_StatusText := 'Capping position fault resolved!';
END_IF

```

Figure IV.149 Récupération automatique si la bouteille est détectée

6. Forçage manuel de l'alarme lumineuse et de l'avertisseur sonore (Figure IV.94)
7. Les transitions et sautes

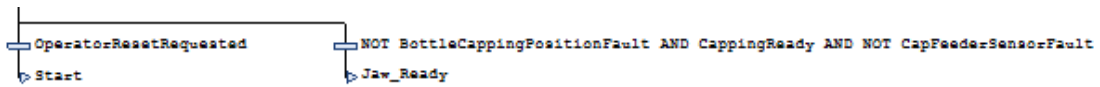


Figure IV.150 Les transitions et sautes d'étape CappingBottlePositionFault

IV.3.5.5.2. L'erreur ConveyorMotorFaultBottleCappingReady

Cette étape d'erreur est une précaution pour la surcharge du moteur du convoyeur pendant que la bouteille est à la station de bouchage, ce qui signifie qu'il aura le même code de (IV.3.5.5 et IV.3.5.7)

- Les transitions et sautes

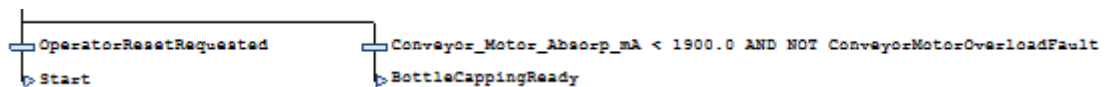


Figure IV.151 Les transitions et sautes d'étape ConveyorMotorFaultBottleCappingReady

IV.3.5.5.3. L'erreur CapFeederMotorFault

1. Transition



Figure IV.152 Transition entre BottleCappingReady et CapFeederSensorFault

2. Message IHM et activation d'alarme et sonore

```
M_HMI_StatusText := 'ERROR: Motor cap feeder overload. Please check the motor and wiring.';
AlarmLight := TRUE;
Buzzzer := TRUE;
```

Figure IV.153 Affichage de l'erreur sur IHM et activation des alarmes

3. Contrôle manuel les vérins d'air et le moteur du bouchon

```

IF ManualOverride_CapFeederMotor THEN
    CapFeederMotor_State := M_HMI_CapFeederMotor_Control;
    IF CapFeederMotor_State THEN
        CapFeederMotor_Speed := M_HMI_CapFeederMotor_Speed;
    ELSE
        CapFeederMotor_Speed := 0.0;
    END_IF
ELSE
    CapFeederMotor_State := FALSE;
    CapFeederMotor_Speed := 0.0;
END_IF
IF ManualOverride_SV_Air_Chuck THEN
    SV_Air_Chuck := M_HMI_SV_Air_Chuck_Control;
ELSE
    SV_Air_Chuck := FALSE;
END_IF
IF ManualOverride_SV_Air_CappingClamps THEN
    SV_Air_CappingClamps := M_HMI_SV_Air_CappingClamps_Control;
ELSE
    SV_Air_CappingClamps := FALSE;
END_IF

// --- Error logging (single-event) ---
IF NOT CapFeederMotor_OverloadFaultLogged THEN
    CapFeederMotor_OverloadFaultLogged := TRUE;
END_IF

```

Figure IV.154 Forçage manuel des actionneurs et moteur

4. Réinitialiser le défaut, le journal et les alarmes, et de remettre les actionneurs en position normale via l'IHM et enregistre l'erreur une seule fois lors de sa détection.

```

IF NOT CapFeederMotor_OverloadFaultLogged THEN
    CapFeederMotor_OverloadFaultLogged := TRUE;
END_IF

IF OperatorResetRequested THEN
    Buzzer := FALSE;
    AlarmLight := FALSE;
    SV_Air_Chuck := FALSE;
    SV_Air_CappingClamps := FALSE;
    CapFeederMotor_OverloadFaultLogged := FALSE;
    CapFeederMotor_OverloadFault := FALSE;
END_IF

```

Figure IV.155 Enregistrement du défaut et Réinitialisation du défaut par l'opérateur

5. Restaure l'état normal et réinitialise le défaut et les alarmes si le courant absorbé par le moteur bouchon repasse sous le seuil critique.

```

IF CapFeederMotor_Absorp_mA < 900.0 THEN
    CapFeederMotor_Speed := 100.0;
    SV_Air_Chuck := TRUE;
    SV_Air_CappingClamps := TRUE;
    CapFeederMotor_OverloadFault := FALSE;
    AlarmLight := FALSE;
    Buzzer := FALSE;
    CapFeederMotor_OverloadFaultLogged := FALSE;
    M_HMI_StatusText := 'Cap feeder motor overload fault resolved!';
END_IF

```

Figure IV.156 Récupération automatique si le courant moteur redevient normal

6. Forçage manuel de l'alarme lumineuse et de l'avertisseur sonore (Figure IV.94)

7. Les transitions et sautes

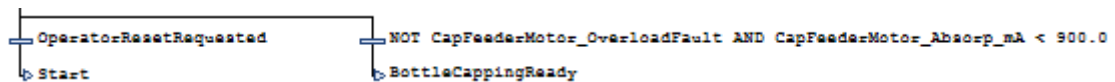


Figure IV.157 Les transition et sautes d'étape CapFeederMotorFault

IV.3.5.5.4. L'erreur CapFeederSensorFault

1. Transition



Figure IV.158 Transition entre BottleCappingReady et CapFeederSensorFault

2. Message IHM et activation d'alarme et sonore

```

M_HMI_StatusText := 'ERROR: Cap not detected at the capping station. Please check the cap supply.!!';
AlarmLight := TRUE;
Buzzer := TRUE;
  
```

Figure IV.159 Affichage de l'erreur sur IHM et activation des alarmes

3. Forçage manuel les vérins d'air et le moteur du bouchon (Figure IV.154)
4. Réinitialiser le défaut, le journal et les alarmes, et de remettre les actionneurs en position normale via l'IHM et enregistre l'erreur une seule fois lors de sa détection.

```

IF NOT CapFeederSensorFaultLogged THEN
  CapFeederSensorFaultLogged := TRUE;
END_IF

IF OperatorResetRequested THEN
  Buzzer := FALSE;
  AlarmLight := FALSE;
  SV_Air_Chuck := FALSE;
  SV_Air_CappingClamps := FALSE;
  CapFeederSensorFaultLogged := FALSE;
  CapFeederSensorFault := FALSE;
END_IF
  
```

Figure IV.160 Enregistrement du défaut et Réinitialisation du défaut par l'opérateur

5. Utilisation d'une temporisation (Filtrage anti-rebond 500 ms) pour vérifier que le bouchon est bien présente avant de réinitialiser le défaut et les alarmes, puis active les vérins pour le capsulage.

```

IF CapFeederSensor THEN
  CapFeederSensorDebounceTimer(IN := TRUE, PT := T#500ms);
  IF CapFeederSensorDebounceTimer.Q THEN
    CapFeederSensorDebounced := TRUE;
  ELSE
    CapFeederSensorDebounced := FALSE;
  END_IF
ELSE
  CapFeederSensorDebounceTimer(IN := FALSE);
  CapFeederSensorDebounced := FALSE;
END_IF

IF CapFeederSensorDebounced THEN
  CapFeederMotor_Speed := 100.0;
  SV_Air_Chuck := TRUE;
  SV_Air_CappingClamps := TRUE;
  CapFeederSensorFault := FALSE;
  AlarmLight := FALSE;
  Buzzer := FALSE;
  CapFeederSensorFaultLogged := FALSE;
  M_HMI_StatusText := 'Capsule sensor fault resolved!';
END_IF

```

Figure IV.161 Récupération automatique si le bouchon est détecté

6. Forçage manuel de l’alarme lumineuse et de l’avertisseur sonore (Figure IV.94)
7. Les transitions et sautes



Figure IV.162 Les transitions et sautes d’étape CapFeederSensorFault

IV.3.5.6. Les erreurs d’étape JawReady

IV.3.5.6.1. L’erreur ChuckDownFault

1. Transition



Figure IV.163 Transition entre JawReady et ChuckDownFault

2. Message IHM et activation d’alarme et sonore

```

M_HMI_StatusText := 'ERROR: Capping chuck did not reach down position. Check mechanism and sensor!';
AlarmLight := TRUE;
Buzzer := TRUE;

```

Figure IV.164 Affichage de l’erreur sur IHM et activation des alarmes

3. Contrôle manuel les vérins d’air pour les pinces, le mandrin et la mâchoire, ou les place automatiquement en position de sécurité lors d’un défaut.

```

IF ManualOverride_ConveyorMotor THEN
    ConveyorMotor_State := M_HMI_ConveyorMotor_Control;
    IF ConveyorMotor_State THEN
        Conveyor_MotorVFD_Speed := M_HMI_ConveyorMotor_Speed;
    ELSE
        Conveyor_MotorVFD_Speed := 0.0;
    END_IF
ELSE
    ConveyorMotor_State := FALSE;
    Conveyor_MotorVFD_Speed := 0.0;
END_IF

IF ManualOverride_SV_Air_Chuck THEN
    SV_Air_Chuck := M_HMI_SV_Air_Chuck_Control;
ELSE
    SV_Air_Chuck := FALSE;
END_IF

IF ManualOverride_SV_Air_CappingClamps THEN
    SV_Air_CappingClamps := M_HMI_SV_Air_CappingClamps_Control;
ELSE
    SV_Air_CappingClamps := FALSE;
END_IF

```

Figure IV.165 Forçage manuel et mise en sécurité des vérins de capsulage

4. Réinitialiser le défaut, le journal et les alarmes, et de remettre les actionneurs en position normale via l'IHM et enregistrer l'erreur une seule fois lors de sa détection.

```

IF NOT BottleCappingPositionFaultLogged THEN
    BottleCappingPositionFaultLogged := TRUE;
END_IF

IF OperatorResetRequested THEN
    Buzzer := FALSE;
    AlarmLight := FALSE;
    SV_Air_Chuck := FALSE;
    SV_Air_CappingClamps := FALSE;
    BottleCappingPositionFaultLogged := FALSE;
    BottleCappingPositionFault := FALSE;
END_IF

```

Figure IV.166 Enregistrement du défaut et Réinitialisation du défaut par l'opérateur

5. Utilisation d'une temporisation (Filtrage anti-rebond 300 ms) pour vérifier que le capteur position basse du mandrin est bien déclenché avant de réinitialiser le défaut et les alarmes.

```

IF CappingConveyorSensor THEN
  BottleCappingSensorDebounceTimer(IN := TRUE, PT := T#500ms);
  IF BottleCappingSensorDebounceTimer.Q THEN
    CappingConveyorSensorDebounced := TRUE;
  ELSE
    CappingConveyorSensorDebounced := FALSE;
  END_IF
ELSE
  BottleCappingSensorDebounceTimer(IN := FALSE);
  CappingConveyorSensorDebounced := FALSE;
END_IF

IF CappingConveyorSensorDebounced THEN
  Conveyor_MotorVFD_Speed := 0.0;
  SV_Air_Chuck := TRUE;
  SV_Air_CappingClamps := TRUE;
  CappingReady := TRUE;
  BottleCappingPositionFault := FALSE;
  AlarmLight := FALSE;
  Buzzer := FALSE;
  BottleCappingPositionFaultLogged := FALSE;
  M_HMI_StatusText := 'Capping position fault resolved!';
END_IF

```

Figure IV.167 Récupération automatique si le capteur position basse est déclenché

6. Forçage manuel de l'alarme lumineuse et de l'avertisseur sonore (Figure IV.94)
7. Les transitions et sautes

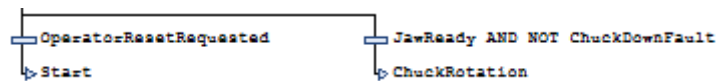


Figure IV.168 Les transitions et sautes d'étape ChuckDownFault

IV.3.5.7. Les erreurs d'étape ChuckRotation

IV.3.5.7.1. L'erreur JawClutchFault

1. Transition

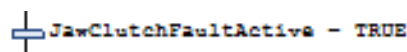


Figure IV.169 Transition entre ChuckRotation et JawClutchFault

2. Message IHM et activation d'alarme et sonore

```

M_HMI_StatusText := 'ERROR: Jaw failed to clutch cap. Check jaw mechanism and cap position!';
AlarmLight := TRUE;
Buzzer := TRUE;

```

Figure IV.170 Affichage de l'erreur sur IHM et activation des alarmes

3. Contrôle manuel les vérins d'air, le mandrin et la mâchoire ou de les placer automatiquement en position de sécurité lors d'un défaut.

```

IF ManualOverride_SV_Air_CappingClamps THEN
    SV_Air_CappingClamps := M_HMI_SV_Air_CappingClamps_Control;
ELSE
    SV_Air_CappingClamps := FALSE;
END_IF

IF ManualOverride_SV_Air_Chuck THEN
    SV_Air_Chuck := M_HMI_SV_Air_Chuck_Control;
ELSE
    SV_Air_Chuck := FALSE;
END_IF

IF ManualOverride_SV_Air_Jaw THEN
    SV_Air_Jaw := M_HMI_SV_Air_Jaw_Control;
ELSE
    SV_Air_Jaw := FALSE;
END_IF

IF ManualOverride_StepperChuck THEN
    Stepper_Chuck_Speed := M_HMI_StepperChuck_Speed;
    Stepper_Chuck_Rotation := M_HMI_StepperChuck_Rotation;
    Stepper_Chuck_State := M_HMI_StepperChuck_On;
ELSE
    Stepper_Chuck_Speed := 0.0;
    Stepper_Chuck_Rotation := 0.0;
    Stepper_Chuck_State := FALSE;
END_IF

```

Figure IV.171 Forçage manuel et mise en sécurité des actionneurs

4. Réinitialiser le défaut, le journal et les alarmes, et de remettre les actionneurs en position normale via l'IHM et enregistre l'erreur une seule fois lors de sa détection.

```

IF NOT JawClutchFaultLogged THEN
    JawClutchFaultLogged := TRUE;
END_IF

IF OperatorResetRequested THEN
    AlarmLight := FALSE;
    Buzzer := FALSE;
    JawClutchFaultLogged := FALSE;
    JawClutchFaultActive := FALSE;
END_IF

```

Figure IV.172 Enregistrement du default et Réinitialisation du défaut par l'opérateur

5. Utilisation d'une temporisation (Filtrage anti-rebond 500ms) pour vérifier que le capteur de serrage mâchoire est bien déclenché avant de réinitialiser le défaut et les alarmes.

```

IF JawClutchSensor THEN
    JawClutchDebounceTimer(IN := TRUE, PT := T#500MS);
    IF JawClutchDebounceTimer.Q THEN
        JawClutchSensorDebounced := TRUE;
    ELSE
        JawClutchSensorDebounced := FALSE;
    END_IF
ELSE
    JawClutchDebounceTimer(IN := FALSE);
    JawClutchSensorDebounced := FALSE;
END_IF

IF JawClutchSensorDebounced THEN
    SV_Air_Jaw := TRUE;
    M_HMI_StatusText := 'Jaw clutch fault resolved!';
    JawClutchFaultActive := FALSE;
    AlarmLight := FALSE;
    Buzzer := FALSE;
    JawClutchFaultLogged := FALSE;
END_IF

```

Figure IV.173 Récupération automatique si le capteur mâchoire est déclenché

6. Forçage manuel de l'alarme lumineuse et de l'avertisseur sonore (Figure IV.94)

7. Les transitions et sautes

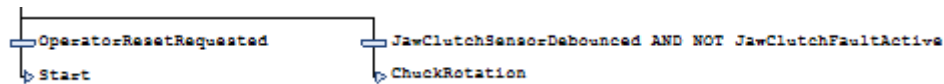


Figure IV.174 Les transitions et sautes d'étape JawClutchFault

IV.3.5.7.2. L'erreur TorqueFault_Active

1. Transition



Figure IV.175 Transition entre ChuckRotation et TorqueFault

2. Message IHM et activation d'alarme et sonore

```

M_HMI_StatusText := 'ERROR: Torque out of range or timeout during capping. Check chuck!';
AlarmLight := TRUE;
Buzzer := TRUE;
M_HMI_MeasuredTorque := MeasuredTorque;
  
```

Figure IV.176 Affichage de l'erreur sur IHM et activation des alarmes

3. Contrôle manuel les vérins d'air, le mandrin et la mâchoire (Figure IV.154)

4. Réinitialiser le défaut, le journal et les alarmes, et de remettre les actionneurs en position normale via l'IHM et enregistre l'erreur une seule fois lors de sa détection.

```

IF NOT TorqueFaultLogged THEN
    TorqueFaultLogged := TRUE;
END_IF

IF OperatorResetRequested THEN
    AlarmLight := FALSE;
    Buzzer := FALSE;
    TorqueFaultLogged := FALSE;
    TorqueFaultActive := FALSE;
END_IF
  
```

Figure IV.177 Enregistrement du défaut et Réinitialisation du défaut par l'opérateur

5. Restaure l'état normal et réinitialise le défaut et les alarmes si le couple mesuré redevient conforme.

```

IF MeasuredTorque >= TorqueMin AND MeasuredTorque <= TorqueMax THEN
    M_HMI_StatusText := 'Torque fault resolved!';
    TorqueFaultActive := FALSE;
    AlarmLight := FALSE;
    Buzzer := FALSE;
    TorqueFaultLogged := FALSE;
END_IF
  
```

Figure IV.178 Récupération automatique si le couple redevient correct

6. Forçage manuel de l'alarme lumineuse et de l'avertisseur sonore (Figure IV.94)

7. Les transitions et sautes

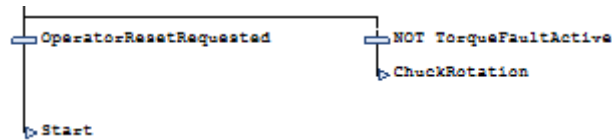


Figure IV.179 Les transitions et sautes d'étape TorqueFault

IV.3.5.8. Les erreurs d'étape Labeling

IV.3.5.8.1. L'erreur LabelTipNotReady

1. Transition



Figure IV.180 Transition entre Labeling et LabelTipNotReadyFault

2. Message IHM et activation d'alarme et sonore

```
M_HMI_StatusText := 'ERROR: Label tip not ready!';
AlarmLight := TRUE;
```

Figure IV.181 Affichage de l'erreur sur IHM et activation des alarmes

3. Control manuel le moteur du convoyeur principal, les convoyeurs latéraux, le moteur d'extrémité d'étiquette et le couteau, ou de les placer automatiquement en position arrêtée lors d'une erreur (Figure IV.67)
4. Réinitialiser le défaut, le journal et les alarmes, et de remettre les actionneurs en position normale via l'IHM et enregistre l'erreur une seule fois lors de sa détection.

```
IF NOT LabelTipNotReadyLogged THEN
  LabelTipNotReadyLogged := TRUE;
END_IF

IF OperatorResetRequested THEN
  AlarmLight := FALSE;
  Buzzer := FALSE;
  LabelTipNotReadyLogged := FALSE;
  LabelTipNotReady := FALSE;
END_IF
```

Figure IV.182 Enregistrement du default et Réinitialisation du défaut par l'opérateur

5. Utilisation d'une temporisation (Filtrage anti-rebond 300 ms) pour vérifier que l'extrémité d'étiquette est bien détectée avant de réinitialiser le défaut et les alarmes.

```
IF LabelTipSensor THEN
  LabelTipDebounceTimer(IN := TRUE, PT := T#300MS);
  IF LabelTipDebounceTimer.Q THEN
    LabelTipSensorDebounced := TRUE;
  ELSE
    LabelTipSensorDebounced := FALSE;
  END_IF
ELSE
  LabelTipDebounceTimer(IN := FALSE);
  LabelTipSensorDebounced := FALSE;
END_IF

IF LabelTipSensorDebounced THEN
  M_HMI_StatusText := 'Label tip fault resolved!';
  LabelTipNotReady := FALSE;
  AlarmLight := FALSE;
  Buzzer := FALSE;
  LabelTipNotReadyLogged := FALSE;
END_IF
```

Figure IV.183 Récupération automatique si le couple redevient correct

6. Forçage manuel de l'alarme lumineuse et de l'avertisseur sonore (Figure IV.94)
7. Les transitions et sautes

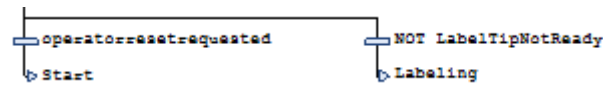


Figure IV.184 Les transitions et sautes d'étape LabelTipNotReady

IV.3.5.8.2. L'erreur LabelGapNotDetected

1. Transition



Figure IV.185 Transition entre Labeling et LabelGapNotDetected

2. Message IHM et activation d'alarme et sonore

```

M_HMI_StatusText := 'ERROR: Label gap not detected!';
AlarmLight := TRUE;
Buzzer := TRUE;
  
```

Figure IV.186 Affichage de l'erreur sur IHM et activation des alarmes

3. Control manuel le moteur du convoyeur principal, les convoyeurs latéraux, le moteur d'extrémité d'étiquette et le couteau, ou de les placer automatiquement en position arrêtée lors d'une erreur (Figure IV.67)
4. Réinitialiser le défaut, le journal et les alarmes, et de remettre les actionneurs en position normale via l'IHM et enregistre l'erreur une seule fois lors de sa détection.

```

IF NOT LabelGapNotDetectedLogged THEN
  LabelGapNotDetectedLogged := TRUE;
END_IF

IF OperatorResetRequested THEN
  AlarmLight := FALSE;
  Buzzer := FALSE;
  LabelGapNotDetectedLogged := FALSE;
  LabelGapNotDetected := FALSE;
END_IF
  
```

Figure IV.187 Enregistrement du défaut et Réinitialisation du défaut par l'opérateur

5. Utilisation d'une temporisation (Filtrage anti-rebond 300 ms) pour vérifier que le capteur gap d'étiquette est bien déclenché avant de réinitialiser le défaut et les alarmes.

```

IF LabelGapSensor THEN
  LabelGapDebounceTimer(IN := TRUE, PT := T#300MS);
  IF LabelGapDebounceTimer.Q THEN
    LabelGapSensorDebounced := TRUE;
  ELSE
    LabelGapSensorDebounced := FALSE;
  END_IF
ELSE
  LabelGapDebounceTimer(IN := FALSE);
  LabelGapSensorDebounced := FALSE;
END_IF

IF LabelGapSensorDebounced THEN
  M_HMI_StatusText := 'Label gap fault resolved!';
  LabelGapNotDetected := FALSE;
  AlarmLight := FALSE;
  Buzzer := FALSE;
  LabelGapNotDetectedLogged := FALSE;
END_IF

```

Figure IV.188 Récupération automatique si le couple redevient correct

6. Forçage manuel de l'alarme lumineuse et de l'avertisseur sonore (Figure IV.94)
7. Les transitions et sautes



Figure IV.189 Les transitions et sautes d'étape LabelGapNotDetected

IV.3.5.8.3. L'erreur BottleArrivalFault

1. Transition

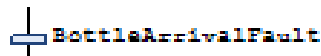


Figure IV.190 Transition entre Labeling et BottleArrivalFault

2. Message IHM et activation d'alarme et sonore

```

M_HMI_StatusText := 'ERROR: Bottle not detected at labeling position!';
AlarmLight := TRUE;
Buzzer := TRUE;

```

Figure IV.191 Affichage de l'erreur sur IHM et activation des alarmes

3. Control manuel le moteur du convoyeur principal, les convoyeurs latéraux, le moteur d'extrémité d'étiquette et le couteau, ou de les placer automatiquement en position arrêtée lors d'une erreur (Figure IV.67)
4. Réinitialiser le défaut, le journal et les alarmes, et de remettre les actionneurs en position normale via l'IHM et enregistre l'erreur une seule fois lors de sa détection.

```

IF NOT BottleArrivalFaultLogged THEN
  BottleArrivalFaultLogged := TRUE;
END_IF

IF OperatorResetRequested THEN
  AlarmLight := FALSE;
  Buzzer := FALSE;
  BottleArrivalFaultLogged := FALSE;
  BottleArrivalFault := FALSE;
END_IF

```

Figure IV.194 Enregistrement du default et Réinitialisation du défaut par l'opérateur

5. Utilisation d'une temporisation (Filtrage anti-rebond 300 ms) pour vérifier que la bouteille est bien détectée avant de réinitialiser le défaut et les alarmes.

```

IF BottleLabellingPositionSensor THEN
  BottleLabellingPositionDebounceTimer(IN := TRUE, PT := T#300MS);
  IF BottleLabellingPositionDebounceTimer.Q THEN
    BottleLabellingPositionSensorDebounced := TRUE;
  ELSE
    BottleLabellingPositionSensorDebounced := FALSE;
  END_IF
ELSE
  BottleLabellingPositionDebounceTimer(IN := FALSE);
  BottleLabellingPositionSensorDebounced := FALSE;
END_IF

IF BottleLabellingPositionSensorDebounced THEN
  M_HMI_StatusText := 'Bottle arrival fault resolved!';
  BottleArrivalFault := FALSE;
  AlarmLight := FALSE;
  Buzzer := FALSE;
  BottleArrivalFaultLogged := FALSE;
END_IF

```

Figure IV.193 Récupération automatique si le couple redevient correct

6. Forçage manuel de l'alarme lumineuse et de l'avertisseur sonore (Figure IV.94)
7. Les transitions et sautes



Figure IV.194 Les transitions et sautes d'étape BottleArrivalFault

IV.3.5.8.4. L'erreur ConveyorMotorFaultLabeling

Cette étape d'erreur est une précaution pour la surcharge du moteur du convoyeur pendant que la bouteille est à la station d'étiquetage, ce qui signifie qu'il aura le même code de (IV.3.5.5 et IV.3.5.7)

- Les transitions et sautes

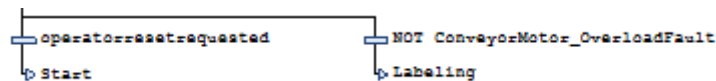


Figure IV.195 Les transitions et sautes d'étape ConveyorMotorFaultLabeling

IV.3.5.8.5. L'erreur SideLeftConveyorFault/ SideRightConveyorFault

1. Transition

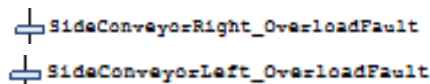


Figure IV.196 Transition entre Labeling et SideLeftConveyorFault/ SideRightConveyorFault

2. Message IHM et activation d'alarme et sonore

```
M_HMI_StatusText := 'ERROR: Side conveyor left motor overload! Inspect conveyor, drive, or wiring.';
AlarmLight := TRUE;
Buzzer := TRUE;

M_HMI_StatusText := 'ERROR: Side conveyor right motor overload! Inspect conveyor, drive, or wiring.';
AlarmLight := TRUE;
Buzzer := TRUE;
```

Figure IV.197 Affichage de l'erreur sur IHM et activation des alarmes

3. Control manuel le moteur du convoyeur latéral droit/gauche ou de le désactiver automatiquement en cas de surcharge.

```
IF ManualOverride_SideConveyorRight THEN
    SideConveyor_Right_State := M_HMI_SideConveyorRight_Control;
    IF SideConveyor_Right_State THEN
        SideConveyor_Right_SpeedSetpoint := M_HMI_SideConveyorRight_Speed;
    ELSE
        SideConveyor_Right_SpeedSetpoint := 0.0;
    END_IF
ELSE
    SideConveyor_Right_State := FALSE;
    SideConveyor_Right_SpeedSetpoint := 0.0;
END_IF

IF ManualOverride_SideConveyorLeft THEN
    SideConveyor_Left_State := M_HMI_SideConveyorLeft_Control;
    IF SideConveyor_Left_State THEN
        SideConveyor_Left_SpeedSetpoint := M_HMI_SideConveyorLeft_Speed;
    ELSE
        SideConveyor_Left_SpeedSetpoint := 0.0;
    END_IF
ELSE
    SideConveyor_Left_State := FALSE;
    SideConveyor_Left_SpeedSetpoint := 0.0;
END_IF
```

Figure IV.198 Forçage manuel du moteur convoyeur latéral droit/gauche

4. Réinitialiser le défaut, le journal et les alarmes, et de remettre les actionneurs en position normale via l'IHM et enregistre l'erreur une seule fois lors de sa détection.

```
IF NOT SideConveyorRight_OverloadFaultLogged THEN
    SideConveyorRight_OverloadFaultLogged := TRUE;
END_IF

IF OperatorResetRequested THEN
    SideConveyorRight_OverloadFault := FALSE;
    SideConveyorRight_OverloadFaultLogged := FALSE;
    AlarmLight := FALSE;
    Buzzer := FALSE;
END_IF

IF NOT SideConveyorLeft_OverloadFaultLogged THEN
    SideConveyorLeft_OverloadFaultLogged := TRUE;
END_IF

IF OperatorResetRequested THEN
    SideConveyorLeft_OverloadFault := FALSE;
    SideConveyorLeft_OverloadFaultLogged := FALSE;
    AlarmLight := FALSE;
    Buzzer := FALSE;
END_IF
```

Figure IV.199 Enregistrement du défaut et Réinitialisation du défaut par l'opérateur

5. Restaure l'état normal et réinitialise le défaut et les alarmes si le courant absorbé par le moteur convoyeur latéral droit/gauche repasse sous le seuil critique.

```

IF SideConveyor_Left_Absorp_mA < 900.0 THEN
    AlarmLight := FALSE;
    Buzzer := FALSE;
    SideConveyorLeft_OverloadFaultLogged := FALSE;
    SideConveyorLeft_OverloadFault := FALSE;
    M_HMI_StatusText := 'Side conveyor left overload resolved!';
END_IF

IF SideConveyor_Right_Absorp_mA < 900.0 THEN
    AlarmLight := FALSE;
    Buzzer := FALSE;
    SideConveyorRight_OverloadFaultLogged := FALSE;
    SideConveyorRight_OverloadFault := FALSE;
    M_HMI_StatusText := 'Side conveyor right overload resolved!';
END_IF

```

Figure IV.200 Récupération automatique si le courant moteur redevient normal

6. Forçage manuel de l'alarme lumineuse et de l'avertisseur sonore (Figure IV.94)
7. Les transitions et sautes



Figure IV.201 Les transitions et sautes d'étape
SideLeftConveyorFault/SideRightConveyorFault

IV.3.5.8.6. L'erreur LabelTipMotorFault

1. Transition

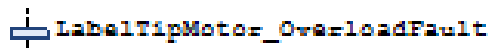


Figure IV.202 Transition entre Labeling et LabelTipMotorFault

2. Message IHM et activation d'alarme et sonore

```

M_HMI_StatusText := 'ERROR: Label tip motor overload! Inspect label tip mechanism, drive, or wiring.';
AlarmLight := TRUE;
Buzzer := TRUE;

```

Figure IV.203 Affichage de l'erreur sur IHM et activation des alarmes

3. Control manuellement le moteur d'extrémité d'étiquette ou de le désactiver automatiquement en cas de surcharge.

```

IF ManualOverride_LabelTipMotor THEN
    Label_Tip_Motor_State := M_HMI_LabelTipMotor_Control;
    IF Label_Tip_Motor_State THEN
        Label_Tip_Motor_SpeedSetpoint := M_HMI_LabelTipMotor_Speed;
    ELSE
        Label_Tip_Motor_SpeedSetpoint := 0.0;
    END_IF
ELSE
    Label_Tip_Motor_State := FALSE;
    Label_Tip_Motor_SpeedSetpoint := 0.0;
END_IF

```

Figure IV.204 Forçage manuel du moteur extrémité d'étiquette

4. Réinitialiser le défaut, le journal et les alarmes, et de remettre les actionneurs en position normale via l'IHM et enregistre l'erreur une seule fois lors de sa détection.

```

IF NOT LabelTipMotor_OverloadFaultLogged THEN
    LabelTipMotor_OverloadFaultLogged := TRUE;
END_IF

IF OperatorResetRequested THEN
    LabelTipMotor_OverloadFault := FALSE;
    LabelTipMotor_OverloadFaultLogged := FALSE;
    AlarmLight := FALSE;
    Buzzer := FALSE;
END_IF

```

Figure IV.205 Enregistrement du défaut et Réinitialisation du défaut par l'opérateur

5. Restaure l'état normal et réinitialise le défaut et les alarmes si le courant absorbé par le moteur extrémité d'étiquette repasse sous le seuil critique.

```

IF Label_Tip_Motor_Absorp_mA < 600.0 THEN
    AlarmLight := FALSE;
    Buzzer := FALSE;
    LabelTipMotor_OverloadFaultLogged := FALSE;
    LabelTipMotor_OverloadFault := FALSE;
    M_HMI_StatusText := 'Label tip motor overload resolved!';
END_IF

```

Figure IV.206 Récupération automatique si le courant moteur redevient normal

6. Forçage manuel de l'alarme lumineuse et de l'avertisseur sonore (Figure IV.94)
7. Les transitions et sautes

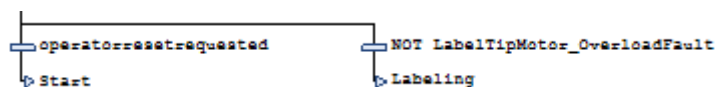


Figure IV.207 Les transitions et sautes d'étape LabelTipMotorFault

IV.4. Conclusion

Dans ce chapitre, nous avons programmé puis simulé une ligne de remplissage automatique de liquide sur CODESYS à l'aide des langages de programmation GRAFCET et ST, nous avons couvert les codes de bon fonctionnement et les erreurs qui peuvent arriver sur chaque étape dans le système.

Conclusion générale

Les machines de remplissage de liquides occupent une place essentielle dans l'industrie manufacturière, où elles contribuent directement à l'efficacité et à la qualité des processus de production. Dans le cadre de ce mémoire, l'objectif n'a pas été la réalisation physique d'une machine, mais la modélisation et la simulation complète d'un système de remplissage automatisé.

Le travail a d'abord porté sur l'étude des éléments constitutifs de la partie opérative, avec une attention particulière aux capteurs, actionneurs, moteurs électriques et organes de convoyage, afin de définir un environnement réaliste de simulation. Ensuite, la partie commande a été développée à travers une programmation structurée dans un automate industriel virtuel, intégrant à la fois les séquences normales et Les scénarios d'erreurs.

Une contribution majeure de ce travail a consisté à détailler les codes d'action associés à chaque étape (par exemple le remplissage, le bouchonnage, l'étiquetage et le contrôle qualité), tout en intégrant des mécanismes de sécurité et des logiques de reprise. Ces étapes ont été modélisées avec un haut niveau de précision, notamment grâce à l'introduction des valeurs caractéristiques des moteurs et des seuils de détection, afin de rapprocher la simulation des conditions industrielles réelles. Ainsi, ce mémoire illustre comment une simulation peut servir de base de validation fonctionnelle avant toute mise en œuvre réelle, permettant d'anticiper les problèmes potentiels et de définir des solutions de correction.

En perspectives, ce travail pourrait être étendu par l'intégration de modèles plus avancés de capteurs intelligents, l'étude énergétique des actionneurs, ou encore la connexion de la simulation à un jumeau numérique pour une exploitation industrielle plus poussée.

Bibliographie

- [1] Bouacem Mourad, Bouhamadouche Hamza, Etude et Automatisation du Système d'Emplissage de Bouteilles de Gaz butane Par l'API S7-300 : université Mouloud Mammeri de Tizi-Ouzou, Faculté de Génie électrique et d'informatique, département d'électronique
- [2] Namaoui Nazha, Automatisation d'un système de remplissage de quatre trémies de sucre, avec supervision HMI, CEVITAL. Université Abderrahmane MIRA 2014
- [3] BELAIDI Ouerdia BELHACENE Lydia, Conception d'une Automatisation et d'une Supervision d'un Four de Cuisson d'Emaillage (E11) au sein de l'ENIEM : Université Mouloud Mammeri De Tizi-Ouzou, Faculté De Génie Electrique Et D'informatique, DEPARTEMENT D'AUTOMATIQUE 2017/2018
- [4] FLITTI Fatima HADBI Rahma, Réalisation d'un système de remplissage automatique de bouteilles en utilisant une carte arduino : université aboubakr belgaid – Tlemcen, département de génie électrique et électronique 2020/2021
- [5] <https://remplissage-pro.fr/conditionnement-liquide/fonctionnement-remplisseuse-manuelle-liquide/>
- [6] <https://ustotc.netlify.app/files/tcs5/CMT/CMT-1-Definitions-Et-Caracteristiques-Des-Capteurs.pdf>
- [7] <https://fr.wikipedia.org/wiki/Capteur>
- [8] <http://dptgeii.iutsd.univ-lorraine.fr/cours/lpsarii/IM/Cours/Capteurs.pdf>
- [9] <https://www.spatialpost.com/active-vs-passive-sensors-remote-sensing/>
- [10] <https://www.cognex.com/fr-fr/blogs/machine-vision/smart-cameras-in-machine-vision>
- [11] https://www.baumer.com/fr/fr/service-assistance/fonctionnement/le-fonctionnement-des-capteurs-de-force/a/Know-how_function_force_sensor
- [12] <https://sensel-measurement.fr/fr/18-capteur-de-couple>
- [13] <https://www.maxicours.com/se/cours/acquerir-l-information---les-differents-capteurs/>
- [14] <https://atlas-scientific.com/blog/analog-vs-digital-sensors-understanding-the-key-differences/>
- [15] http://lgt.garnier.free.fr/espace_ee_fichiers/capteurs/capteurs.pdf
- [16] <https://www.yoctopuce.com/FR/article/comment-brancher-des-capteurs-npn-et-pnp>
- [17] http://gcedidactic.free.fr/automatisme/ch_acquisition/TOR/captor.htm#capacitif
- [18] <https://www.mytopschool.net/mysti2d/activites/polynesie2/ETT/C044/21/Pneumatique/files/Documents/pdf/distributeurs.pdf>
- [19] <https://www.3dfindit.com/fr/engiclopedia/differents-types-de-distributeurs-et-leur-application>
- [20] <http://infarab.free.fr/technologie2/lesrelais.pdf>
- [21] <https://www.electricity-magnetism.org/fr/relais-electromagnetiques/>
- [22] <https://www.electricalclassroom.com/contactor-working-application-types/>
- [23] <https://www.mytopschool.net/mysti2d/activites/polynesie2/ETT/C044/21/Pneumatique/files/Documents/pdf/verins.pdf>

- [24] <https://engineerfix.com/mechanical/pneumatic-cylinders/a-complete-guide-to-pneumatic-cylinders/>
- [25] <https://www.multi-moteur.com/blog/qu-est-ce-qu-un-moteur-electrique-monophasé-n9>
- [26] <https://energieplus-lesite.be/techniques/ascenseurs7/moteur-asynchrone/>
- [27] <https://www.renesas.com/en/support/engineer-school/brushless-dc-motor-01-overview?srsId=AfmBOopYcaUqkr0dbhertux6gUyJ-frsFfS1whvhVPr0tVfGITdFzfwE>
- [28] https://www.orientalmotor.com/products/pdfs/SteppingMotors/SectionBIntro_B2-B42.pdf
- [29] <https://www.thomasnet.com/articles/machinery-tools-supplies/what-is-a-3-phase-motor-and-how-does-it-work/>
- [30] <https://www.gamak.com/en/what-is-a-reducer-and-what-are-its-types>
- [31] <https://hal.science/hal-00979032/document>.
- [32] <https://universalfilling.com/how-important-are-nozzles-on-a-liquid-filling-machine/>
- [33] https://www.researchgate.net/publication/332947247_Design_of_Automated_Mechanical_System_for_Filling_and_Capping_of_water_Bottle
- [34] <https://assetpackaging.com.au/articles/how-do-cap-tightening-machines-work/>
- [35] https://ceking.com/products/cs3030027a-3-jaw-chuck-assembly?srsId=AfmBOooPK-Hsc2SPYe81B0Eo_cKVjCLz1z6Bfjj-S-TGPtSbtbEcyvRh
- [36] <https://www.se.com/id/en/product-category/2100-hmi-terminals-and-industrial-pc/>
- [37] <https://ojautomation.com/content/the-importance-of-alarm-management-in-industrial-automation>
- [38] Meharzi Anas, Conception et Réalisation d'une Ligne Automatique de Remplissage pour Liquides : Université d'Oran II Mohamed Ben Ahmed, Institut de Maintenance et de sécurité industrielle, Département de Maintenance en Electromécanique (2023/2024)
- [39] https://fac.umc.edu.dz/ista/pdf/cours/Chapter_1_papier_02.pdf
- [40] <https://theautomization.com/plc-working-principle-and-plc-scan-cycle/>
- [41] https://infosys.beckhoff.com/index_en.htm
- [42] <https://control.com/technical-articles/an-overview-of-sequential-function-chart-sfc-programming/>
- [43] <https://en.wikipedia.org/wiki/CODESYS>