

RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE
MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR ET DE LA
RECHERCHE SCIENTIFIQUE
UNIVERSITÉ IBN KHALDOUN DE TIARET



FACULTÉ DES MATHÉMATIQUE ET D'INFORMATIQUE
DÉPARTEMENT D'INFORMATIQUE

Mémoire de fin d'études en vue de l'obtention du diplôme de Master(LMD)

Spécialité : Génie Informatique

Réalisé par :

ADIM Hakim

Encadré par :

Dr. BELARBI Mostefa

Sujet du mémoire

**CONCEPTION D'UN NOYAU DE CALCUL POUR CONTROLE-
COMMANDE TEMPS REEL DE LA MACHINE-TOUR**

Soutenu publiquement le

devant le jury composé de :

Dr. MERATI Medjeded

Président

Dr. BENGHANI Abdelmalek

Examineur

Dr. BELARBI Mostefa

Encadreur

PROMOTION : 2019/2020

Remerciements

Ce travail a été réalisé au Laboratoire de mathématique et informatique (LIM) de l'université Ibn Khaldoun de Tiare, sous la supervision du Dr. BELARBI Mostefa, enseignant à l'université Ibn Khaldoun de Tiaret.

Je remercie chaleureusement mon encadreur Dr. BELARBI Mostefa qui a été attentif à l'évolution de mes recherches et a apporté toute sa contribution pour mener à bien ce travail. Ses qualités humaines et scientifiques, Ses conseils ainsi que son enthousiasme, été comme de leçons d'un père à son fils, m'ont été très bénéfiques durant cette année de master.

Je tiens également à remercier Dr. BENGHENI Abdelmalek et Dr. MERATI Medjeded qui ont accepté d'examiner mon travail, et exprime mon honneur ainsi que ma gratitude de les avoir comme jury dans ma soutenance.

Je tiens toute ma reconnaissance aux, Dr. BENAÏSSA Trariet, Dr. HARICHE Abdelhamid, pour leur aide dans la réalisation de ce travail.

Je voudrais remercier tous les membres du Laboratoire de mathématique et informatique (LIM) de l'université Ibn Khaldoun de Tiaret avec qui j'ai passé d'agréables moments.

A toute ma famille, mes collègues et à tous mes enseignants du département d'informatique de l'Université Ibn Khaldoun de Tiaret, je dirai merci pour les encouragements et le soutien dont ils n'ont cessé de me donner tout au long de mon cursus universitaire.

Résumé

Ce mémoire présente une méthodologie pour le développement de systèmes temps réel complexes. Cette méthodologie utilise des mécanismes incrémentiels afin d'établir les paramètres de mesure du processus de développement inter domaine. C'est un objectif important de développer un cadre inter-domaines qui répond aux exigences et aux contraintes de plusieurs types de modèles industriels. En promouvant un style de conception strict basé sur les incréments et en identifiant les incréments qui peuvent être déployés dans différents domaines d'application, les coûts de conception et de production de nouvelles applications peuvent être considérablement réduits en réutilisant des composants. Nous fournissons dans Ce mémoire un aperçu des paramètres de mesure du processus de développement inter domaine basé sur SART (Analyse Structurée en Temps Réel) à l'aide d'études de cas industriels pour le développement de machines-outils.

Introduction général

Avec l'évolution de la technologie dans le secteur industriel, militaire, aéronautique, télécommunications, etc., la demande est plus en plus accrue de systèmes temps réel complexes et fiables, ces derniers s'efforcent, à travers des méthodes de développement avancées, des solutions informatiques plus élaborées et, de contraintes temporelles rigoureuses, pour subvenir aux besoins.

Poussés par la curiosité de connaître le fonctionnement de tels systèmes, nous essayerons de réaliser un noyau de calcul pour l'acquisition et traitement de données générés par des machines de contrôle-commande, la conception du noyau exploite les fonctionnalités d'un exécutif temps réel pour contrôler la synchronisation des différents traitements issues des différents sous-systèmes de la machine, en illustrant et en mettant en relief tout au long de ce mémoire qui accompagnera ce travail, toutes les étapes nécessaires.

La philosophie qui sera adoptée, se basera, elle aussi sur la simplicité des mécanismes qui seront mis en œuvre, ceci afin de permettre, une meilleure compréhension de principes et politiques utilisés d'une part, et d'autre part, offrir un noyau, simple à modéliser, en vue de lui intégrer un certain aspect pédagogique, justifiant ainsi, le choix de la création au lieu d'une modification ou amélioration d'un système déjà existant.

Ce mémoire présente, des rappels, des définitions et recommandations sur les différents types de matériels et méthodologie utiliser pour réaliser ce travail, le premier chapitre contient, une présentation des appareils de mesure, les différents modèles et leur fonctionnement et quelque notion de base sur les composants électroniques, le deuxième chapitre complète les concepts présentés au premier chapitre, en rappelant les systèmes embarqués et les systèmes temps réel et aussi la relation entre eux, le troisième chapitre présente, le cycle de vie et la méthode de développement d'un système temps réel embarqué, le quatrième chapitre, présente toute la procédure de réalisation de notre travail, nous donne une vue détaillée de notre système, et présenter les différents étapes de développement du noyau conçu.

Mots clé : appareil de mesure, capteur, microcontrôleur, système embarqué en temps réel, exécutif temps réel, méthode SART, langage LACATRE.

Table des matières

Remerciements	1
Résumé	2
Introduction général	3
Table des matières	4
Liste des figures	7
List des tableaux	9
Liste des abréviations	10
Chapitre 01 : Les systèmes de mesure embarqués	11
1.1 Introduction	11
1.2 Les systèmes de mesure	12
1.2.1 Définition	12
1.2.2 Quelque exemple d'appareils de mesure	12
1.3 Les microcontrôleurs	13
1.3.1 Définition	13
1.3.2 Les capteurs :	13
1.3.3 Interfaçage des capteurs avec les microcontrôleurs	14
1.4 Conclusion	14
Chapitre 02 : Les systèmes temps réel embarqués	15
2.1 Introduction	15
2.2 Les systèmes embarqués	15
2.2.1 Définition	15
2.2.2 Architecture	15
2.2.3 Domaines d'applications	16
2.2.4 Les concepts principaux d'un système embarqué :	17
2.2.5 Caractéristiques de système embarqué	17
2.3 Les systèmes temps réel	18
2.3.1 Définition 01	18
2.3.2 Définition 02	18
2.4 Les systèmes temps réel embarqués	19
2.4.1 Contraintes des systèmes embarqués en temps réel :	19
2.5 La notion du multitâche	20
2.5.1 Notion de processus	20
Remarque	21
2.6 Conclusion	21
Chapitre 03 : Méthode de développement d'un système temps réel embarqué	22
3.1 Introduction	22
3.2 Développement des systèmes informatiques	23

3.2.1 Cycle de vie d'un système	23
3.2.2 Les modèles classiques de cycle de vie :	23
3.3 Les méthodes de spécification pour un système temps réel	25
3.3.1 La méthode SA-RT	25
3.3.2 Objectif de méthode SA-RT	26
3.4 Les langages de conception pour un système temps réel	27
3.4.1 Le langage LACATRE	27
3.5 Les outils d'implémentation pour un système temps réel	29
3.5.1 Exécutif temps réel	29
3.5.2 Les Services offerts par un exécutif temps réel	29
3.6 Conclusion	30
Chapitre 04 : Les plateformes matérielles et logicielles utilisées	31
4.1 Introduction	31
4.2 L'exécutif temps réel FreeRTOS.....	31
4.2.1 Pourquoi FreeRTOS ?.....	33
4.2.2 Fonctionnalités de FreeRTOS :	34
4.2.2Algorithmes d'ordonnancement de FreeRTOS :	35
4.3 Les cartes ARDUINO.....	36
4.3.1 Pourquoi choisir Arduino ?.....	37
4.3.2 Types de cartes.....	39
4.4 Conclusion	39
Chapitre 05 : Étude de cas.....	40
5.1 Introduction	40
5.2 Spécification informelle	40
5.2.1 Aspect matériel :	41
5.2.2 Aspect fonctionnel :	41
5.2.3 Les Paramètres de coupe [21] :.....	42
5.2.4 Remarque :	43
5.3 Spécification formelle.....	44
5.3.1 Noyau :.....	44
5.3.1.1Spécification :.....	44
5.3.1.2Conception	45
5.3.1.3 Test.....	46
5.3.2 Transaction du noyau à l'incrément 01 :	47
5.3.2.1 Les nouvelles fonctionnalités :.....	47
5.3.3 Incrément 01 :	47
5.3.3.1 Spécification :.....	47
5.3. 3.2 Conception	49

5.3.3.3 Test	49
5.3.3 Transaction du l'incrément 01à l'incrément 02	50
5.3.3.1 Les nouvelles fonctionnalités	50
5.3.4 Incrément 02	50
5.3.4.1 Spécification :.....	50
5.3.4.2 Conception	51
5.3.4.3 Test.....	52
5.3.5 Transaction du l'incrément 02 à l'incrément 03	53
5.3.5.1 Les nouvelles fonctionnalités	53
5.3.5.2 Système de lubrification.....	53
5.3.6 Incrément 03	54
5.3.6.1 Spécification :.....	54
5.3.6.2 Conception	55
5.3.6.3 Test.....	56
5.4 Mesure des métriques de complexité et de qualité	57
5.5 Quelles sont les limites acceptables ?.....	57
Conclusion générale	59
Bibliographié.....	60

Liste des figures

Figure 01 : Représentation schématique d'un appareil de mesure.....	12
Figure 02 : composant d'un microcontrôleur	13
Figure 03 : Représentation schématique d'un capteur.	14
Figure 04 : Symbole électrique d'un convertisseur analogique-numérique.....	14
Figure 05 : Représentation schématique d'un système embarqué avec son environnement	16
Figure 06 : Graphe à barres présent l'état de marché [3].....	16
Figure 07 : Représentation schématique d'un système en temps réel avec son environnement. .	19
Figure 08 : Etats des processus.	21
Figure 09 : Modèle incrémental.	24
Figure 10 : représentation graphique de différent aspect de SA-RT.....	26
Figure 11 : symbole des objets configurable.....	28
Figure 12 : symbole des objets programmable.	28
Figure 13 : exemple d'un exécutif temps réel Vx Works	29
Figure 14 : représentation schématique d'un exécutif temps réel.....	30
Figure 15 : le système de fichiers du noyau « FreeRTOS ».....	32
Figure 16 : les états du tache sure FreeRTOS [18].	33
Figure 17 : Observation de marché des OS et RTOS pour l'embarqué [19].	34
Figure 18 : exemple de « Prioritized Preemptive Scheduling with Time Slicing ».....	35
Figure 19 :« Carte Arduino UNO ».....	36
Figure 20 : « Arduino 1.8.1 IDE».....	36
Figure 21 : Observation de marché des cartes pour les systèmes embarqués [19].	38
Figure 22 :« les différents modèles des carte ARDUINO ».....	39
Figure 23 : Principe de l'usinage.	40
Figure 24 : les organes principaux de la machine tour.....	41
Figure 25 : Schéma des mouvements d'un tour parallèle.....	42
Figure 26 : Chaîne cinématique d'un tour.....	42
Figure 27 : diagramme de contexte.....	44
Figure 28 : diagramme de flot de données.....	45
Figure 29 : diagramme de flot de contrôle.....	45
Figure 30 : schéma multitâche.....	46
Figure 31 : schéma de câblage.....	46

Figure 32 : le chronogramme de la séquence d'exécution.....	46
Figure 33 : la tâche afficher.	47
Figure 34 : la boîte au lettre.	47
Figure 35 : diagramme de contexte.....	48
Figure 36 : diagramme de flot de données.	48
Figure 37 : diagramme de flot de contrôle.....	48
Figure 38 : schéma multitâche.	49
Figure 39 : schéma de câblage.	49
Figure 40 : le chronogramme de la séquence d'exécution.....	49
Figure 41 : la tâche calculer température.	50
Figure 42 : digramme de contexte.....	50
Figure 43 : digramme de flot de données.....	51
Figure 44 : diagramme de flot de contrôle.....	51
Figure 45 : schéma multitâche.	52
Figure 46 : schéma de câblage.	52
Figure 47 : le chronogramme de la séquence d'exécution.....	52
Figure 48 : la tâche lubrificateur.	53
Figure 49 : un système de lubrification basic.....	53
Figure 50 : diagramme de contexte.....	54
Figure 51 : diagramme de flot de données.....	54
Figure 52 : diagramme de flot de contrôle.....	55
Figure 53 : schéma multitâche.....	55
Figure 54 : schéma de câblage.....	56
Figure 55 : le chronogramme de la séquence d'exécution.....	56

List des tableaux

Tableau 01 : Tableau des contraintes de système embarqué selon secteur d'activité	20
Tableau 02 : Les différents éléments graphiques de la méthode SA-RT.....	25
Tableau 03 : raccourcis de barre d'actions.	37
Tableau 04 : les métriques des lignes de code LOC	57
Tableau 05 : les résultats de nos codes	58

Liste des abréviations

SA-RT : « Structured Analysis for Real Time ».

LACATRE : « Langage d'Aide à la Conception d'Applications Temps Réel ».

EMS: « Embedded Measurement Systems ».

FPGA: « field-programmable gate array ».

MEMS: « Micro Electro Mechanical System ».

RTD: « resistive thermal device ».

MCU: « microcontroller unit ».

CPU: « central processing unit ».

ECG: « electrocardiography ».

RAM: « Random-access memory ».

E / S: « entrée/sortie».

ADC: « analog-to-digital converter ».

DAC: « Digital-to-analog converter ».

TCP: « Transmission Control Protocol».

IP: « Internet Protocol ».

WEB: « World Wide Web ».

ROM: «Read-only memory ».

EPROM: «erasable programmable ROM ».

EEPROM: «electrically erasable programmable ROM ».

GAB: « guichet automatique bancaire».

UART: «Universal Asynchronous Receiver Transmitter ».

PDA: « personal digital assistant».

LED: «light emitting diode ».

ICE: «In Circuit Emulator ».

JTAG: « Joint Test Action Group».

JSD: « Jackson System Design ».

DARTS: « Design Approach for Real-Time Systems ».

UML-RT: « Unified Modeling Language - Real Time ».

MSMC : « Modélisation Simulation des Machines Cybernétiques ».

DDC : « diagramme de contexte ».

DFD : « diagramme de flot de données ».

DFC : « le diagramme de flot de contrôle ».

SDL-RT: « Specification and Description Language - Real Time ».

ROOM: « Real-Time Object-Oriented Modeling ».

ETR : « exécutif temps réel ».

RTD : « résistance température detector ».

Chapitre 01 : Les systèmes de mesure embarqués

1.1 Introduction

Les systèmes de mesure deviennent plus sophistiqués pour satisfaire les besoins énormes de données, qui sont collectés et traités par les laboratoires de recherche dans un cadre académique ou industriel, aujourd'hui les systèmes de mesure sont représentés par la fusion des systèmes embarqués et systèmes de mesure électriques, chimique, etc. c'est ce que l'on appelle EMS (Embedded Measurement Systems) Cela indique que les EMS sont des systèmes matériels-logiciels dédiés à mesurer une ou quelques grandeurs physiques ou bien chimique [1].

Les systèmes numériques embarqués comme microcontrôleurs et les circuits logiques programmables FPGA, en automatique et systèmes de mesure informatisés ont été utilisés dans systèmes de mesure depuis l'introduction du premier microcontrôleur par Intel en 1976 et sont aujourd'hui largement utilisés dans les applications de mesure.

Les applications sont nombreuses; Les EMS mesurent : la température dans les réfrigérateurs, congélateurs, fers à repasser, fours et moteurs à combustion d'automobiles, les vibrations dans les alarmes d'inclinaison et les consoles de jeux, le débit d'air dans les moteurs et les systèmes de ventilation ils mesurent l'impact des chocs dans les détecteurs de collision et sont utilisés comme enregistreurs de chocs et de température pendant le transport des marchandises, la pression de l'air dans les cabines d'avion, l'humidité dans l'air conditionné environnements, les niveaux de liquide dans les réservoirs de carburant, la fumée dans les alarmes incendie, la viscosité de l'huile lubrifiante dans les moteurs, la vitesse de rotation des roues moteur, le couple dans les moteurs et sont utilisés comme détecteurs de fréquence cardiaque et d'ECG en médecine, etc.

La demande commerciale de produits toujours moins chers et les législations environnementales mondiales imposent aux fournisseurs à rechercher en permanence des solutions plus rentables, avec précision et moins gourmandes en énergie pour leurs systèmes de mesure embarqués, l'attention croissante dans le monde sur les problèmes environnementaux, santé, l'énergie réduction de la consommation et des coûts, oblige les concepteurs de systèmes mesures embarquées pour rechercher en permanence des solutions plus rentables et moins énergivores, les systèmes de mesure embarquée sont intéressants, car le coût d'une solution intégrée est généralement bien inférieur au coût d'un instrument informatisés commercial.

1.2 Les systèmes de mesure

1.2.1 Définition

Un système de mesure est le matériel et / ou le logiciel nécessaire pour mesurer l'ampleur de la quantité d'intérêt. Il peut être mécanique, électrique, chimique ou hybride, un baromètre à mercure est un système de mesure mécanique, tandis qu'un multi numérique mètre est un système de mesure électrique, un accéléromètre MEMS intégré (Micro Electro Mechanical System) est un exemple de système hybride.

Tous les systèmes de mesure commencent par un capteur ou un élément de détection sensible, comme indiqué dans la figure suivant, un capteur traduit une variation d'une quantité physique en une variation de une certaine quantité électrique, par exemple un dispositif thermique résistif RTD, change sa résistance lorsque la température change, les capteurs sont actifs ou passif selon qu'ils ont besoin ou non d'un support d'alimentation externe, la quantité électrique qui varie dans un capteur passif est la résistance, l'échantillon analogique est transformé en numérique conviviale avec l'élément de traitement (microprocesseur, microcontrôleur).

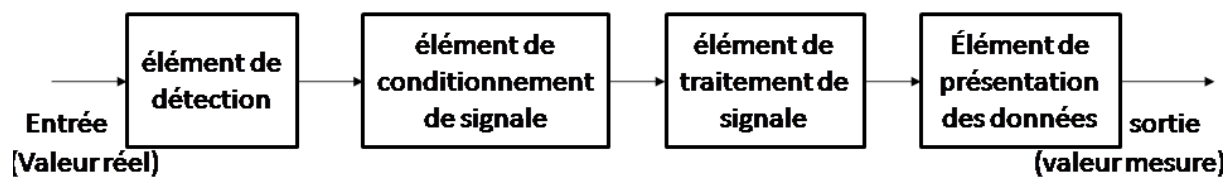


Figure 01 : Représentation schématique d'un appareil de mesure.

1.2.2 Quelques exemples d'appareils de mesure

Il existe plusieurs types d'appareil de mesure pour les différents domaines d'utilisation, mécanique, électrique, ou bien chimique, comme les exemples suivants.

Ampèremètre : Un ampèremètre est un appareil de mesure de l'intensité d'un courant électrique dans un circuit. L'unité de mesure de l'intensité est l'ampère.

Analyseur de spectre : Un analyseur de spectre est un instrument de mesure destiné à afficher les différentes fréquences contenues dans un signal ainsi que leurs amplitudes respectives. Les signaux peuvent être de natures diverses : électrique, optique, sonore, radioélectrique.

Conductimètre : Un conductimètre, est un appareil permettant de mesurer une propriété de conductivité.

1.3 Les microcontrôleurs

1.3.1 Définition

Un microcontrôleur MCU est une seule puce intégrée qui contient tout matériel nécessaire pour fonctionner comme un ordinateur autonome, par rapport à un microprocesseur, qui ne contient que le CPU, un microcontrôleur contient à la fois des données mémoire RAM et de la mémoire programme flash et du matériel d'E / S, compteurs, ADC, ports série, etc.

Un microcontrôleur est conçu pour fonctionner dans un système embarqué, tandis qu'un microprocesseur est généralement l'unité centrale dans un ordinateur à usage général.

Les microcontrôleurs sont plus compacts et plus rentables, tandis que les microprocesseurs sont plus flexibles.

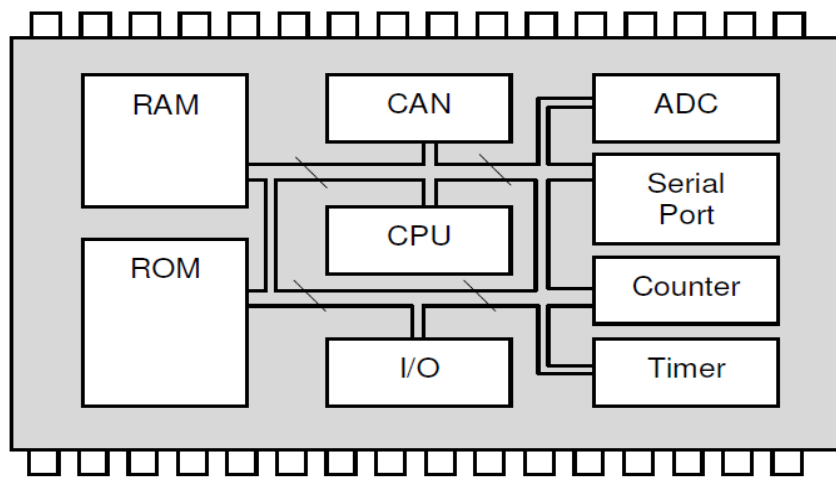


Figure 02 : composants d'un microcontrôleur [1]

Il existe plusieurs familles industrielles et académiques disponibles pour les chercheurs et les grands fabricants, on cite les plus disponibles dans le marché algérien comme la famille Atmel AT91, ARM Cortex-M, Atmel AVR, Siemens, Intel 80, PIC.

1.3.2 Les capteurs :

Dans de nombreux domaines (industrie, recherche scientifique, services, loisirs...), on a besoin de contrôler des paramètres physiques (température, force, position, vitesse, luminosité...), le capteur est l'élément indispensable à la détection de ces grandeurs physiques, est un organe de prélèvement d'informations qui élabore à partir d'une grandeur physique, une autre grandeur physique de nature différente (souvent électrique), cette grandeur représentative de la grandeur prélevée est utilisable à des fins de command ou de mesure comme notre cas d'étude.

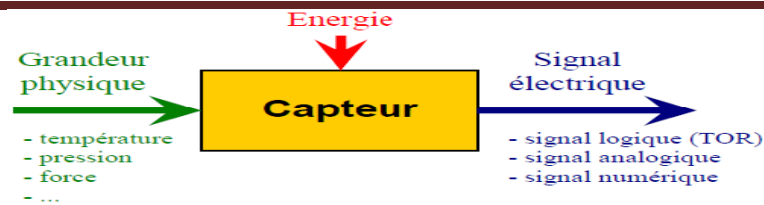


Figure 03 : Représentation schématique d'un capteur [1].

1.3.3 Interfaçage des capteurs avec les microcontrôleurs

le problème d'interfaçage du capteur analogique avec le microcontrôleur numérique, impose le développement d'un grand nombre de techniques par exemple les convertisseurs analogiques-numériques ADC, la grande variété de méthodes d'interfaçage est motivée par le fait que différentes applications ont des priorités différentes, en fonction, de l'application et l'environnement dans lesquels l'EMS est destiné à fonctionner, il existe plusieurs paramètres différents pour optimiser la conception, ces paramètres incluent par exemple: coût, taille, poids, consommation électrique, fiabilité, disponibilité et fabricabilité.



Figure 04 : Symbole électrique d'un convertisseur analogique-numérique [1].

1.4 Conclusion

Le sujet des systèmes de mesure embarqués est un mélange interdisciplinaire de la physique, de la technique de mesure électrique et de la technologie des micro-ordinateurs, nous nous sommes concentrés sur l'étude de systèmes de mesure électroniques avec l'utilisation d'un microcontrôleur peu coûteux pour fournir des solutions rentables, avec objectif principal de démontrer qu'un ordinateur de bureau coûteux et énergivore, peut être souvent remplacé par un appareil embarqué peu coûteux et de faible puissance. L'acquisition de données en temps réel provient de différents types d'appareils de mesures (analogique, TCP-IP, ...), de plusieurs protocoles de communication de plusieurs marques de constructeurs différents. Il y'a un besoin d'analyse et validation automatique des données en temps réel en fonction du contexte d'acquisition de la donnée, plus le transfert automatique des données vers des logiciels ou vers d'autre système, la gestion automatique des transmissions instantanée des données, des alertes en cas de dysfonctionnement de l'appareil ou des systèmes, serveur WEB avec interface ergonomique et intuitive pour une prise en main et un contrôle distant du système. Tous ces besoins nous obligent à développer et perfectionner ce type de système temps réel embarqué.

Chapitre 02 : Les systèmes temps réel embarqués

2.1 Introduction

Les systèmes embarqués font partie intégrante de notre vie, la majorité des individus possèdent aujourd'hui un système embarqué que ce soit chez soi, dans sa maison, véhicule ou bien directement sur soi. Nous pouvons actuellement en compter plus d'un milliard ayant chacun ses spécificités, recouvrant tous les domaines que ce soit l'aéronautique, l'électroménager, la téléphonie ou encore l'automobile.

L'amélioration de la qualité des produits et l'ajout constant de nouvelles fonctionnalités sont directement dues à l'intégration de systèmes embarqués, cela a permis, dans le cadre du développement économique, de créer de la valeur ajoutée, tout en améliorant la compétitivité de l'entreprise fabricante. Ainsi, les systèmes embarqués apportent une plus-value certaine aux caractéristiques du produit permettant de se différencier dans des marchés de masse tout en apportant une disponibilité de fonctions techniques et complexes. L'industrie est en perpétuelle évolution avec une concurrence massive obligeant l'innovation et la démarcation des principaux acteurs du marché, l'intégration de nouveaux systèmes embarqués plus puissants, plus performants et pouvant proposer tout type de fonctionnalités, permet de répondre aux besoins des consommateurs de plus en plus exigeants.

2.2 Les systèmes embarqués

2.2.1 Définition

Un système embarqué est un système électronique et informatique autonome composé de matériel informatique et de logiciels, et peut-être d'autres pièces mécaniques ou autres, conçues pour exécuter une fonction spécifique dédiée à une tâche précise, souvent en temps réel, possédant des ressources limitées (encombrement réduit) et ayant une consommation énergétique restreinte [2], un système embarqué est un système informatique dans lequel le logiciel est encapsulé par le matériel qu'il contrôle

2.2.2 Architecture

Les systèmes embarqués utilisent généralement des microprocesseurs à basse consommation d'énergie ou des microcontrôleurs, dont la partie logicielle est en partie ou entièrement programmée dans le matériel, généralement en mémoire morte (ROM), EPROM, EEPROM, FLASH, etc.

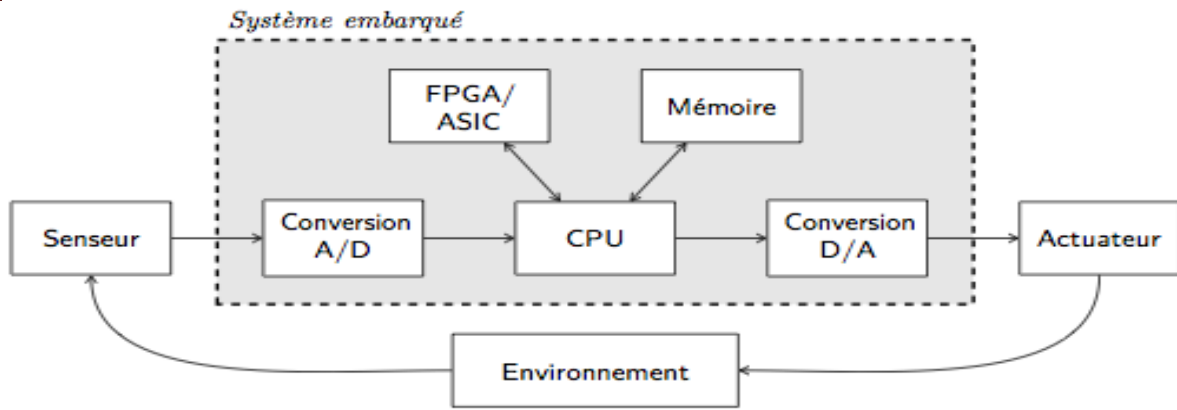


Figure 05 : Représentation schématique d'un système embarqué avec son environnement

Les systèmes embarqués sont la plupart du temps dans des machines qui doivent fonctionner en continu pendant de nombreuses années, sans erreurs et, dans certains cas, réparer eux-mêmes les erreurs quand elles arrivent. C'est pourquoi les logiciels sont toujours développés et testés avec plus d'attention que ceux pour les ordinateurs. Les pièces mobiles non fiables (par exemple les lecteurs de disques, boutons ou commutateurs) sont proscrites.

2.2.3 Domaines d'applications

Les domaines dans lesquels on trouve des systèmes embarqués sont de plus en plus nombreux : astronautique (fusée, satellite artificiel, sonde spatiale), automate programmable industriel, électroménager (télévision, four à micro-ondes), environnement (station de prédiction météo), équipement médical, guichet automatique bancaire (GAB), impression, informatique (disque dur, Lecteur de disquette), militaire (Missile), multimédia (console de jeux vidéo, assistant personnel), télécommunication (routeur, Téléphone portable), transport (Automobile, Aéronautique), etc.

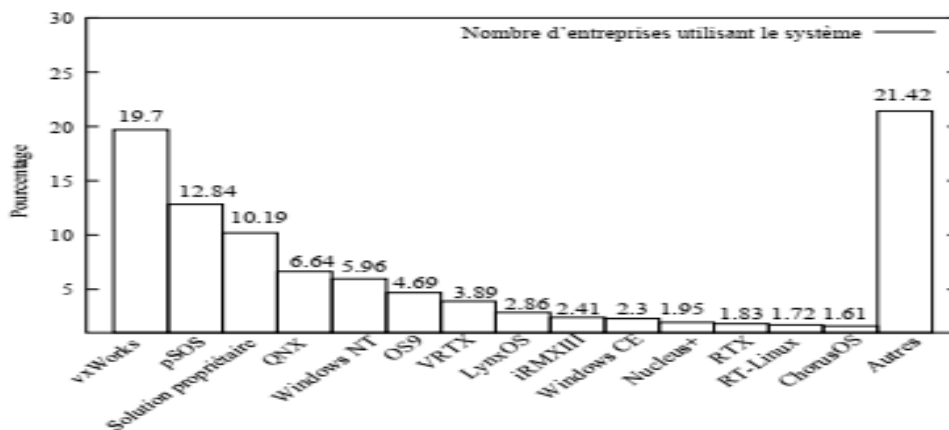


Figure 06 : Graphe à barres présent l'état de marché [3].

Le graphe à barres précédent représente le pourcentage d'utilisation des systèmes temps réel par les grandes entreprises.

2.2.4 Les concepts principaux d'un système embarqué :

Entrée / sortie à usage général : Ce sont exactement comme les ports d'E / S d'ordinateur, mais beaucoup plus simples. Ils peuvent soit sortir l'une des deux valeurs (0 logique correspond à 0V, 1 logique correspond à la tension d'alimentation la plupart du temps, disons 3,3V) ou lire la valeur sur le port (en tant que valeur logique).

Les interruptions : Les interruptions sont utiles lorsque vous essayez d'exécuter un morceau de code après que quelque chose se soit produit. Contrairement à l'interrogation, où nous attendons à l'intérieur d'une boucle qu'un événement se produise (tel qu'un drapeau défini), nous activons une interruption matérielle et écrivons un gestionnaire d'interruption - ici nous mettons le code que nous voulons exécuter lorsque l'événement arrive.

L'horloge : Les horloges de base comptent en continu, en augmentant et en descendant, et peuvent être utilisées pour les délais d'attente, en comptant la durée d'un processus, en créant une horloge en temps réel et bien d'autres.

Convertisseurs analogique-numérique : Les convertisseurs analogique-numérique nous permettent de convertir une tension en un nombre avec lequel nous pouvons travailler dans un logiciel. Nous l'utiliserions pour mesurer la tension d'une batterie, telle que la propre batterie du système ou pour lire certains capteurs. Le contraire d'un ADC est un DAC ou un convertisseur numérique-analogique, qui délivre une certaine tension en fonction du code que nous fournissons au DAC.

Les interfaces série : Les interfaces série nous permettent de communiquer avec d'autres circuits intégrés ou même avec un ordinateur. L'interface série la plus basique généralement disponible est UART, qui signifie récepteur / émetteur asynchrone universel et ne nécessite que 2 fils de données - TX et RX. Il signale le début d'une communication via un bit de démarrage, suivi de 8 bits de données, éventuellement un bit de parité pour le contrôle d'erreur de base et un bit d'arrêt.

2.2.5 Caractéristiques de système embarqué

Contrairement aux systèmes universels effectuant plusieurs tâches, les systèmes embarqués sont étudiés pour effectuer des tâches précises. Certains doivent répondre à des contraintes de temps

réel pour des raisons de fiabilité et de rentabilité. D'autres ayant peu de contraintes au niveau performances permettent de simplifier le système et de réduire les coûts de fabrication.

Les systèmes embarqués ne sont pas toujours des modules indépendants. Le plus souvent ils sont intégrés dans le dispositif qu'ils contrôlent, le logiciel créé pour les systèmes embarqués est appelé firmware. Il est stocké dans la mémoire en lecture seule ou la mémoire flash plutôt que dans un disque dur. Il fonctionne le plus souvent avec des ressources matérielles limitées : écran et clavier de tailles réduites, voire absent, peu de mémoire, capacités de calcul relativement faibles.

Certains systèmes embarqués peuvent ne pas avoir d'interface utilisateur (ils sont alors spécialisés dans une seule tâche). Mais cette interface peut également être similaire à celle d'un système d'exploitation d'ordinateur (par exemple un PDA), ces systèmes les plus simples comportent uniquement des boutons, des LED, Les systèmes les plus complexes peuvent avoir un écran tactile ou encore un écran comportant des boutons de façon à minimiser l'espace. La signification des boutons change selon l'écran et la sélection se fait naturellement en pointant la fonction désirée.

2.3 Les systèmes temps réel

2.3.1 Définition 01

Un système temps réel est un système qui interagit avec un environnement physique en remplissant souvent des missions critiques pour lesquelles une faute du système peut avoir des conséquences graves. Il sera dit correct s'il possède les bonnes fonctionnalités et si celles-ci sont réalisées à temps, c'est-à-dire avec le respect des contraintes temporelles imposées par l'environnement ou par une certaine qualité de service offerte à un utilisateur. La validation fonctionnelle et temporelle des systèmes temps réel est ainsi une nécessité forte. Toutes les situations, tous les comportements du système doivent être envisagés pour que la validation fournisse des résultats fiables [4].

2.3.2 Définition 02

En informatique, on parle d'un système temps réel lorsque ce système informatique contrôle un procédé physique à une vitesse adaptée à l'évolution du procédé contrôlé. Ce système doit réagir continuellement à des événements en provenance du procédé contrôlé avec des contraintes temporelles dont le respect est aussi important que l'exactitude du résultat. Une réaction ne

respectant pas son échéance pourrait causer une catastrophe humaine et financière. Un tel système comporte deux parties fondamentales : le matériel et le logiciel [5].

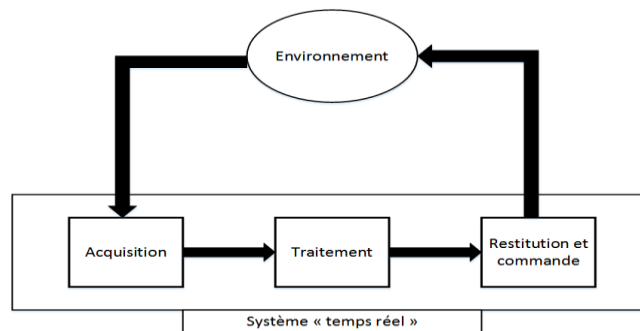


Figure 07 : Représentation schématique d'un système en temps réel avec son environnement.

2.4 Les systèmes temps réel embarqués

Ce sont des systèmes embarqués dans lequel l'exactitude des applications ne dépend pas seulement du résultat mais aussi du temps auquel ce résultat est produit. Si les contraintes temporelles de l'application ne sont pas respectées, on parle de défaillance du système.

2.4.1 Contraintes des systèmes embarqués en temps réel :

Une puissance de calcul définie au plus juste afin de répondre aux besoins tout en respectant les contraintes temporelles et spatiales - l'objectif étant d'éviter les surcoûts et les éventuelles surconsommations d'énergie.

Ayant un espace mémoire limité de l'ordre de quelques mégaoctets maximums, bien que la taille vienne à être de moins en moins limitée grâce à la miniaturisation des éléments. Il convient de concevoir des systèmes embarqués qui répondent aux besoins au plus juste pour éviter un surcoût.

La consommation énergétique doit être la plus faible possible, due à l'utilisation de batteries et/ou, de panneaux solaires voire de pile à combustible pour certains prototypes.

Temporel, dont les temps d'exécution et l'échéance temporelle d'une tâche sont déterminés (les délais sont connus ou bornés a priori).

De sûreté de fonctionnement. Car s'il arrive que certains de ces systèmes embarqués subissent une défaillance, ils mettent des vies humaines en danger ou mettent en péril des investissements importants. Ils sont alors dits « critiques » et ne doivent jamais faillir. Par « jamais faillir », il

faut comprendre toujours donner des résultats justes, pertinents et ce dans les délais attendus par les utilisateurs (machines et/ou humains) des dits résultats.

Une sécurité indispensable pour assurer la confidentialité des données utilisées, notamment pour les systèmes employés au service de la santé.

Secteur d'activité	Contraintes
Équipements scientifiques	Performances, fiabilité, coût
Équipements militaires et aérospatiaux	Performances, fiabilité, pérennité, intégration
Transports	Fiabilité, coût, interactivité
Informatique industrielle	Fiabilité, coût, pérennité
Matériel de bureau	Performance, coût, standardisation
Réseau et télécommunications	Performance, fiabilité, intégration
Électronique grand public	Performance, coût, design / intégration

Tableau 01 : Tableau des contraintes de système embarqué selon secteur d'activité

2.5 La notion du multitâche

Lorsqu'on parle sur les systèmes temps réel, il faut toujours aborder la notion de multitâche, un système informatique est dit, en multitâche, s'il permet alternativement dans le temps, l'exécution de plusieurs programmes coexistant en mémoire, tout en assurant leur intégrité.

Un système multitâche est souvent concerné par un partage de ressources, dans la plupart des cas, ces ressources peuvent être utilisées que par un seul processus à la fois et, leur utilisation, ne peut être interrompu, tandis que la ressource CPU peut, contrer par elle-même la simultanéité d'accès, les procédures dans lesquelles le code est partageable en mémoire ne peuvent y pallier. Un tel code est dit, section critique, si deux ou plusieurs processus accèdent à la fois, à la section critique, une erreur fatale voire catastrophique se produira.

2.5.1 Notion de processus

C'est une entité dynamique qui naît, qui vit et qui meurt, par opposition à la notion de programme qui est une entité statique qui occupe un espace en mémoire ou sur disque et qui, tant qu'elle n'est pas exécutée, ne produit aucune action.

D'autre part, un programme peut être vu comme un ensemble de modules (sous-programmes) et l'exécution d'un module donne naissance à un processus.

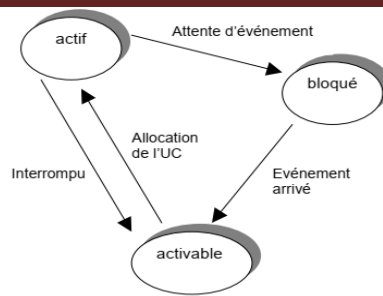


Figure 08 : Etats des processus.

Remarque

Les notions présentées pour les processus sont, également valable pour les tâches. Cependant, nous aurons tendance à mesure que nous progressions dans ce document, d'utiliser le terme tâche.

2.6 Conclusion

Pour concevoir un système embarqué et temps réel, il faut généralement combiner des compétences en électronique, en informatique et en automatique, parmi le matériel nécessaire pour réaliser un système embarqué et temps réel on trouve :

- La documentation (datasheet) sur les composants utilisés. C'est la première source d'informations pour le développement !
- L'outillage de base de l'électronicien (fer à souder, etc....)
- Les outils d'analyse temporelle : oscilloscope, analyseur logique...
- Des composants de base (résistances, condensateurs...)
- Un microprocesseur ou un microcontrôleur
- Un compilateur croisé (dit aussi en anglais cross-compiler)
- Un programmeur de microcontrôleur ou un programmeur in-situ

Chapitre 03 : Méthode de développement d'un système temps réel embarqué

3.1 Introduction

Avec l'évolution de la technologie, le développement de certains systèmes se pose comme une obligation, cette opération consiste à transformer le principe d'un nouveau produit, procédé ou service en réalisation industrielle prête à être commercialisée ou exploitée.

Le développement de systèmes embarqués composés de logiciels, de l'électronique et des composants mécaniques qui fonctionnent dans un monde physique est un défi très complexe, parmi les systèmes embarqués, le système de mesures a la particularité d'avoir un couplage fort entre l'aspect matériel et logiciel.

Pour le développement de ce type de système, on utilise la méthode SART ce qui est programmée dans notre formation master, et le cycle de vie généralement utilisé pour écrire le développement des systèmes embarqués est le modèle incrémental, pour le découpage du processus de développement et raffinements successifs, chaque développement qui s'ajoute pour l'amélioration du système enrichit l'existant tout en évitant les coûts d'une solution commerciale nouvelle. De plus, le temps de formation du personnel est réduit [6].

Chapitre 3 : ...Méthode de développement d'un système temps réel embarqué

3.2 Développement des systèmes informatiques

3.2.1 Cycle de vie d'un système

Le cycle de vie d'un système est constitué d'un ensemble d'étapes.

1. Excrétion des besoins Il s'agit de rédiger un cahier des charges relatif aux besoins du client en matière de matériels, de logiciels et de personnel.
2. Conception préliminaire Elle comporte deux aspects, l'un fonctionnel et l'autre organisationnel. L'aspect fonctionnel concerne la description des services que le système devra fournir (quoi), l'aspect organisationnel, l'intérêt est porté sur comment ces services vont être réalisé.
3. Conception détaillée IL s'agit d'affiner (détailler) les aspects fonctionnel et organisationnel du système.
4. Validation du système cette phase a lieu sur le site, son objectif est de démontrer au client que le système développé répond effectivement aux besoins exprime dans le cahier des charges fonctionnel (résultat de la conception préliminaire) et aux spécifications techniques des besoins.
5. Maintenance du système c'est une étape qui nécessite une collaboration entre le concepteur du système et l'utilisateur. Pour qu'un système reste opérationnel, il est nécessaire de l'améliorer, de l'adapter et de le corriger.

3.2.2 Les modèles classiques de cycle de vie :

Le modèle en cascade, le modèle en V, le modèle incrémental, le modèle en spirale. Tous ces modèles ne sont pas les mêmes, mais l'idée est toujours la même : le logiciel est développé en phases. Sans vouloir faire un expose exhaustif de ces modèles, il est intéressant de citer une démarche de développement.

Pour ce projet on ait choisir le modèle incrémental, ce modèle de cycle de vie prend en compte le fait qu'un logiciel peut être construit étape par étape, les besoins sont figés et clairement identifiables et Le logiciel est spécifié et conçu dans son ensemble.

La réalisation se fait par incréments de fonctionnalités. Chaque incrément est intégré à l'ensemble des précédents et à chaque étape le produit est testé exploité et maintenu dans son ensemble. Ce cycle de vie permet de prendre en compte l'analyse de risques et de faire accepter progressivement un logiciel par les utilisateurs plutôt que de faire un changement brutal des habitudes [7].

Chapitre 3 : ...Méthode de développement d'un système temps réel embarqué

Le modèle incrémental a été proposé dans les années 1980. Les premiers incréments peuvent être des maquettes (jetables s'il s'agit juste de comprendre les exigences des utilisateurs) ou des prototypes (réutilisables pour passer au prochain incrément en les complétant et/ou en optimisant leur implantation) [8].

En effet, lorsqu'un système est destiné à piloter un procédé, l'analyse des besoins se déduit naturellement des lois de commande conçues pour assurer ce pilotage et des caractéristiques physiques de l'installation à contrôler. De plus, une approche par étape est statique et donc bien adaptée à la certification (ce qui est une nécessité pour la majorité des systèmes critiques), de plus, dans les domaines comme l'industrie [9].

Ce modèle de cycle de vie prend en compte le fait qu'un logiciel peut être construit étape par étape. Le logiciel est spécifié et conçu dans son ensemble. La réalisation se fait par incréments de fonctionnalités. Chaque incrément est intégré à l'ensemble des précédents et à chaque étape le produit est testé exploité et maintenu dans son ensemble. Ce cycle de vie permet de prendre en compte l'analyse de risques et de faire accepter progressivement un logiciel par les utilisateurs plutôt que de faire un changement brutal des habitudes.

Exemples:

Un Scheduler (ordonnanceur) ou un gestionnaire de fichiers peuvent constituer des incréments d'un système d'exploitation.

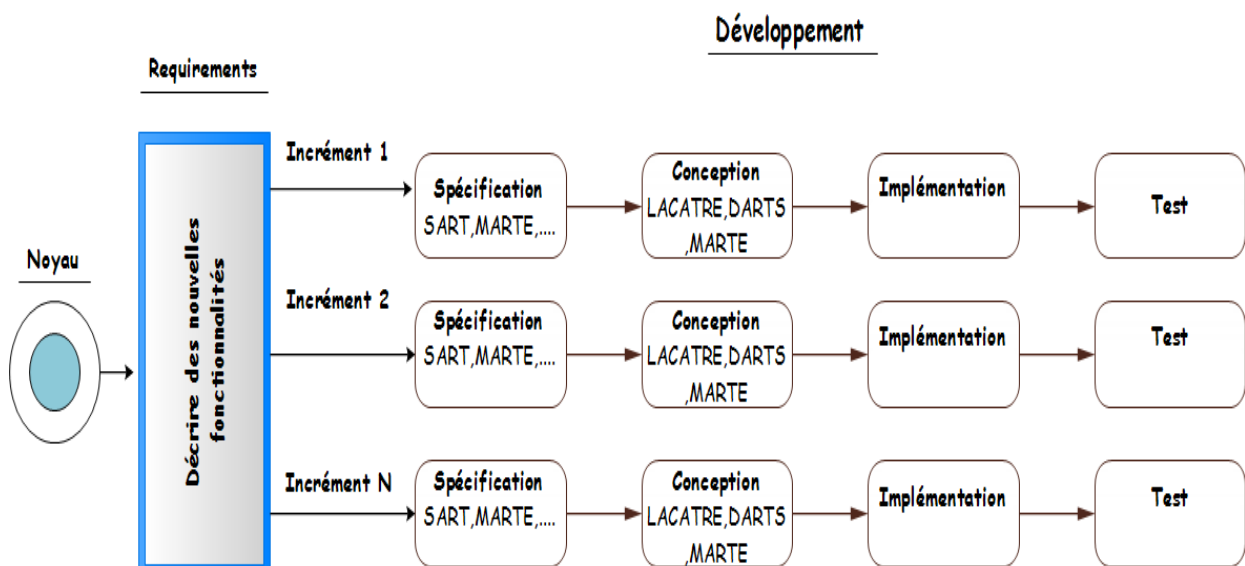


Figure 09 : Modèle incrémental.

Chapitre 3 : ...Méthode de développement d'un système temps réel embarqué

3.3 Les méthodes de spécification pour un système temps réel

Etant donné la complexité croissante des systèmes et l'apparition de contraintes temps réel de plus en plus sophistiquées plusieurs méthodes ont été mise au point pour l'analyse, spécification et conception des systèmes temps réel parmi lesquelles on peut citer.

- **JSD** : « Jackson System Design ».
- **SA-RT**: « Structured Analysis Real Time » [10].
- **DARTS**: « Design Approach for Real-Time Systems ».
- **SDL**: « Specification and Description Language ».
- **UML-RT**: « Unified Modeling Language - Real Time » [11].
- **MSMC** : « Modélisation Simulation des Machines Cybernétiques ».

Le travail présenté dans cette mémoire est basé sur la méthode d'analyse structuré pour les systèmes temps réel SA-RT

3.3.1 La méthode SA-RT

La méthode SA-RT est une méthode d'analyse fonctionnelle et opérationnelle des applications temps réel. Cette méthode permet de réaliser une description graphique et textuelle de l'application en termes de besoins, c'est-à-dire de « ce que l'on doit faire » ou le « quoi », cette mise en forme du « cahier des charges » de l'application est formelle dans le sens où la méthodologie (ensemble des documents à élaborer) et l'expression (syntaxe graphique) sont définies [4].

En revanche, elle ne permet pas d'effectuer une vérification de propriétés de l'application à partir des seules descriptions SA-RT.


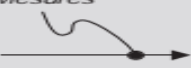
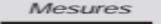

Fonction	Signification	Représentation graphique	
Traitement ou process	Unité de travail qui réalise la transformation des données d'entrée en données de sortie	- Cercle ou bulle - Action décrite par : verbe + nom	
Flot de données	Vecteur nommé reliant deux process, sur lequel circule un ensemble de données de même nature	- Flèche en trait plein - Donnée nommée	
Unité de stockage ou réservoir	Entité ou zone de rangement de données	- Deux traits parallèles - Entité nommée	
Entité externe ou terminateur	Provenance, source ou destination des données	- Rectangle - Entité nommée	

Tableau 02 : Les différents éléments graphiques de la méthode SA-RT.

Chapitre 3 : ...Méthode de développement d'un système temps réel embarqué

La SA-RT considère la spécification des systèmes temps réel selon 3 points de vue orthogonaux :

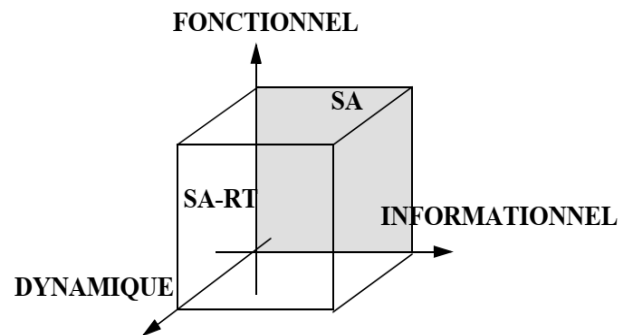


Figure 10 : représentation graphique de différents aspects de SA-RT.

Aspect fonctionnel : décompose le système en une hiérarchie de fonctions présentée par le diagramme de contexte DDC.

Aspect informationnel : définit les données manipulées par les fonctions présentées par le diagramme de flot de données DFD.

Aspect dynamique : exprime le contrôle du flux des données et l'activation des fonctions présentées par le diagramme de flot de contrôle DFC. [12]

3.3.2 Objectif de méthode SA-RT

L'objectif de SA-RT est d'obtenir un modèle des besoins permettant le développement :

- Conformité aux besoins réels de l'utilisateur.
- Communicabilité (i.e. interprétables sans erreurs).
- Faisabilité.
- Modifiabilité.
- Validabilité.
- Complétude.
- Cohérence.

Chapitre 3 : ...Méthode de développement d'un système temps réel embarqué

3.4 Les langages de conception pour un système temps réel

Lors de la conception d'un système temps réel, plusieurs approches sont possibles, qui peuvent s'appuyer sur des outils classiques standards comme langage C/ C++ et quelquefois l'assembleur et des exécutif multitâches temps réel tels que FreeRTOS, Vx Works, POSIX, etc.

Afin de s'abstraire des détails d'implémentation et de ne pas se lier à une plateforme cible précise, des langages de haut niveau ont été proposés, utilisés pour la conception tels que [4] :

LACATRE : Langage d'Aide à la Conception d'Applications Temps Réel [13].

SDL-RT: « Specification and Description Language - Real Time » [14].

UML-RT: « Unified Modeling Language - Real Time » [15].

ROOM: « Real-Time Object-Oriented Modeling» [16].

Remarque : la méthode prescrite dans notre programme de formation master est LACATRE, qui est utilisée dans la conception de notre système.

3.4.1 Le langage LACATRE

LACATRE (Langage d'Aide à la Conception d'Applications Temps Réel) est un langage graphique dédié à la description des objets élémentaires utilisés dans les exécutifs multitâches temps réel.

Présentant un modèle graphique et un modèle textuel destinés à faciliter la conception préliminaire et détaillée d'application basée sur la mise en œuvre d'exécutifs multitâches temps réel.

LACATRE fait un usage intensif et rigoureux du symbolisme graphique pour la représentation des mécanismes de communication / synchronisation, permettant ainsi de donner une vision synthétique, et précise à la fois, de l'application multitâche temps réel.

Le langage LACATRE se présente comme une surcouche aux exécutifs temps réel, il opère ainsi sur des entités qui sont en partie l'image des objets manipulés par les exécutifs multitâche temps réel, avec quelques extensions, ces entités appelées les objets LACATRE, sont : la tâche, les routines, le sémaphore, la boîte aux lettres, le message, l'évènement, la ressource, l'alarme et l'interruption.

Chapitre 3 : ...Méthode de développement d'un système temps réel embarqué

Un schéma LACATRE est un ensemble d'objets de LACATRE qui sont reliés par des primitives LACATRE, il y a deux type d'objets, configurables et objets programmables.

Les objets configurables de LACATRE sont ceux dont le comportement est parfaitement défini après avoir configuré leurs paramètres de fonctionnement, les paramètres de configuration permettent, par exemple, la définition du mode de gestion des files d'attente des boîtes aux lettres, les objets configurables de LACATRE sont le sémaphore, la boîte aux lettres, le message, l'évènement, la ressource.

Les objets configurables ne connaissent que deux états : inexistant (avant création et /ou après destruction) et existant (après création).

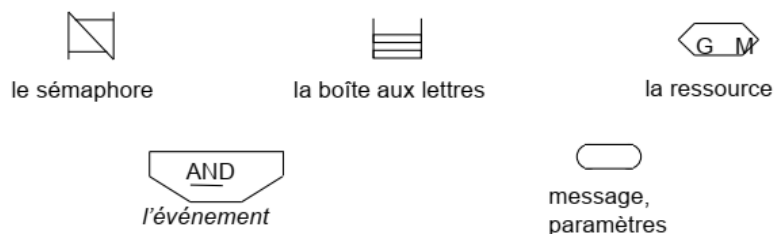


Figure 11 : symbole des objets configurable.

Les objets programmables ont un comportement qui doit être défini par le programmeur sous forme de séquence d'appel de primitives LACATRE.

LACATRE possède trois objets programmables, la tâche, la routine d'interruption et la routine d'alarme, ceux-ci possèdent deux états actif et inactif, l'état actif est l'état dans lequel l'objet programmable est en mesure d'exécuter des primitives LACATRE.



Figure 12 : symbole des objets programmable.

Les primitives LACATRE sont les relations que le programmeur d'application a sa disposition pour créer des liens entre les objets LACATRE.

Chapitre 3 : ...Méthode de développement d'un système temps réel embarqué

Au niveau de la formalisation, plusieurs études ont été réalisées pour valider la conception faite en LACATRE, ici l'idée est de coupler LACATRE avec une méthode de spécification outillée comme SART [17].

3.5 Les outils d'implémentation pour un système temps réel

Pour l'implémentation d'un système temps réel on utilise ce qu'on appelle exécutif temps réel.

3.5.1 Exécutif temps réel

Un exécutif temps réel (ETR) est en définitive, un noyau temps réel, qui gère les interruptions matérielles, crée les tâches et ou processus, gère la mémoire, se sert d'une interruption horloge pour faire l'ordonnancement des tâches et des processus, plus les pilotes de périphériques pour la gestion des entrées/sorties, protocoles réseaux.

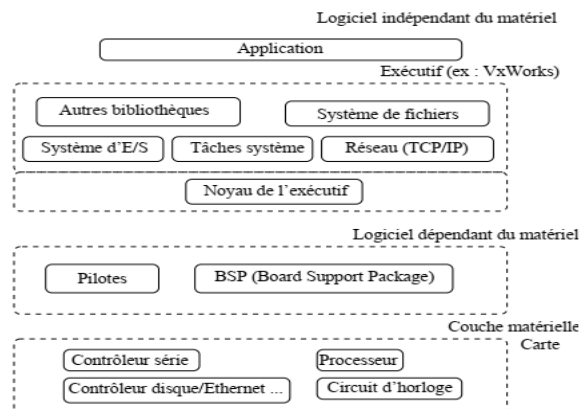


Figure 13 : exemple d'un exécutif temps réel Vx Works

Contrairement aux systèmes d'exploitations généralistes Unix, Linux, Microsoft Windows, etc. sont très consommateurs de ressources (processeur, mémoire, ...), les exécutifs temps réel comme FreeRTOS, Vx Works, POSIX, RTOS, etc. sont peu consommateurs de ressources et embarquables.

3.5.2 Les Services offerts par un exécutif temps réel

- Ordonnancement et tâches.
- Synchronisation : les sémaphores.
- Communication : Les interfaces série, file de messages, etc.
- Manipulation du temps.
- Gestion des interruptions.

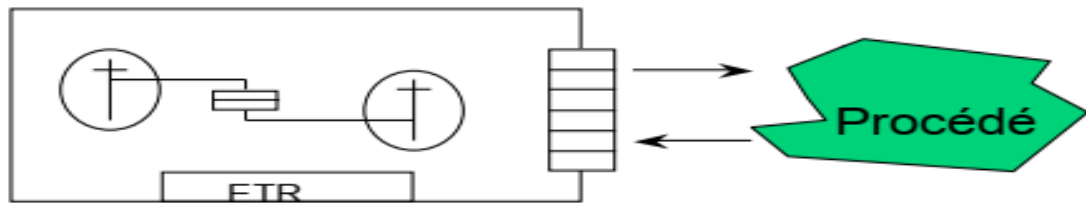


Figure 14 : représentation schématique d'un exécutif temps réel

3.6 Conclusion

Le langage SA-RT est très répandu et très utilisé pour spécifier les systèmes temps réel, à cause de sa simplicité d'utilisation et sa capacité d'expression, graphique en particulier, le SA-RT possède l'avantage d'être bien lisible car la spécification est faite d'une manière hiérarchique, cependant cette spécification est informelle et ne permet pas de vérifier directement le modèle.

LACATRE est un langage de conception et de programmation multitâche. Il est très proche de l'implémentation, et permet de générer du code exécutable pour plusieurs types de cibles FreeRTOS, Vx Works, etc. LACATRE fournit une boîte à outils de haut niveau pour le concepteur du système temps réel et permet de s'abstraire des détails d'implémentation.

Une lacune importante concerne la prise en compte des aspects temps réel et formels, les travaux menés s'appliquent à fixer une sémantique au langage pour autant, actuellement, peu de travaux sont en cours et peu d'outils formels disponibles.

Tous les systèmes d'exploitation utilisent des ordonnanceurs pour allouer du temps CPU aux tâches ou aux threads. Parce qu'un seul processus peut s'exécuter à un moment donné sur le processeur dans le cas d'un processeur à un seul cœur. Le microcontrôleur à faible coût est livré avec un seul processeur et FreeRTOS est spécialement conçu pour les microcontrôleurs à ressources et mémoire limitées. Par conséquent, l'ordonnanceur fournit le mécanisme pour décider quelle tâche doit être exécutée sur le processeur.

Chapitre 04 : Les plateformes matérielles et logicielles utilisées

4.1 Introduction

Pour la réalisation de notre projet on utilise l'exécutif temps réel FreeRTOS avec les carte ARDUINO, et exploiter la simplicité d'utilisation et la licence libre, ces plateformes ont été développé par des professionnels, strictement contrôlé de qualité, robuste, pris en charge, ne contient aucune ambiguïté de propriété intellectuelle et est vraiment gratuit à utiliser dans des applications commerciales sans aucune obligation d'exposer votre code source propriétaire.

4.2 L'exécutif temps réel FreeRTOS

FreeRTOS est un système d'exploitation en temps réel pour microcontrôleurs développé en partenariat avec les principaux fabricants de puces électronique au monde sur une période de 15 ans, et maintenant téléchargé toutes les 170 secondes, leader du marché pour les microcontrôleurs et les petits microprocesseurs, distribué gratuitement sous la licence open source , FreeRTOS comprend un noyau et un ensemble croissant de bibliothèques adaptées à une utilisation dans tous les secteurs industriels. FreeRTOS est construit en mettant l'accent sur la fiabilité et la facilité d'utilisation.

FreeRTOS est un RTOS assez limité par rapport aux grands RTOS commerciaux (tels que Vx Works, Windows CE, QNX, etc.) Nous avons choisi de travailler avec FreeRTOS car il fournit du code source, est relativement facile à comprendre et à utiliser, est gratuit pour un usage commercial avec des problèmes minimes, et largement utilisé, l'inconvénient majeur de son utilisation est le manque de prise en charge de fonctions plus complexes.

FreeRTOS est parfaitement adapté aux applications en temps réel profondément embarquées qui utilisent des microcontrôleurs ou de petits microprocesseurs.

FreeRTOS, comme n'importe quel autre noyau, est un outil purement logiciel, ce n'est qu'un système de fichiers. FreeRTOS nous propose une API de programmation pour gérer un environnement multitâche, vous trouverez ci-dessous le système de fichiers du noyau :

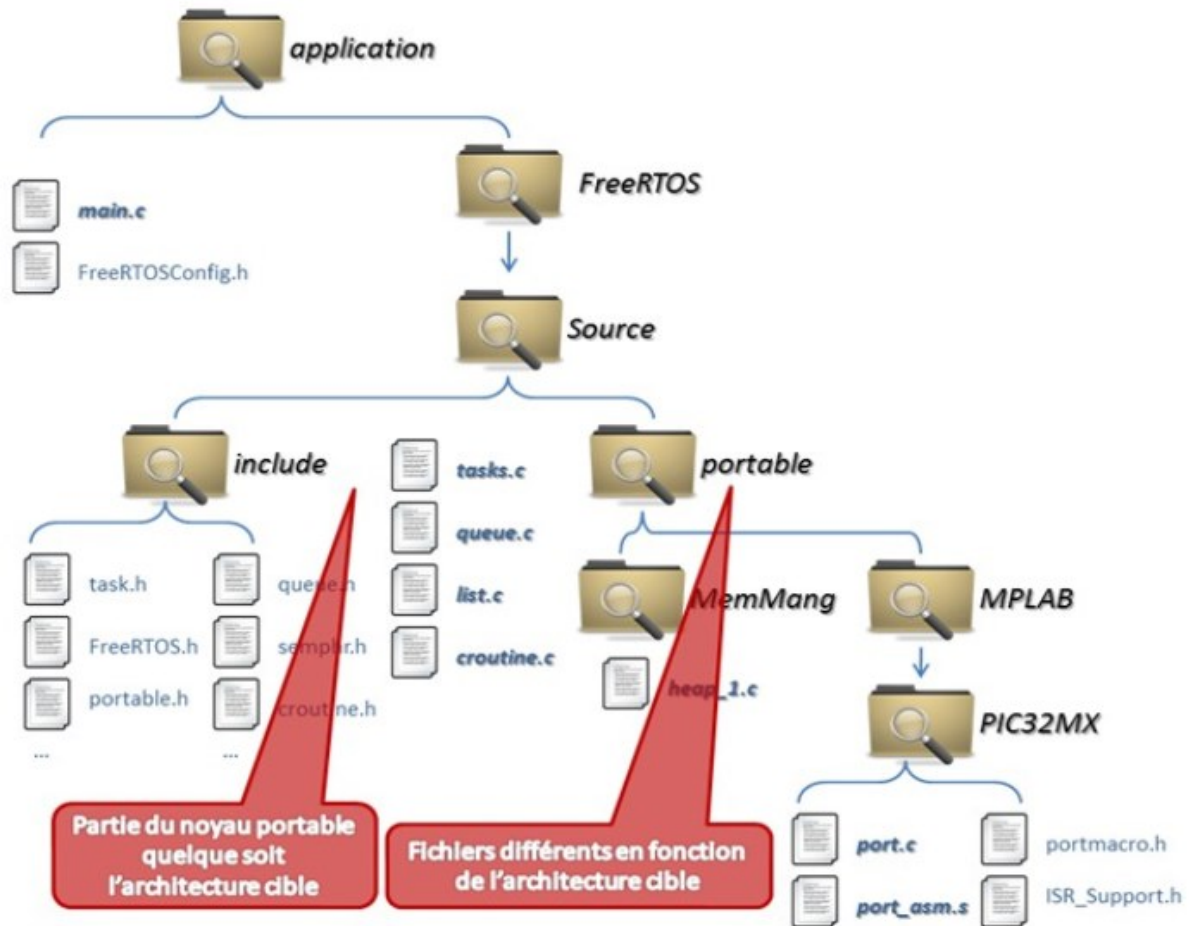


Figure 15 : le système de fichiers du noyau « FreeRTOS ».

En son cœur, FreeRTOS est un ensemble de bibliothèques C et en particulier un planificateur de tâches. À chaque « tick » (réglé à 15 ms sur l'Arduino), le planificateur lance une interruption et considère toutes les tâches « prêtes » à s'exécuter. Il exécute la tâche prête à la priorité la plus élevée. S'il existe une égalité pour la tâche de priorité la plus élevée prête à être exécutée, il utilise la planification à tour de rôle pour basculer entre les tâches de cette priorité. Le planificateur peut également « bloquer » les tâches et les faire supprimer de la liste prête jusqu'à ce qu'un événement se produise (des exemples d'événements incluent un sémaphore en cours de définition et un certain temps qui s'écoule). Il peut également « suspendre » des tâches et les rendre imprévisibles jusqu'à leur reprise explicite. Voir le diagramme ci-dessous pour une représentation visuelle.

- **Running**
 - Task is actually executing
- **Ready**
 - Task is ready to execute but a task of equal or higher priority is Running.
- **Blocked**
 - Task is waiting for some event.
 - **Time**: if a task calls `vTaskDelay()` it will block until the delay period has expired.
 - **Resource**: Tasks can also block waiting for queue and semaphore events.
- **Suspended**
 - Much like blocked, but not waiting for anything.
 - Tasks will only enter or exit the suspended state when explicitly commanded to do so through the `vTaskSuspend()` and `xTaskResume()` API calls respectively.

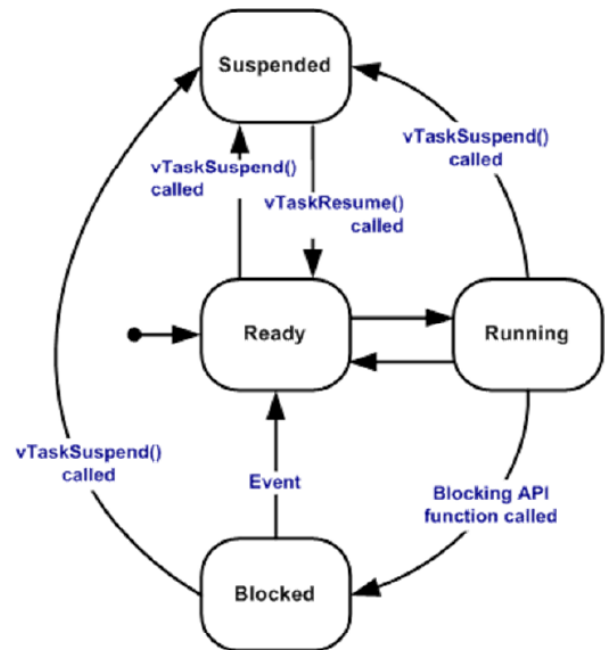


Figure 16 : les états du tache sure FreeRTOS [18].

4.2.1 Pourquoi FreeRTOS ?

Noyau de confiance : Avec une robustesse éprouvée, un faible encombrement et une large prise en charge des périphériques, le noyau FreeRTOS est considéré par les entreprises de renommée mondiale comme la norme de facto pour les microcontrôleurs et les petits microprocesseurs.

Accélérez la mise sur le marché : Grâce aux démos préconfigurées détaillées et aux intégrations de référence de l'Internet des objets, il n'est pas nécessaire de déterminer comment configurer un projet. Téléchargez, compilez et accédez instantanément au marché plus rapidement.

Prise en charge d'un large écosystème : Notre écosystème de partenaires offre un large éventail d'options, notamment des contributions de la communauté, un support professionnel, ainsi que des outils intégrés d'IDE et de productivité.

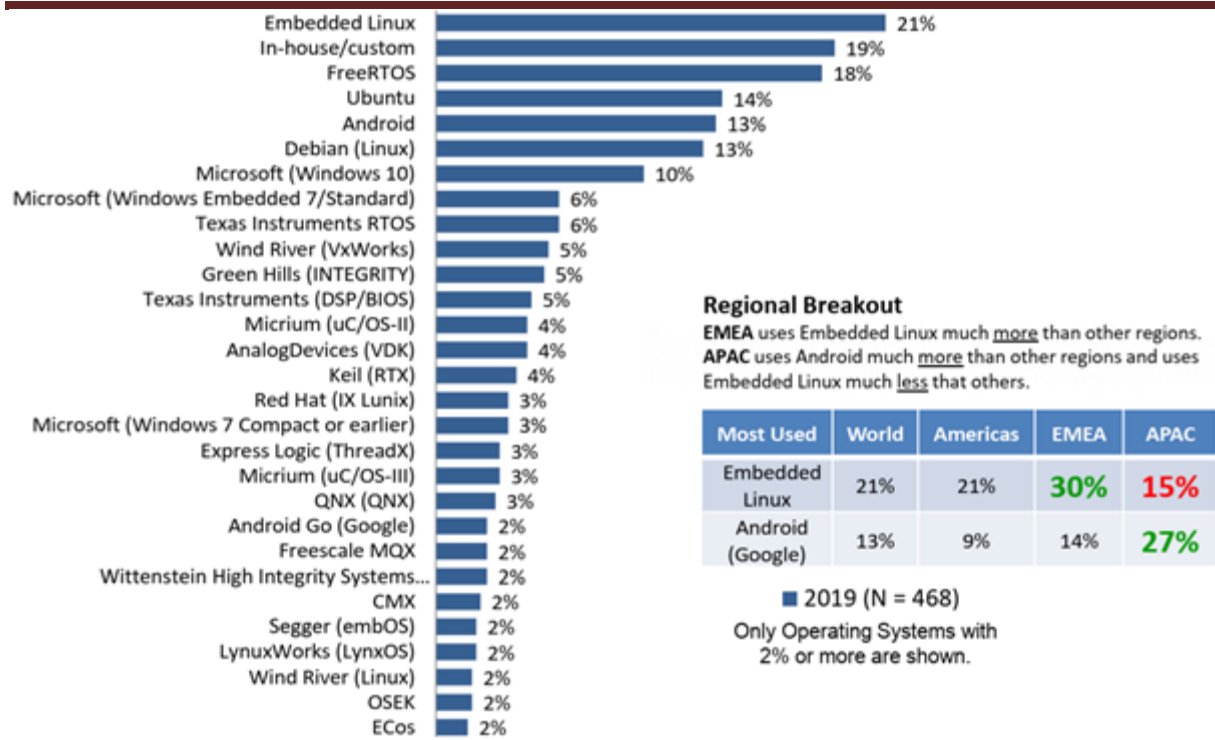


Figure 17 : Observation de marché des OS et RTOS pour l'embarqué [19].

Remarque :

EMEA, est une appellation que certaines entreprises utilisent pour désigner la région économique qui regroupe les pays d'Europe, du Moyen-Orient et de l'Afrique.

APAC est un ensemble géographique constitué de l'Extrême-Orient, du sous-continent indien et de l'Océanie.

4.2.2 Fonctionnalités de FreeRTOS :

- Opération préventive ou coopérative
- Affectation des priorités de tâches très flexible
- Mécanisme de notification des tâches flexible, rapide et léger
- Files d'attente
- Sémaphores
- Mutex
- Minuteurs
- Groupes d'événements
- Vérification du dépassement de la pile
- Enregistrement de trace

- Collecte des statistiques d'exécution des tâches
- Licence commerciale et assistance en option
- Modèle d'imbrication d'interruption complète (pour certaines architectures)
- Pile d'interruption gérée par logiciel le cas échéant (cela peut aider à économiser la RAM)

4.2.2 Algorithmes d'ordonnement de FreeRTOS :

Il existe un certain nombre d'algorithmes de planification assez standard qui sont utilisés dans les RTOS, l'algorithme idéal serait capable de : planifier tout ensemble de tâches pour lequel il existe un calendrier, échouer normalement lorsque les tâches ne sont pas planifiables et seraient simples à utiliser (ayant des priorités fixes ou "statiques" pour chaque tâche serait un bon début pour être simple...). Bien sûr, un tel algorithme n'existe pas, nous avons donc un nombre d'algorithmes différents que nous pouvons envisager, chacun avec son propre ensemble d'avantages et d'inconvénients.

FreeRTOS n'utilise que deux algorithmes d'ordonnement :

- « **Round-robin Scheduling** », Dans cet algorithme, toutes les tâches de priorité égale obtiennent le processeur en parts égales de temps processeur.
- « **Fixed Priority Preemptive Scheduling** », cet algorithme sélectionne les tâches en fonction de leur priorité. En d'autres termes, une tâche à haute priorité obtient toujours le processeur avant une tâche à faible priorité. Une tâche de faible priorité s'exécute uniquement lorsqu'il n'y a pas de tâche de haute priorité à l'état prêt.

Remarque : dans notre projet, nous utilisons l'ordonnement en mélangeant les deux algorithmes et il est connu sous le nom « Prioritized Preemptive Scheduling with Time Slicing ».

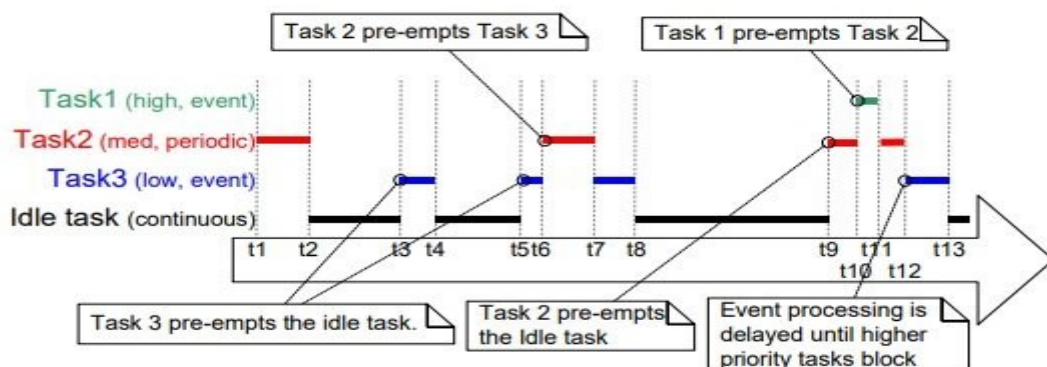


Figure 18: exemple de « Prioritized Preemptive Scheduling with Time Slicing ».

4.3 Les cartes ARDUINO

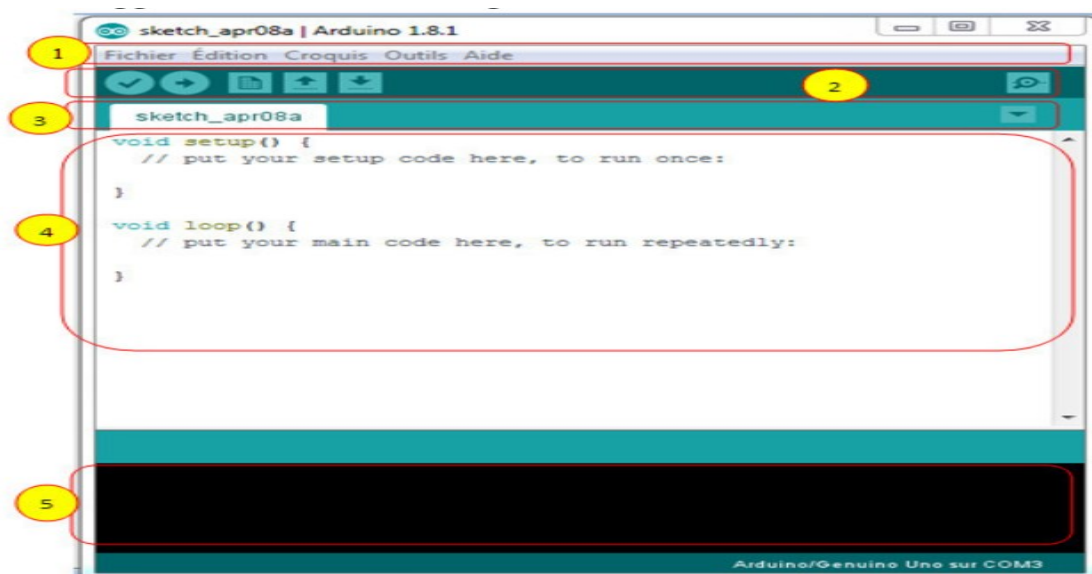
Arduino est une plate-forme de prototypage d'objets interactifs à usage créatif constituée d'une carte électronique et d'un environnement de programmation. Sans tout connaître ni tout comprendre de l'électronique, cet environnement matériel et logiciel permet à l'utilisateur de formuler ses projets par l'expérimentation directe avec l'aide de nombreuses ressources disponibles en ligne.

Le matériel : Il s'agit d'une carte électronique basée autour d'un microcontrôleur Atmega du fabricant Atmel, dont le prix est relativement bas pour l'étendue possible des applications.



Figure 19 :« Carte Arduino UNO ».

Le logiciel : Le logiciel va nous permettre de programmer la carte Arduino en langage C. Il nous offre une multitude de fonctionnalités.



*

Figure 20: « Arduino 1.8.1 IDE».

1. Un menu.
2. Une barre d'actions.
3. Un ou plusieurs onglets correspondant aux sketches.
4. Une fenêtre de programmation.
5. Une console qui affiche les informations et erreurs de compilation et de Téléversement du programme.






	Bouton « Verify » (Vérifier) ; il permet de compiler votre programme et de vérifier si des erreurs s'y trouvent. Cette procédure prend un certain temps d'exécution et lorsque est terminée, elle affiche un message de type « Binary sketch size : ... » indiquant la taille du sketch téléversé.
	Pour transmettre le sketch compilé avec succès sur la carte Arduino dans le microcontrôleur.
	Bouton « New » (Nouveau) ; ce bouton permet de créer un nouveau sketch.
	Bouton « Open » (Ouvrir) ; il fait apparaître un menu qui permet d'ouvrir un sketch qui figure dans votre dossier de travail ou des exemples de sketches intégrés au logiciel.
	Bouton « Save » (Sauvegarder) ; il permet de sauvegarder votre sketch.

Tableau 03 : raccourcies de barre d'actions.

Le système Arduino nous permet de réaliser un grand nombre des systèmes électronique complexes, qui ont des applications dans tous les domaines, nous pouvons donner quelques exemples :

- Contrôler et/ou télécommander les appareils domestiques
- Contrôler et/ou télécommander les machines industrielles
- Communiquer avec des capteurs et / ou un ordinateur
- Contrôler et/ou télécommander des appareils mobiles (modélisme) etc.
- Fabriquer votre propre robot.

Avec Arduino, nous allons faire des systèmes électroniques tels qu'un système de contrôle électronique, un système de lubrification, etc. Tous ces systèmes seront conçus basé sur une carte Arduino, des capteurs et d'autres composants électroniques.

4.3.1 Pourquoi choisir Arduino ?

Il existe pourtant dans le commerce, une multitude de plateformes qui permettent de faire la même chose, notamment les microcontrôleurs « PIC » du fabricant Micro chip, nous allons voir pour quoi choisir l'Arduino.

a. Le prix En vue des performances qu'elles offrent, les cartes Arduino sont relativement peu coûteuses, ce qui est un critère majeur pour le débutant.

b. La liberté C'est un bien grand mot, mais elle définit de façon assez concise l'esprit de l'Arduino, elle constitue en elle-même deux choses Le logiciel : gratuit et open source, développé en c, dont la simplicité d'utilisation relève du savoir cliquer sur la souris. Le matériel : cartes électroniques dont les schémas sont en libre circulation sur internet. Cette liberté a une condition : le nom « Arduino » ne doit être employé que pour les cartes « officielles », en somme, vous ne pouvez pas fabriquer votre propre carte sur le modèle Arduino et lui assigner le nom « Arduino».

Les cartes non officielles, on peut les trouver et les acheter sur Internet et sont pour la quasi-totalité compatibles avec les cartes officielles Arduino.

c. La compatibilité Le logiciel, tout comme la carte, est compatible sous les plateformes les plus courantes (Windows, Linux et Mac), contrairement aux autres outils de programmation du commerce qui ne sont, en général, compatibles qu'avec Windows.

d. La communauté La communauté Arduino est impressionnante et le nombre de ressources à son sujet est en constante évolution sur internet. De plus, on trouve les références du langage Arduino ainsi qu'une page complète de tutoriels sur le site arduino.cc.

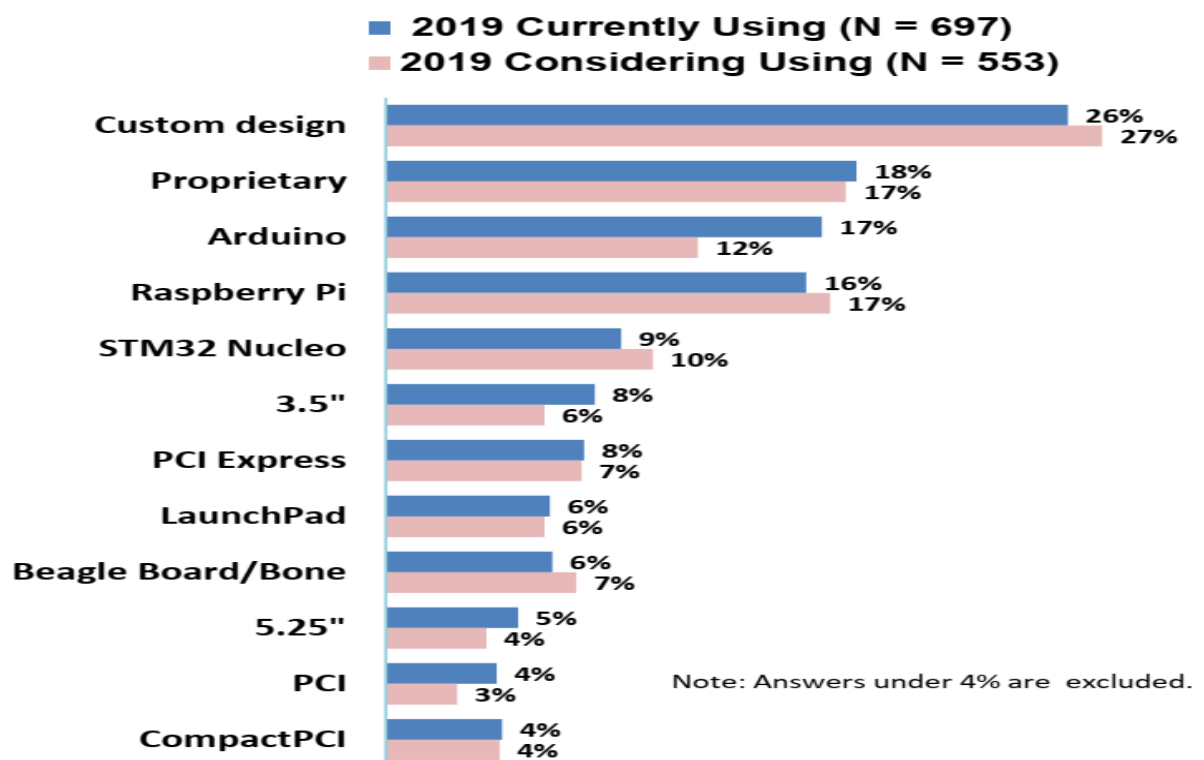


Figure 21 : Observation de marché des cartes pour les systèmes embarqués [19].

4.3.2 Types de cartes

Il y a trois types de cartes :

- Les cartes dites « officielles » qui sont fabriquées en Italie par le fabricant officiel : Smart Projects.
- Les cartes dites « compatibles » qui n'ait pas fabriqués par Smart Projects, mais qui sont totalement compatibles avec les Arduino officielles.
- Les « autres » fabriquées par diverse entreprise et commercialisées sous un nom différent (Freeduino, Seeduino, Femtoduino, ...).

4.3.3 Différentes cartes

Des cartes Arduino il en existe beaucoup, on représenter les plus connues dans l'image suivante:



Figure 22 :« les différents modèles des carte ARDUINO ».

4.4 Conclusion

Comme pont tendu entre le monde réel et le monde numérique, Arduino permet d'étendre les capacités de relations humain/machine ou environnement/machine. Arduino est un projet en source ouverte : la communauté importante d'utilisateurs et de concepteurs permet à chacun de trouver les réponses à ses questions.

FreeRTOS est un environnement exécutif à très faible empreinte mémoire quelque KB fournissant un ordonnanceur et les mécanismes associés pour le partage de ressources entre tâches, en utilisant une bibliothèque libre et compatible avec plusieurs architectures.

Chapitre 05 : Étude de cas

5.1 Introduction

Dans le cadre de Conception et Optimisation d'un noyau de calcul pour traitement temps réel d'équipements de mesure, on prendra comme cas d'étude la procédure de tournage sur la machine de tour.

Le tournage mécanique est un procédé d'usinage par enlèvement de matière qui consiste en l'obtention de pièces de forme cylindrique à l'aide d'outils coupants sur des machines appelées tour, la pièce à usiner est fixée dans une pince, dans un mandrin, ou entre pointes, Il est également possible de percer sur un tour, le mouvement de coupe est obtenu par rotation de la pièce serrée entre les mors d'un mandrin, alors que le mouvement d'avance est obtenu par le déplacement de l'outil coupant, l'outil est serré dans le porte outil et reçoit les mouvements rectilignes longitudinal ou transversal, la combinaison des mouvements de la pièce et de l'outil permet de réaliser des surfaces variées : cylindriques, coniques, sphériques, hélicoïdales....etc. qui peuvent être extérieures ou intérieures.



Figure 23 : Principe de l'usinage.

Toutes fonctionnalités de la machine tour se fait manuellement, notre objective c'est, le rendre semi-automatique avec l'aide de capteurs, microcontrôleurs et un système de contrôle simple.

Pour modéliser le système de contrôle de la machine industrielle tour on utilise une spécification informelle et une spécification formelle selon la méthode SA-RT avec une approche incrémentale.

5.2 Spécification informelle

La spécification informelle est un type de cahier de charge pour exprimer les besoins de fonctionnement de ce système en présentant une description de l'architecture matérielle du système et une description des fonctions du système et les paramètres utilisés.

5.2.1 Aspect matériel :

Malgré la diversité des conceptions, tous les tours présentent beaucoup d'ensembles et d'éléments similaires.

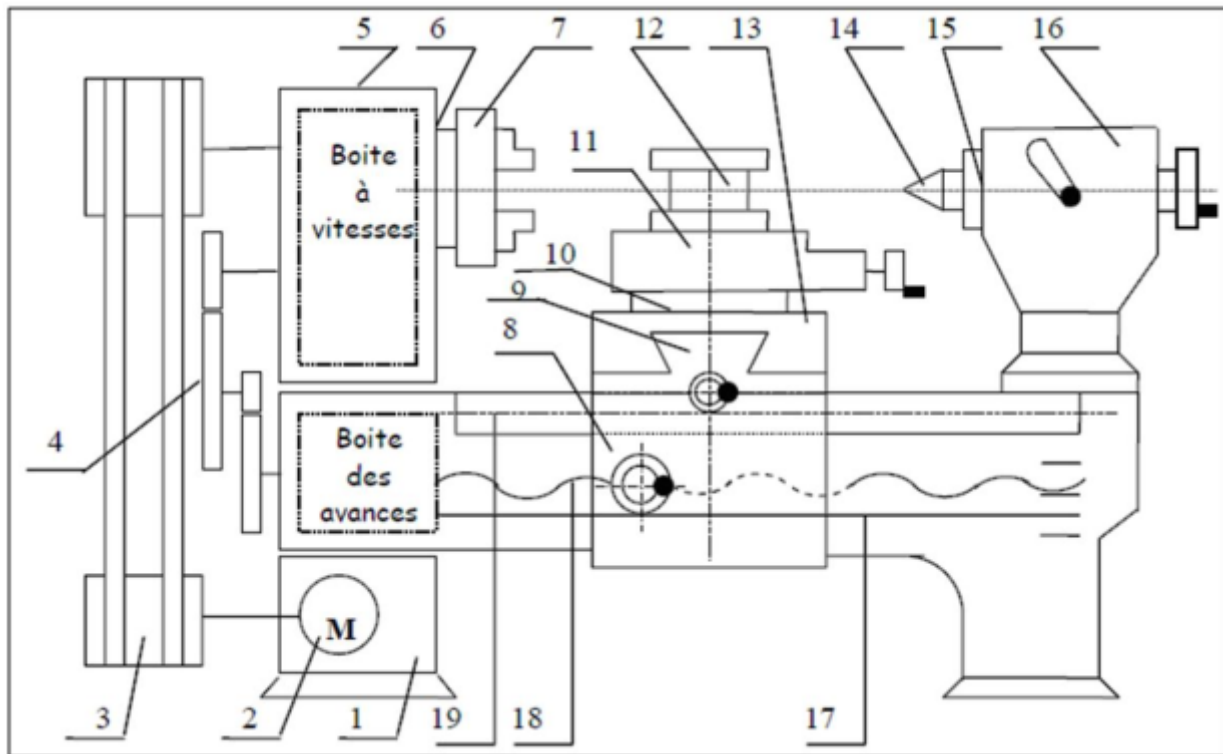


Figure 24 : les organes principaux de la machine tour.

1 : Bâti, 2 : Moteur, 3 : Transmission par courroie, 4 : Inverseur tête de cheval, 5 : Poupée fixe, 6 : Broche, 7 : Mandrin, 8 : Tablier, 9 : Traînard, 10 : Chariot intermédiaire, 11 : Chariot supérieur, 12 : Porte-outil (touple), 13 : Chariot inférieur, 14 : Contre pointe, 15 : Fourreau, 16 : Poupée mobile, 17 : Barre de commande (barre de chariotage), 18 : Vis de commande (vis mère, vis de filetage), 19 : crémaillère. [20]

5.2.2 Aspect fonctionnel :

La réalisation de tout type de forme possible est une combinaison des trois mouvements suivant, un mouvement de rotation (mouvement de coupe) transmis par la broche, l'outil peut se déplacer en translation suivant deux directions(mouvement d'avance), ces deux directions, perpendiculaires entre elles, appartiennent à un plan auquel l'axe de la broche est parallèle, le premier mouvement de translation est parallèle à l'axe de la broche, le deuxième mouvement de translation est perpendiculaire à l'axe de la broche [21].

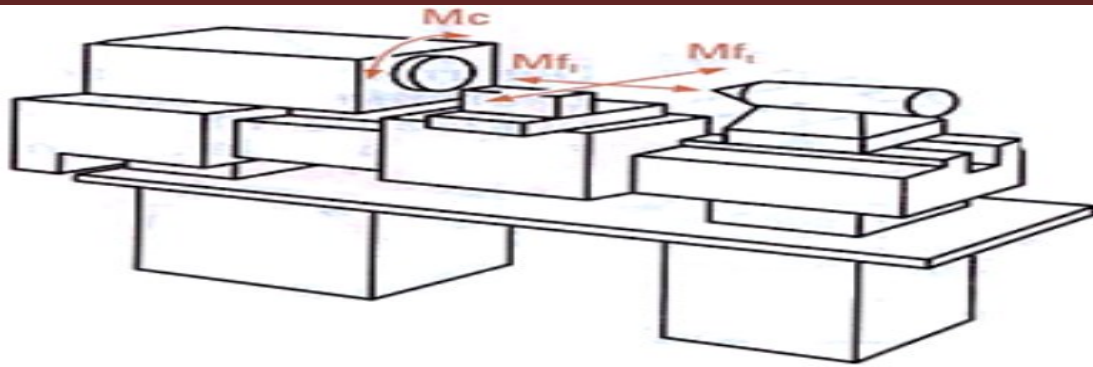


Figure 25 : Schéma des mouvements d'un tour parallèle.

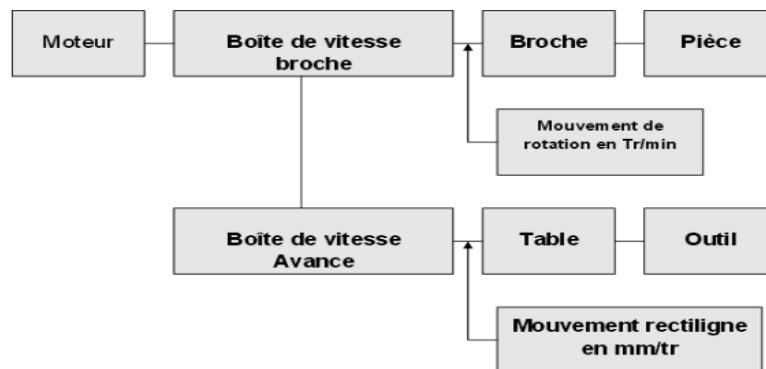


Figure 26 : Chaîne cinématique d'un tour.

5.2.3 Les Paramètres de coupe [21] :

1-Vitesse de coupe : est la vitesse relative de l'outil par rapport à la pièce. Il s'agit donc de la vitesse tangentielle au point de la pièce coïncidente avec la pointe de l'outil. Cette vitesse qui s'exprime toujours en mètres par minute (m/min) se calcule ainsi :

$$V_c = \frac{\pi \times d \times N}{1000}$$

Avec :

V_c : vitesse de coupe en m/min.

d : diamètre en mm au point d'usinage.

N : vitesse de rotation de la pièce en tours par minute.

2-Vitesse de rotation : en permutant les termes de la formule précédente, on obtient la vitesse de rotation N que l'on règle sur la machine leur valeur est le plus souvent issue de méthodes empiriques.

$$N = \frac{1000 \times V_c}{\pi \times d}$$

3-Vitesse d'avance : en tournage, l'avance est la vitesse avec laquelle progresse l'outil suivant l'axe de rotation pendant une révolution de la pièce, cette vitesse est déterminée expérimentalement en fonction des critères précédemment cités, cela correspond, en première approximation à l'épaisseur du copeau, on règle l'avance directement sur la machine, pour calculer la vitesse d'avance de l'outil, on applique cette formule :

$$Vf = f \times N$$

Avec :

f : L'avance en millimètre/tour.

N : fréquence de rotation réglée sur la machine en tour/minute.

4-Puissance nécessaire : un tour ne dispose pas d'une puissance illimitée, celle-ci lui est fournie par voie électrique, lors de l'usinage d'une pièce, il est impératif de s'assurer que la machine est capable de réaliser l'opération demandée, sans quoi on risque d'abîmer, l'outil, le tour, ou la pièce à usiner, on définit pour cela le débit de tournage, quantité de matière enlevée par l'outil dans un temps donné, ce débit Q_T s'exprime comme :

$$Q_T = f \times a_p \times V_c$$

En cm^3/min avec :

f : L'avance en millimètre/tour

a_p : La profondeur de passe en millimètre

V_c : La vitesse de coupe en mètre /minute

Grâce à ce débit, on exprime la puissance nécessaire à la passe demandée :

$$P = \frac{k_c \times Q_T}{60} = \frac{k_c \times a_p \times f \times V_c}{60}$$

En watts avec :

k_c : La pression de coupe, ou effort spécifique de coupe, en newtons par millimètre carré.

5.2.4 Remarque : La vitesse de coupe est déterminée en fonction de différents facteurs :

- La matière à usiner : en général plus elle est tendre et plus la vitesse est élevée
- La matière de l'outil de coupe
- La géométrie de l'outil de coupe
- Le type d'usinage : ébauche, finition, filetage, etc.
- Le lubrifiant, qui permet une augmentation de la vitesse
- La qualité du tour : plus il est rigide, plus il supportera des vitesses élevées

La puissance de fonctionnement, et la puissance max sont données par le constructeur du tour, l'ajustement de la profondeur de passe permettra de diminuer la puissance nécessaire à l'étape demandée, sans jouer sur la vitesse de coupe ni sur l'avance, critères primordiaux à la bonne réalisation de la pièce.

5.3 Spécification formelle

La spécification formelle ait faite selon la méthode d'analyse fonctionnelle et opérationnelle des applications temps réel SA-RT, cette méthode permet de réaliser une description graphique et textuelle de l'application en termes de besoins selon un modèle incrémental.

La SA-RT considère la spécification des systèmes temps réel selon trois points de vue orthogonaux :

- Fonctionnel
- Informationnel
- Dynamique

5.3.1 Noyau :

On va commencer avec un petit noyau simple, en suit-on vas ajouter une seule fonctionnalité à la fois.

5.3.1.1 Spécification :

Aspect fonctionnel : La décomposition de système de contrôle de la machine industriel tour en une hiérarchie de fonction selon la méthode SA-RT est représentée par le diagramme de contexte suivant.

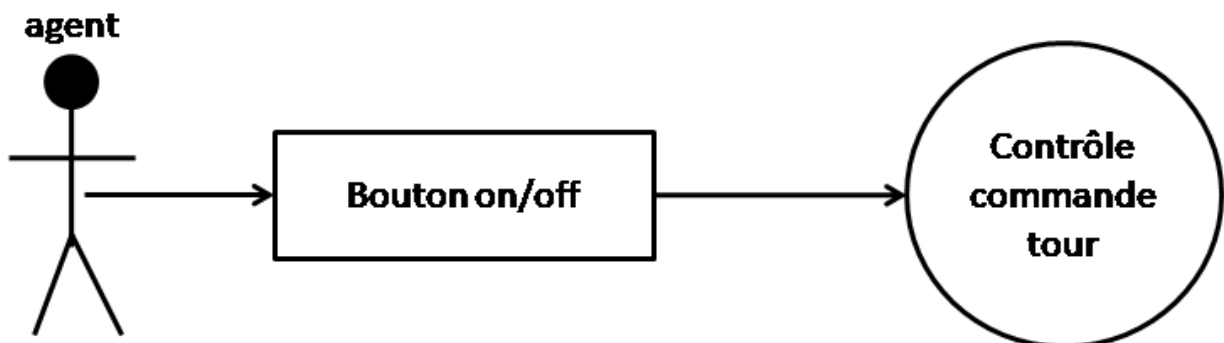


Figure 27 : diagramme de contexte.

Aspect informationnel : Le cheminement de flot de données qui transporte les valeurs d'une certaine information qui manipuler par les fonctions est représentée par le diagramme de flot de données suivant.

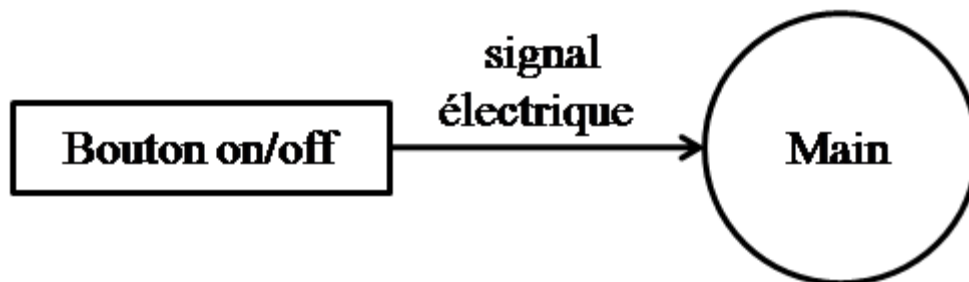


Figure 28 : diagramme de flot de données.

Aspect dynamique : Le contrôle du flux des donnée et l'activation des fonctions est exprimer par le flot de contrôle qui transporte les événements qui conditionnent directement ou indirectement l'exécution des processus de transformation de données, selon la méthode SA-RT est représenter par le diagramme de flot de contrôle. Les événements sont généralement liés à l'activation ou à la désactivation :

- **E** pour activation « Enable ».
- **D** pour désactivation « Disable ».
- **T** pour déclenchement « Trigger ».

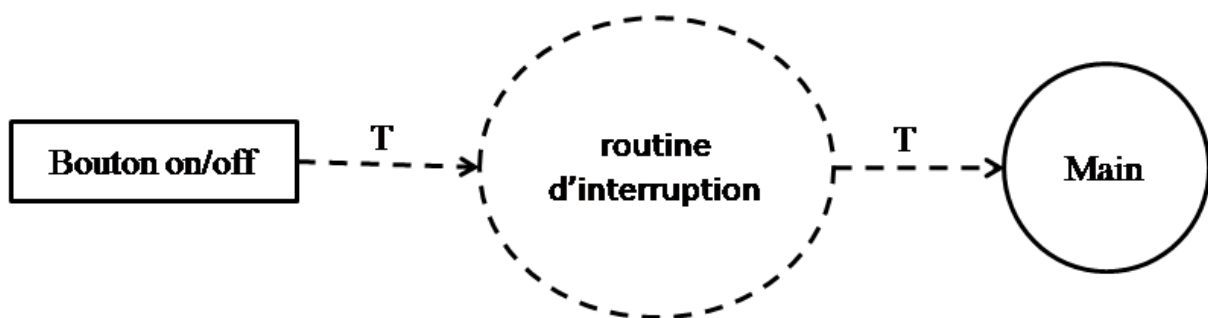


Figure 29 : diagramme de flot de contrôle.

5.3.1.2 Conception

On utilise la méthode de conception multitâche « LACATRE » pour permettre le passage d'un modèle de spécification base sure un analyse fonctionnel et structuré qui été réalisée avec la méthode SA-RT aux des entités de programmes dans un langage exécutable.

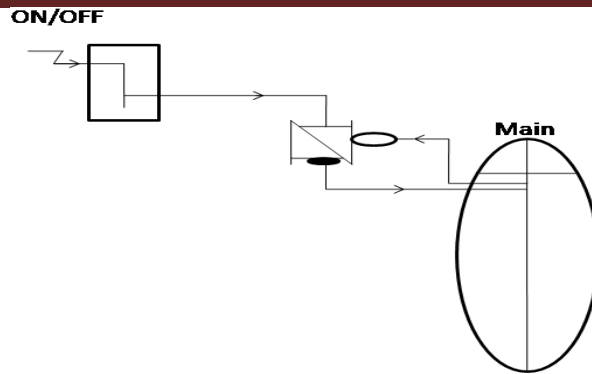


Figure 30 : schéma multitâche.

5.3.1.3 Test

Le test se fait en assemblant les composants et en programmant la carte Arduino, les images suivantes montrent comment cela a été fait.

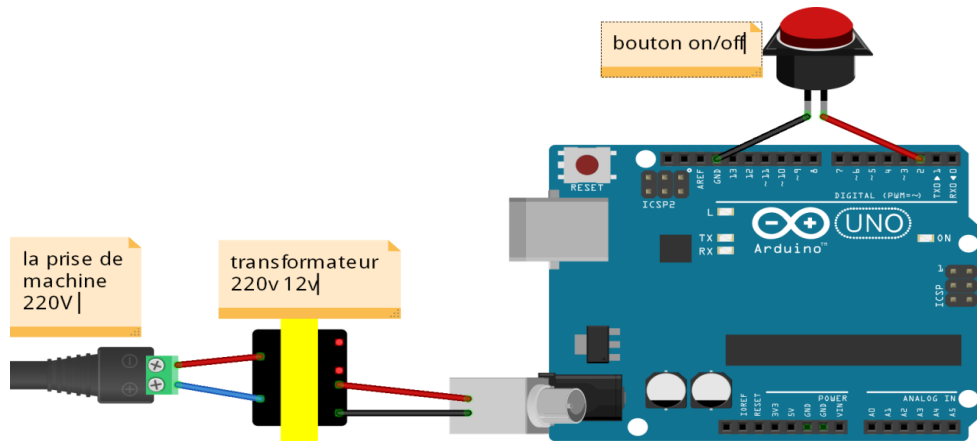


Figure 31 : schéma de câblage.

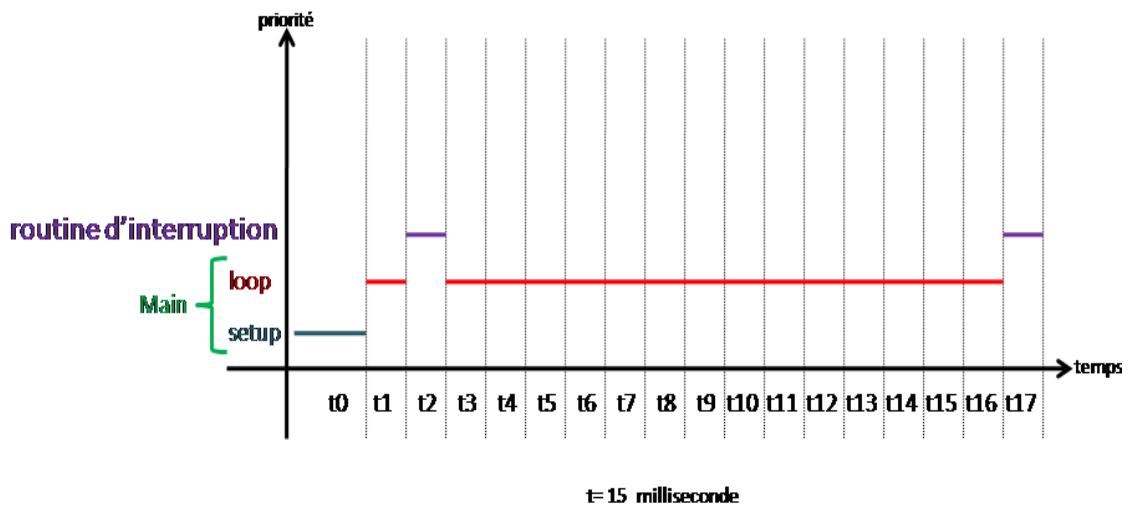


Figure 32 : le chronogramme de la séquence d'exécution.

5.3.2 Transaction du noyau à l'incrément 01 :

On va ajouter un afficheur pour afficher l'état de la machine, et un boîtier à lettre pour la communication entre les tâches.

5.3.2.1 Les nouvelles fonctionnalités :

01- Afficher :

Pour afficher la température, avec les paramètres suivant :

Input : une chaîne de caractères.

Output : un signal numérique.

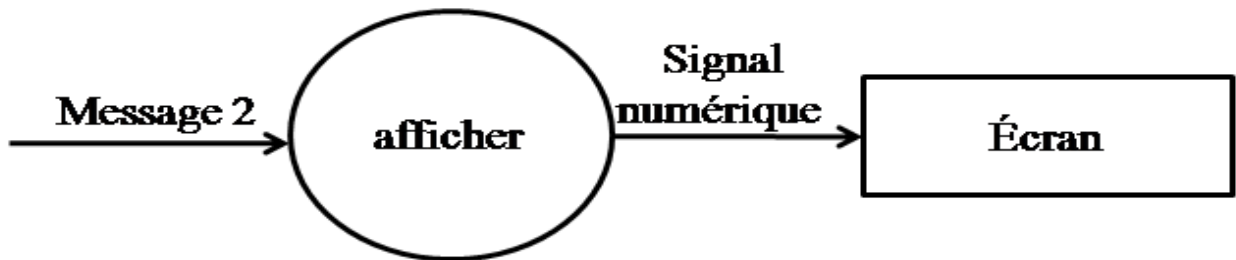


Figure 33 : la tâche afficher.

02- Boîte au lettre :

Pour stocker les données a utilisé ultérieurement par les tâches avec les paramètres suivant.

Input : une chaîne de caractères.

Output : une chaîne de caractères.

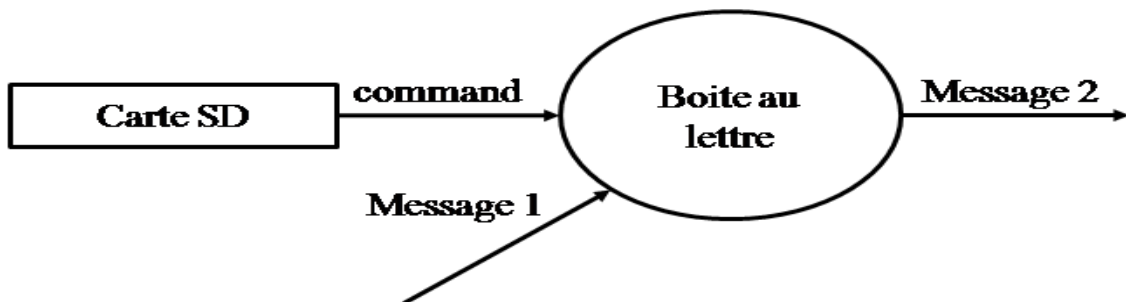


Figure 34 : la boîte au lettre.

5.3.3 Incrément 01 :

Dans cet incrément on va ajouter un lecteur de carte sd pour faciliter l'entrée des commandes à exécuter sur la machine

5.3.3.1 Spécification :

Aspect fonctionnel : La décomposition de système de contrôle de la machine industriel tour en une hiérarchie de fonction selon la méthode SA-RT est représentée par le diagramme de contexte suivant.

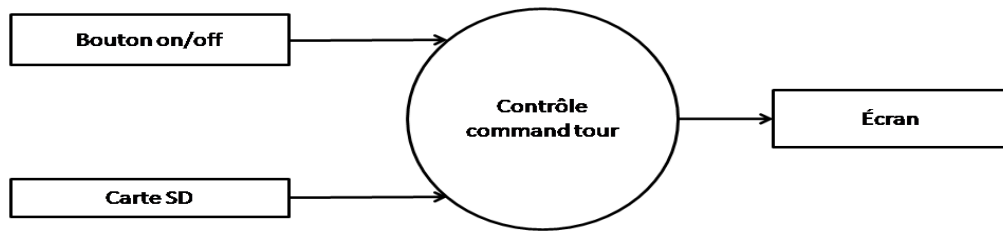


Figure 35 : diagramme de contexte.

Aspect informationnel : Le cheminement de flot de données qui transporte les valeurs d’une certaine information qui manipuler par les fonctions est représentée par le diagramme de flot de données suivant.

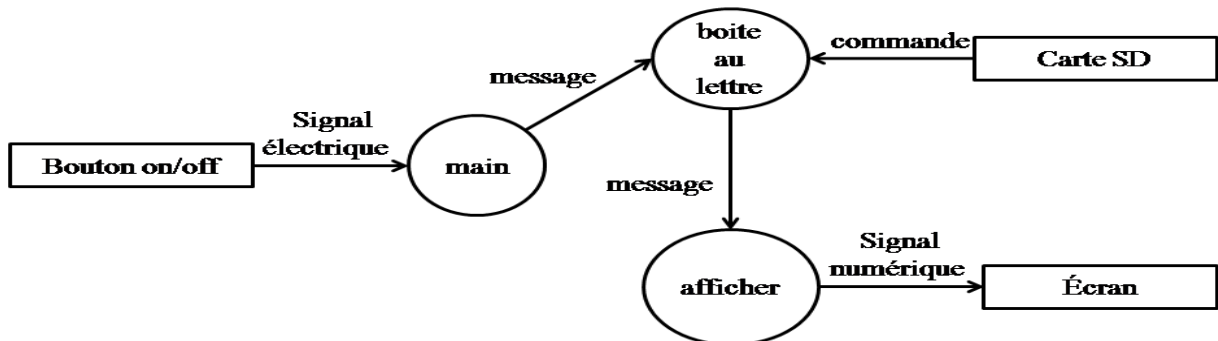


Figure 36 : diagramme de flot de données.

Aspect dynamique : Le contrôle du flux des donnée et l’activation des fonctions est exprimer par le flot de contrôle qui transporte les événements qui conditionnent directement ou indirectement l’exécution des processus de transformation de données, selon la méthode SA-RT est représenter par le diagramme de flot de contrôle.

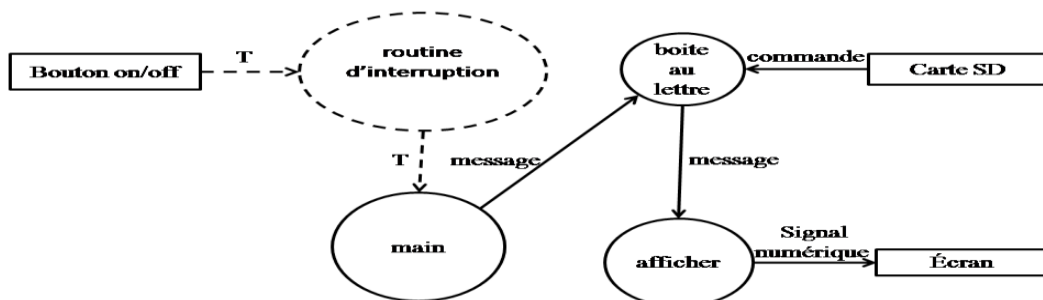


Figure 37 : diagramme de flot de contrôle.

5.3.3.2 Conception

On utilise la méthode de conception multitâche « LACATRE » pour permettre le passage d'un modèle de spécification base sure un analyse fonctionnel et structuré qui été réalisée avec la méthode SA-RT aux des entités de programmes dans un langage exécutable.

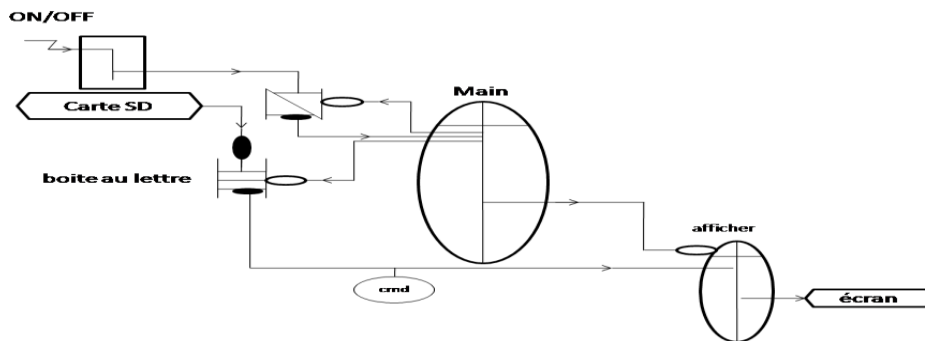


Figure 38 : schéma multitâche.

5.3.3.3 Test

Le test se fait en assemblant les composants et en programmant la carte Arduino, les images suivantes montrent comment cela a été fait.

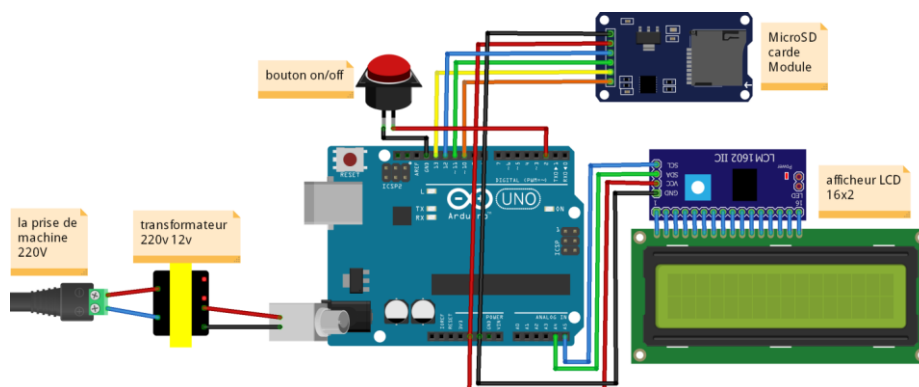


Figure 39 : schéma de câblage.

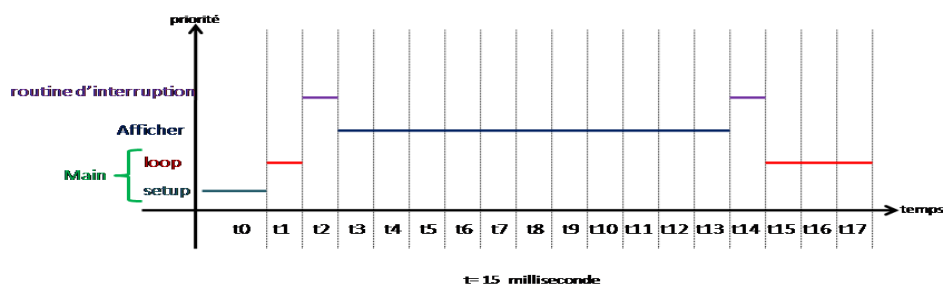


Figure 40 : le chronogramme de la séquence d'exécution.

5.3.3 Transaction du l'incrément 01à l'incrément 02

5.3.3.1 Les nouvelles fonctionnalités

01- Calculer température :

Pour contrôler le capteur de température.

Input : un signal analogique.

Output : une valeur numérique de température en Celsius.

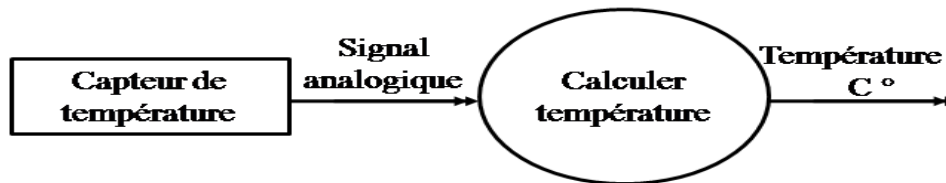


Figure 41 : la tâche calculer température.

5.3.4 Incrément 02

5.3.4.1 Spécification :

Aspect fonctionnel : La décomposition de système de contrôle de la machine industriel tour en une hiérarchie de fonction selon la méthode SA-RT est représentée par le diagramme de contexte suivant.

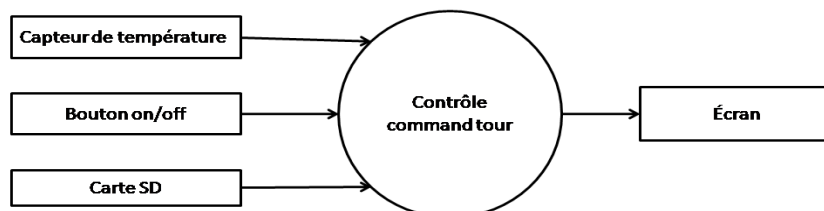


Figure 42 : digramme de contexte.

Aspect informationnel : Le cheminement de flot de données qui transporte les valeurs d'une certaine information qui manipuler par les fonctions est représentée par le diagramme de flot de données suivant.

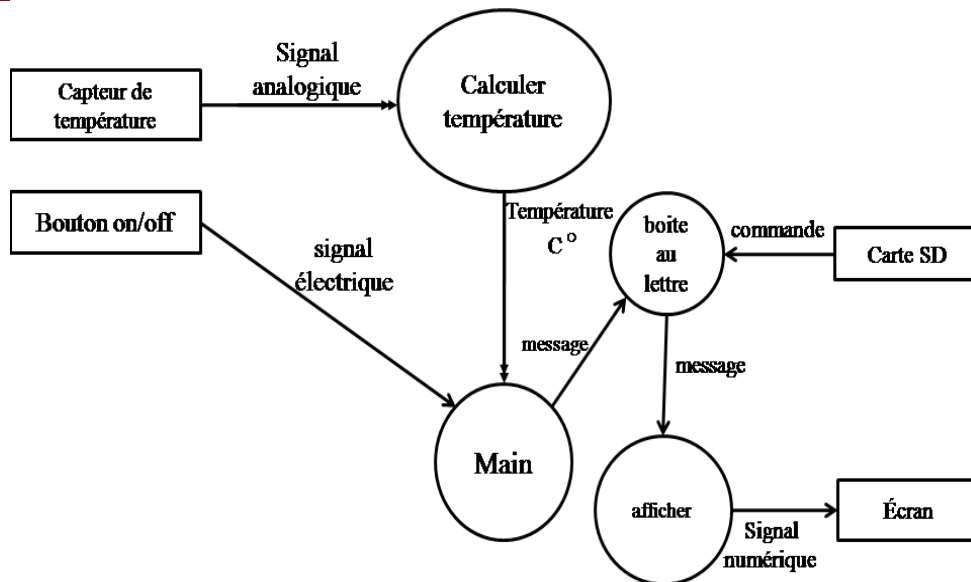


Figure 43 : digramme de flot de données.

Aspect dynamique : Le contrôle du flux des donnée et l’activation des fonctions est exprimer par le flot de contrôle qui transporte les événements qui conditionnent directement ou indirectement l’exécution des processus de transformation de données, selon la méthode SA-RT est représenter par le diagramme de flot de contrôle.

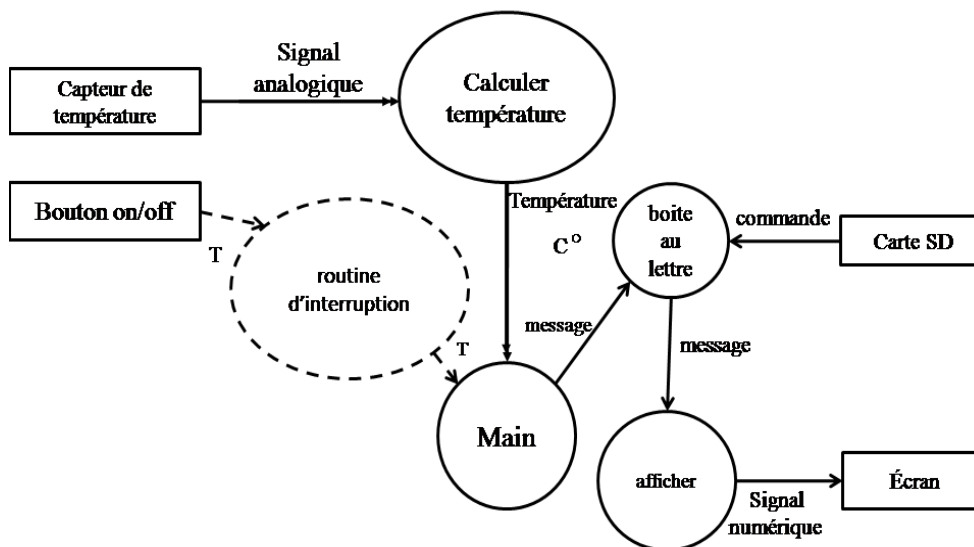


Figure 44 : diagramme de flot de contrôle.

5.3.4.2 Conception

On utilise la méthode de conception multitâche « LACATRE » pour permettre le passage d’un modèle de spécification base sure un analyse fonctionnel et structuré qui été réalisée avec la méthode SA-RT aux des entités de programmes dans un langage exécutable.

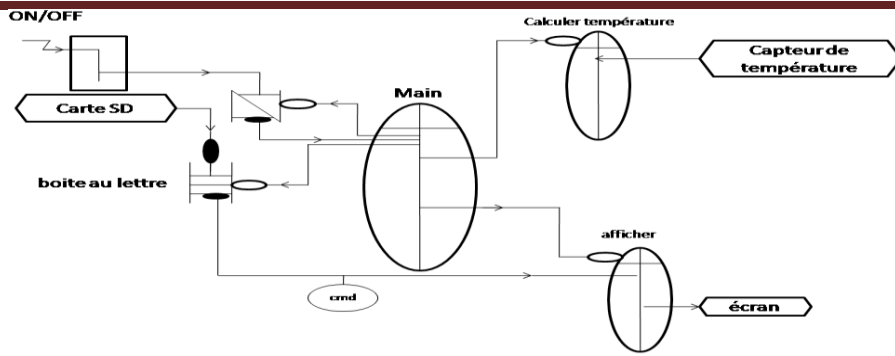


Figure 45 : schéma multitâche.

5.3.4.3 Test

Le test se fait en assemblant les composants et en programmant la carte Arduino, les images suivantes montrent comment cela a été fait.

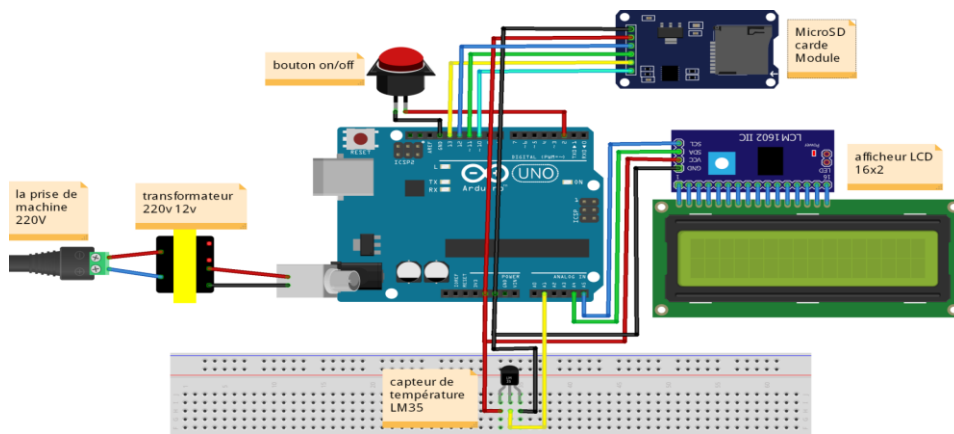


Figure 46 : schéma de câblage.

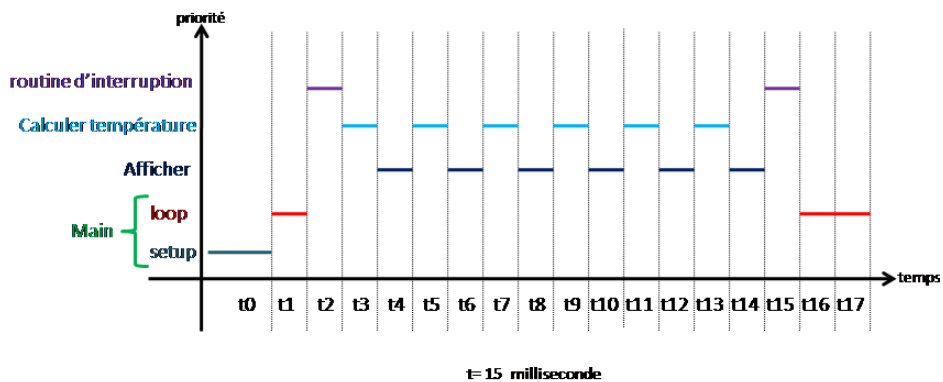


Figure 47 : le chronogramme de la séquence d'exécution.

5.3.5 Transaction du l'incrément 02 à l'incrément 03

5.3.5.1 Les nouvelles fonctionnalités

01- lubrificateur :

Pour contrôler la pompe de lubrification.

Input : une valeur numérique de température en Celsius.

Output : un signal électrique.

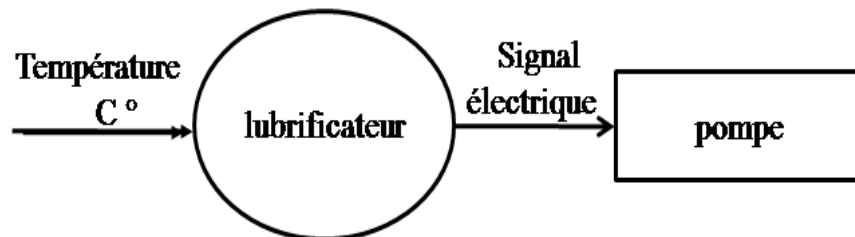


Figure 48 : la tâche lubrificateur.

5.3.5.2 Système de lubrification

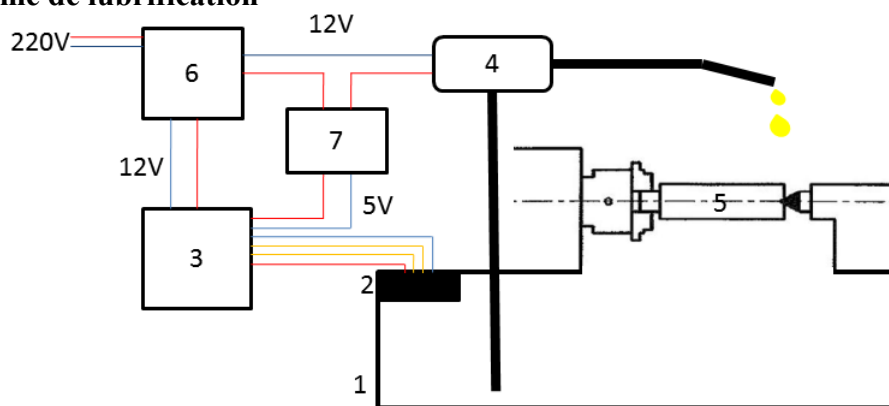


Figure 49 : un système de lubrification basic.

1. Réservoir
2. Capteur de niveau ultrason
3. Carte Arduino
4. Pompe 12V
5. La pièce
6. Le transformateur 220V vers 12V
7. Relais électromécanique

5.3.6 Incrément 03

5.3.6.1 Spécification :

Aspect fonctionnel : La décomposition de système de contrôle de la machine industriel tour en une hiérarchie de fonction selon la méthode SA-RT est représentée par le diagramme de contexte suivant.

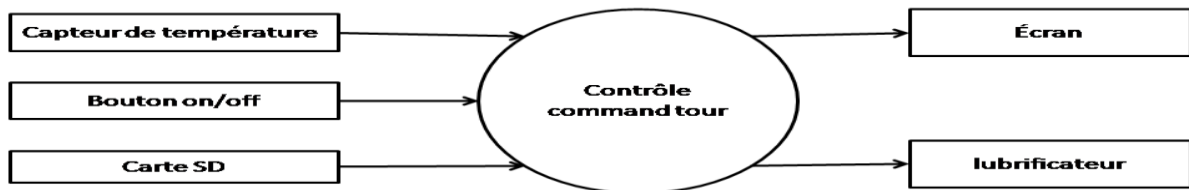


Figure 50 : diagramme de contexte.

Aspect informationnel : Le cheminement de flot de données qui transporte les valeurs d'une certaine information qui manipuler par les fonctions est représentée par le diagramme de flot de données suivant.

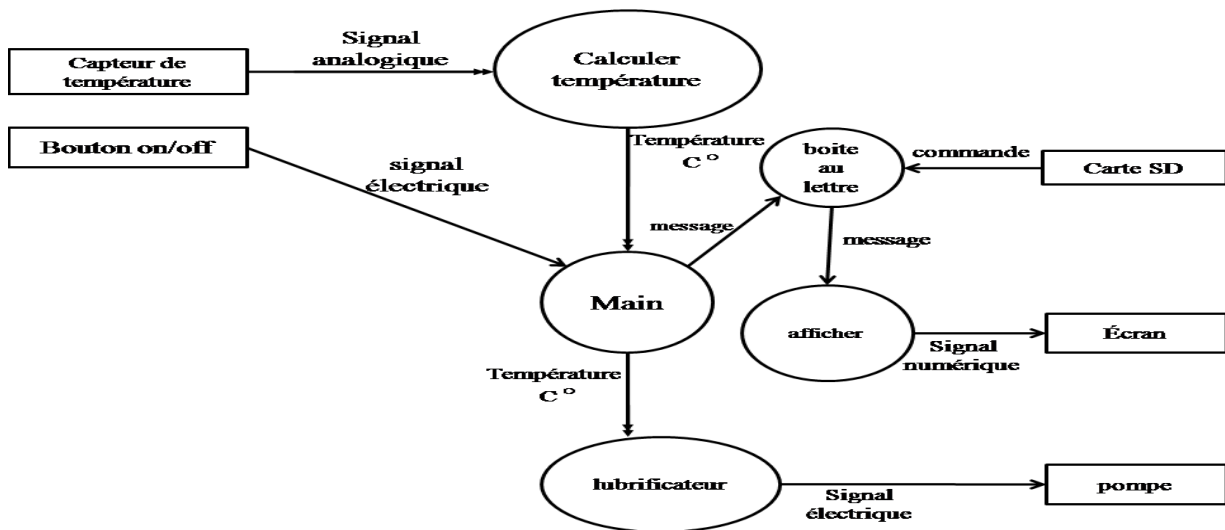


Figure 51 : diagramme de flot de données.

Aspect dynamique : Le contrôle du flux des donnée et l'activation des fonctions est exprimer par le flot de contrôle qui transporte les événements qui conditionnent directement ou indirectement l'exécution des processus de transformation de données, selon la méthode SA-RT est représenter par le diagramme de flot de contrôle.

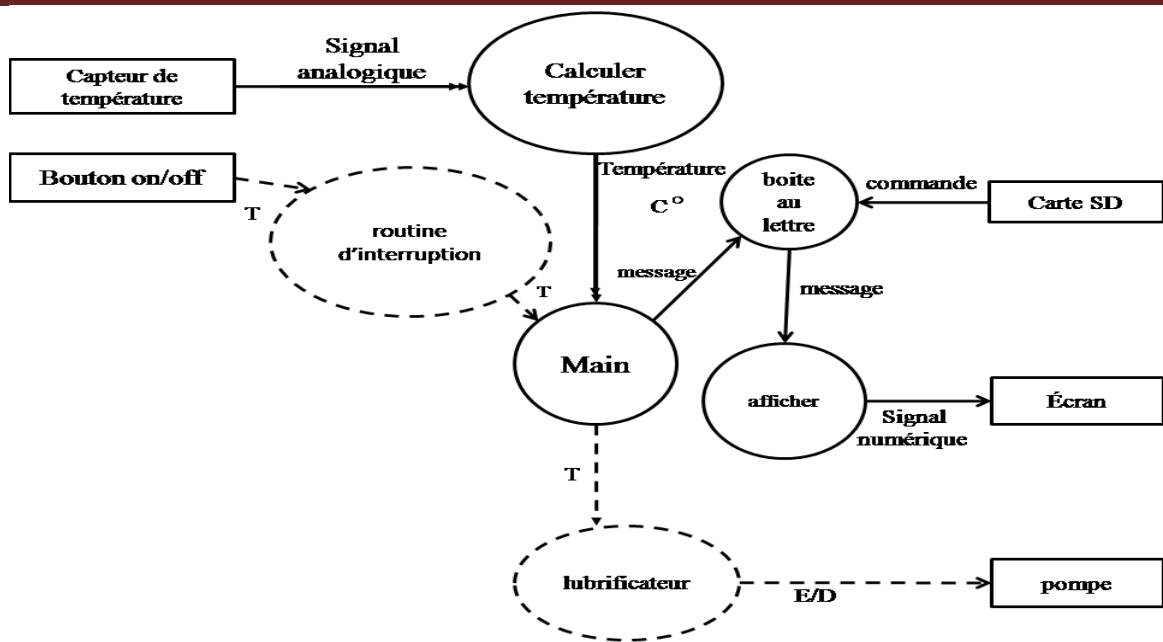


Figure 52 : diagramme de flot de contrôle.

5.3.6.2 Conception

On utilise la méthode de conception multitâche « LACATRE » pour permettre le passage d'un modèle de spécification basé sur une analyse fonctionnelle et structurée qui a été réalisée avec la méthode SA-RT aux entités de programmes dans un langage exécutable.

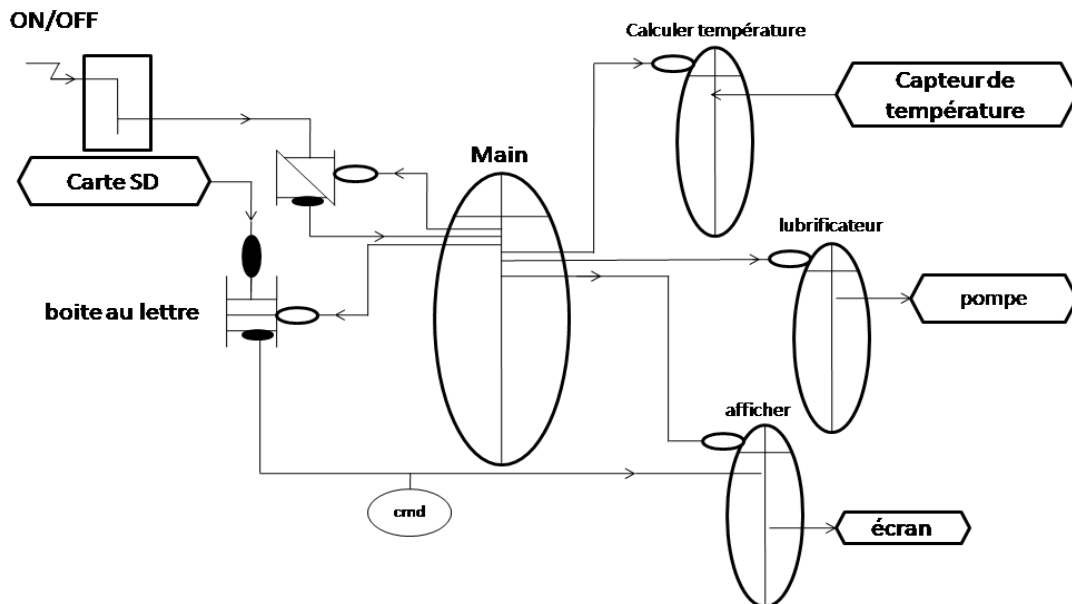


Figure 53 : schéma multitâche.

5.3.6.3 Test

Le test se fait en assemblant les composants et en programmant la carte Arduino, les images suivantes montrent comment cela a été fait.

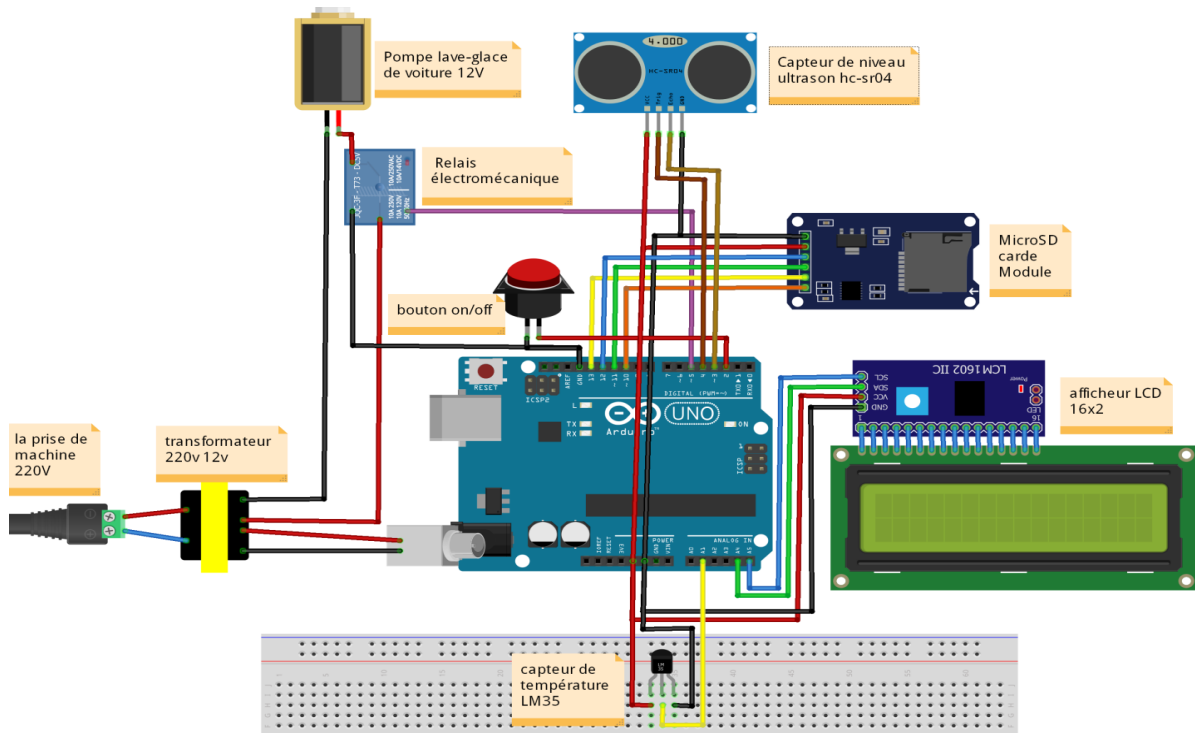


Figure 54 : schéma de câblage.

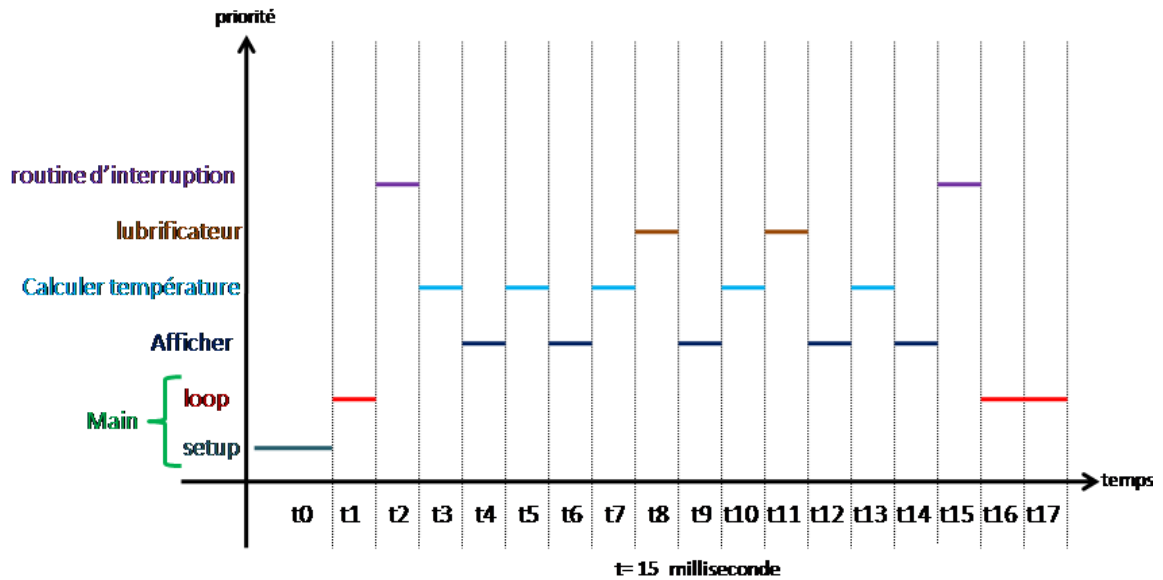


Figure 55 : le chronogramme de la séquence d'exécution.

5.4 Mesure des métriques de complexité et de qualité [22]

Dans ce mémoire, nous avons proposé une approche incrémentale afin d'établir les paramètres de mesure des performances des méthodologies appliquées aux systèmes temps réel embarqué, les métriques mesurent les qualités du logiciel sont les métriques traditionnelles et les métriques orientées objet, les métriques orientées objet prennent en considération les relations entre éléments de programme (classes, méthodes), les métriques traditionnelles se divisent en deux groupes : les métriques mesurant la taille et la complexité, et les métriques mesurant la structure du logiciel, les métriques mesurant taille et complexité les plus connus sont les métriques de ligne de code ainsi que les métriques de Halstead, les métriques mesurant la structure d'un logiciel comme la complexité cyclomatique de McCabe se basent sur des organigrammes de traitement ou des structures de classe.

Le tableau suivant présente les métriques des lignes de code LOC (line of code) :

LOCphy : nombre de lignes physiques (total des lignes des fiches source).

LOCpro : nombre de lignes de programme (déclaration, définition, directives, et code).

LOCcom : nombre de lignes des commentaires.

LOCbl : nombre de lignes vides.

Incrément	LOCphy	LOCpro	LOCcom	LOCbl	Nouvel lignes	fonctions	Max LOC par fonction	Boucle et bronchement
Les normes	Inferieur a 400	4..400	30%..75%	Inferieur a 10%	Inferieur a 40	+1 par incrément	4..40	
00	36	24	08	04	00	03	16	03
01	106	78	21	07	70	06	24	08
02	125	92	25	08	19	07	27	09
03	164	123	31	10	39	08	31	13

Tableau 04 : les métriques des lignes de code LOC

5.5 Quelles sont les limites acceptables ?

La longueur des fonctions devrait être de 4 à 40 lignes de programme. Une définition de fonction contient au moins un prototype, une ligne de code, et une paire d'accolades, qui font 4 lignes, une fonction plus grand que 40 lignes de programme probablement implémente beaucoup de fonctions, les fonctions contenant un état de sélection avec beaucoup de branches sont une exception à cette règle, les décomposer en des fonctions plus petites réduit souvent la lisibilité.

La longueur du fichier devrait être de 4 à 400 lignes de programme. La plus petite entité qui peut raisonnablement occuper un fichier source complet est une fonction, et la longueur minimum d'une fonction est de 4 lignes. Les fichiers plus longs que 400 lignes de programme (10...40 fonctions) sont habituellement trop longs pour être compris en totalité.

Au minimum 30 % et au maximum 75 % d'un fichier devrait être commenté, si moins d'un tiers du fichier est commenté, le fichier est soit très trivial, soit pauvrement expliqué, si plus de trois quarts du fichier est commenté, le fichier n'est plus un programme, mais un document.

Le tableau suivant présente les résultats de nos codes :

Numéro d'incrément	Pourcentages LOCpro	Pourcentages LOCcom	Pourcentages LOCbl
Les normes	25%..70%	30%..75%	Inferieur a 10%
00	66.67%	22.22%	11.11%
01	73.58%	19.82%	6.6%
02	73.6%	20%	6.4%
03	75%	18.9%	6.1%

Tableau 05 : les résultats de nos codes

Conclusion générale

Ce projet avait pour objectif d'appliquer les notions de la méthodologie de conception SA-RT selon une approche incrémentale pour concevoir un system de contrôle command pour les machines manuelles de tournage afin de le rendre semi-automatique et d'élaborer une étude qui permettra de réaliser une meilleure conception et vérifier sa flexibilité maintenabilité et efficacité.

Notre expérimentation au sein du Laboratoire LIM a permis de montrer qu'il est utilisable et pratique d'utiliser cette approche pour réaliser des projets inter domaine grâce à sa flexibilité, les résultats obtenus permettront de faire le contrôle de qualité des produits pour développer la recherche au sein de tout domaine utilisant cette approche, comme perspectives de ce travail, notons qu'il y a des travaux de recherche à faire pour améliorer les résultats obtenus.

Ce projet de fin d'études nous a été d'une grande utilité, il vient de compléter nos compétences et notre formation académique en nous permettant, outre que fréquenter le milieu industriel, de réaliser une étude technique et de mettre en pratique nos connaissances théoriques.

A travers ce travail, nous espérons avoir contribué à apporter une valeur ajoutée à la connaissance du l'équipe du laboratoire LIM avec laquelle nous avons eu l'honneur de travailler pendant cette période de formation.

Par ailleurs, ce projet a été pour nous une opportunité enrichissante pour appliquer les méthodes puissantes de la conception et exploiter ainsi ce que nous avons acquis durant notre formation dans notre établissement.

Bibliographié

- [1] L. BENGTSSON, «Embedded Measurement Systems,» Department of Physics University of Gothenburg, Gothenburg, 2013.
- [2] C. John, «Designing Embedded Hardware,» 2005.
- [3] M. Timmerman, «RTOS Market survey: preliminary result,» *Dedicated System Magazine*, 2000.
- [4] A. ALKHODRE, «Thèse Développement formel de systèmes temps réel à l'aide de SDL,» L'Institut National des Sciences Appliquées de Lyon, Lyon, 2004.
- [5] A. Shaout, A. Elmousa et K. Mattar, «Specification and Modeling of HW/SW CO-Design for Heterogeneous Embedded Systems,» chez *Proceedings of the World Congress on Engineering*, 2009.
- [6] k. LAREDJ, «Développement des approches incrémentales pour la conception et l'implémentation des systèmes contrôle-commande,» Université Ibn Khaldoun Tiaret Faculté de mathématiques et d'informatique Laboratoire LIM, Tiaret.
- [7] A. Leva et L. Piroddi, «FPGA-based implementation of an active vibration controller,» chez *Proceedings of the 17th World Congress The International Federation of Automatic Control*, 2008.
- [8] A. Checkburn, «Using Both Incremental and Iterative Development,» chez *Proceedings of the 17th World Congress The International Federation of Automatic Control*, 2008.
- [9] B. Ahmad, «Thèse de doctorat Compilation pour système temps réel,» L'Institut National des Sciences Appliquées de Lyon, Lyon, 2004.
- [10] M. Lakhoua et Z. SALAH, «Application of Structured Analysis Real Time Method on a Natural Gas Station,» 2013.
- [11] C. GRIMM, Languages for system specification Selected Contributions on UML, SystemC, SystemVerilog, Mixed-Signal Systems, and Property Specification from FDL, 2004.
- [12] M. LeGoc, «cour Spécification et conception temps réel Principes de base de SA-RT,» Projet Sachem I. U. S. P. I. M, 1996.
- [13] J. Schwarz, K. Jelemenska, Z. Huang, R. Aubry et J. Babau, «From CRSM specification to a real-time multitasking execution model,» *IEEE International Symposium on Industrial Electronics (Real-Time session)*, July 1999.
- [14] «SDL-RT: Specification and description language Real-Time,» 2018. [En ligne]. Available: <http://www.sdl-rt.org/>. [Accès le 2020].
- [15] J. Masse, S. Kim et S. Hong, «Tool set implementation for scenario based multithreading of UML-RT models and experimental validation,» chez *Proceeding of the IEEE Real Time and Embedded Technology and Applications Symposium*, 2003.
- [16] B. Selic, T. Gullekson et P. Ward, *Oriented Modeling*, New York: Hardcover, 1994.
- [17] J. Schwarz, J. Skubich et R. Aubry, «Lacatre The basis for a Real Time Software Engineering Workshop,» PEARL Boppard, Germany, 1991.
- [18] B. Richard, «Mastring the FreeRTOS real time kernel,» 2020. [En ligne]. Available: <http://www.FreeRTOS.org>. [Accès le mai 2020].

- [19] «ASPENCORE 2019 Embedded Markets Study Integrating IoT and Advanced Technology Designs, Application Development & Processing Environments,» 2019. [En ligne]. Available: www.eetimes.com. [Accès le 2020].
- [20] I. Amara, «Support de Cours : fabrication mécanique et métrologie,» Département de Génie Mécanique, Université Mentouri Constantine, Constantine, 2001.
- [21] A. Debih, «Techniques de Fabrication mécanique S4 TCT,» Faculté de Technologie de M'sila Département Génie Mécanique, M'sila, 2019.
- [22] K. LAMBERTZ et X. N. Cullmann, «Complexité et qualité Comment mesurer la complexité d'un logiciel,» www.msccoder.org/fr, Allemagne, 2007.