

III.1 Introduction

En électronique de puissance, on utilise de plus en plus des convertisseurs utilisant des interrupteurs électroniques fonctionnant soit à l'état passante (ou fermée), soit à l'état bloquée (ou ouverte). Le passage d'un état à l'autre s'effectue périodiquement. Les améliorations apportées à l'onduleur se situent essentiellement au niveau de la commande, tout d'abord, la génération d'une onde purement sinusoïdale, mais également la surveillance de la tension d'entrée et de la tension de sortie, dans ce passage on a besoin d'un générateur ou transmetteur de signal. C'est ainsi que l'on expose les commandes des convertisseurs à partir d'un composant analogique (amplificateur opérationnel) appelée commande analogique ou commande à base d'un (PIC, ARDUINO, DSP ou FPGA) appelée commande numérique. On obtient ainsi les signaux Qui Vont aller vers les composants de puissance. On va utiliser dans ce projet une carte ARDUINO de type Mēga 2560 pour générer des signaux de commande [20].

III.2 Commande numérique

La commande numérique permet également la réalisation de signaux de commande de manière plus fiable qu'avec des montages utilisant l'électronique analogique. Elle fait appel à des circuits intégrés programmables. Elle doit aussi inclure des « entrées » de contrôle (Courant, Tension et de Vitesse, ...) sous forme de signal logique ou numérique, pour protéger le système commandé (Fig.III.1).

On distingue deux types de circuits Programmables :

- Les circuits utilisant des processeurs tels que microcontrôleur, le PIC (Programmable Interrupt Contrôler) ou le DSP (Digital Signal Processor) ou une carte ARDUINO. Ce Qui caractérise ces circuits, c'est que la programmation s'effectue soit en Langage assembleur, soit en langage « C ». L'architecture de ces composants étant Déjà réalisée par le constructeur, on ne peut que modifier le programme pour Adapter le composant à la commande souhaitée du convertisseur de puissance.
- Les circuits intégrés directement programmables de type FPGA (Field Programmable Gate Array). Ce qui caractérise ces circuits, c'est que la programmation s'effectue généralement en langage VHDL (Very High Density Language). L'utilisateur «Construit» lui-même l'architecture de son composant en prévoyant les Fonctions de commande et de contrôle souhaitées.

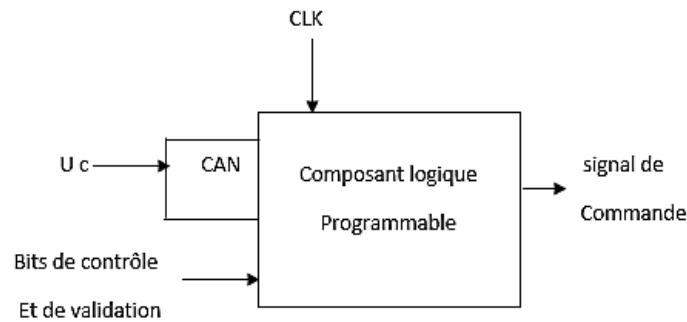


Figure III.1 : Principe de la commande numérique.

III.3 Description de la carte 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. 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. Des dizaines de milliers d'artistes, de designers, d'ingénieurs, de chercheurs, d'enseignants et même d'entreprises l'utilisent pour réaliser des projets incroyables dans de multiples domaines :

- Prototypage rapide de projets innovants utilisant l'électronique.
- Captation et analyse de données scientifiques.
- Installations d'arts numériques.
- Projets pédagogiques.

III 3.1 Partie matérielle

La carte Arduino repose sur un circuit intégré (un miniordinateur appelé également microcontrôleur) associée à des entrées et sorties qui permettent à l'utilisateur de brancher différents types d'éléments externes :

- Côté entrées, des capteurs qui collectent des informations sur leur environnement comme la variation de température via une sonde thermique, le mouvement via un détecteur de présence ou un accéléromètre, le contact via un bouton-poussoir, etc.

- Côté sorties, des actionneurs qui agissent sur le monde physique telle une petite Lampe qui produit de la lumière, un moteur qui actionne un bras articulé, etc.

Comme le logiciel Arduino, le circuit électronique de cette plaquette est libre et ses plans sont disponibles sur internet. On peut donc les étudier et créer des dérivées. Plusieurs constructeurs proposent ainsi différents modèles de circuits électroniques programmables et utilisables avec le logiciel Arduino.

Il existe plusieurs variétés de cartes Arduino UNO, DUE et MEGA chaque une a des caractéristiques spécifiques.

III 3.2 Partie logicielle

L'environnement de programmation ARDUINO (IDE en anglais) est une application écrite en Java, dérivée du langage Processing. L'IDE permet d'écrire et de modifier les codes et les convertir en une série d'instructions compréhensibles par la carte. On y retrouve les éléments de la fenêtre de travail de Processing : intitulés des menus, boutons, zone d'édition, console, etc. comme dans Processing, le programme est constitué d'une série d'instructions saisie dans la fenêtre du logiciel. Le programme qui envoyé vers la carte ARDUINO est enregistré de manière permanente, ce programme est activé à chaque fois que l'on alimente la carte.

III.4 La carte ARDUINO Mēga2560

Le modèle Mega Arduino est une carte microcontrôleur basée sur l'ATmega2560. Il dispose de 54 broches numériques d'entrée / sortie (dont 14 peuvent être utilisées comme sorties PWM), 16 entrées analogiques, 4 UART (ports série matériels), un oscillateur en cristal de 16 MHz, d'une connexion USB, une prise d'alimentation, d'une embase ICSP, et un bouton de réinitialisation. Il contient tout le nécessaire pour soutenir le microcontrôleur suffit de le brancher à un ordinateur avec un câble USB ou de la puissance avec un adaptateur ou la batterie AC -DC pour commencer. Le Mega est compatible avec la plupart des blindages conçus pour l'Arduino Duemilanove ou Diecimila [20].

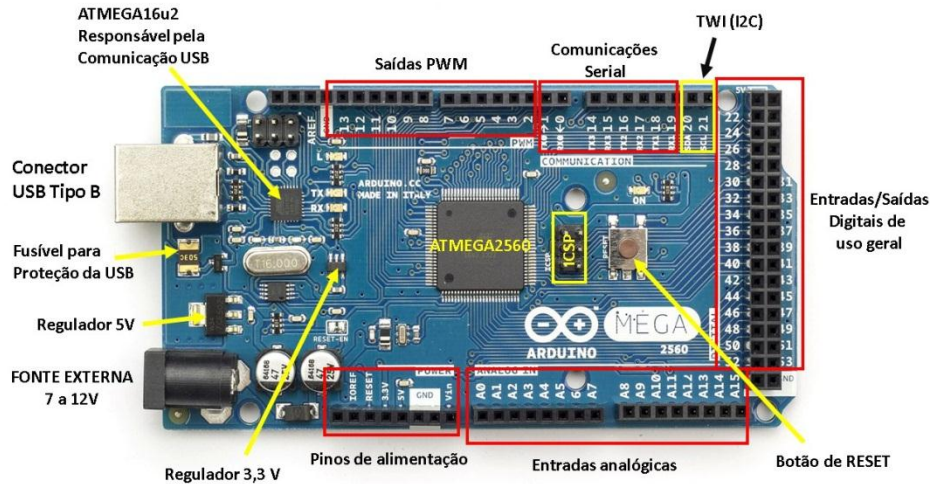


Figure .III.2 : Modèle de l'ARDUINO MEGA 2560.

Programmables via un langage proche du "C" (disponible en libre téléchargement), les modules Arduino peuvent fonctionner de façon autonome ou en communiquant avec un logiciel "tournant" sur un ordinateur (Flash, MaxMSP...).

Les modules Arduino sont des plateformes de prototypage micro contrôlées "open source" spécialement conçu pour les artistes, les concepteurs ou les hobbyistes.

III.4.1 Les caractéristiques de la carte ARDUINO MEGA2560

Microcontrôleur	ATmega2560
Tension de fonctionnement	5 V
Tension d'alimentation (recommandée)	7- 12 V
Tension d'alimentation (limites)	6 - 20V
Nombre d'E/S	54 (où 14 pouvant générer des signaux PWM)
Nb ports "Analogique "	16
Courant max. par E/S	40 mA
Courant pour broches 3.3 V	50 mA
Mémoire Flash	256 KB dont 8 KB utilisé parle Bootloader
SRAM	8 KB
EEPROM	4 KB
Vitesse horloge	16 MHz

Tableau III.1 : Caractéristiques de l'Arduino MEGA2560 [20].

III.4.2 Partie programme

Une telle carte d'acquisition qui se base sur sa construction sur un microcontrôleur doit être dotée d'une interface de programmation comme est le cas de notre carte. L'environnement de programmation open-source pour Arduino peut être téléchargé gratuitement (pour Mac OS X, Windows, et Linux) [21].

III.4.2.1 L'environnement de la programmation

Le logiciel de programmation de la carte Arduino sert d'éditeur de code (langage proche du C). Une fois, le programme tapé ou modifié au clavier, il sera transféré et mémorisé dans la carte au travers de la liaison USB. Le câble USB alimente à la fois en énergie la carte et transporte aussi l'information CE programme appelé IDE Arduino. [22][23]

III.4.2.2 Structure générale du programme (IDE Arduino)

Comme n'importe quel langage de programmation, une interface souple et simple est exécutable sur n'importe quel système d'exploitation Arduino basé sur la programmation en C.

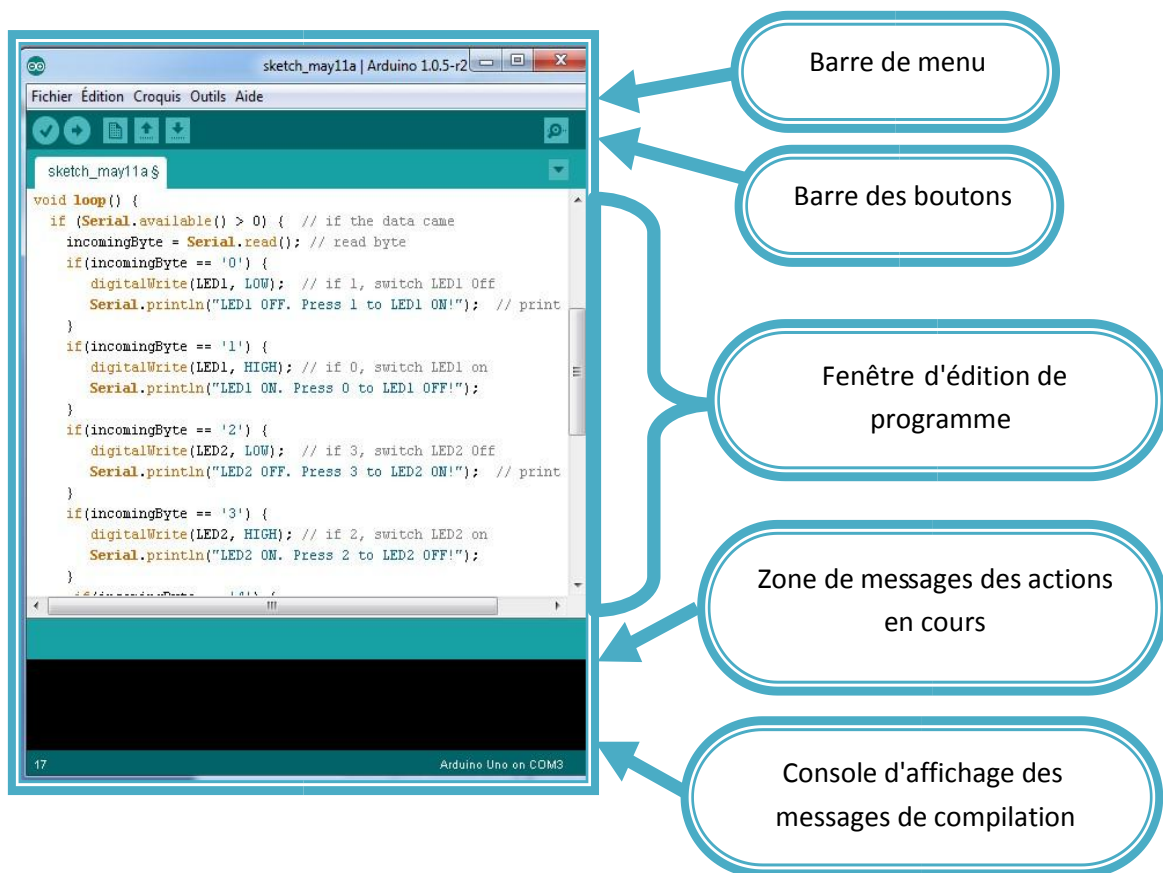


Figure III.3 : Interface IDE Arduino.

III.4.2.3 Commandes de structure du programme

Des programmes d'Arduino peuvent être divisés dans trois parts principales : structure, valeurs (variables et constantes), et fonctions [24].

Structure générale	Opérations booléennes	Pointer Access Operators	
setup () loop ()	&& (and) (or) ! (not)	* dereference operator & reference operator	
Contrôle et conditions	Variables Constants	Data Types	
if if...else for switch case while do... while break continue return goto	HIGH LOW INPUT OUTPUT INPUT_PULLUP true false integer constants floating point constants	boolean / char unsigned char / byte int / unsigned int word / long unsigned long short / float double string - char array String - object array	
Autres commandes	Arithmetic Operators	Opérations de comparaison	
(semicolon) { } (curly braces) // (single line comment) /* */(multi-line comment) #define #include	= (assignment operator) + (addition) - (subtraction) * (multiplication) / (division) % (modulo)	== (equal to) != (not equal to) < (less than) > (greater than) <= (less than or equal to) >= (greater than or equal to)	
Bitwise Operators	Conversion	Variable Scope & Qualifiers	
& (bitwise and) (bitwise or)	char() word() byte() long() float() int()	variable scope static	const volatile
Random Numbers	E/S analogique	Fonctionne E/S numérique	
randomSeed() random()	analogReference() analogRead() analogWrite() - PWM	pinMode() digitalWrite() digitalRead()	
Bits and Bytes	E/S Avancé	Trigonométrie	
lowByte() highByte() bitRead() bitWrite() bitSet()	tone() noTone() shiftOut() shiftIn() pulseIn()	sin() cos() tan() bitClear() bit()	
E/S Avancé	Gestion du temps	Mathématiques	
tone() noTone() shiftOut() shiftIn() pulseIn()	millis() micros() delay() delayMicroseconds()	min() max() abs() sqrt()	constrain() map() pow()

Tableau III.2: Listes des instructions d'un programme sous Arduino [24].

III.5 Création un signal MLI à partir de l'ARDUINO

III.5.1 Principe de création

Commande MLI ou PWM en anglais est une méthode très utilisée pour commander la puissance de l'autre côté des charges. Cette méthode est très facile à mettre en œuvre et à le rendement élevé. Signal PWM est essentiellement une onde carrée haute fréquence (typiquement supérieure à 1 kHz). Le cycle d'utilisation de cette onde carrée est varié afin de faire varier la puissance fournie à la charge [20]. Le cycle est généralement indiqué en pourcentage et il peut être exprimé par l'équation :

$$\text{Duty (\%)} = (\text{TON} / (\text{TON} + \text{TOFF})) * 100 \quad (\text{III.1})$$

Où TON est le temps pendant lequel l'onde carrée est en haute et TOFF est le temps pendant lequel l'onde carrée est en basse. Où le cycle de droit est augmenté le pouvoir a chuté à travers la charge augmentée et lorsque le cycle de service est réduit, la puissance à travers la charge diminue. Le schéma de principe d'un système typique de commande de puissance de PWM est représenté sur Figure III.5.

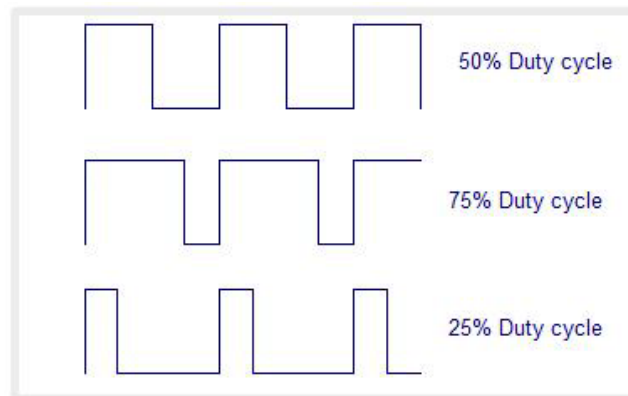


Figure III.4 : les cycles du signal PWM.

III.6 Simulation de la commande par modulation d'un onduleur triphasé

Après avoir rappelé le principe de fonctionnement de la MLI, nous allons là modéliser sous l'environnement du logiciel MATLAB/SIMULINK.

La simulation du système à l'aide de MATLAB/SIMULINK est tout à fait justifiable en raison de ses divers avantages uniques comme la facilité et l'efficacité de programmation élevées, l'interface graphique élégante et son architecture ouverte permettant l'adaptation à un grand nombre de cas.

III.6.1 La présentation des blocs sous Simulink de la MLI vectorielle

On simule le système pour une durée de 1/ 2.5s (5 périodes) à l'aide de l'algorithme de résolution des équations différentielles ode1 et un pas d'échantillonnage de 1/385000s.

Le schéma bloc de cette simulation apparaît sur la Fig.III.5 suivante:

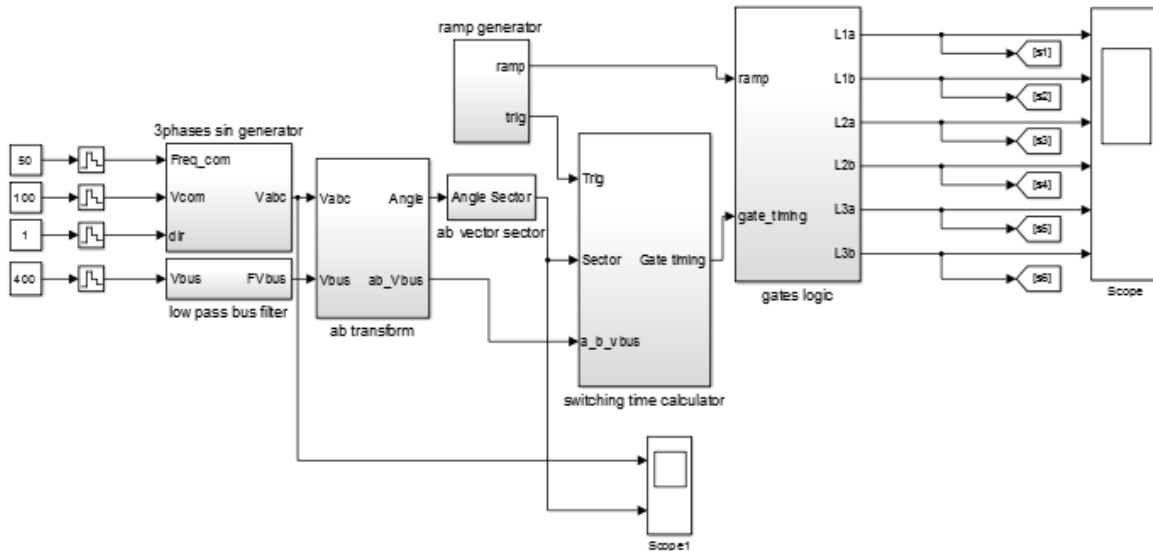


Figure III.5 : Schéma bloc de la commande MLI vectorielle sous l'environnement MATLAB/SIMULINK.

L'onduleur de puissance de la Figure III.7 est également utilisé pour la modulation vectorielle, et les résultats de simulation seront présentés dans cette section.

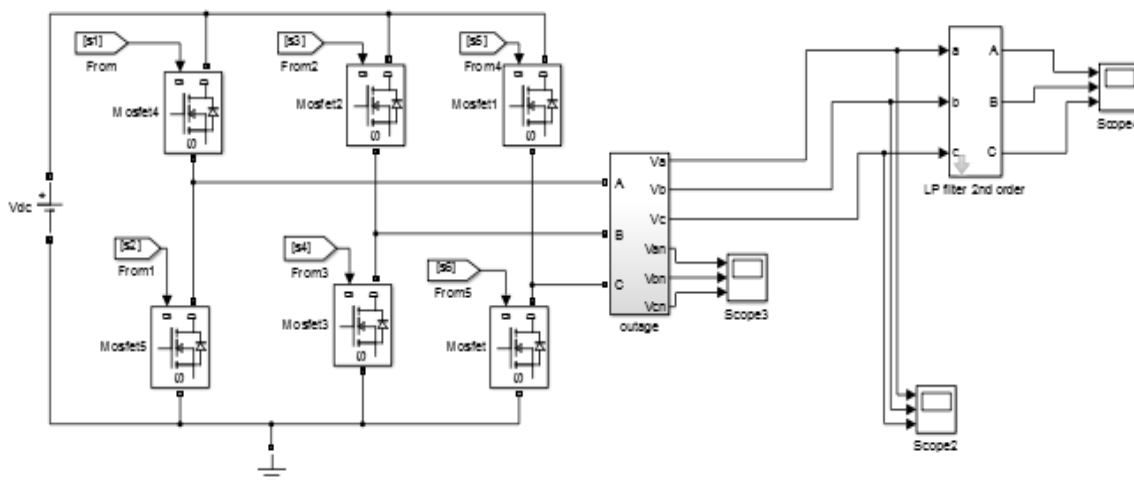


Figure III.6 : Schéma bloc de l'onduleur à MLI triphasé.

III.6.2 Résultats de simulation de la commande MLI vectorielle

Les résultats de simulation seront présentés dans cette section. Les différentes formes d'ondes ont été relevées dans les conditions de fonctionnement suivantes:

Tension d'entrée 400V.

Charge triphasée RL : $R=72.95 \Omega$ et $L=38.7\text{mH}$

Fréquence du signal de sortie $F_m=50 \text{ Hz}$

Fréquence de commutation $F_c=3000 \text{ Hz}$

Indice de modulation: $m=0.8$

La fréquence de commutation a été fixée à 3000Hz ce qui représente 60 fois la fréquence du signal de sortie. Nous remarquons qu'à cette fréquence la qualité des signaux de sortie est très acceptable et les pertes par commutation sont réduites.

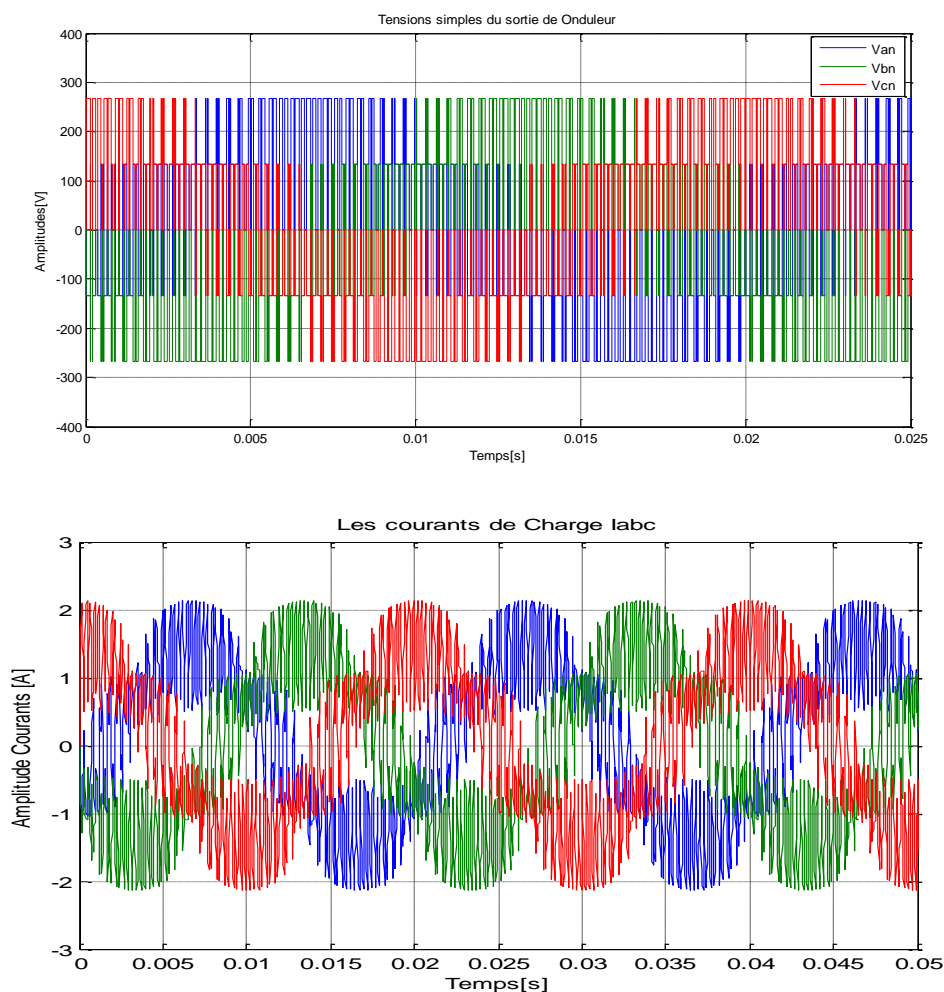


Figure III.7 : Tensions simples et courants triphasés.

Nous remarquons que dans le cas de la modulation vectorielle, les résultats sont légèrement meilleurs que ceux des autres modulations. Ce résultat nous surprend en quelque sorte, car on s'attendait à ce que cette modulation donne de meilleurs résultats vu que dans cette technique les durées de commutation sont déterminées selon la position du vecteur spatiale. Nous nous sommes donc intéressés à vérifier les performances de la modulation vectorielle dans le cas où $m=0.8$. Les résultats de simulation sont donnés à la Figure.III.8.

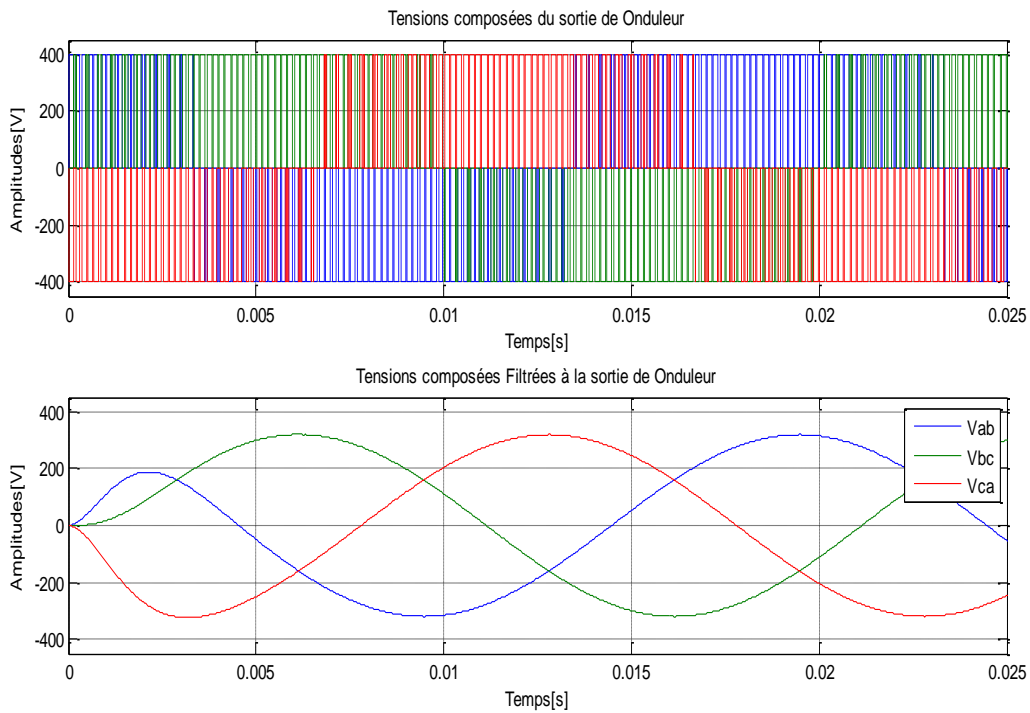


Figure III.8 : Tensions composés devant et après le filtrage.

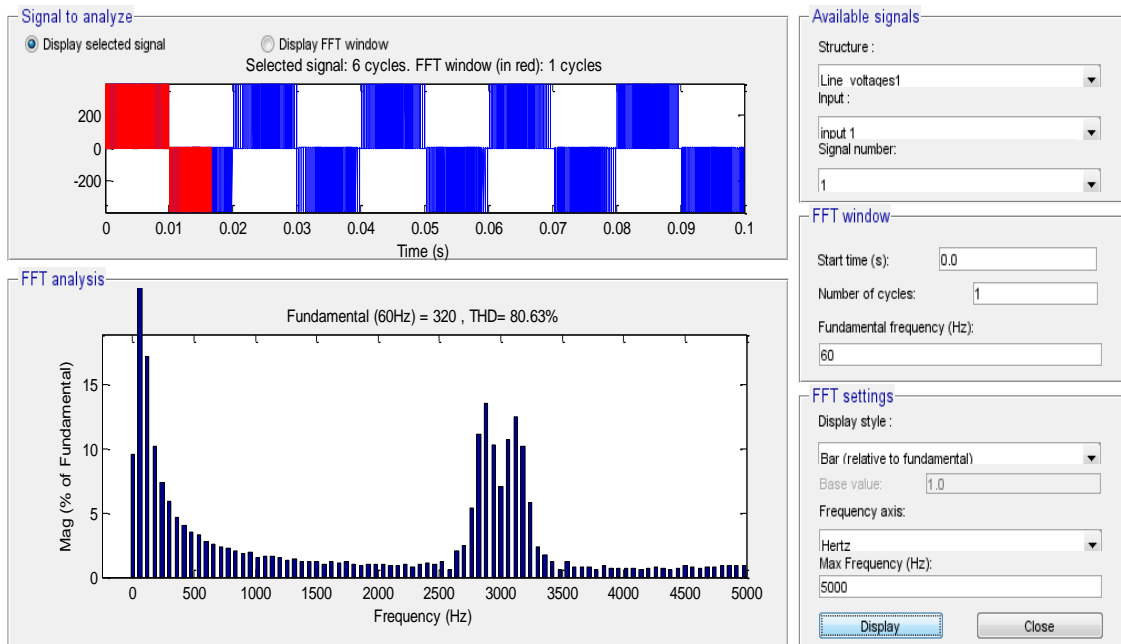


Figure III.9 : Forme d'onde et spectre de tension U_{ab} .

Les résultats obtenus du point de vue forme d'onde. Une meilleure comparaison Peut avoir lieu en utilisant l'analyse harmonique. La fonction FFT de la Figure III.9 nous montre la forme d'onde et le spectre de Van, avec un THD de 80.63% et un fondamental (50Hz) de 320.

III.7 MATLAB Arduino

Le choix académique actuel en matière de logiciel de simulation multi-physique se porte actuellement sur MATLAB. Il convient alors de s'ouvrir aux possibilités actuelles que cette solution logicielle permet à ce jour en matière de communication avec l'extérieur d'autant que la situation évolue de jour en jour. Avec Support Package MATLAB pour Arduino matériel, vous pouvez utiliser MATLAB pour communiquer de manière interactive avec une carte Arduino via un câble USB. Le package vous permet d'effectuer des tâches telles que :

- Acquérir des données de capteurs analogiques et numériques à partir de la carte Arduino.
- Contrôler d'autres appareils avec sorties numériques et MLI
- Conduisez DC, servo et moteurs pas à pas et des capteurs périphériques d'accès connectés via I2C ou SPI
- Construire des Add-ones à l'interface avec les bibliothèques matérielles et logicielles supplémentaires

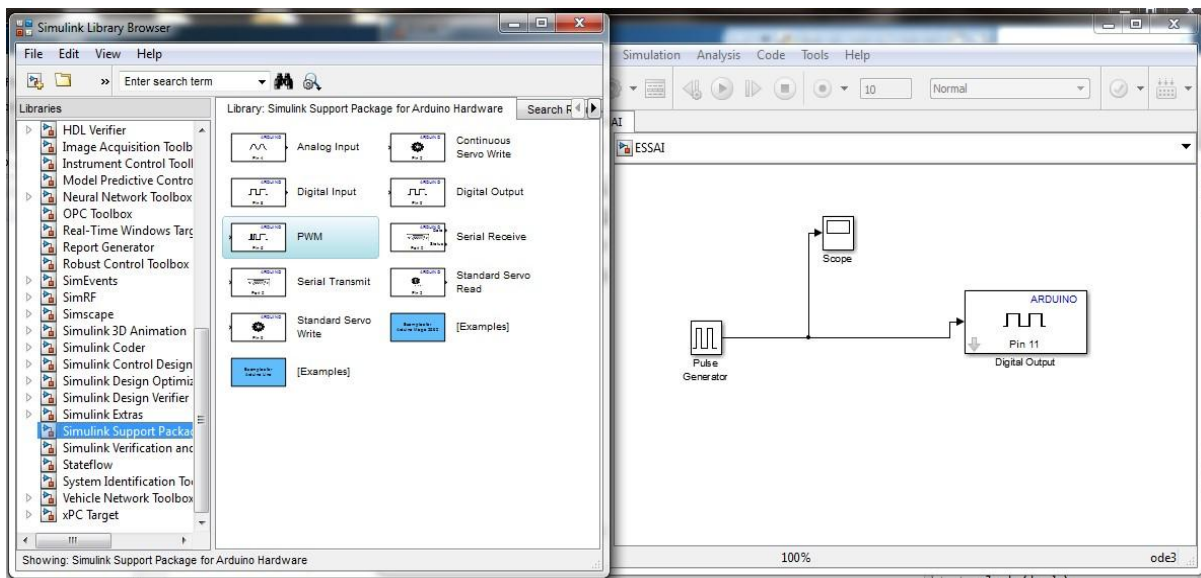


Figure III.10 : Communication de carte Arduino avec MATLAB.

Puisque MATLAB est un langage de haut niveau interprété, nous pouvons voir les résultats des instructions d'entrée / Sortie immédiatement, sans compiler. MATLAB comprend des milliers de mathématiques intégrés, l'ingénierie et des fonctions graphiques que vous pouvez utiliser pour analyser et visualiser rapidement les données recueillies à partir de notre Arduino [25]. Cette

solution consiste d'une part à utiliser les fonctions offert par le langage Arduino qui permet d'envoyer et d'acquérir des données binaires via le port série (USB) et d'autre part à développer sous Simulink un programme pour traiter ou visualiser ces données.

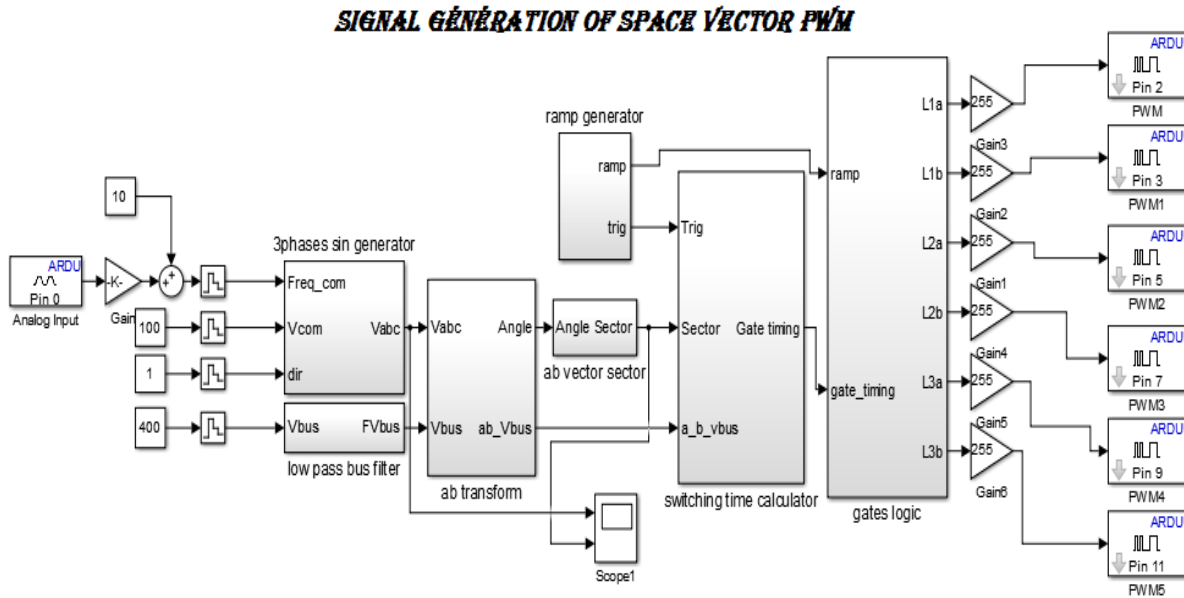


Figure III.11 : Modèle SIMULINK pour l'envoi des signaux SVPWM sous Arduino.

Les résultats que nous avons obtenus à partir de ce bloc SIMULINK montre une limitation en fréquence qui peut attein jusqu'au 500Hz présentée dans la figure III.12.

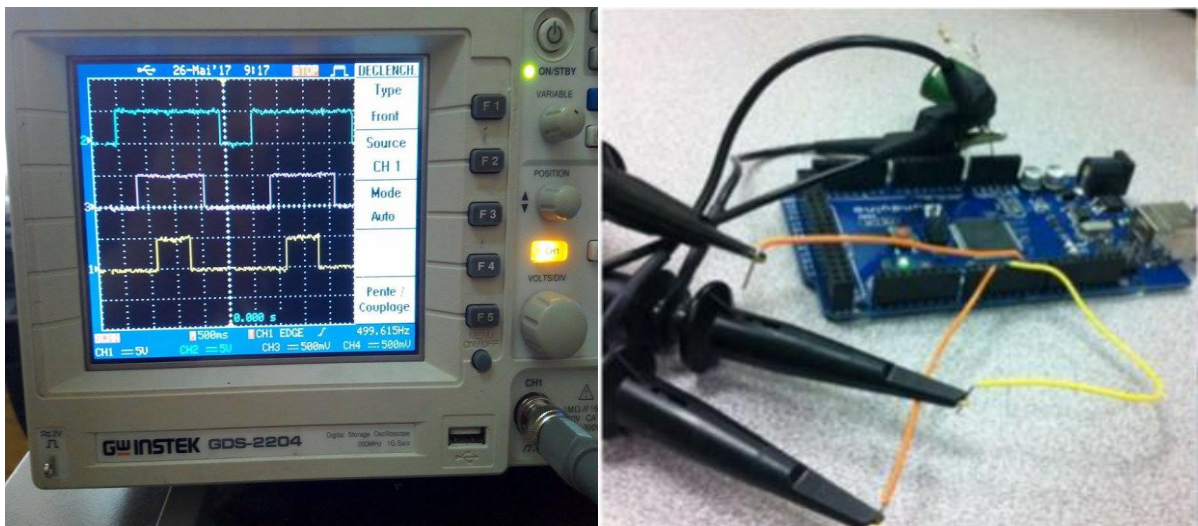


Figure III.12 : Signal de sortie du l'interface MATLAB/Arduino.

C'est la raison pour laquelle nous utilisons la programmation sous Arduino IDE software pour la mise en œuvre de l'onduleur de tension triphasé dans les étapes sont les suivantes :

- Générer une onde sinusoïdale à l'aide de la table de consultation.
- Initialiser les registres de contrôle de base de temps où l'opération PWM alignée au centre a lieu en sélectionnant le mode continu de comptage / décomptage.
- Initialiser les valeurs du registre où les cycles de charge sont chargés.
- Calculer V_s et φ sur la base de l'exigence de V_α et V_β .
- Compte tenu d'une période T d'onde porteuse, calculez T_1 , T_2 et T_0 .
- Calculer les cycles de service T_a , T_b et T_c dans les registres TIM1, TIM2 et TIM3. Au début d'un nouveau cycle (interruption PWM) T_1 , T_2 et T_0 sont calculés. Et puis, le temps actif de S_1 , S_2 , . . . , Les commutateurs S_6 sont donc écrits dans les registres TIM1, TIM2 et TIM3 du microcontrôleur Atmega2560.

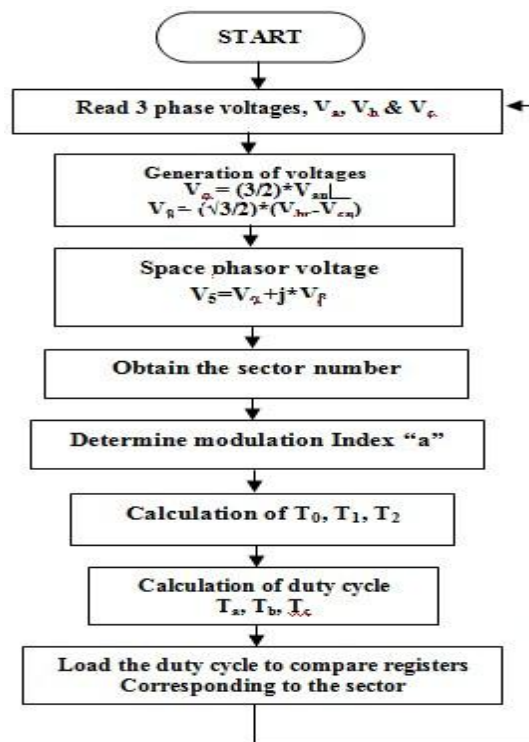


Figure III.13 : Diagramme d'écoulement pour l'implémentation de la Modulation Vectorielle.

III.8 Conclusion

Dans ce chapitre, nous avons exposé une description sur la carte ARDUINO MEGA et comment l'utiliser et puis on a expliqué la technique de création un signal MLI à partir de la carte ARDUINO. Cette technique est réalisable grâce aux caractéristiques internes de cette carte et selon l'utilisation, après l'écriture du programme final, on va réaliser le circuit de l'onduleur et utiliser ce programme pour le commander, c'est ce que nous allons expliquer dans le chapitre qui suivent.