RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE

Ministere de l'Enseignement Superieur et de la Recherche Scientifique

École Nationale Polytechnique

Département d'Automatique

Spécialité Automatique

Projet de Fin d'Étude

En vue de l'obtention du

Diplôme d'Ingénieur d'Etat en Automatique

Thème :

Implémentation de la "SVM" à deux niveaux et à trois niveaux par la carte "ARDUINO DUE"

Travail réalisé par :

KHERROUBI Zine elabidine

Dirigé par :

Pr. El Madjid BERKOUK

Promotion Juin 2015

Abstract

In this work, a space vector pulse width modulation (SVPWM) for both two-level inverter and three-level neutral point clamped inverter are studied and simulated using Matlab/Simulink environment. In further stage, a real-time digital implementation of the developed algorithms is carried out using ARDUINO Due development board. The main contribution in this work is that the implementation was performed by using directly the internal PWM module of the ARM cortex M3 processor of the ARDUINO DUE. By mean of this, the computation time and the resources utilization of the processor are greatly reduced. The other contribution is that this implementation may reduce power losses and harmonics by minimizing the number of switching and generating symmetrical signals. To validate the implemented algorithms, an experimental validation of the SVPWM for two level inverters is carried out by using a semikron multi-function converter. The SVPWM for three-level neutral point clamped Inverter is validated using the switching signals generated from the arduino DUE board and the Matlab/Simulink environment. The results obtained from the experimentation are closer to that of simulation, which confirms the validity of the implemented algorithms.

Key words: Space Vector Pulse Width Modulation (SVPWM), Two Level Inverter, Three Level Neutral Point Clamped Inverter, ARDUINO DUE BOARD.

ملخص

في هذا العمل، قمنا بدراسة تقنية التحكم الشعاعي لكل من المموجات ذات مستويين و ثلاث مستويات بإستعمال برنامج ماتلاب-سيميلنك. في مرحلة ثانية، قمنا بتثبيت هذه التقنية على بطاقة التحكم أردوينوديو. الإضافة الأساسية في هذا العمل تثمثل في إستغلال الوحدة الداخلية للحاسب أردوينوديو بطريقة سمحت بالتقليل من وقت تنفيذ البرنامج و إذخار مصادر لوحة التحكم. هذه الطريقة مكنت أيضا من التقليل من ضياع الطاقة في المموج و التقليل من التوافقيات، و ذلك بالتقليل من عدد التبديلات و توليد إشارات متناسقة. التحقق من صحة خوارزمية التحكم الشعاعي ذو مستويين تم بواسطة المموج سوميكرون. التحقق من صحة خوارزمية التحكم الشعاعي ذات ثلاث مستويات تم بإستعمال إشارات التحكم الصادرة من بطاقة أردوينوديو و البرنامج ماتلاب-سيميلنك. النتائج التجريبية المحصلة كانت قريبة جدا من النتائج النظرية، مما يتبث صحة العمل المنجز.

الكلمات المفتاحية: **التحكم الشعاعي، مموج ذو مستويين، مموج ذو ثلاث مستويات، بطاقة التحكم أردوينوديو**

Résumé

Le but de ce travail est d'étudier et d'implémenter la modulation vectorielle (SVM) pour les onduleurs à deux niveaux et les onduleurs à trois niveaux à structure NPC. Une étude théorique a été faite grâce a l'outil MATLAB/SIMULINK. Une implémentation temps réel a était réalisée par la carte ARDUINO DUE. L'apport principal de ce projet est qu'on a réussi à exploiter le module PWM de la carte ARDUINO DUE d'une manière à optimiser le temps d'exécution et les ressources matérielles de la carte. On a réussi aussi à réduire les pertes en puissance et les harmoniques, par la minimisation du nombre de commutations des interrupteurs et la génération des signaux de commande symétriques. La validation de l'algorithme SVM à deux niveaux a été faite en utilisant l'onduleur multifonctionnel SEMIKRON. La validation de l'algorithme SVM à trois niveaux a été faite en utilisant les signaux de commande générés par la carte ARDUINO DUE et l'environnement Matlab/Simulink. Les résultats expérimentaux obtenus étaient très proches des résultats de simulation, ce qui confirme la validité de notre implémentation.

Mot clés : Modulation vectorielle (SVM), Onduleur à deux niveaux, Onduleur à trois niveaux à structure NPC, Carte ARDUINO DUE.

Remerciements

Louange à ALLAH, Seigneur de l'univers, le tout miséricordieux, le très miséricordieux, pour m'avoir accordé le savoir, le courage, la patience, la volonté et la force nécessaire pour affronter toutes les difficultés et les obstacles, durant toutes mes années d'études..

La réalisation de ce mémoire a été possible grâce au concours de plusieurs personnes à qui je voudrais témoigner toute ma reconnaissance.

Je voudrais tout d'abord remercie mon promoteur Monsieur ***El Madjid Berkouk****, professeur à l'École Nationale Polytechnique, pour le privilège qu'il ma a fait en acceptant de diriger ce travail.*

Je remercie très chaleureusement les membres du jury pour l'honneur qu'ils m'ont fait en acceptant d'examiner mon travail.

Je souhaite aussi remercier Monsieur ***Kermadi Mostafa*** *et* ***Fethi Akel*** *pour toute l'aide qu'ils m'ont apportée.*

Je profite à remercier du cœur mes professeurs qui m'ont transmis le tison de la connaissance depuis l'école primaire.

Dédicace

Je dédie ce modeste travail :

A mes très chers parents,

A mes très chères sœurs,

A mes profs,

A mes Amis,

A toute ma famille de proche ou de loin.

ZINOU

Table des matières

Liste des Figures iv

Liste des Tableaux v

Abréviations et symboles vi

1 Description de l'environnement ARDUINO 3
- 1.1 Introduction . . . 3
  - 1.1.1 Architecture ARM . . . 4
- 1.2 Avantages de l'environnement ARDUINO . . . 4
- 1.3 ARDUINO HARDWARE . . . 5
  - 1.3.1 ARDUINO DUE HARDWARE . . . 6
  - 1.3.2 Caractéristiques de la carte *ARDUINO DUE* . . . 8
- 1.4 L'environnement de développement intégré (ARDUINO IDE) . . . 9
- 1.5 Le langage ARDUINO . . . 11
- 1.6 Exemples d'application . . . 12
  - 1.6.1 Exemple 1 . . . 14
  - 1.6.2 Exemple 2 . . . 15
- 1.7 Conclusion . . . 17

2 La modulation vectorielle à deux niveaux 18
- 2.1 Introduction . . . 18
- 2.2 La structure d'un onduleur à deux niveaux . . . 19
- 2.3 États d'un bras de l'onduleur . . . 19
- 2.4 États de l'onduleur à deux niveaux . . . 21
- 2.5 Modélisation de l'onduleur à deux niveaux . . . 21
- 2.6 Transformation triphasée biphasée . . . 23
- 2.7 Séquence de commutation des interrupteurs . . . 27
- 2.8 Simulation numérique . . . 29
  - 2.8.1 Tensions de sortie . . . 30
  - 2.8.2 Analyse Harmonique . . . 32
  - 2.8.3 Courbe de réglage . . . 34
  - 2.8.4 Conclusion . . . 35

3 La modulation vectorielle à trois niveaux 36

3.1 Introduction . . . 36
3.2 La topologie d'un onduleur à trois niveaux à structure NPC . . . 37
3.3 Fonctions de commutation . . . 37
3.4 États d'un bras de l'onduleur à trois niveaux à structure NPC . . . 38
3.5 Fonction de connection . . . 40
3.6 États de l'onduleur a trois niveaux . . . 40
3.7 Modélisation de l'onduleur à trois niveaux à structure NPC . . . 42
3.8 Vecteur tension de référence et diagramme vectoriel . . . 42
3.9 Équations des régions de l'onduleur à trois niveaux . . . 45
3.10 Séquence des états de l'onduleur . . . 46
3.11 Durée d'application des états X,Y et Z . . . 47
3.12 Ordre d'application des états X,Y et Z . . . 48
3.13 Simulation numérique . . . 49
3.13.1 Tensions de sortie . . . 50
3.13.2 Analyse Harmonique . . . 52
3.13.3 Courbe de réglage . . . 54
3.14 Conclusion . . . 55

4 Étude expérimentale **57**
4.1 Introduction . . . 57
4.2 Description du banc d'essais utilisé pour la validation expérimentale . . . 58
4.3 Étude expérimentale de la SVM à deux niveaux . . . 63
4.3.1 Implémentation de la SVM à deux niveaux sur la carte ARDUINO DUE 63
4.3.2 Description du module PWM . . . 63
4.3.3 Initialisation du module PWM . . . 66
4.3.4 Algorithme de la SVM a deux niveaux sur la carte ARDUINO DUE . 66
4.3.5 Signaux de commande . . . 68
4.3.6 Étude de l'onduleur avec une charge résistive R . . . 70
4.3.7 Étude de l'onduleur avec une charge R-L . . . 73
4.3.8 Influence de la fréquence d'échantillonnage . . . 75
4.4 Étude expérimentale de la SVM à trois niveaux . . . 78
4.4.1 Détermination de la phase du vecteur tension de référence . . . 78
4.4.2 Détermination du secteur et de la région . . . 78
4.4.3 Détermination des durées relatives . . . 78
4.4.4 Arrangement des vecteurs X, Y et Z . . . 79
4.4.5 Codage des états à appliquer . . . 80
4.4.6 Implémentation sur la carte ARDUINO DUE . . . 85
4.4.7 Signaux de commande . . . 86
4.4.8 Méthode de validation . . . 88
4.4.9 Résultats obtenus . . . 89
4.4.10 Courbe de réglage . . . 90
4.4.11 Courbe du taux de distorsion d'harmonique . . . 90
4.4.12 Influence de la fréquence d'échantillonnage . . . 91

4.5 Conclusion 93

5 Conclusion générale 94

6 Bibliographie 96

7 Annexe 98

Table des figures

1.1 Circuit ARDUINO UNO . . . 3
1.2 Environnement de développement intégré (ARDUINO IDE) . . . 4
1.3 ARDUINO GSM SCHIELD . . . 5
1.4 ARDUINO MEGA 2560 . . . 6
1.5 Carte ARDUINO DUE . . . 7
1.6 Ports USB de la carte ARDUINO DUE . . . 7
1.7 L'interface d'utilisateur ARDUINO . . . 10
1.8 Sélection de la carte . . . 12
1.9 Sélection du port COM . . . 12
1.10 choix du programme . . . 13
1.11 chargement du programme . . . 13

2.1 Structure d'un onduleur à deux niveaux . . . 19
2.2 Etat 1 de l'onduleur à deux niveaux . . . 20
2.3 Etat 0 de l'onduleur à deux niveaux . . . 20
2.4 États d'un onduleur à deux niveaux . . . 21
2.5 Diagramme vectorielle de l'onduleur à deux niveaux . . . 25
2.6 États des ordres de commande pour le secteur 1 . . . 27
2.7 États des ordres de commande pour le secteur 2 . . . 27
2.8 États des ordres de commande pour le secteur 3 . . . 28
2.9 États des ordres de commande pour le secteur 4 . . . 28
2.10 États des ordres de commande pour le secteur 5 . . . 28
2.11 États des ordres de commande pour le secteur 6 . . . 29
2.12 Modéle de simulation de la SVM à deux niveaux . . . 29
2.13 Tension V_{ao} ($r = 0.8$, $F_e = 4KHz$) . . . 30
2.14 Tension V_{an} ($r = 0.8$, $F_e = 4KHz$) . . . 30
2.15 Tension V_{ab} ($r = 0.8$, $F_e = 4KHz$) . . . 31
2.16 Tensions V_{an}, V_{bn} et V_{cn} ($r = 0.8$, $F_e = 4KHz$) . . . 31
2.17 Spectre du tension V_{an} (r=0.8, F_e=4 KHz) . . . 32
2.18 Spectre du tension V_{an} (r=1.2, F_e=4 KHz) . . . 32
2.19 Taux de distorsion d'harmonique . . . 33
2.20 Courbe de réglage . . . 34
2.21 Trajectoire du $\vec{V}_{ref}$ lorsque $r = \frac{2\sqrt{3}}{3}$ . . . 34

3.1 Topologie d'un onduleur à trois niveaux à structure NPC 37
3.2 Etat P d'un bras de l'onduleur à trois niveaux à structure NPC 38
3.3 Etat O d'un bras de l'onduleur à trois niveaux à structure NPC 39
3.4 Etat N d'un bras de l'onduleur à trois niveaux à structure NPC 39
3.5 États d'un onduleur à trois niveaux 41
3.6 Diagramme vectoriel de l'onduleur à trois niveaux 43
3.7 Régions du premier secteur 45
3.8 États X,Y et Z de l'onduleur à trois niveaux 47
3.9 Modèle de simulation de la SVM à 3 niveaux 50
3.10 Tension V_{1o} (r=0.8, F_e=4 KHz) 50
3.11 Tension V_{1n} (r=0.8, F_e=4 KHz) 51
3.12 Tension V_{12} (r=0.8, F_e=4 KHz) 51
3.13 Tensions V_{1n}, V_{2n} et V_{3n} (r=0.8, F_e=4 KHz) 51
3.14 Spectre du tension V_{1n} (r=0.8, F_e=4 KHz) 52
3.15 Spectre du tension V_{1n} (r=1.2, F_e=4 KHz) 53
3.16 Taux de distorsion d'harmonique 53
3.17 Taux de distorsion d'harmonique (comparaison) 54
3.18 Courbe de réglage 54
3.19 Trajectoire du $\vec{V}_{ref}$ lorsque $r = \frac{2\sqrt{3}}{3}$ 55

4.1 La plateforme expérimentale 58
4.2 Onduleur à deux niveaux SEMIKRON 59
4.3 Interface d'adaptation 60
4.4 Retard généré par l'interface d'adaptation 60
4.5 Alimentation stabilisée 61
4.6 Charge R-L 61
4.7 Analyseur de spectre 62
4.8 Unité de commande 62
4.9 Module PWM de la carte ARDUINO DUE 64
4.10 Module PWM de la carte ARDUINO DUE en mode Center Aligned 65
4.11 Module PWM de la carte ARDUINO DUE en mode Left Aligned 65
4.12 Diagramme SVM 2 niveaux 68
4.13 signaux de commande pour le secteur 1 68
4.14 signaux de commande pour le secteur 2 69
4.15 signaux de commande pour le secteur 3 69
4.16 signaux de commande pour le secteur 4 69
4.17 signaux de commande pour le secteur 5 70
4.18 signaux de commande pour le secteur 6 70
4.19 Tension V_{ao}, V_{bo} et V_{co} (r=1, $F_e = 4KHz$) : expérimentale 71
4.20 Tension V_{ao}, V_{bo} et V_{co} (r=1, $F_e = 4KHz$) : simulation 71
4.21 Tension V_{ab} avec son spectre (r=1, $F_e = 4KHz$) : expérimentale 71
4.22 Tension V_{ab} avec son spectre (r=1, $F_e = 4KHz$) : simulation 72
4.23 Variation du fondamental normalisé de V_{ab} en fonction de r 72

4.24 Tension V_{an} avec son spectre (r=1, $F_e = 4KHz$) : expérimentale 73
4.25 Tension V_{an}, Courant i_a (r=1, $F_e = 4KHz$) : expérimentale 73
4.26 Tension V_{an}, Courant i_a (r=1, $F_e = 4KHz$) : simulation 74
4.27 Variation du fondamental normalisé de V_{an} en fonction du taux de réglage r . 74
4.28 THD (%) = f (r) 75
4.29 Tension V_{an} (r=1, F_e=1 KHz) 76
4.30 Tension V_{an} (r=1, F_e=5 KHz) 76
4.31 Tension V_{an} (r=1, F_e=10 KHz) 76
4.32 Fondamental de V_{an} en fonction de F_e (r=1) 77
4.33 Fréquence de V_{an} en fonction de F_e (r=1) 77
4.34 Signaux des interrupteurs pour la première région du premier secteur 81
4.35 Signaux des interrupteurs pour la quatrième région du sixième secteur 82
4.36 Tableau de codage pour l'interrupteur S_{11} 82
4.37 Tableau de codage pour l'interrupteur S_{21} 83
4.38 Tableau de codage pour l'interrupteur S_{12} 83
4.39 Diagramme SVM 3 niveaux 86
4.40 Signaux de commande des interrupteurs S_{11} et S_{21} : expérimentale 86
4.41 Signaux de commande des interrupteurs S_{11} et S_{21} : simulation 87
4.42 Etat P 87
4.43 Etat O 88
4.44 Etat N 88
4.45 Tension V_{1n} (r=0.8, F_e=4 KHz) 89
4.46 Spectre du tension V_{1n} (r=0.8, F_e=4 KHz) 89
4.47 Courbe de réglage 90
4.48 Courbe de THD(%) 90
4.49 Tension V_{1n} (r=1, F_e=1 KHz) 91
4.50 Tension V_{1n} (r=1, F_e=3 KHz) 91
4.51 Tension V_{1n} (r=1, F_e=6 KHz) 92
4.52 fondamental de la tension V_{1n} en fonction de F_e 92

Liste des tableaux

2.1 États d'un bras de l'onduleur à deux niveaux 20
2.2 Vecteurs de l'onduleur à deux niveaux dans le repère (α,β) 24

3.1 États d'un bras de l'onduleur à trois niveaux à structure NPC 39
3.2 Vecteur tension de référence pour l'onduleur à trois niveaux 44
3.3 Durée d'application des états X,Y et Z 48
3.4 Séquence des états de l'onduleur à trois niveaux 49

4.1 Arrangement des durées T_x, T_y et T_z pour la région 1 79
4.2 Arrangement des durées T_x, T_y et T_z pour la région 2 80
4.3 Arrangement des durées T_x, T_y et T_z pour la région 3 80
4.4 Arrangement des durées T_x, T_y et T_z pour la région 4 80
4.5 Tableau de codage pour l'interrupteur S_{22} 83
4.6 Tableau de codage pour l'interrupteur S_{13} 84
4.7 Tableau de codage pour l'interrupteur S_{23} 84

Abréviations et symboles

SVM	Modulation vectorielle
IDE	Environnement de développement intégré
PWM	Modulation a largeur d'impulsion
PIN	Broche de la carte ARDUINO
S_{a1},S_{b1},S_{c1}	Interrupteurs supérieurs de l'onduleur à deux niveaux
S_{a2},S_{b2},S_{c2}	Interrupteurs inférieurs de l'onduleur à deux niveaux
V_{dc}	Tension du bus continu
V_{ao}, V_{bo}, V_{co}	Tension entre bras et neutre de la source de l'onduleur à deux niveaux
V_{an}, V_{bn}, V_{cn}	Tension entre bras et neutre de la charge de l'onduleur à deux niveaux
$\vec{V}_a^*$, $\vec{V}_b^*$, $\vec{V}_c^*$	Vecteur tension de référence de chaque bras dans le repère *(abc)*
$\vec{V}_{ref}$	Vecteur tension de référence dans le repère *(α,β)*
V_{ref}	Amplitude du vecteur tension de référence
ω	Pulsation du vecteur tension de référence
θ	Phase du vecteur tension de référence
α	Phase du vecteur tension de référence par rapport au premier secteur
V_α,V_β	Composants du vecteur tension de référence dans le repère *(α,β)*
T_s	Période d'échantillonnage
$\hat{V}_{ref}$	Amplitude du $\vec{V}_{ref}$ en valeur relative a V_{dc}
F_e	Fréquence d'échantillonnage
r	Taux de réglage = $\dfrac{V_{ref}}{V_{dc}/2}$
THD(%)	Taux de distorsion d'harmonique en %
V_{eff}	Valeur efficace
V_{eff1}	Valeur efficace du fondamental

S_{11},S_{21},S_{31},S_{41}	Interrupteurs du bras 1 de l'onduleur à trois niveaux à structure NPC
S_{12},S_{22},S_{32},S_{42}	Interrupteurs du bras 2 de l'onduleur à trois niveaux à structure NPC
S_{13},S_{23},S_{33},S_{43}	Interrupteurs du bras 3 de l'onduleur à trois niveaux à structure NPC
V_{1o}, V_{2o}, V_{3o}	Tension entre bras et neutre de la source de l'onduleur à trois niveaux
V_{1n}, V_{2n}, V_{3n}	Tension entre bras et neutre de la charge de l'onduleur à trois niveaux
V_{1n}^*, V_{2n}^* *et* V_{3n}^*	Tensions de référence de l'onduleur à trois niveaux
V_d, V_q	Composants du vecteur tension de référence dans le plan *(d,q)*
$\hat{V}_d$, $\hat{V}_q$	V_d, V_q en valeur relative a V_{dc}
T_x,T_y,T_z	Durées d'application des vecteurs X,Y et Z
d_x,d_y,d_z	Durées relative d'application des vecteurs X,Y et Z
R	Résistance
L	Inductance
PWM_CCNTx	Registre de compteur de la chaine *x* du module PWM
PWM_CPRDx	Registre de période de la chaine *x* du module PWM
PWM_CDTYUPDx	Registre tampon du rapport cyclique de la chaine *x* du module PWM

Introduction générale

L'électronique de puissance est la technologie qui facilite la conversion d'énergie entre une source et une charge, et qui combine entre des techniques d'énergétique, d'électronique et de commande. A cause de la variété des sources d'énergies, et les exigences des applications modernes, cette conversion est essentielle pour assurer une énergie adéquate et efficace lors de son utilisation. Un système d'électronique de puissance consiste en un convertisseur et un contrôleur. Le convertisseur est un circuit composé de semi-conducteurs de grande puissance, d'un système de stockage d'énergie et un transformateur magnétique. Le processus de conversion commence lorsque le contrôleur, qui est un circuit numérique ou analogique de faible puissance, commande les semi-conducteurs du convertisseur, suivant une stratégie qui contrôle la stabilité et le flux d'énergie du système globale [10].

Parmi les formes de conversion la conversion continue-alternative (DC-AC) est la plus utilisées. Elle est assurée par deux grandes familles de convertisseurs : les onduleurs de tensions et les onduleurs de courant. Les onduleurs de tension constituent une fonction incontournable de l'électronique de puissance, présente dans les domaines d'applications les plus variés, dont le plus connu est sans doute celui de la variation de la vitesse des machines à courants alternatifs [2].

Ces convertisseurs de puissance DC/AC font toujours l'objet de recherche de plusieurs travaux. Toutes les approches utilisées pour leurs commandes ont des avantages et des inconvénients, reliées principalement à la génération d'harmoniques, la complexité, l'efficacité, la souplesse, la fiabilité, la sécurité et le cout. À l'heure actuelle, carrier based Modulation (CBM) et la modulation vectorielle (SVM) sont considérés comme les stratégies de modulation les plus courantes pour les convertisseurs de puissance [7], en raison de leur fonctionnement à fréquence fixe.

Dans le cas du CBM, la tension de sortie de chaque bras du convertisseur est déterminée par la comparaison d'une porteuse de forme de dents de scie, à une tension de référence sinusoïdale. Une méthode alternative à base de modulation est la modulation vectorielle (SVM), qui a été proposée dans de nombreuses publications [6], [13]. Cette méthode offre plusieurs avantages par rapport aux autres méthodes de modulation : génération de moins d'harmoniques et implémentation complète sur un calculateur numérique [3]. Un élément clé dans le contrôle d'un convertisseur de puissance par cette méthode est sa mise en œuvre numérique en temps réel, en utilisant soit des microprocesseurs, soit des circuits intégrés spécifiques.

C'est dans ce contexte, que s'inscrit notre travail, dont le but est d'implémenter la modulation vectorielle pour commander des onduleurs à deux niveaux et des onduleur à trois niveaux à structure NPC. Cette implémentation est faite sur le calculateur numérique *ARDUINO DUE*, qui est un microcontrôleur 32-bit de core ARM.

Notre travail est divisé en quatre parties :
Dans le premier chapitre, on va voir la carte de commande utilisée, ses caractéristiques, ainsi que l'environnement de développement intégrée (IDE) associé à son utilisation.
Dans le second chapitre, on va étudier la modulation vectorielle appliqué à des onduleurs à deux niveaux, et analyser sa performance.
Dans le troisième chapitre, on va étendre cette technique (SVM) aux onduleurs à trois niveaux à structure NPC.
Le dernier chapitre montre l'implémentation expérimentale de la SVM à deux et à trois niveaux par la carte *ARDUINO DUE*.
A la fin, une comparaison entre les résultats expérimentaux et les résultats de simulation, va nous permettre d'évaluer notre travail.

Description de l'environnement ARDUINO

1.1 Introduction

Arduino est un environnement de développement open-source [12], composé autour d'un circuit imprimée (figure 1.1), qui contient un microcontrôleur *(généralement un AVR)*, d'un langage de programmation *(C++)*, et d'un environnement de développement intégré *(IDE)* (figure 1.2) développé en *JAVA* autour de *PROCESSING (c'est une librairie java et un environnement de développement libre)*. Le matériel (HARDWARE) et le logiciel (SOFTWARE) Arduino sont Open Source. Le matériel consiste d'un circuit réalisé autour d'un microcontrôleur Atmel AVR de 8-bit avec une horloge de 16 MHz, un régulateur de tension 5 V et une interface USB. Il existe plusieurs versions d'ARDUINO disponibles. Les plus utilisées sont : Arduino Uno, Mega 2560. La dernière version est l'Arduino Due qui utilise un microcontrôleur 32-bit d'architecture ARM, et qui fonctionne a une fréquence d'horloge de 84 MHz, avec une mémoire de programme de 512KB [8].

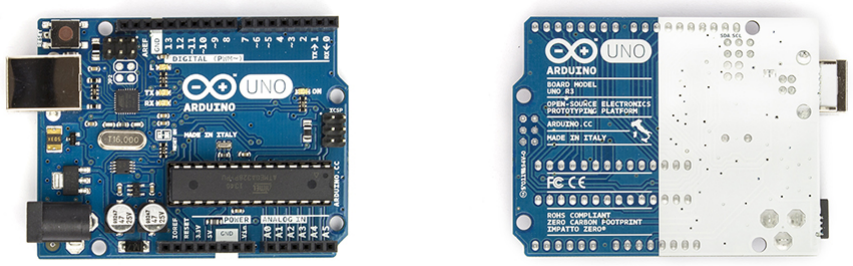


FIGURE 1.1: Circuit ARDUINO UNO [14]

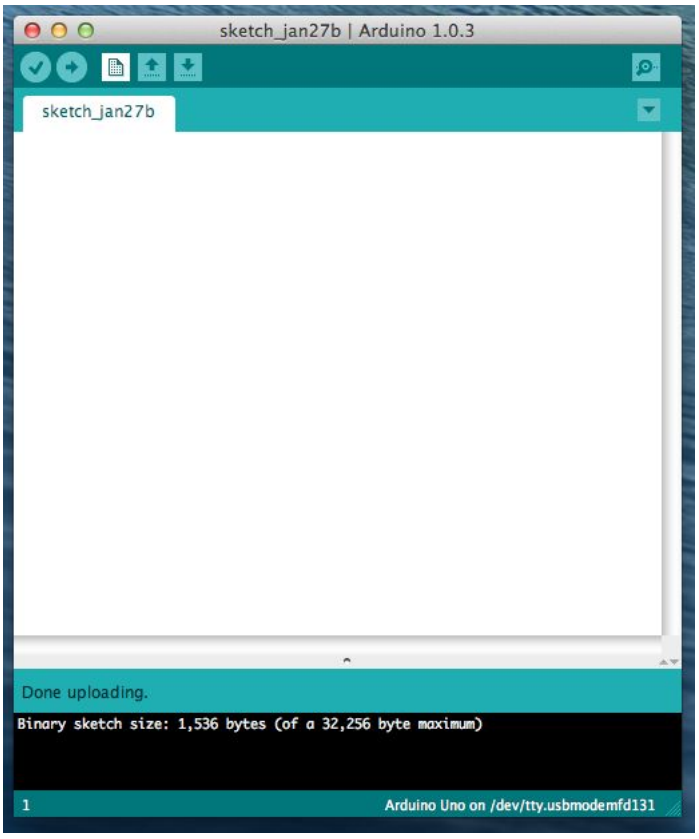


FIGURE 1.2: Environnement de développement intégré (ARDUINO IDE)

1.1.1 Architecture ARM

Les architectures ARM sont des architectures RISC 32 bits (ARMv1 à ARMv7) et 64 bits (ARMv8)1 développées par ARM Ltd depuis 1990 et introduites à partir de 1983 par Acorn Computers.

Dotés d'une architecture relativement plus simple que d'autres familles de processeurs, et bénéficiant d'une faible consommation, les processeurs ARM sont devenus dominants dans le domaine de l'informatique embarquée, en particulier la téléphonie mobile et les tablettes [17].

1.2 Avantages de l'environnement ARDUINO

Arduino offre aux utilisateurs les avantages suivants :

- Environnement multiplateforme : *WINDOWS*, *MAC* et *LINUX*.

- Environnement matériel et logiciel open-source et extensible.

- *IDE* gratuit.

- Environnement clair et simple pour les amateurs.

- Environnement très flexible pour les professionnels.

- Connexion simple avec le *PC* par *USB* pour charger le programme, communiquer et debugger.

- Des librairies disponibles avec des fonctions prédéfinies.

- Existence de *"SCHIELD"* : ceux sont des cartes qui se connectent directement sur le circuit *ARDUINO*, et qui offrent des fonctions HARDWARE supplémentaires. Exemple : *SD CARD*, *ETHERNET*, *XBEE*,*GSM* ... etc.

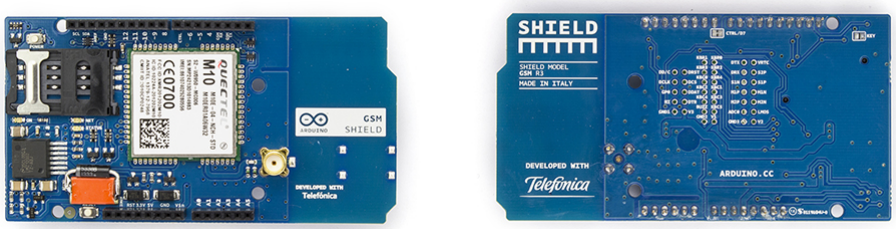


FIGURE 1.3: ARDUINO GSM SCHIELD [14]

1.3 ARDUINO HARDWARE

Il existe plusieurs versions des cartes ARDUINO. Les plus utilisées sont :

1- Arduino Uno : C'est la carte ARDUINO la plus utilisée (figure 1.1). Elle offre 6 entrées analogique et 14 broches d'entrées/sorties numérique, parmi lesquelles 6 peuvent être utilisées comme sortie de modulation a largeur d'impulsion *PWM*. Elle contient un régulateur de tension de 5 V et une interface de communication USB. La carte ARDUINO UNO fonctionne avec une fréquence d'horloge de 16 MHz. La carte contient aussi 6 convertisseurs analogique numérique (CAN) avec une résolution de 10 bit, et 6 sorties de modulation a largeur d'impulsion *PWM* avec une résolution de 8 bit.

2- Arduino Mega 2560 : La carte Arduino Mega 2560 est construite autour du microcontrôleur ATmega 2560 (figure 1.4). Elle possède 54 entrées/sorties numérique (parmi lesquelles 15 peuvent être utilisées comme sortie de modulation a largeur d'impulsion *PWM*), 16 entrées analogiques, 4 ports de communication série (UART), une horloge de 16 MHz et une interface de communication USB. La carte Arduino Mega 2560 est compatible avec la plupart des *shields* dédiées a Arduino [11].

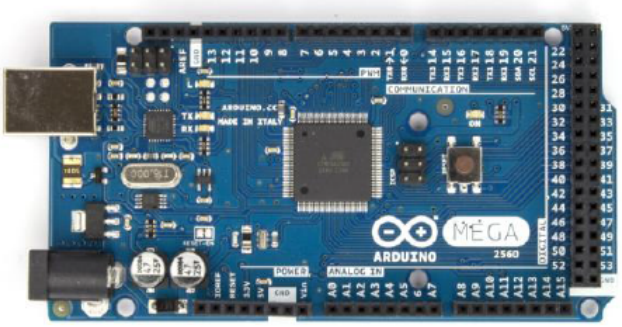


FIGURE 1.4: ARDUINO MEGA 2560 [14]

3- Arduino Due : C'est la dernière version des cartes ARDUINO. C'est la carte utilisée dans ce travail, et qui sera étudiée en détail dans la section suivante.

1.3.1 ARDUINO DUE HARDWARE

Arduino Due (figure 1.5) est une carte de la famille *ARDUINO*, réalisée autour du microcontrôleur *Atmel SAM3X8E ARM Cortex-M3 CPU*. C'est le premier *ARDUINO* réalisé autour d'un microcontrôleur 32-bit de core *ARM*. Elle possède 54 entrées/sorties numériques (dont 12 peuvent être utilisés comme des sorties de modulation a largeur d'impulsion), 12 entrées analogique, 4 module de communication *UART* (Universal Asynchronous Receiver Transmitter), une horloge de *84 MHz*, 2 convertisseur analogique numérique, 2 modules de communication *TWI*. Il peut être alimenté par *USB* ou par une source d'alimentation externe [1].

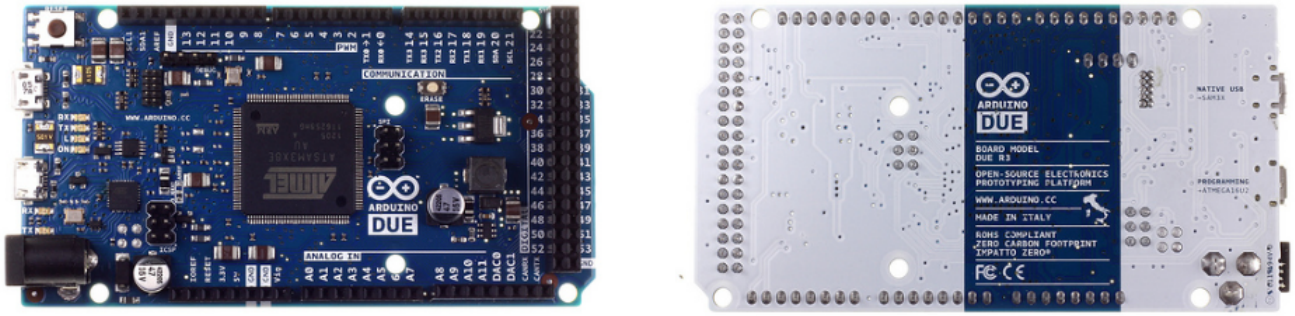


FIGURE 1.5: Carte ARDUINO DUE [14]

ARDUINO DUE possède deux PORT USB (figure 1.6) :

- **The Programming port :** il est connecté à un *ATmega16U2*, qui se comporte comme un *PORT COM* virtuel par rapport à l'ordinateur. Le microcontrôleur *ATMEGA 16U2* est aussi utilisé pour programmer l'*ARDUINO DUE* à travers les portes *RX0* et *TX0*. L'environnement *ARDUINO* contient un moniteur série pour le transfert des données textuelles depuis et vers le circuit.
- **The Native USB port :** il est connecté directement à *SAM3X*, et il permet la communication série à travers l'*USB*. Il est également utilisé comme un émulateur ou un hôte *USB* d'une souris ou d'un clavier [14].

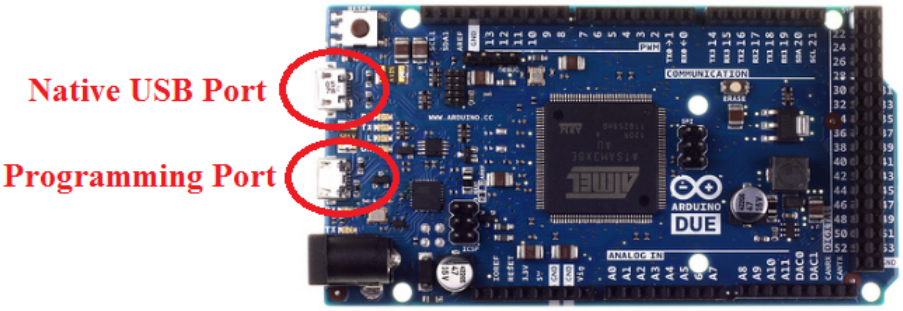


FIGURE 1.6: Ports USB de la carte ARDUINO DUE

1.3.2 Caractéristiques de la carte *ARDUINO DUE*

Les principaux caractéristiques de la carte *ARDUINO DUE* sont [1] :

- **Microcontrôleur :** *AT91SAM3X8E.*
- **Tension de fonctionnement :** *3.3V.*
- **Fréquence d'horloge :** *84 MHz.*
- **Broches d'entrée/sortie numérique :** *54.*
- **Broches de sorties PWM :** *12.*
- **Broches d'entrée analogique :** *12.*
- **Broches de sortie analogique :** *2 (CAN).*
- **Courants de sortie de chaque broche :** *130 mA.*
- **Mémoire Flash :** *512 KB disponible pour les applications d'utilisateurs.*
- **Mémoire SRAM :** *96 KB.*
- **Ports série UART :** *4.*
- **Ports USB :** *2.*
- **Temporisateur/Compteur :** *9.*
- **Module TWI :** *2.*
- **Module SPI :** *6.*
- **Module CAN :** *2.*
- **ETHERNET :** *1.*
- **Longueur :** *101.52 mm.*
- **Largeur :** *53.3 mm.*
- **Poids :** *36 g.*

1.4 L'environnement de développement intégré (ARDUINO IDE)

L'environnement de développement intégré *(IDE ARDUINO)* est une application multiplateforme développée en *JAVA* par les environnements *PROCESSING* et *WIRING*. Cet environnement inclue une bibliothèque appelée 'Wiring', qui facilite plusieurs opérations d'entrée sortie [8]. L'environnement de développement intégré (IDE) ARDUINO comporte :

- **Un éditeur de texte :** c'est un éditeur de code source *C/C++*, qui facilite le développement de programmes et qui offre de nombreuses caractéristiques comme la coloration syntaxique et la séparation automatique.
- **Un compilateur :** c'est le *GNU toolchain* et *AVR Libc*.
- **AVR Uploader Downloader (AVRDUDE) :** c'est un programme pour charger le programme exécutable depuis et vers la carte *ARDUINO*.
- **Zone de message :** pour visualiser l'état et les erreurs de compilation.
- **Console de texte :** pour visualiser le type de la carte *ARDUINO* et le port *COM* utilisée.
- **Une barre d'outils :** pour vérifier le code, compiler et charger le programme, créer un nouveau programme, ouvrir un programme existant et sauvegarder le programme courant.
- **Une barre de menu :** qui offre des différentes options comme le formatage automatique, la sélection du type de la carte et le port *COM*.
- **Un moniteur série :** permet de transférer et de visualiser des données textuelles depuis et vers la carte.

La figure suivante montre l'interface d'utilisateur [16] :

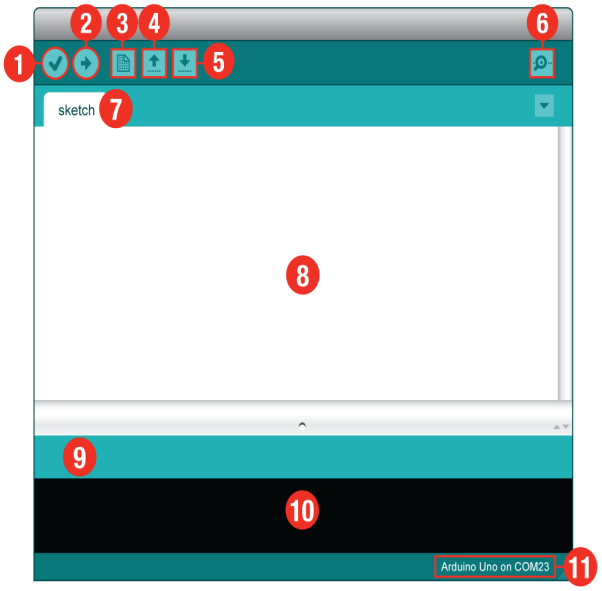


FIGURE 1.7: L'interface d'utilisateur ARDUINO [16]

1 - Vérifier (Verify) : Compiler et approuver le code. Il détecte les erreurs de syntaxes *(exemple : manque des points-virgules ou des parenthèses)*.

2 - Charger le programme (Upload) : Permet de charger le programme vers la carte ARDUINO cible.

3 - Nouveau (New) : Permet d'ouvrir une nouvelle fenêtre de code.

4 - Ouvrir (Open) : Permet d'ouvrir un programme existant.

5 - Enregistrer (Save) : Permet de sauvegarder le programme courant.

6 - Moniteur série (Serial Monitor) : Permet d'ouvrir une fenêtre de communication avec la Carte ARDUINO. Elle est très utile pour le *debugging*.

7 - Nom du programme (Sketch Name) : Permet de visualiser le nom du programme courant.

8 - Zone de code (Code Area) : Permet de visualiser et de construire le code du programme courant.

9 - Zone de messages (Message Area) : Permet de visualiser s'il existe des erreurs de code.

10 - Console de text (Text Console) : Permet de visualiser les messages d'erreurs.

10 - Type de carte et port série (Board and Serial Port) : Permet de visualiser le type de carte et le port série.

1.5 Le langage ARDUINO

Le langage *ARDUINO* est implémenté en *C/C++*, et basé sur *WIRING*. Un programme *ARDUINO* – appelé sketch - utilise implicitement la bibliothèque *WIRING*, qui est incluse dans l'*IDE*.
Un sketch *ARDUINO* utilise deux fonctions principales [14] :

– **SETUP ( ) :** c'est une fonction appelé une seule fois au début d'exécution. Elle est généralement utilisée pour initialiser les variables, définir la polarité des entrée-sorties et configurer les différents modules de la carte (initialisation des registres).

– **LOOP ( ) :** c'est une fonction appelé cycliquement et représente le *CORE* du programme. Elle contient la tache principale à exécuter par la carte.

Les principales fonctions incluses dans l'*Environnement de développement intégré (IDE) ARDUINO* sont :

– **pinMode (PIN, MODE) :** permet de configurer le port (*PIN*) comme une entrée (MODE=INPUT), ou une sortie (MODE=OUTPUT).

– **digitalRead (PIN) :** retourne l'état du port (*PIN*) : *HIGH* s'il est à 5 V ou *LOW* s'il est à 0 V.

– **digitalWrite (PIN,STATE) :** permet d'assigner au port (*PIN*) l'état *HIGH* 5 V ou l'état *LOW* (*0 V*).

– **analogRead (PIN) :** permet de convertir la tension analogique au port (*PIN*) sur un nombre définie de bits.

– **analogWrite (PIN, VALUE) :** permet de générer une onde *PWM* sur le port *PIN* avec un rapport cyclique de valeur *VALUE* par rapport à 255.

– **delay (ms) :** permet de garder l'état actuelle de la carte pendant un nombre de millisecondes passés comme argument a cette fonction.

– **attachInterrupt ( ) :** permet de définir une fonction a appelé (*ISR*), généré par une source d'interruption.

1.6 Exemples d'application

Avant d'utiliser la carte ARDUINO, on doit tout d'abord :

1- Sélectionner le type de la carte utilisée :

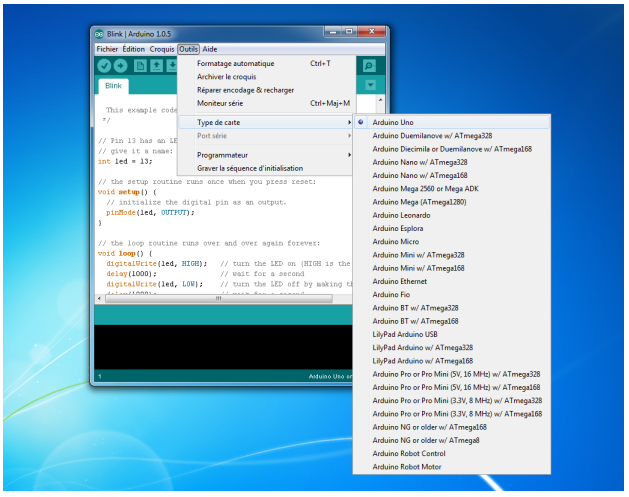


FIGURE 1.8: Sélection de la carte

2- Sélectionner le port COM :

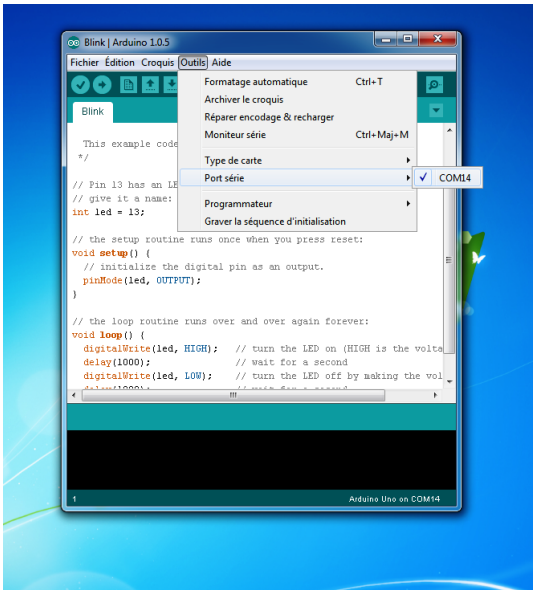


FIGURE 1.9: Sélection du port COM

3- Choix du programme :

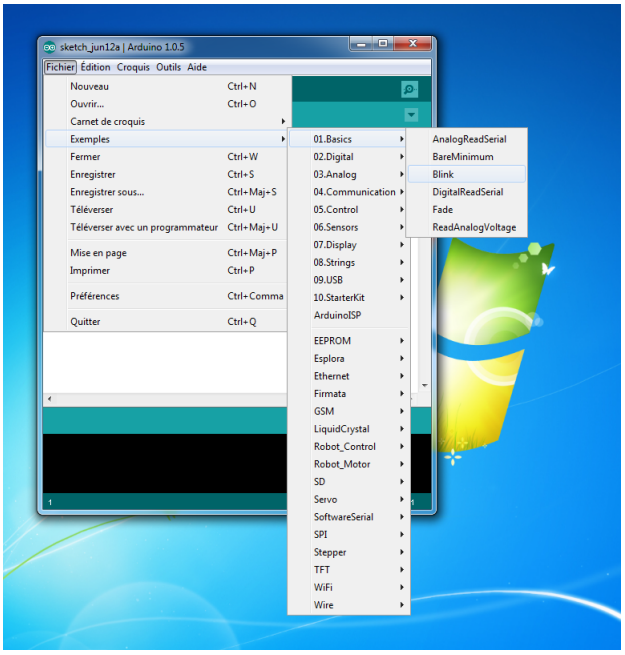


FIGURE 1.10: choix du programme

4- Chargement du programme vers la carte :

FIGURE 1.11: chargement du programme

ARDUINO IDE contient divers exemples. Pour expliquer l'utilisation de la language ARDUINO, on va choisit l'exemple de LED BLINK.

1.6.1 Exemple 1

Cet exemple à pour but de clignoter la LED orange, connectée au PIN 13 pour la plupart des cartes ARDUINO. Le programme complet est le suivant [14] :

```
int led = 13;
void setup() {
pinMode(led, OUTPUT);
}
void loop() {
digitalWrite(led, HIGH);
delay(1000);
digitalWrite(led, LOW);
delay(1000);
}
```

On va maintenant expliquer ce programme ligne par ligne :

- *int led = 13;* déclare une variable **led** qui porte le numéro du PIN 13.

- *void setup()* le programme de configuration qui s'exécute lorsque le bouton *reset* est pressé. Il permet de configurer les entrées/sorties de la carte.

- *pinMode(led, OUTPUT);* déclare le pin 13 (led) comme sortie.

- *void loop()* fonction principale qui s'exécute d'une façon cyclique.

- *digitalWrite(led,HIGH);* associe l'état HIGH (5 V) au pin 13, ce qui allume la led.

- *delay(1000);* maintenir l'état de la carte pendant une seconde.

- *digitalWrite(led,LOW);* associe l'état LOW (0 V) au pin 13, ce qui éteint la led.

- *delay(1000) ;* maintenir l'état de la carte pendant une seconde.

1.6.2 Exemple 2

Cet exemple à pour but de surveiller la température en indiquant son état normal par une led verte. une autre led rouge sera utiliser comme alarme. Le programme complet est le suivant :

```
int LED_rouge=10 ;
int LED_green=11 ;
int capteur_temp=0 ;
void setup() {
pinMode(LED_green,OUTPUT) ;
pinMode(LED_rouge,OUTPUT) ;
}
void loop() {
int temperature= analogRead(capteur_temp) ;
if ((temperature < 75) || (temperature> 125))
{
digitalWrite(LED_green, HIGH) ;
digitalWrite(LED_rouge, LOW) ;
}
else
{
digitalWrite(LED_green, LOW) ;
digitalWrite(LED_rouge, HIGH) ;
}
}
```

On va maintenant expliquer ce programme ligne par ligne :

- *int LED_rouge=10 ;* déclare une variable **LED_rouge** qui porte le numéro du PIN 10.

- *int LED_green=11 ;* déclare une variable **LED_green** qui porte le numéro du PIN 11.

- *int capteur_temp=0 ;* déclare une variable **capteur_temp** qui porte le numéro du PIN 0.

- *void setup()* le programme de configuration qui s'exécute lorsque le bouton *reset* est pressé. Il permet de configurer les entrées/sorties de la carte.

- *pinMode(LED_green,OUTPUT) ;* déclare le pin 11 (led vert) comme sortie.

- *pinMode(LED_rouge,OUTPUT) ;* déclare le pin 10 (led rouge) comme sortie.

- *void loop()* fonction principale qui s'exécute d'une façon cyclique.

- *int temperature=analogRead(capteur_temp) ;* lire la valeur analogique depuis le pin 0 relié au capteur de température, ce qui nous donne une valeur entre 0 (image de 0 V) et 1023 (image de 5 V).

- *if ((temperature < 75) || (temperature> 125))* vérifier l'intervalle du température.

- *{digitalWrite(LED_green, HIGH) ;digitalWrite(LED_rouge, LOW) ;}* mode normal : allumer la led vert et éteindre la led rouge.

- *else{digitalWrite(LED_green, LOW) ;digitalWrite(LED_rouge, HIGH) ;}* mode alarm : allumer la led rouge et éteindre la led vert.

1.7 Conclusion

Dans ce chapitre, on a vu l'environnement ARDUINO sur son coté matériel et logiciel.

On a vu que la carte ARDUINO DUE est une carte développé autour d'un microcontrôleur de CORE ARM, avec de très bonne fonctionnalités HARDWARE.
On a aussi vu que l'interface d'utilisation ARDUINO est une interface simple et très facile à utiliser.
La syntaxe de l'environnement ARDUINO est éventuellement claire et compréhensible.

Donc, on peut dire que l'environnement ARDUINO est un environnement compacte, qui combine entre la simplicité, la flexibilité et l'efficacité.

La modulation vectorielle à deux niveaux

2.1 Introduction

La modulation vectorielle (SVM) est une méthode avancée, utilisé par les convertisseurs de puissance DC/AC, qui délivre en sortie une tension de forme non-sinusoïdal, mais avec des propriétés fréquentiel très proches.
Cette méthode offre plusieurs avantages par rapport aux autres méthodes de modulation.

Tout d'abord, elle génère moins d'harmoniques avec une utilisation efficace du bus d'alimentation continue [4].
De plus, Elle facilite l'implémentation sur un circuit numérique, surtout avec le développement récent des microprocesseurs et l'accroissement de leurs puissances de calcul.
La modulation vectorielle (SVM) est, aussi, une méthode très flexible. Elle peut être adaptée aux convertisseurs DC/AC de structure multiniveaux. Malgré que la complexité de l'algorithme augmente avec l'augmentation de nombre des niveaux, La méthode peut être étendue pour des convertisseurs de niveaux supérieurs.

L'onduleur à deux niveaux est le premier modèle ou la SVM a été appliquée. C'est le modèle le plus simple et le plus ancien.

Dans ce chapitre on va voir la structure de l'onduleur à deux niveaux, son principe de fonctionnement, sa modélisation ainsi que l'algorithme de la SVM à deux niveaux.

2.2 La structure d'un onduleur à deux niveaux

L'onduleur de tension triphasé à deux niveaux est composé de trois bras. Chaque bras a deux interrupteurs bidirectionnels, qui fonctionnent de manière complémentaire pour éviter le court circuit de la source de tension continue à l'entrée de l'onduleur.
L'interrupteur bidirectionnel peut être réalisé par un transistor de puissance en antiparallèle avec une diode de puissance [9].

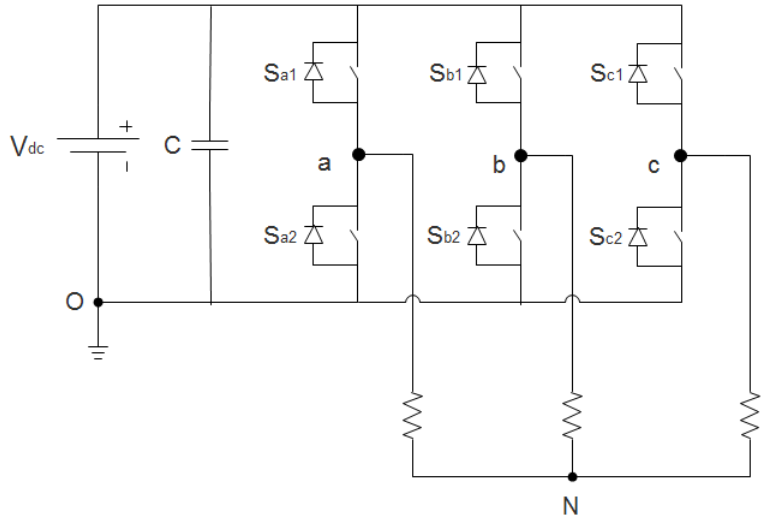


FIGURE 2.1: Structure d'un onduleur à deux niveaux

2.3 États d'un bras de l'onduleur

Chaque bras de l'onduleur a (comme son nom l'indique) deux états possibles :
- Etat 1 : L'interrupteur du haut $\mathbf{S_{x1}}$ **(x=a, b ou c)** est fermé, tandis que l'interrupteur du bas $\mathbf{S_{x2}}$ est ouvert.La tension du bras par rapport au neutre **(o)** de la source est $\mathbf{V_{dc}}$.
- Etat 0 : L'interrupteur du haut $\mathbf{S_{x1}}$ ($\mathbf{x = a, b\ \ ou\ \ c}$) est ouvert, tandis que l'interrupteur du bas $\mathbf{S_{x2}}$ est fermé.La tension du bras par rapport au neutre **(o)** de la source est **0 V**.

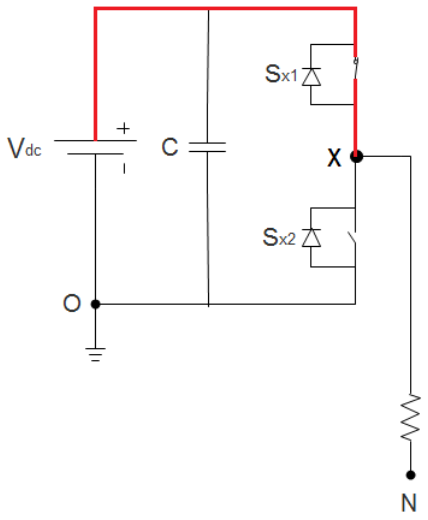


FIGURE 2.2: Etat 1 de l'onduleur à deux niveaux

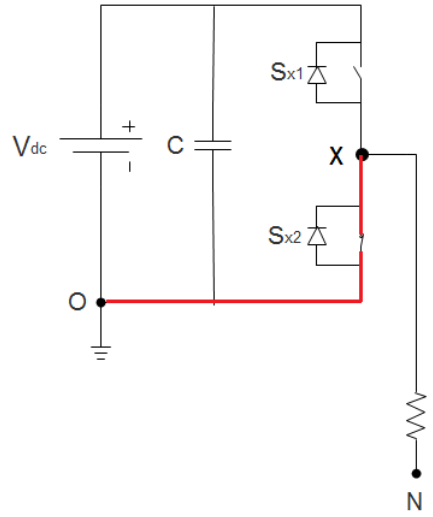


FIGURE 2.3: Etat 0 de l'onduleur à deux niveaux

Le tableau 2.1 résume les différents états d'un bras de l'onduleur à deux niveaux.

Etat	Bras a		Bras b		Bras b		Tension de sorties		
	S_{a1}	S_{a2}	S_{b1}	S_{b2}	S_{c1}	S_{c2}	V_{ao}	V_{bo}	V_{co}
1	1	0	1	0	1	0	$+V_{dc}$	$+V_{dc}$	$+V_{dc}$
0	0	1	0	1	0	1	0	0	0

TABLE 2.1: États d'un bras de l'onduleur à deux niveaux

2.4 États de l'onduleur à deux niveaux

Étant donné que chaque bras peut avoir deux états, l'onduleur entier possède $2^3 = 8$ états : 100,110,010,011,001,101,111,000.
Par exemple : l'état 100 indique que le bras de la phase **'a'** est a l'état 1,le bras de la phase **'b'** est a l'état 0 et le bras de la phase **'c'** est a l'état 0.
La configuration de l'onduleur pour chacune de ces états est représentée à la figure 2.4.

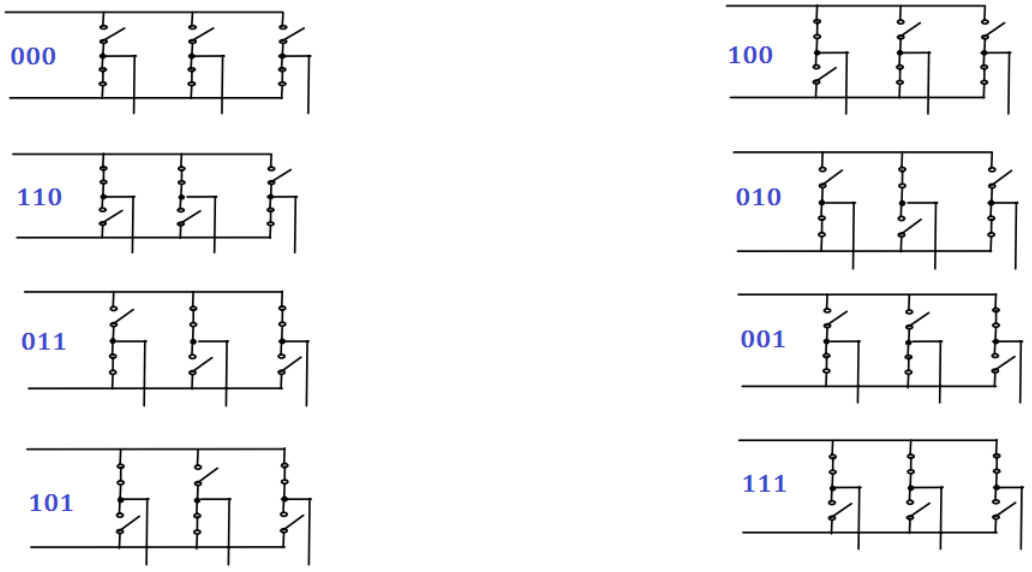


FIGURE 2.4: États d'un onduleur à deux niveaux

2.5 Modélisation de l'onduleur à deux niveaux

On définit les fonctions de commutations $\mathbf{S_a}$, $\mathbf{S_b}$ et $\mathbf{S_c}$ telle que :

- $\mathbf{S_a} = \mathbf{1}$ si l'interrupteur $\mathbf{S_{a1}}$ est fermé et l'interrupteur $\mathbf{S_{a2}}$ est ouvert.
- $\mathbf{S_a} = \mathbf{0}$ si l'interrupteur $\mathbf{S_{a2}}$ est fermé et l'interrupteur $\mathbf{S_{a1}}$ est ouvert.
- $\mathbf{S_b} = \mathbf{1}$ si l'interrupteur $\mathbf{S_{b1}}$ est fermé et l'interrupteur $\mathbf{S_{b2}}$ est ouvert.

– $\mathbf{S_b = 0}$ si l'interrupteur $\mathbf{S_{b2}}$ est fermé et l'interrupteur $\mathbf{S_{b1}}$ est ouvert.

– $\mathbf{S_c = 1}$ si l'interrupteur $\mathbf{S_{c1}}$ est fermé et l'interrupteur $\mathbf{S_{c2}}$ est ouvert.

– $\mathbf{S_c = 0}$ si l'interrupteur $\mathbf{S_{c2}}$ est fermé et l'interrupteur $\mathbf{S_{c1}}$ est ouvert.

On a ainsi :

– $\mathbf{V_{ao} = S_a.V_{dc}}$

– $\mathbf{V_{bo} = S_b.V_{dc}}$

– $\mathbf{V_{co} = S_c.V_{dc}}$

Avec : $\mathbf{V_{ao}}$, $\mathbf{V_{bo}}$ et $\mathbf{V_{co}}$ sont les tension entre les bras **a**, **b**, **c** et le neutre de la source **(o)**.

Soit $\mathbf{V_{an}}$,$\mathbf{V_{bn}}$ et $\mathbf{V_{cn}}$ les tensions entre les bras **a**, **b**, **c** et le neutre de la charge **(N)**.

Les tensions de phases $\mathbf{V_{an}}$, $\mathbf{V_{bn}}$ et $\mathbf{V_{cn}}$ sont liées aux tensions des trois bras de l'onduleur $\mathbf{V_{ao}}$, $\mathbf{V_{bo}}$ et $\mathbf{V_{co}}$ par :

– $\mathbf{V_{ao} = V_{an} + V_{no}}$

– $\mathbf{V_{bo} = S_{bn} + V_{no}}$

– $\mathbf{V_{co} = S_{cn} + V_{no}}$

Sachant que :

$$\mathbf{V_{an} + V_{bn} + V_{cn} = 0} \quad (\textbf{charge triphasée équilibrée}) \tag{2.1}$$

On déduit que :

$$\mathbf{V_{no}} = \frac{\mathbf{1}}{\mathbf{3}}.(\mathbf{V_{ao} + V_{bo} + V_{co}}) \tag{2.2}$$

Donc :

- $\mathbf{V_{an} = \frac{1}{3}(2.V_{ao} - V_{bo} - V_{co})}$
- $\mathbf{V_{bn} = \frac{1}{3}(V_{ao} - 2.V_{bo} - V_{co})}$
- $\mathbf{V_{cn} = \frac{1}{3}(V_{ao} - V_{bo} - 2.V_{co})}$

Finalement :

$$\begin{bmatrix} V_{an} \\ V_{bn} \\ V_{cn} \end{bmatrix} = V_{dc} \begin{bmatrix} \frac{2}{3} & \frac{1}{3} & \frac{1}{3} \\ -\frac{1}{3} & \frac{2}{3} & -\frac{1}{3} \\ -\frac{1}{3} & -\frac{1}{3} & \frac{2}{3} \end{bmatrix} \begin{bmatrix} S_a \\ S_b \\ S_c \end{bmatrix} \tag{2.3}$$

On remarque bien que les tensions de sortie de l'onduleur (V_{an}, V_{bn} *et* V_{cn}) peuvent prendre cinq niveaux de tension différents : $-\frac{2}{3}V_{dc}$, $-\frac{1}{3}V_{dc}$, 0, $\frac{1}{3}V_{dc}$ et $\frac{2}{3}V_{dc}$.

2.6 Transformation triphasée biphasée

Les grandeurs triphasées, dont la somme des composantes est nulle dans le référentiel fixe *(abc)*, peuvent être représentées par une grandeur biphasé dans un repère fixe (α,β), en utilisant la transformation qui maintient le module invariant [5] :

$$V_{\alpha}^{*} = \frac{2}{3}(V_{a}^{*} - \frac{1}{2}(V_{b}^{*} - V_{c}^{*})) \tag{2.4}$$

$$V_{\beta}^{*} = \frac{\sqrt{3}}{3}(V_{b}^{*} - V_{c}^{*}) \tag{2.5}$$

Cette transformation peut être mise sous une forme compacte en utilisant l'opérateur de rotation :

$$a = e^{j2\pi/3} = -\frac{1}{2} + j\frac{\sqrt{3}}{2} \tag{2.6}$$

$$\vec{V}_{ref} = \frac{2}{3}(V_a^* + aV_b^* + a^2V_c^*) \tag{2.7}$$

Si les références forment un système sinusoïdal triphasé équilibré, caractérisé par l'amplitude V_{ref} et la pulsation ω, au vecteur tension de référence $\vec{V}_{abc}^* = [\vec{V}_a^* \ \ \vec{V}_b^* \ \ \vec{V}_c^*]$ dans le référentiel fixe *(abc)*, correspond le vecteur $\vec{V}_{ref}$ dans le repère fixe (α,β), d'amplitude constante et tournant a la vitesse angulaire ω.
Cette transformation peut être appliquée aux huit états de l'onduleur à deux niveaux, ce qui conduit à huit vecteurs $\vec{V}_1$ a $\vec{V}_8$ dans le repère (α,β) (figure 2.2).

Vecteurs	$[S_a \ S_b \ S_c]$	$V_{an} \ V_{bn} \ V_{cn}$	$\vec{V}_{ref}$	θ
V_1	[1 0 0]	$\frac{2V_{dc}}{3} \quad -\frac{V_{dc}}{3} \quad -\frac{V_{dc}}{3}$	$\frac{2V_{dc}}{3}$	0
V_2	[1 1 0]	$\frac{V_{dc}}{3} \quad \frac{V_{dc}}{3} \quad -\frac{2V_{dc}}{3}$	$\frac{2V_{dc}}{3}$	$\frac{\pi}{3}$
V_3	[0 1 0]	$-\frac{V_{dc}}{3} \quad \frac{2V_{dc}}{3} \quad -\frac{V_{dc}}{3}$	$\frac{2V_{dc}}{3}$	$\frac{2\pi}{3}$
V_4	[0 1 1]	$-\frac{2V_{dc}}{3} \quad \frac{V_{dc}}{3} \quad \frac{V_{dc}}{3}$	$\frac{2V_{dc}}{3}$	π
V_5	[0 0 1]	$-\frac{V_{dc}}{3} \quad -\frac{V_{dc}}{3} \quad \frac{2V_{dc}}{3}$	$\frac{2V_{dc}}{3}$	$\frac{-4\pi}{3}$
V_6	[1 0 1]	$\frac{V_{dc}}{3} \quad -\frac{2V_{dc}}{3} \quad \frac{V_{dc}}{3}$	$\frac{2V_{dc}}{3}$	$\frac{-5\pi}{3}$
V_7	[0 0 0]	0 0 0	0	-
V_8	[1 1 1]	0 0 0	0	-

TABLE 2.2: Vecteurs de l'onduleur à deux niveaux dans le repère (α,β)

Les extrémités des vecteurs $\vec{V}_1$ a $\vec{V}_6$ forment les sommets d'un hexagone, et les 2 vecteurs $\vec{V}_7$ et $\vec{V}_8$ forment l'origine du repère (α,β) (figure 2.5).

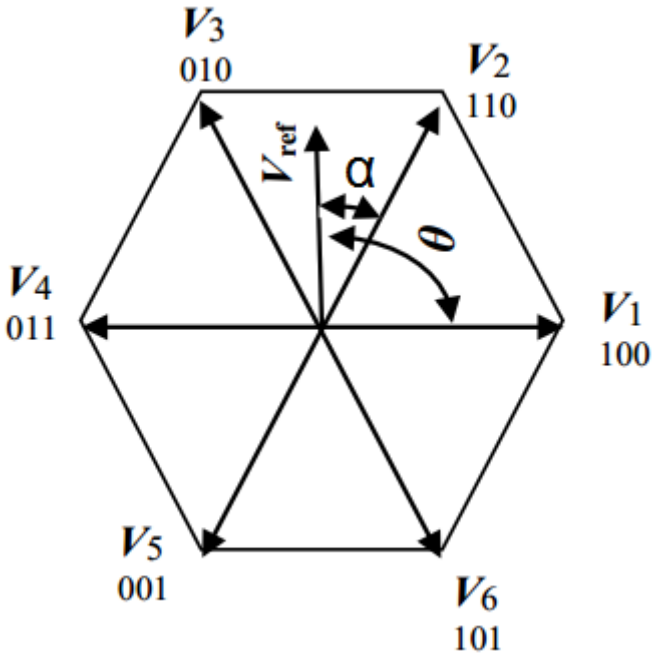


FIGURE 2.5: Diagramme vectorielle de l'onduleur à deux niveaux [5]

L'objectif de la modulation vectorielle est d'approximer le vecteur de référence $\vec{V}_{ref}$ par les 8 vecteurs ($\vec{V}_i$,i=1,8) disponibles pour l'onduleur à 2 niveaux. Cependant, si le vecteur $\vec{V}_{ref}$ est situé dans l'un des 6 secteurs de l'hexagone on n'utilise que les 2 vecteurs les plus proches et le vecteur zéro ($\vec{V}_z = \vec{V}_7$ *ou* $\vec{V}_8$). Pour que la valeur moyenne de la tension de référence ($\vec{V}_{ref}$) sur une période de découpage T_d soit la même que celle due a l'application des tensions $\vec{V}_i$, $\vec{V}_{i+1}$ et $\vec{V}_z$ pendant les durées T_i, T_{i+1} et T_z respectivement, l'équation vectorielle suivante doit être vérifiée :

$$\vec{V}_{ref}.T_s = \vec{V}_i.T_i + \vec{V}_{i+1}.T_{i+1} + \vec{V}_z.T_z \tag{2.8}$$

Comme le vecteur $\vec{V}_z$ est nul, la projection de cette équation sur l'axe $\vec{V}_i$ puis sur l'axe $\vec{V}_{i+1}$ donne :

$$V_{ref}.T_s.cos(\alpha) = V.T_i + V.T_{i+1}.cos(\pi/3) \tag{2.9}$$

$$V_{ref}.T_s.cos(\pi/3 - \alpha) = V.T_i.cos(\pi/3) + V.T_{i+1} \tag{2.10}$$

Comme $cos(\pi/3) = \frac{1}{2}$ et $V = \frac{2}{3}V_{dc}$, Ces deux équations peuvent se mettre sous la forme matricielle :

$$\begin{bmatrix} 1 & \frac{1}{2} \\ \frac{1}{2} & 1 \end{bmatrix} \begin{bmatrix} T_i \\ T_{i+1} \end{bmatrix} = \frac{3}{2} \begin{bmatrix} \hat{V}_{ref}.T_s.cos\alpha \\ \hat{V}_{ref}.T_s.cos(\pi/3 - \alpha) \end{bmatrix} \tag{2.11}$$

La solution est donc :

$$\begin{bmatrix} T_i \\ T_{i+1} \end{bmatrix} = \frac{4}{3} \begin{bmatrix} 1 & -\frac{1}{2} \\ -\frac{1}{2} & 1 \end{bmatrix} \frac{3}{2} \begin{bmatrix} \hat{V}_{ref}.T_s.cos\alpha \\ \hat{V}_{ref}.T_s.cos(\pi/3 - \alpha) \end{bmatrix} \tag{2.12}$$

Ou encore :

$$T_i = 2.\hat{V}_{ref}.T_s(cos\alpha - \frac{1}{4}cos\alpha - \frac{\sqrt{3}}{4}sin\alpha) = \hat{V}_{ref}.T_s.\sqrt{3}.sin(\frac{\pi}{3} - \alpha) \tag{2.13}$$

$$T_{i+1} = 2.\hat{V}_{ref}.T_s(-\frac{1}{2}cos\alpha + \frac{1}{2}cos\alpha + \frac{\sqrt{3}}{2}sin\alpha) = \hat{V}_{ref}.T_s.\sqrt{3}.sin\alpha \tag{2.14}$$

Finalement la solution est la suivante :

$$T_i = \hat{V}_{ref}.T_s.\sqrt{3}.sin(\frac{\pi}{3} - \alpha) \tag{2.15}$$

$$T_{i+1} = \hat{V}_{ref}.T_s.\sqrt{3}.sin\alpha \tag{2.16}$$

Il faut noter que $\hat{V}_{ref}$ est exprimé en valeur relative à V_{dc}.
Si la durée $T_i + T_{i+1} < T_s$, la période est complétée par le vecteur zéro pendant le reste de la période donc la séquence (0 0 0) est appliquée pendant la durée T_z telle que :

$$T_z = T_s - (T_i + T_{i+1}) \tag{2.17}$$

2.7 Séquence de commutation des interrupteurs

Afin de minimiser le nombre de commutations des interrupteurs, et réduire les harmoniques de tension [9], on va arranger les impulsions et le temps de commutation comme indiqué suivant :

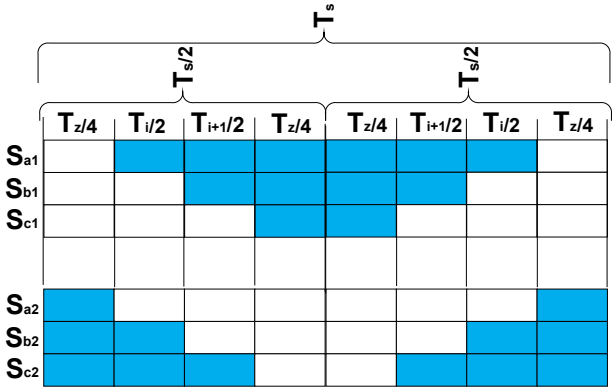


FIGURE 2.6: États des ordres de commande pour le secteur 1

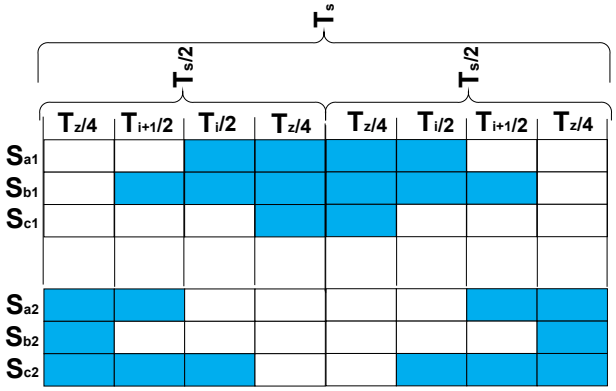


FIGURE 2.7: États des ordres de commande pour le secteur 2

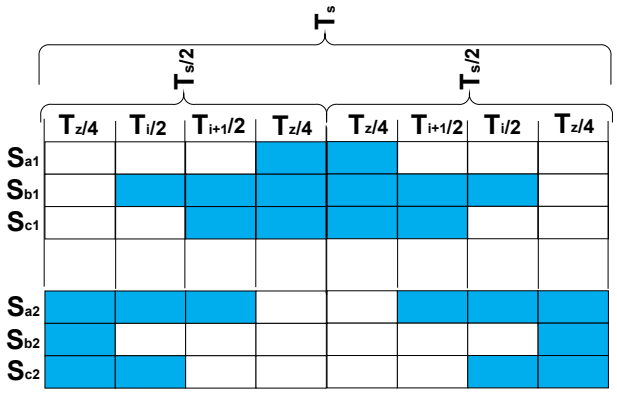


FIGURE 2.8: États des ordres de commande pour le secteur 3

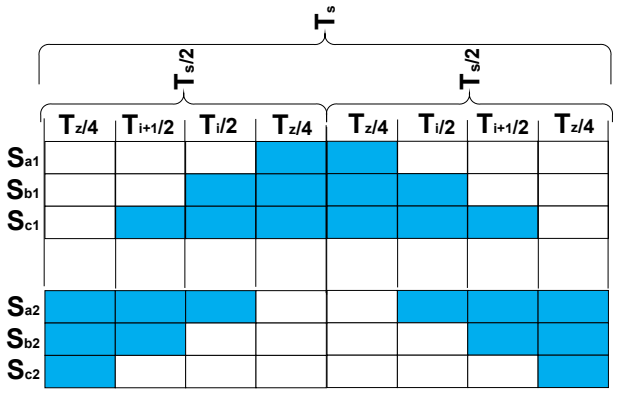


FIGURE 2.9: États des ordres de commande pour le secteur 4

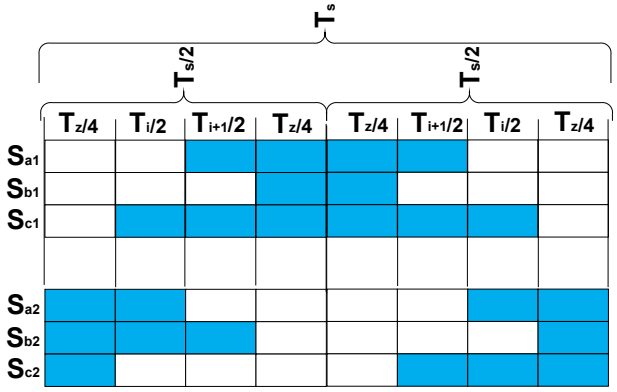


FIGURE 2.10: États des ordres de commande pour le secteur 5

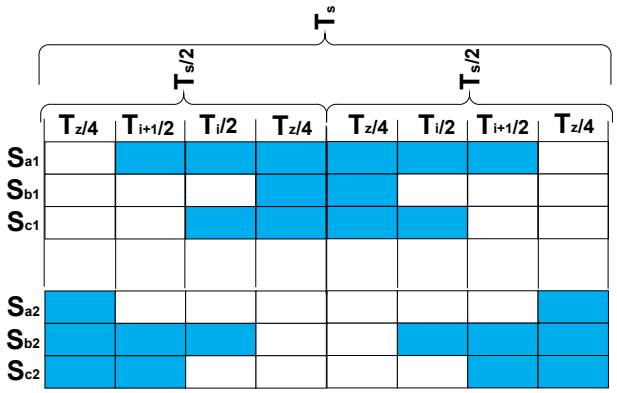


FIGURE 2.11: États des ordres de commande pour le secteur 6

On remarque que les impulsions des interrupteurs sont symétriques par rapport au demi-période. De plus, les vecteurs sont appliqués dans l'ordre :V_z, V_i, V_{i+1} si le secteur est impair *(1,3 et 5)*, et dans l'ordre :V_z, V_{i+1}, V_i si le secteur est pair *(2,4 et 6)*.

2.8 Simulation numérique

La simulation sous l'environnement Matlab/Simulink est un outil nécessaire pour réduire le temps, le risque et le cout d'étude d'un système d'électronique de puissance. Le modèle de simulation comporte deux parties principales (figure 2.12) :

1- Un bloc S-Function Builder : contient l'algorithme de la SVM à deux niveaux codé en C, et représente le modèle de la carte ARDUINO DUE.

2- Un bloc Matlab-Function : contient le modèle mathématique de l'onduleur à deux niveaux.

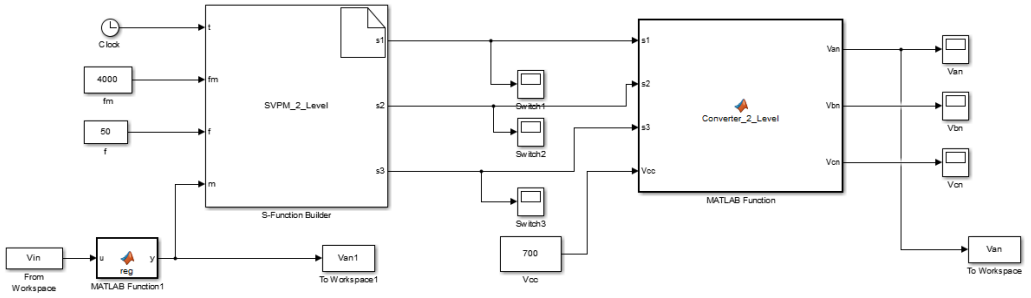


FIGURE 2.12: Modéle de simulation de la SVM à deux niveaux

2.8.1 Tensions de sortie

On va simuler les tensions de sortie pour un r=0.8 (r=$\frac{V_{ref}}{V_{dc}/2}$), une fréquence d'échantillonnage (F_e) de 4 KHz, et une tension d'alimentation a l'entrée de l'onduleur de 700 V.

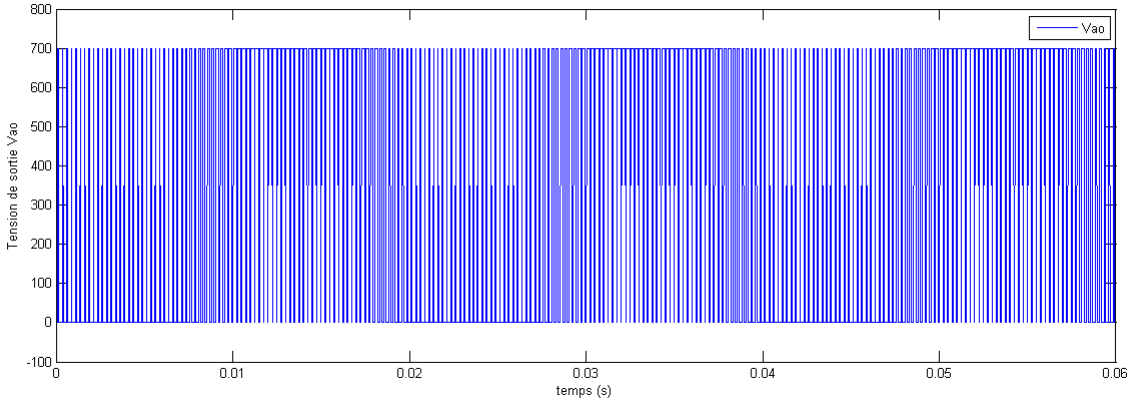


FIGURE 2.13: Tension V_{ao} ($r = 0.8$, $F_e = 4KHz$)

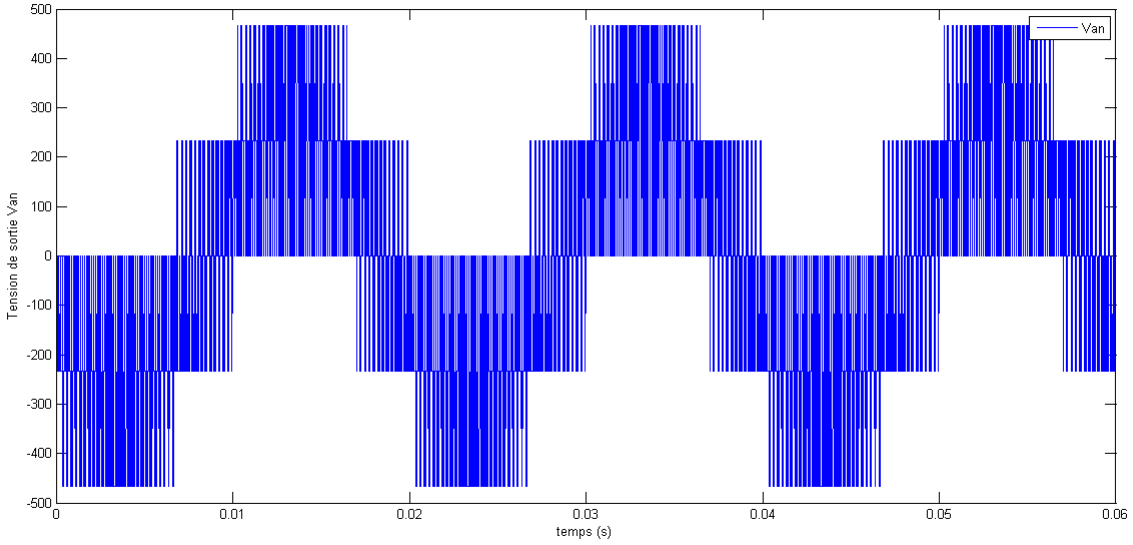


FIGURE 2.14: Tension V_{an} ($r = 0.8$, $F_e = 4KHz$)

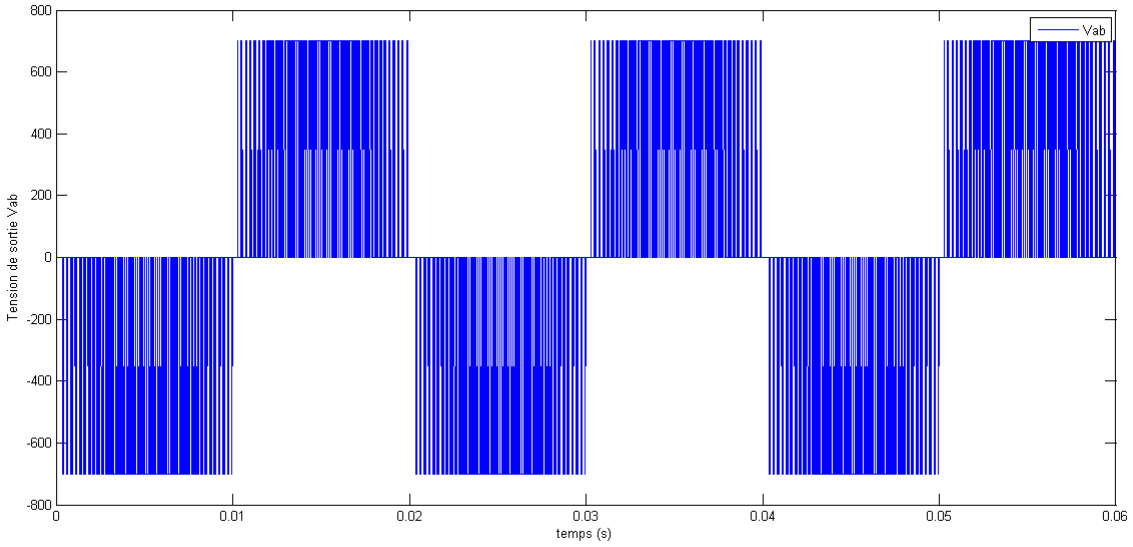


FIGURE 2.15: Tension V_{ab} ($r = 0.8$, $F_e = 4KHz$)

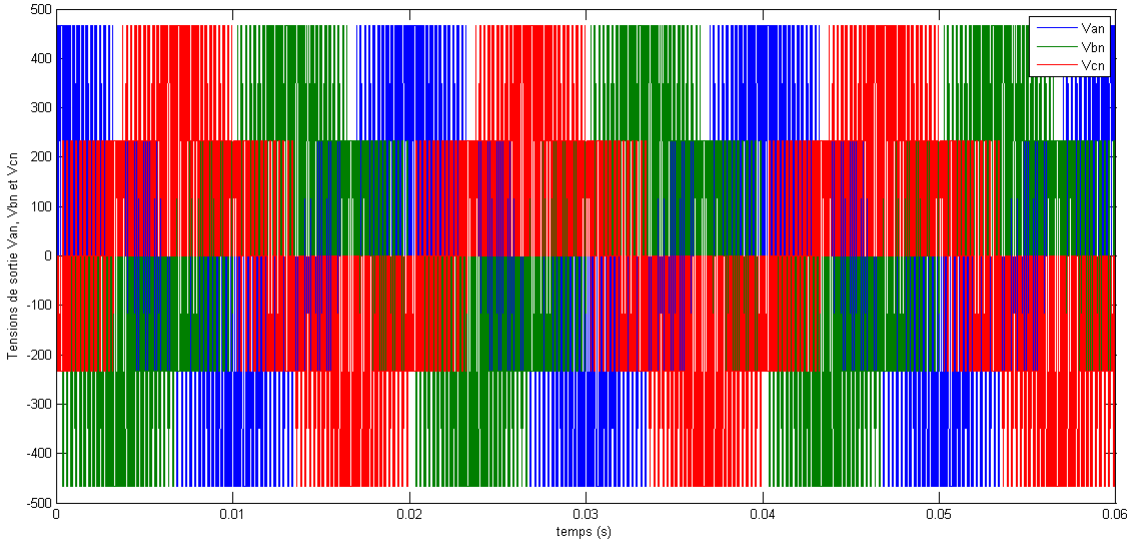


FIGURE 2.16: Tensions V_{an}, V_{bn} et V_{cn} ($r = 0.8$, $F_e = 4KHz$)

La figure 2.13 montre la tension V_{ao} entre le bras **'a'** et le neutre de la source d'alimentation **'o'**. On remarque que cette tension prend deux valeurs : 0 V ou 700 V (V_{dc}), d'où son appellation : "Onduleur à deux niveaux".

La figure 2.14 montre la tension V_{an} entre le bras **'a'** et le neutre de la charge **'N'**. On

remarque que cette tension prend cinq valeurs différentes : 467 V ($\frac{2V_{dc}}{3}$), 234 V ($\frac{V_{dc}}{3}$), 0 V, -234 V ($\frac{-V_{dc}}{3}$), -467 V ($\frac{-2V_{dc}}{3}$).

La figure 2.15 montre la tension V_{ab} entre le bras **'a'** et le bras **'b'**. On remarque que cette tension prend deux valeurs différentes : 700 V ($+V_{dc}$) ou -700 V ($-V_{dc}$).

La figure 2.16 montre les tensions simples des trois bras V_{an}, V_{bn} et V_{cn}. On remarque que ces trois tensions approximent la forme de trois tensions sinusoïdale, de fréquence 50 Hz et déphasées entre elles de $\frac{2\pi}{3}$ l'une a l'autre.

2.8.2 Analyse Harmonique

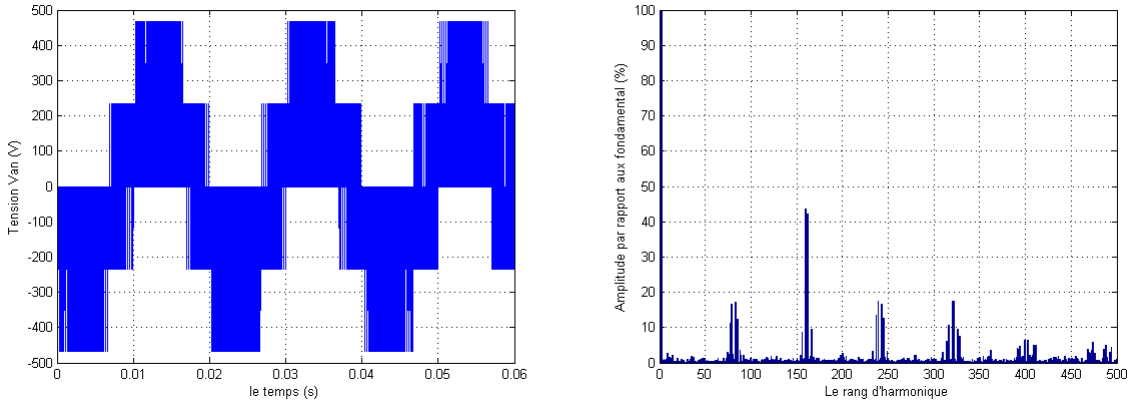


FIGURE 2.17: Spectre du tension V_{an} (r=0.8, F_e=4 KHz)

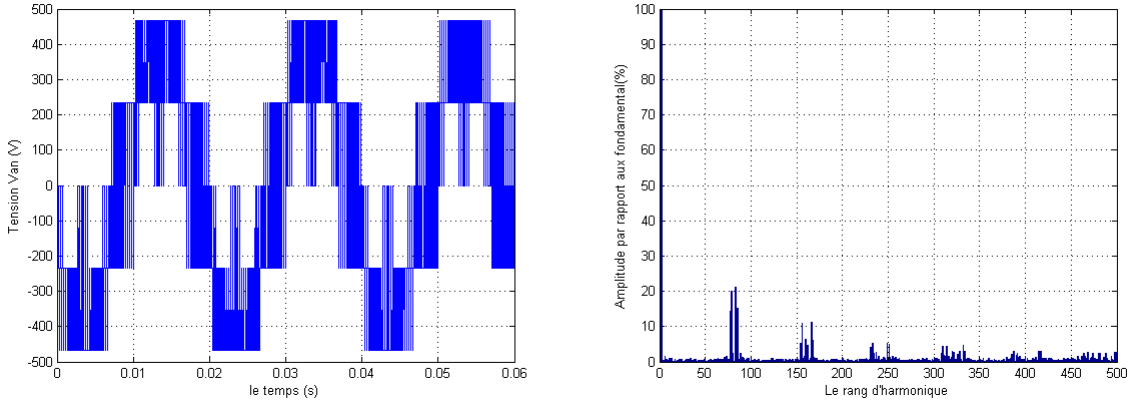


FIGURE 2.18: Spectre du tension V_{an} (r=1.2, F_e=4 KHz)

On remarque que les harmoniques de la tension simple V_{an} se regroupent en familles autour des multiples de rang 80, c'est-à-dire autour des multiples de la fréquence 80×50 Hz=4000 Hz, qui est la fréquence d'échantillonnage.
La figure 2.19 montre la variation du taux de distorsion d'harmonique (THD%) en fonction du taux de réglage r.
Le THD% est défini par :

$$THD\% = 100 \times \frac{\sqrt{(V_{eff}^2 - V_{eff1}^2)}}{V_{eff}}$$

Avec :
- V_{eff} valeur efficace de la tension de sortie.
- V_{eff1} valeur efficace du fondamental de la tension de sortie.

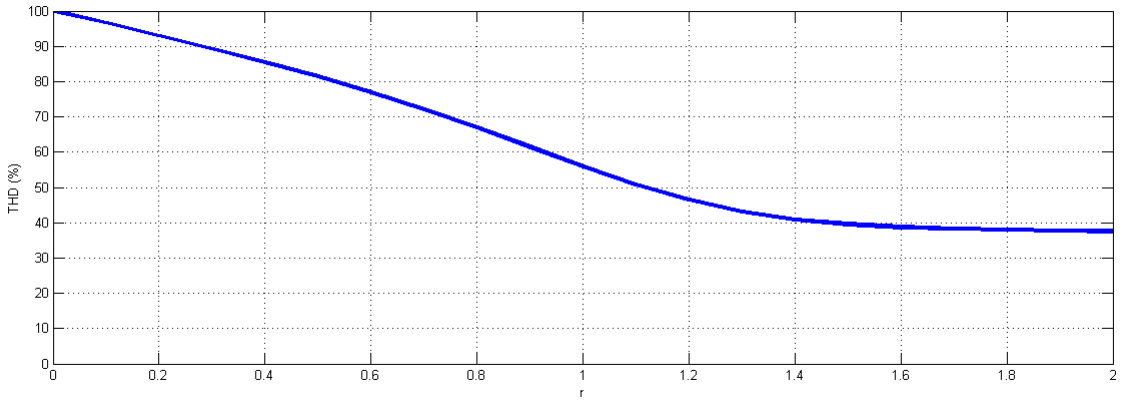


FIGURE 2.19: Taux de distorsion d'harmonique

On remarque que le taux de distorsion d'harmonique diminue avec l'augmentation de r.

2.8.3 Courbe de réglage

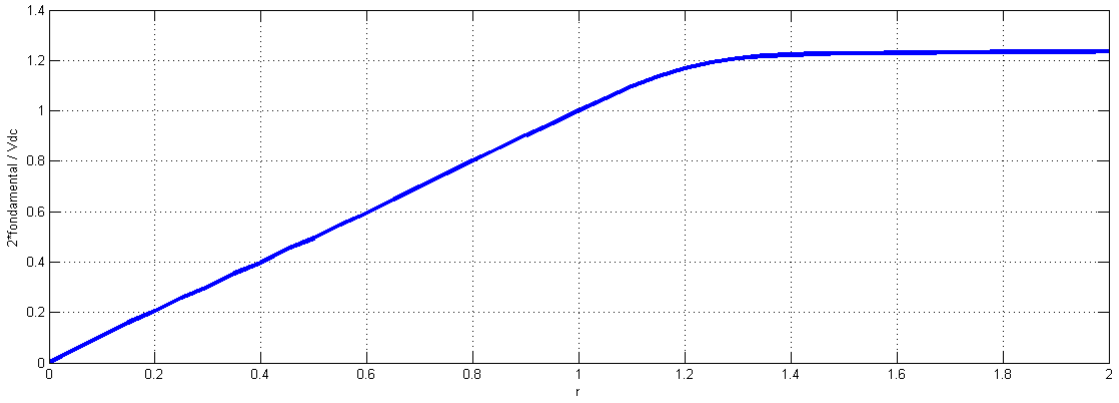


FIGURE 2.20: Courbe de réglage

On remarque que le fondamental de V_{1n} en valeur relative a $\frac{V_{dc}}{2}$ augmente linéairement avec l'augmentation de r.

On constate une saturation lorsque $r = \frac{2\sqrt{3}}{3} \approx 1.15$.

Ceci est du au fait que lorsque $r > \frac{2\sqrt{3}}{3}$, le vecteur tension de référence $\vec{V}_{ref}$ se situe en dehors du cercle délimité par l'hexagone du diagramme vectoriel de l'onduleur à deux niveaux (Figure 2.21).

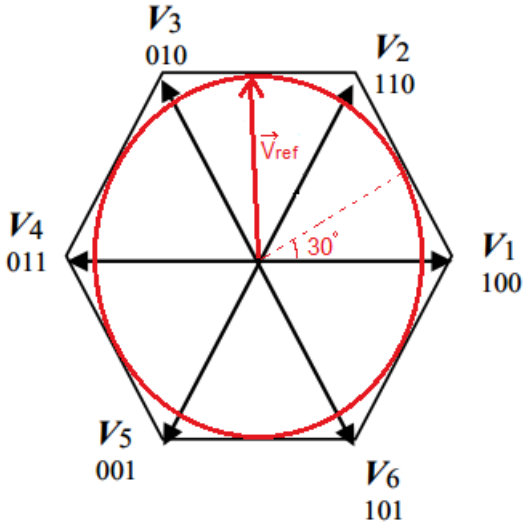


FIGURE 2.21: Trajectoire du $\vec{V}_{ref}$ lorsque $r = \frac{2\sqrt{3}}{3}$

2.8.4 Conclusion

Dans ce chapitre, on a vu la structure de l'onduleur à deux niveaux, son principe de fonctionnement, sa modélisation, ainsi que ses états dans le diagramme vectoriel.
On a vu que la modulation vectorielle à deux niveaux (SVM à deux niveaux) est une méthode qui consiste à approximer un vecteur tension de référence par les 8 vecteurs disponible pour l'onduleur à deux niveaux.
La simulation montre que cette méthode génère une tension simple qui approxime une référence sinusoïdale par cinq valeurs de tension différentes. L'amplitude de son fondamental est égale à celle du vecteur tension de référence jusqu'à la valeur $r = \frac{2\sqrt{3}}{3} \approx 1.15$.
Dans le chapitre suivant, on va étendre cette méthode aux onduleurs à trois niveaux à structure NPC.

La modulation vectorielle à trois niveaux

3.1 Introduction

On a vu dans le chapitre précédent que l'onduleur à deux niveaux est le plus simple dans sa structure et dans son algorithme de commande. Son inconvénient majeur est que la tension de chaque bras ne peut prendre que deux valeurs : 0 V ou V_{dc}. De plus la tension inverse de non-conduction de chaque interrupteur est V_{dc}, ce qui impose une limitation due à la technologie disponible.

Pour remédier à ces deux contraintes, les onduleurs à trois niveaux à structure NPC sont a présent les plus utilisés. Ils présentent plusieurs avantages : coût réduit, faible poids et taille compacte[4].

Dans le bus continu de l'onduleur à trois niveaux à structure NPC, le condensateur est divisé en deux, ce qui offre un point neutre. Les diodes connectées au point neutre sont dites diodes flottantes[4]. Elles connectent les bras de l'onduleur au point neutre, ce qui réduit la tension inverse de non-conduction de chaque interrupteur à $\frac{V_{dc}}{2}$. Chaque bras de l'onduleur peut prendre trois valeurs de tension : 0 V, $\frac{V_{dc}}{2}$ ou – $\frac{V_{dc}}{2}$, ce qui permet d'approximer au mieux la tension de sortie à la sinusoïde, et d'avoir une meilleure qualité d'énergie.

Dans ce chapitre, on va voir la structure de l'onduleur à trois niveaux à structure NPC, son principe de fonctionnement, sa modélisation ainsi que l'algorithme de la *SVM* à trois niveaux.

3.2 La topologie d'un onduleur à trois niveaux à structure NPC

L'onduleur à trois niveaux à structure NPC (Figure 3.1) est composé de trois bras, chaque bras ayant quatre interrupteurs bidirectionnels, réalisés par la mise en antiparallèle d'un transistor et une diode. Pour éviter le court-circuit de la source continue à l'entrée de l'onduleur, on doit éviter de fermer ou d'ouvrir simultanément les quatre interrupteurs d'un bras [9].

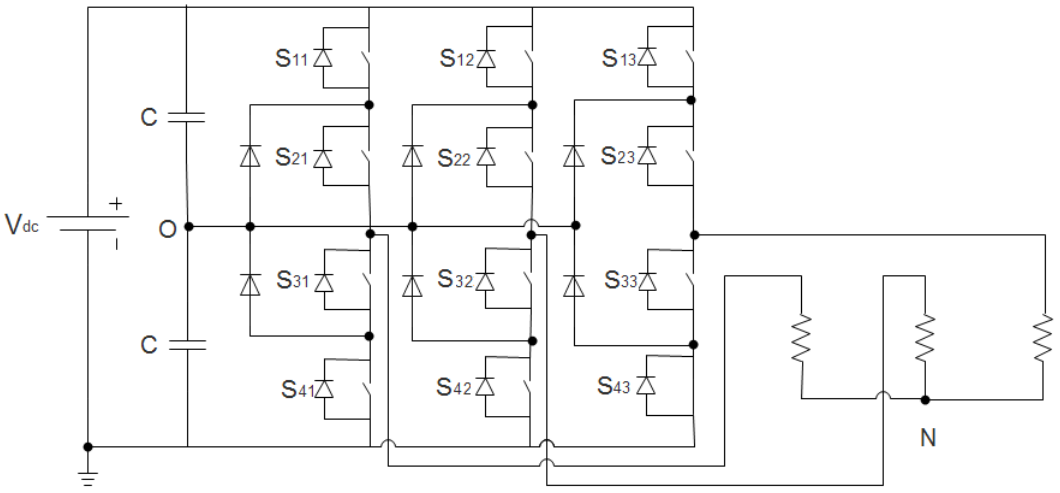


FIGURE 3.1: Topologie d'un onduleur à trois niveaux à structure NPC

On suppose que la tension V_{dc} est divisée en égalité entre les deux capacités :

$$U_{c1} = U_{c2} = \frac{V_{dc}}{2} \tag{3.1}$$

3.3 Fonctions de commutation

Pour chaque interrupteur S_{ij} *(i=1,2,3,4 ; j=1,2,3)*, ou j est le numéro du bras et i est le numéro d'interrupteur, on définit une fonction de commutation F_{ij} de la manière suivante [9] :

$$F_{ij} = \begin{cases} 1 \;\; si \;\; S_{ij} \;\; est \;\; fermé \\ 0 \;\; si \;\; S_{ij} \;\; est \;\; ouvert \end{cases} \tag{3.2}$$

Les interrupteurs de chaque bras sont complémentaires deux à deux :

$$F_{ij} = 1 - F_{(i-2)j} \qquad i=3,4\,;\, j=1,2,3 \tag{3.3}$$

3.4 États d'un bras de l'onduleur à trois niveaux à structure NPC

Chaque bras de l'onduleur a (comme son nom l'indique) trois états possibles :

- Etat P : Les deux interrupteurs du haut $\mathbf{S_{1x}}$ et $\mathbf{S_{2x}}$ **(x=1,2 ou 3)** sont fermés, tandis que les deux interrupteurs du bas $\mathbf{S_{3x}}$ et $\mathbf{S_{4x}}$ **(x=1,2 ou 3)** sont ouverts. La tension de sortie par rapport au neutre de la source **(o)** est $\frac{\mathbf{V_{dc}}}{\mathbf{2}}$.
- Etat O : Les deux interrupteurs du milieu $\mathbf{S_{2x}}$ et $\mathbf{S_{3x}}$ **(x=1,2 ou 3)** sont fermés, tandis que les deux interrupteurs des extrémités $\mathbf{S_{1x}}$ et $\mathbf{S_{4x}}$ **(x= 1,2 ou 3)** sont ouverts. La tension de sortie par rapport au neutre de la source **(o)** est **0** V.
- Etat N : Les deux interrupteurs du bas $\mathbf{S_{3x}}$ et $\mathbf{S_{4x}}$ **(x=1,2 ou 3)** sont fermés, tandis que les deux interrupteurs du haut $\mathbf{S_{1x}}$ et $\mathbf{S_{2x}}$ **(x= 1,2 ou 3)** sont ouverts. La tension de sortie par rapport au neutre de la source **(o)** est $\frac{-\mathbf{V_{dc}}}{\mathbf{2}}$.

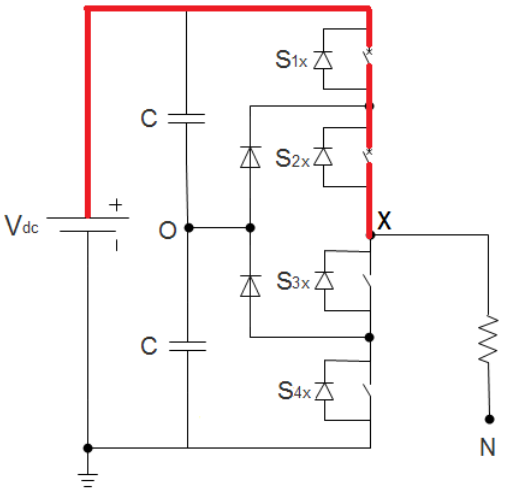


FIGURE 3.2: Etat P d'un bras de l'onduleur à trois niveaux à structure NPC

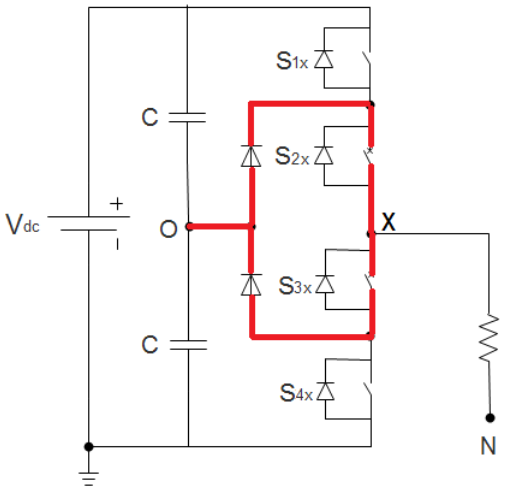


FIGURE 3.3: Etat O d'un bras de l'onduleur à trois niveaux à structure NPC

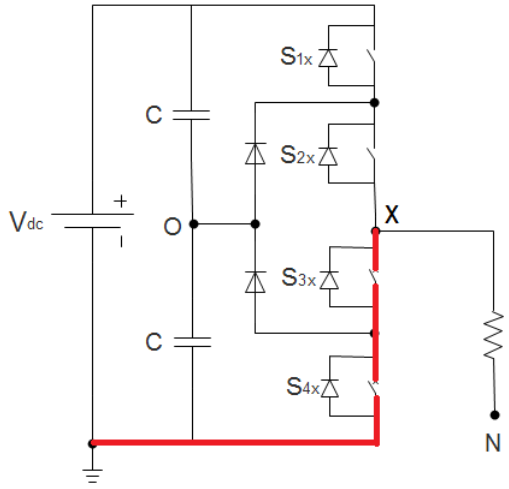


FIGURE 3.4: Etat N d'un bras de l'onduleur à trois niveaux à structure NPC

Le tableau 3.1 résume les différents états d'un bras de l'onduleur à trois niveaux à structure NPC.

Etat	S_{1x}	S_{2x}	S_{3x}	S_{4x}	Tension V_{xo}
P	1	1	0	0	$+V_{dc}/2$
O	0	1	1	0	0
N	0	0	1	1	$-V_{dc}/2$

TABLE 3.1: États d'un bras de l'onduleur à trois niveaux à structure NPC

3.5 Fonction de connection

On définit pour chaque bras *(j)* trois fonctions de connections, correspondantes au trois états du bras [9] :

$$\begin{cases} F_{c1j} = F_{1j}.F_{2j} \quad pour \quad l'état \quad P \\ F_{c2j} = F_{2j}.F_{3j} \quad pour \quad l'état \quad O \qquad j=1,2,3 \\ F_{c3j} = F_{3j}.F_{4j} \quad pour \quad l'état \quad N \end{cases} \tag{3.4}$$

3.6 États de l'onduleur a trois niveaux

Étant donné que chaque bras peut avoir trois états, l'onduleur entier possède $3^3 = 27$ états possibles :
PPP, POP, PNP, OPP, OOP, ONP, NPP, NOP, NNP, PPO, POO, PNO, OPO, OOO, ONO, NPO, NOO, NNO, PPN, PON, PNN, OPN, OON, ONN, OPN, NON et NNN.

Par exemple : l'état ONP indique que le bras de la phase **1** est a l'état O, le bras de la phase **2** est a l'état N et le bras de la phase **3** est a l'état P.
l'état POP indique que le bras de la phase **1** est a l'état P, le bras de la phase **2** est a l'état O et le bras de la phase **3** est a l'état P.

La configuration de l'onduleur pour chacune de ces états est représentée à la figure 3.5.

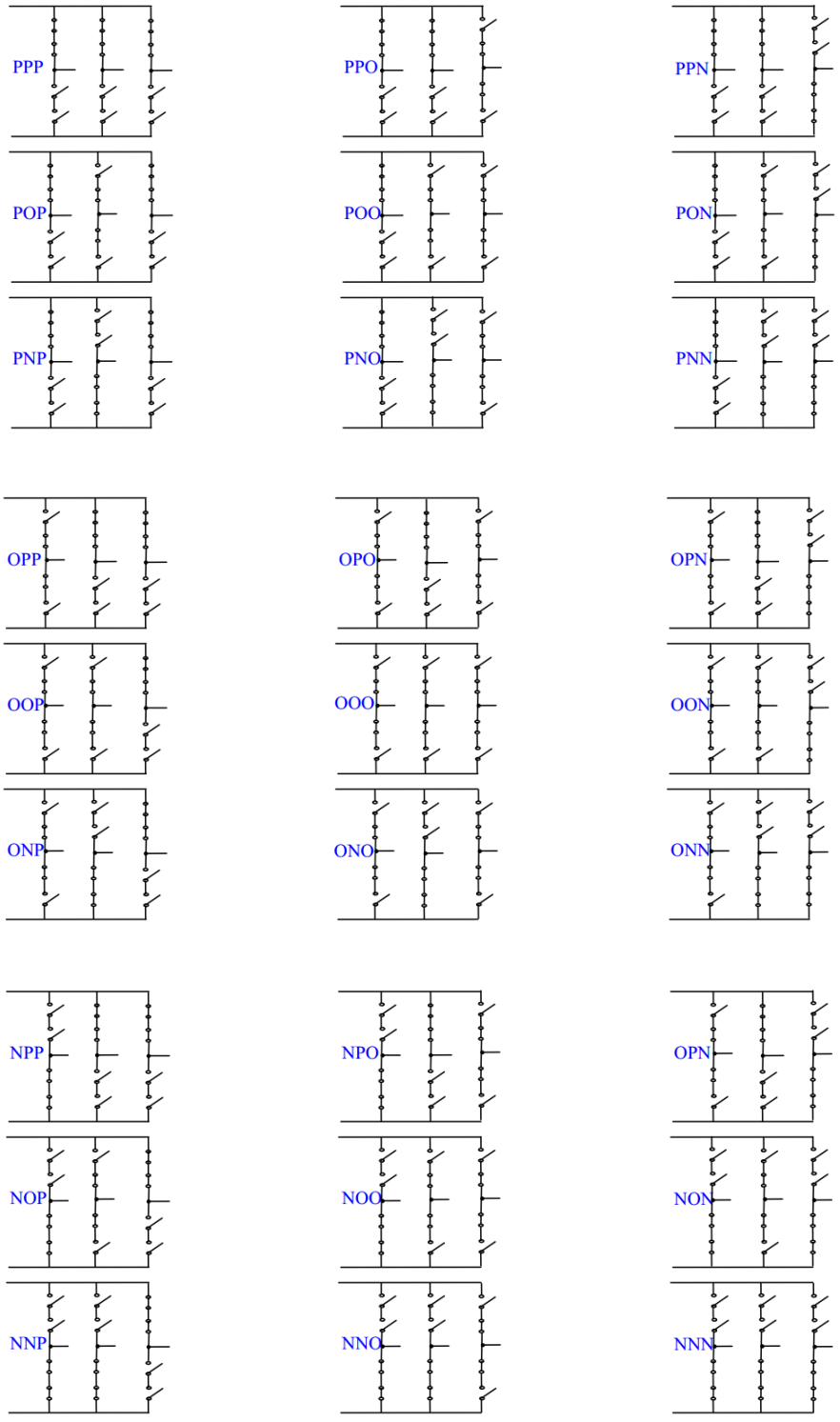


FIGURE 3.5: États d'un onduleur à trois niveaux

3.7 Modélisation de l'onduleur à trois niveaux à structure NPC

Les tensions de sortie par rapport au point neutre de la source continue **(o)** sont exprimées par [9] :

$$\begin{bmatrix} V_{1o} \\ V_{2o} \\ V_{3o} \end{bmatrix} = \begin{bmatrix} F_{c11} & F_{c21} & F_{c31} \\ F_{c12} & F_{c22} & F_{c32} \\ F_{c13} & F_{c23} & F_{c33} \end{bmatrix} \begin{bmatrix} V_{dc}/2 \\ 0 \\ -V_{dc}/2 \end{bmatrix} \tag{3.5}$$

A un instant donné, une seule des trois fonctions de connections des bras prend la valeur 1. Les autres fonctions sont a zéro. Ainsi, on peut avoir trois niveaux de tension pour chacune des tensions V_{1o}, V_{2o} et V_{3o}, ce qui est a l'origine de l'appellation : onduleur à trois niveaux. Les tensions composées entre les phases de la charge sont :

$$\begin{bmatrix} V_{12} \\ V_{23} \\ V_{31} \end{bmatrix} = \begin{bmatrix} V_{1o} - V_{2o} \\ V_{2o} - V_{3o} \\ V_{3o} - V_{1o} \end{bmatrix} = \begin{bmatrix} F_{c11} - F_{c12} & F_{c21} - F_{c22} & F_{c31} - F_{c32} \\ F_{c12} - F_{c13} & F_{c22} - F_{c23} & F_{c32} - F_{c33} \\ F_{c13} - F_{c11} & F_{c23} - F_{c21} & F_{c33} - F_{c31} \end{bmatrix} \begin{bmatrix} V_{dc}/2 \\ 0 \\ -V_{dc}/2 \end{bmatrix} \tag{3.6}$$

Dans le cas d'une charge équilibrée, les tensions de sortie par rapport au neutre de la charge sont exprimées par :

$$\begin{bmatrix} V_{1n} \\ V_{2n} \\ V_{3n} \end{bmatrix} = \frac{1}{3} \begin{bmatrix} V_{12} - V_{31} \\ V_{23} - V_{12} \\ V_{31} - V_{23} \end{bmatrix} = \frac{1}{3} \begin{bmatrix} 2F_{c11} - F_{c12} - F_{c13} & 2F_{c21} - F_{c22} - F_{c23} & 2F_{c31} - F_{c32} - F_{c33} \\ 2F_{c12} - F_{c11} - F_{c13} & 2F_{c22} - F_{c21} - F_{c23} & 2F_{c32} - F_{c31} - F_{c33} \\ 2F_{c13} - F_{c11} - F_{c12} & 2F_{c23} - F_{c21} - F_{c22} & 2F_{c33} - F_{c31} - F_{c32} \end{bmatrix} \begin{bmatrix} V_{dc}/2 \\ 0 \\ -V_{dc}/2 \end{bmatrix} \tag{3.7}$$

3.8 Vecteur tension de référence et diagramme vectoriel

Le vecteur tension de référence est défini par les équations :

$$\vec{V}_{ref} = V_{1n}^{*}.e^{j0} + V_{2n}^{*}.e^{-j2\pi/3} + V_{3n}^{*}.e^{j2\pi/3} = V_d + j.V_q \tag{3.8}$$

V_{1n}^{*}, V_{2n}^{*} et V_{3n}^{*} sont les tensions de référence des tensions de sorties V_{1n}, V_{2n} et V_{3n} respectivement.

V_d et V_q sont les composantes du vecteur $\vec{V}_{ref}$ dans le plan d-q.
Le tableau 3.2 résume la relation entre le vecteur tension de référence et les différents états possible de l'onduleur. On peut remarquer que le vecteur tension de référence génère dix-huit positions discrètes dans le plan d-q. Ces positions sont indiquées sur le diagramme vectoriel de l'onduleur (Figure 3.6).

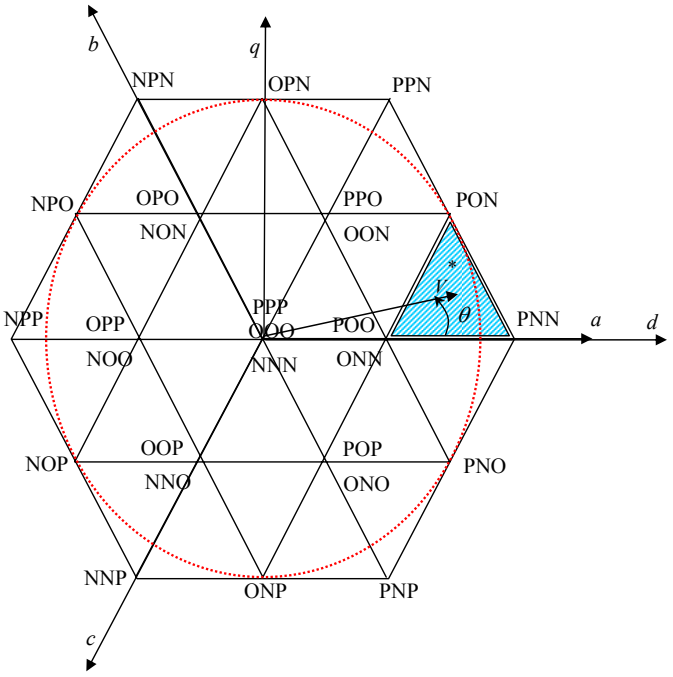


FIGURE 3.6: Diagramme vectoriel de l'onduleur à trois niveaux [9]

Les états de l'onduleur sont rangées sur deux hexagones : douze positions sur l'hexagone externe (correspondantes aux états PNN, PON, PPN, OPN, NPN, NPO, NPP, NOP, NNP, ONP, PNP et PNO), six positions sur l'hexagone interne (correspondantes aux états (POO-ONN), (PPO-OON), (OPO-NON), (OPP-NOO), (OOP-NNO) et (POP-ONO)), en plus d'une position à l'origine du diagramme, correspondante aux états PPP, OOO et NNN.
Chaque position sur l'hexagone interne est produite par deux états distincts. Ces états sont appelés états redondants, et les vecteurs correspondants sont appelés vecteurs redondants.
Les positions discrètes du vecteur tension de référence $\vec{V}_{ref}$ divise le diagramme vectoriel en six secteurs triangulaires, les sommets de ces secteurs triangulaires étant l'origine du diagramme vectoriel et les apex de l'hexagone externe. Chaque secteur est composé de quatre régions triangulaires. On a ainsi 24 régions dans la totalité du diagramme vectoriel[9].

Groupe	Etat	V_{ref}	θ	Tensions de phase		
				V_{1n}	V_{2n}	V_{3n}
0	[NNN]	0	0	0	0	0
	[OOO]					
	[PPP]					
1	[POO]	$V_{dc}/3$	0	$V_{dc}/3$	-$V_{dc}/6$	-$V_{dc}/6$
	[ONN]					
2	[PPO]	$V_{dc}/3$	$\pi/3$	$V_{dc}/6$	$V_{dc}/6$	-$V_{dc}/3$
	[OON]					
3	[OPO]	$V_{dc}/3$	$2\pi/3$	-$V_{dc}/6$	$V_{dc}/3$	-$V_{dc}/6$
	[NON]					
4	[OPP]	$V_{dc}/3$	π	-$V_{dc}/3$	$V_{dc}/6$	$V_{dc}/6$
	[NOO]					
5	[OOP]	$V_{dc}/3$	$4\pi/3$	-$V_{dc}/6$	-$V_{dc}/6$	$V_{dc}/3$
	[NNO]					
6	[POP]	$V_{dc}/3$	$5\pi/3$	$V_{dc}/6$	-$V_{dc}/3$	$V_{dc}/6$
	[ONO]					
7	[PON]	$\sqrt{3}V_{dc}/3$	$\pi/6$	$V_{dc}/2$	0	-$V_{dc}/2$
8	[OPN]	$\sqrt{3}V_{dc}/3$	$\pi/2$	0	$V_{dc}/2$	-$V_{dc}/2$
9	[NPO]	$\sqrt{3}V_{dc}/3$	$5\pi/6$	-$V_{dc}/2$	$V_{dc}/2$	0
10	[NOP]	$\sqrt{3}V_{dc}/3$	$7\pi/6$	-$V_{dc}/2$	0	$V_{dc}/2$
11	[ONP]	$\sqrt{3}V_{dc}/3$	$3\pi/2$	0	-$V_{dc}/2$	$V_{dc}/2$
12	[PNO]	$\sqrt{3}V_{dc}/3$	$3\pi/2$	$V_{dc}/2$	-$V_{dc}/2$	0
13	[PNN]	$2V_{dc}/3$	0	$2V_{dc}/3$	-$V_{dc}/3$	-$V_{dc}/3$
14	[PPN]	$2V_{dc}/3$	$\pi/3$	$V_{dc}/3$	$V_{dc}/3$	-$2V_{dc}/3$
15	[NPN]	$2V_{dc}/3$	$2\pi/3$	-$V_{dc}/3$	$2V_{dc}/3$	-$V_{dc}/3$
16	[NPP]	$2V_{dc}/3$	π	-$2V_{dc}/3$	$V_{dc}/3$	$V_{dc}/3$
17	[NNP]	$2V_{dc}/3$	$4\pi/3$	-$V_{dc}/3$	-$V_{dc}/3$	$2V_{dc}/3$
18	[PNP]	$2V_{dc}/3$	$5\pi/3$	$V_{dc}/3$	-$2V_{dc}/3$	$V_{dc}/3$

TABLE 3.2: Vecteur tension de référence pour l'onduleur à trois niveaux

3.9 Équations des régions de l'onduleur à trois niveaux

En se référent sur le premier secteur (Figure 3.7), les équations délimitant chaque région sont données par :

– **Pour la région 1 :**

$$\hat{V}_q + \sqrt{3}.\hat{V}_d - \frac{\sqrt{3}}{3} < 0 \tag{3.9}$$

– **Pour la région 2 :**

$$\hat{V}_q - \frac{\sqrt{3}}{6} < 0 \tag{3.10}$$

– **Pour la région 3 :**

$$\hat{V}_q - \sqrt{3}.\hat{V}_d + \frac{\sqrt{3}}{3} < 0 \tag{3.11}$$

– **Pour la région 4 :**

$$\hat{V}_q - \frac{\sqrt{3}}{6} > 0 \tag{3.12}$$

Notant que $\hat{V}_d$ et $\hat{V}_q$ sont exprimés en valeur relative a V_{dc}, c'est-a-dire qu'ils sont normalisés.

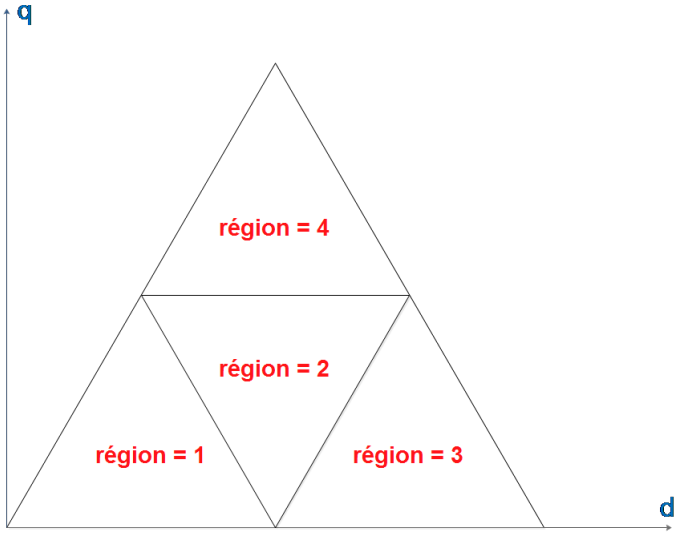


FIGURE 3.7: Régions du premier secteur

3.10 Séquence des états de l'onduleur

Sur une période d'échantillonnage T_s, le vecteur tension de référence $\vec{V}_{ref}$ doit coïncider avec la moyenne des vecteurs $\vec{V}_x$, $\vec{V}_y$ et $\vec{V}_z$ représentants les apex du triangle contenant $\vec{V}_{ref}$ [9] :

$$\vec{V}_{ref} = V_{ref}.e^{i\alpha} = r.\frac{V_{dc}}{2}.e^{i\alpha} = \frac{T_x.\vec{V}_x + T_y.\vec{V}_y + T_z.\vec{V}_z}{T_s} = d_x.\vec{V}_x + d_y.\vec{V}_y + d_z.\vec{V}_z \tag{3.13}$$

- α c'est la position angulaire du vecteur $\vec{V}_{ref}$ à l'intérieur d'un secteur : α=$\theta[\pi/3]$.
- V_{ref} c'est l'amplitude du vecteur tension de référence exprimé en valeur relative a V_{dc}.
- r c'est le taux de réglage.
- V_{dc} c'est la tension d'alimentation.
- T_x,T_y et T_z sont les durées d'application des vecteurs $\vec{v}_x$,$\vec{v}_y$ et $\vec{v}_z$ respectivement à la sortie de l'onduleur. Ils sont reliés par l'équation :

$$T_x + T_y + T_z = T_s \quad et \quad d_x + d_y + d_z = 1 \tag{3.14}$$

Les vecteurs $\vec{V}_x$,$\vec{V}_y$ et $\vec{V}_z$ dépendent de la position du vecteur $\vec{V}_{ref}$ dans le plan d-q. La Figure 3.8 indique les états X, Y et Z pour les différentes régions du diagramme vectoriel.

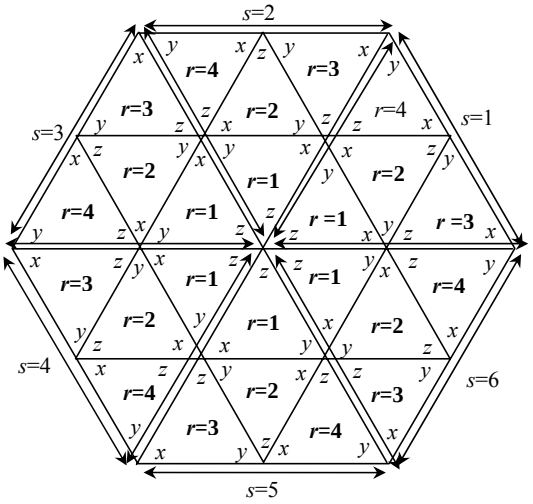


FIGURE 3.8: États X,Y et Z de l'onduleur à trois niveaux [9]

3.11 Durée d'application des états X,Y et Z

En se référent sur le premier secteur par le reste de la division entière de la phase (θ) du vecteur tension de référence ($\vec{V}_{ref}$) par $\pi/3$ (α=$\theta[\pi/3]$), et après projection de l'équation (3.13) sur les axes (d,q), on obtient les équations suivantes :

Région = 1 :

$$\begin{cases} \hat{V}_{ref}.cos\alpha = \frac{1}{3}.d_x + \frac{1}{6}.d_y \\ \hat{V}_{ref}.sin\alpha = \frac{\sqrt{3}}{6}.d_y \end{cases} \tag{3.15}$$

Région = 2 :

$$\begin{cases} \hat{V}_{ref}.cos\alpha = \frac{1}{6}.d_x + \frac{1}{3}.d_y + \frac{1}{2}.d_z \\ \hat{V}_{ref}.sin\alpha = \frac{\sqrt{3}}{6}.(d_x + d_y) \end{cases} \tag{3.16}$$

Région = 3 :

$$\begin{cases} \hat{V}_{ref}.cos\alpha = \frac{2}{3}.d_x + \frac{1}{2}.d_y + \frac{1}{3}.d_z \\ \hat{V}_{ref}.sin\alpha = \frac{\sqrt{3}}{6}.d_y \end{cases} \tag{3.17}$$

Région = 4 :

$$\begin{cases} \hat{V}_{ref}.cos\alpha = \frac{1}{2}.d_x + \frac{1}{3}.d_y + \frac{1}{6}.d_z \\ \hat{V}_{ref}.sin\alpha = \frac{\sqrt{3}}{6}.(d_x + d_z) + \frac{\sqrt{3}}{3}.d_y \end{cases} \tag{3.18}$$

La résolution du système formé par les équations précédentes, et la relation : d_x+d_y+d_z=1 , donne les expressions de d_x , d_y et d_z dans le tableau .

	région=1	région=2	région=3	région=4
d_x	$3\hat{V}_d$ - $\frac{d_y}{2}$	$\frac{3}{2}$ - $3\hat{V}_d$ - $\frac{d_y}{2}$	$3\hat{V}_d$ - 1 - $\frac{d_y}{2}$	$3\hat{V}_d$ - $\frac{1}{2}$ - $\frac{d_y}{2}$
d_y	$6\frac{\hat{V}_q}{\sqrt{3}}$	1 - $6\frac{\hat{V}_q}{\sqrt{3}}$	$6\frac{\hat{V}_q}{\sqrt{3}}$	$6\frac{\hat{V}_q}{\sqrt{3}}$ - 1
d_z	1-d_x-d_y			

TABLE 3.3: Durée d'application des états X,Y et Z

3.12 Ordre d'application des états X,Y et Z

Du tableau 3.2, on remarque que certains états X, Y et Z de l'onduleur à trois niveaux sont des états redondants, C'est-a-dire ils produisent les mêmes tensions de sortie V_{1n}, V_{2n} et V_{3n}. Dans ce cas, on peut choisir l'état redondant à appliquer pour minimiser le nombre de commutation des interrupteurs, ce qui conduit a la minimisation des pertes en puissance de l'onduleur [9].
De plus pour réduire les harmoniques de la tension de sortie, on choisit une séquence des états X, Y et Z qui consiste à :

- Appliquer les trois états dans un ordre donné durant la première demi période, puis dans l'ordre inverse durant le reste de la période.
- De diviser le temps approprié à chaque état sur toute les redondances.

Le tableau 3.4 résume la succession des états pendant la première demi-période d'échantillonnage pour toutes les régions du diagramme vectoriel. Ces états s'appliquent dans l'ordre inverse sur la $2^{éme}$ demi période.

Secteur	Région 1							Région 2					Région 3				Région 4			
1	N	O	O	O	P	P	P	O	O	P	P	P	O	P	P	P	O	P	P	P
	N	N	O	O	O	P	P	N	O	O	O	P	N	N	O	O	O	O	P	P
	N	N	N	O	O	O	P	N	N	N	O	O	N	N	N	O	N	N	N	O
2	P	P	O	O	O	N	N	P	O	O	O	N	P	P	O	O	O	O	N	N
	P	P	P	O	O	O	N	P	P	P	O	O	P	P	P	O	P	P	P	O
	P	O	O	O	N	N	N	O	O	N	N	N	O	N	N	N	O	N	N	N
3	N	N	N	O	O	O	P	N	N	N	O	O	N	N	N	O	N	N	N	O
	N	O	O	O	P	P	P	O	O	P	P	P	O	P	P	P	O	P	P	P
	N	N	O	O	O	P	P	N	O	O	O	P	N	N	O	O	O	O	P	P
4	P	O	O	O	N	N	N	O	O	N	N	N	O	N	N	N	O	N	N	N
	P	P	O	O	O	N	N	P	O	O	O	N	P	P	O	O	O	O	N	N
	P	P	P	O	O	O	N	P	P	P	O	O	P	P	P	O	P	P	P	O
5	N	N	O	O	O	P	P	N	O	O	O	P	N	N	O	O	O	O	P	P
	N	N	N	O	O	O	P	N	N	N	O	O	N	N	N	O	N	N	N	O
	N	O	O	O	P	P	P	O	O	P	P	P	O	P	P	P	O	P	P	P
6	P	P	P	O	O	O	N	P	P	P	O	O	P	P	P	O	P	P	P	O
	P	O	O	O	N	N	N	O	O	N	N	N	O	N	N	N	O	N	N	N
	P	P	O	O	O	N	N	P	O	O	O	N	P	P	O	O	O	O	N	N

TABLE 3.4: Séquence des états de l'onduleur à trois niveaux [9]

3.13 Simulation numérique

On va simuler le système onduleur à trois niveaux à structure NPC commandé par la SVM a 3 niveaux. Le modèle de simulation comporte deux parties principales :

1- Un bloc S-Function Builder : contient l'algorithme de la SVM à trois niveaux codé en C, et représente le modèle de la carte ARDUINO DUE.

2- Un bloc Matlab-Function : contient le modèle mathématique de l'onduleur à trois niveaux à structure NPC.

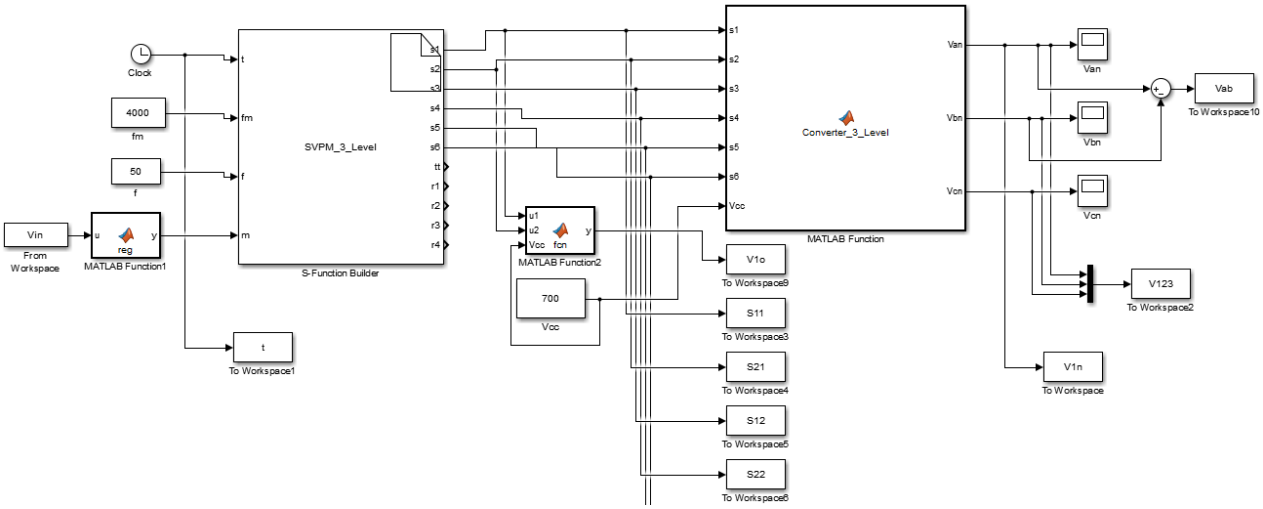


FIGURE 3.9: Modèle de simulation de la SVM à 3 niveaux

3.13.1 Tensions de sortie

On va simuler les tensions de sortie pour r=0.8, une fréquence d'échantillonnage de 4 KHz, et une tension d'alimentation à l'entrée de l'onduleur de 700 V.

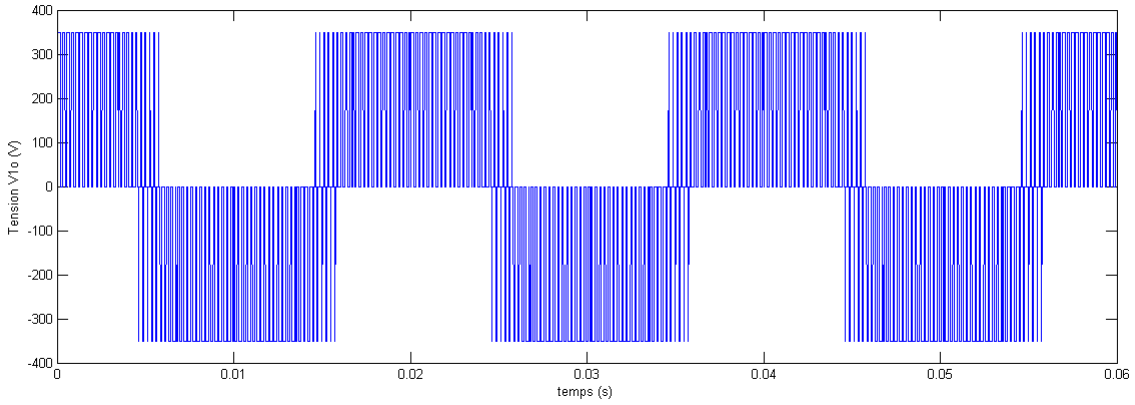


FIGURE 3.10: Tension V_{1o} (r=0.8, F_e=4 KHz)

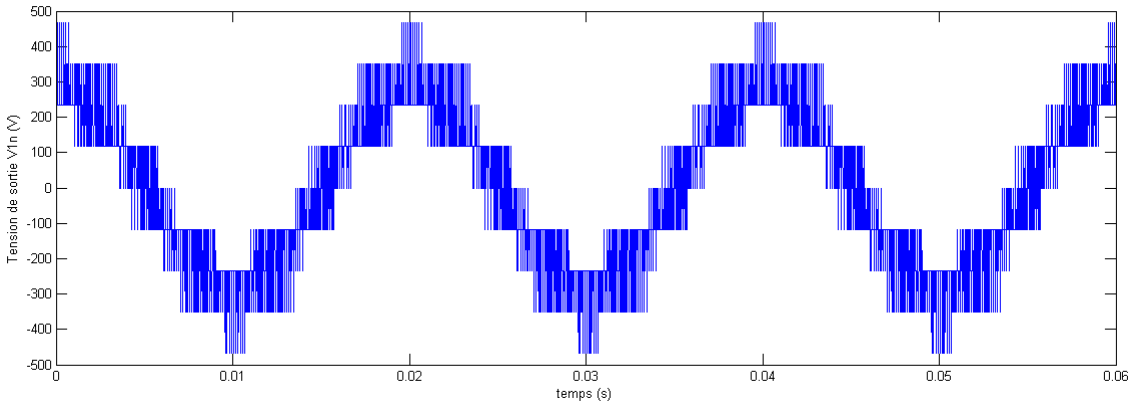


FIGURE 3.11: Tension V_{1n} (r=0.8, F_e=4 KHz)

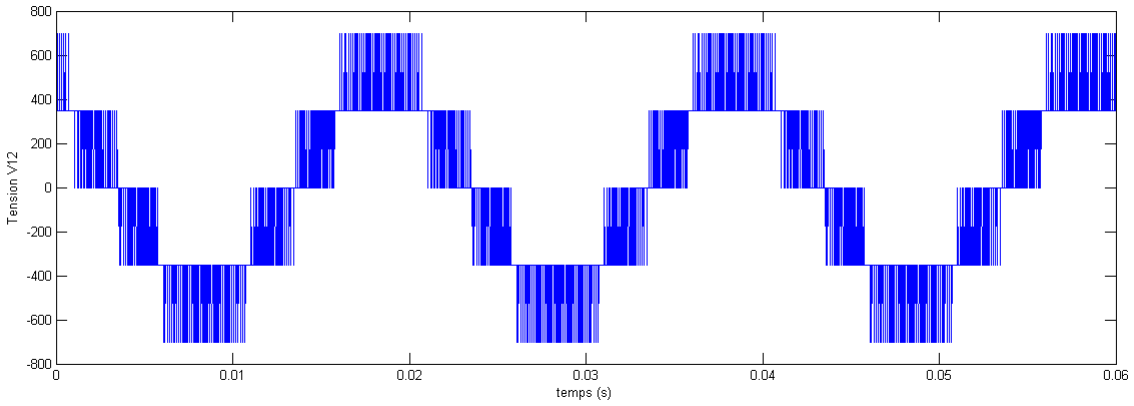


FIGURE 3.12: Tension V_{12} (r=0.8, F_e=4 KHz)

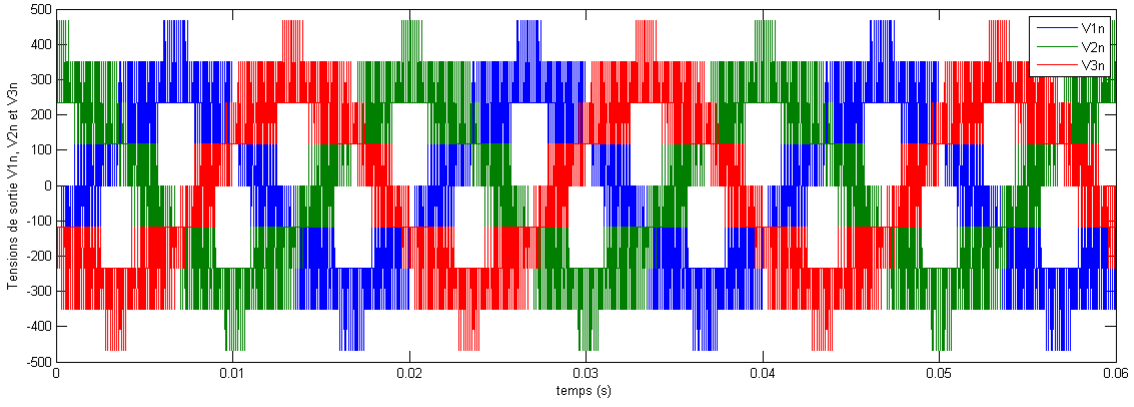


FIGURE 3.13: Tensions V_{1n}, V_{2n} et V_{3n} (r=0.8, F_e=4 KHz)

La figure 3.10 montre la tension V_{1o} entre le bras **'1'** et le neutre de la source d'alimentation **'o'**. On remarque que cette tension peut prendre trois valeurs différentes : -700 V (- V_{dc}), 0 V ou 700 V (+ V_{dc}), d'où son appellation : "Onduleur à trois niveaux".
La figure 3.11 montre la tension V_{1n} entre le bras **'1'** et le neutre de la charge **'N'**. On remarque que cette tension prend neuf valeurs différents : 467 V ($2V_{dc}/3$), 350 V ($V_{dc}/2$), 234 V ($V_{dc}/3$),167 V ($V_{dc}/6$),0 V, -167 V (- $V_{dc}/6$), - 234 V (- $V_{dc}/3$), - 350 V (- $V_{dc}/2$),- 467 V (- $2V_{dc}/3$).

La figure 3.12 montre la tension V_{12} entre le bras **'1'** et le bras **'2'**. On remarque que cette tension prend cinq valeurs différents : 700 V ($+V_{dc}$), 350 V ($+V_{dc}/2$), 0 V, -350 V ($-V_{dc}/2$),- 700 V ($-V_{dc}$).

La figure 3.13 montre les tensions simples des trois bras : V_{1n}, V_{2n} et V_{3n}. On remarque que ces trois tensions approximent la forme de trois tensions sinusoïdales, de fréquence 50 Hz et déphasés entre elles de $\frac{2\pi}{3}$.

3.13.2 Analyse Harmonique

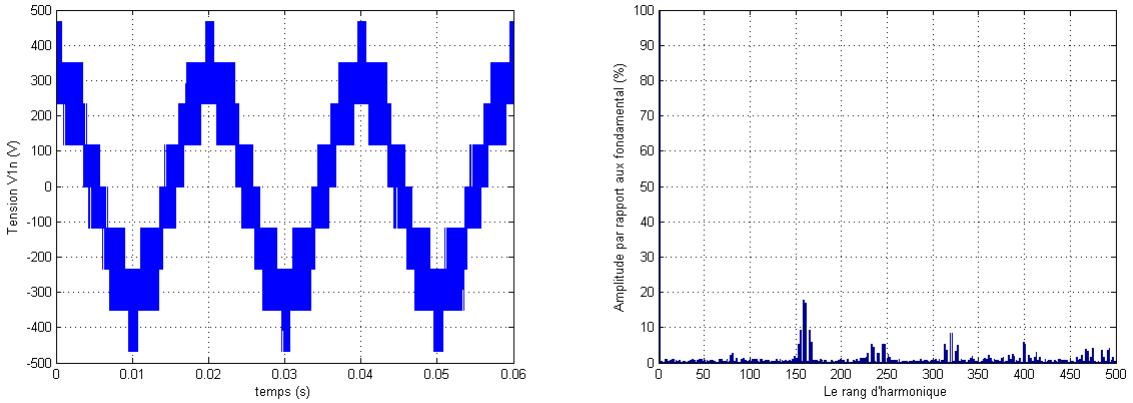


FIGURE 3.14: Spectre du tension V_{1n} (r=0.8, F_e=4 KHz)

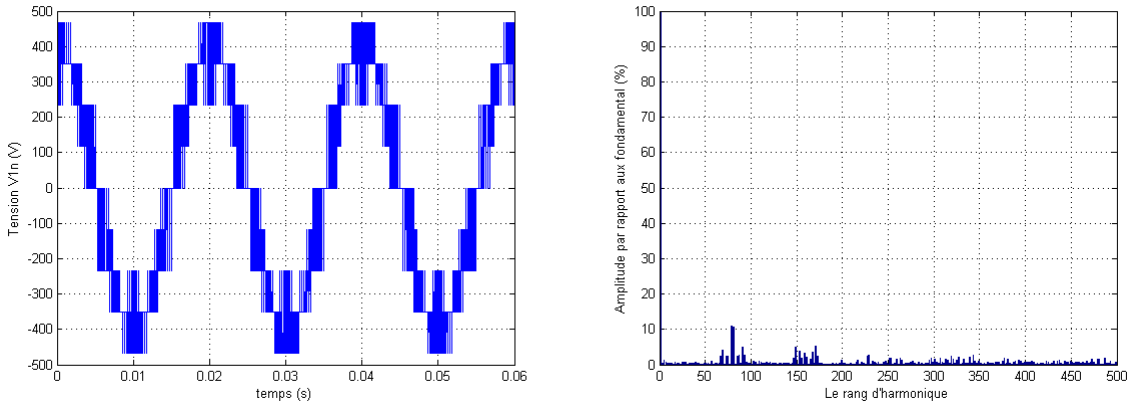


FIGURE 3.15: Spectre du tension V_{1n} (r=1.2, F_e=4 KHz)

On remarque que les harmoniques de la tension de sortie se regroupent en familles autour des multiples de rang 80, c'est-a-dire autour des multiples de la fréquence 80×50 Hz=4000 Hz, qui est la fréquence d'échantillonnage. De la figure 3.14 et 3.15, on remarque que l'amplitude des harmoniques diminue avec l'augmentation du rang d'harmonique. La figure 3.16 montre la variation du taux de distorsion d'harmonique (THD%) en fonction de r.

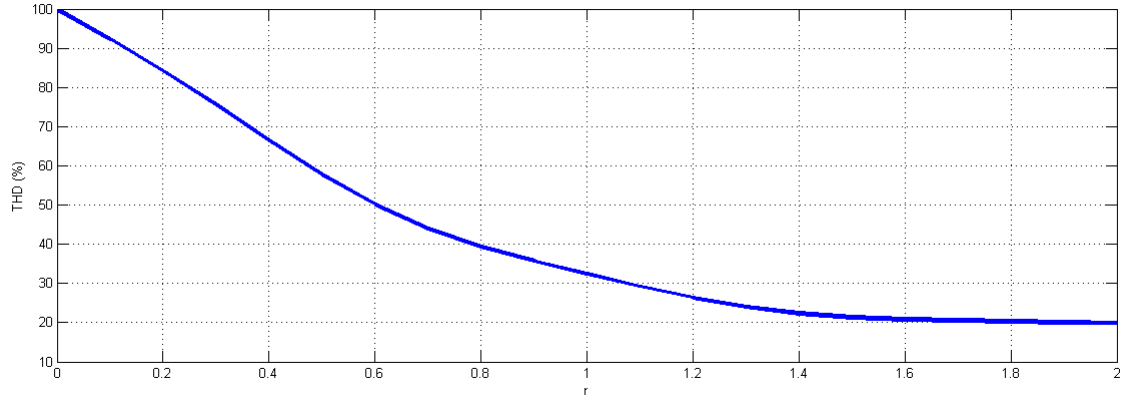


FIGURE 3.16: Taux de distorsion d'harmonique

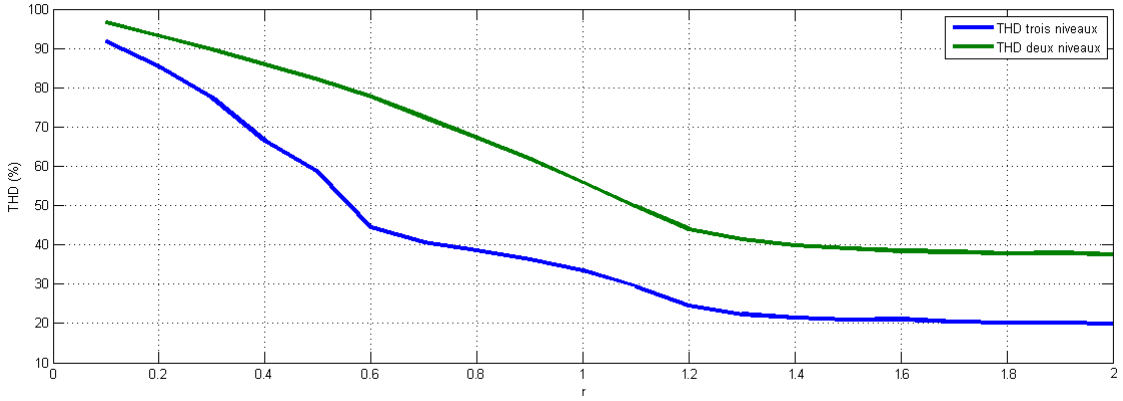


FIGURE 3.17: Taux de distorsion d'harmonique (comparaison)

La figure 3.17 donne une comparaison entre la variation du THD(%) de l'onduleur à deux niveaux et celle de l'onduleur à trois niveaux. On constate que pour la même valeur de r, le taux de distorsion d'harmonique de l'onduleur à deux niveaux est toujours supérieur a celui de l'onduleur à trois niveaux. Ce dernier approxime la tension de sortie simple par neuf valeurs de tension différentes, tant que l'onduleur à deux niveaux ne l'approxime que par cinq valeurs de tension différentes.

3.13.3 Courbe de réglage

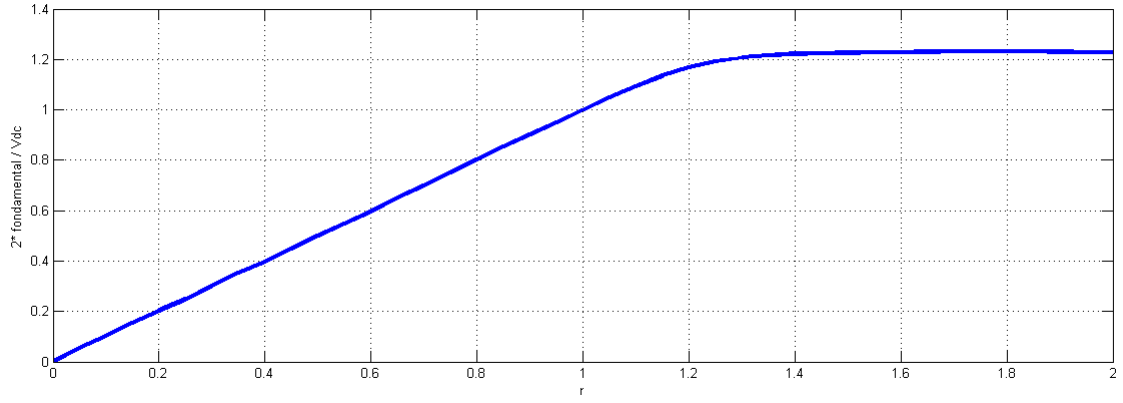


FIGURE 3.18: Courbe de réglage

On remarque que le fondamental de V_{1n} en valeur relative a $\frac{V_{dc}}{2}$ augmente linéairement avec l'augmentation de r pour r$\leq$1.15 ; puis au-delà de r=1.15 la valeur se sature.

On constate une saturation pour r>1.4, parce que au delà de cette valeur, le vecteur tension de référence $\vec{V}_{ref}$ se situe en dehors du cercle délimité par l'hexagone externe du diagramme vectoriel de l'onduleur à trois niveaux (Figure 3.19).

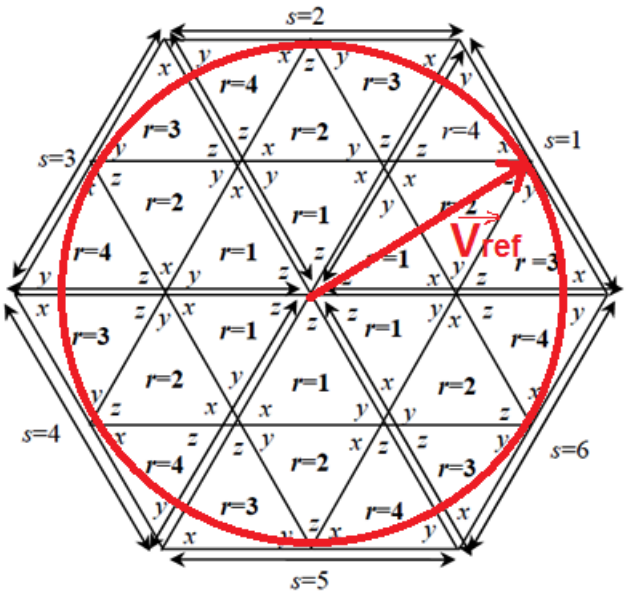


FIGURE 3.19: Trajectoire du $\vec{V}_{ref}$ lorsque $r = \frac{2\sqrt{3}}{3}$

3.14 Conclusion

Dans ce chapitre, on a vu la structure de l'onduleur à trois niveaux, son principe de fonctionnement, sa modélisation, ainsi que ses états dans le diagramme vectoriel.

On a vu que la modulation vectorielle à trois niveaux (SVM à trois niveaux) est une méthode qui consiste à approximer un vecteur de référence par les 27 vecteurs disponibles pour l'onduleur à trois niveaux.
La simulation montre que cette méthode génère une tension de sortie qui approxime une référence sinusoïdale par neuf valeurs de tension différentes. Son fondamental est proportionnel avec l'amplitude du vecteur tension de référence jusqu'à la valeur $r = \frac{2\sqrt{3}}{3}$.
L'analyse fréquentielle montre que cette méthode génère moins d'harmoniques par rapport a la modulation vectorielle à deux niveaux au prix d'une complexité d'algorithme.

Dans le chapitre suivant, on va voir l'implémentation de la SVM à deux et à trois niveaux

sur la carte ARDUINO DUE, puis on va faire une étude expérimentale pour valider cette implémentation et déterminer ces performances et ces limitations.

Étude expérimentale

4.1 Introduction

Dans les chapitres 2 et 3, on a étudié la théorie de la SVM à deux et à trois niveaux. On a utilisé l'outil MATLAB/SIMULINK pour simuler ces deux méthodes, et analyser ses performances : tensions de sorties, courbe de réglage, analyse fréquentiel.

L'utilisation de cette technique *(SVM)* sur un système physique *(onduleur à deux et à trois niveaux)* nécessite son implémentation sur un calculateur numérique, qui est la carte ARDUINO DUE utilisée dans notre travail. Cette implémentation doit respecter la notion du "temps réel". C'est-a-dire que l'exécution de l'algorithme de la SVM ne doit pas affecter la génération des signaux de commande de la modulation vectorielle.

Dans ce chapitre, on va voir l'implémentation de la SVM à deux niveaux sur la carte ARDUINO DUE pour la commande d'un onduleur à deux niveaux SEMISTACK–IGBT de SEMIKRON.
On va aussi implémenter la SVM à trois niveaux sur la même carte, puis on va récupérer les signaux de commande pour les utiliser sur le modèle d'un onduleur à trois niveaux à structure NPC idéale, sous l'environnement MATLAB.
Tout au long de ce chapitre, on va comparer les résultats expérimentaux et celle de simulation afin de valider notre travail.

4.2 Description du banc d'essais utilisé pour la validation expérimentale

Un montage expérimental est mis en œuvre pour évaluer et valider les différentes stratégies de commande vectoriel développées. La plateforme expérimentale conçue autour d'un convertisseur SEMISTACK–IGBT du SEMIKRON a été élaborée au sein de l'unité de développement des équipements solaires située à Bousmail.
Dans cette section, nous allons détailler les différentes parties constituant le banc d'essais : la partie puissance (le système onduleur et la charge R-L), instrumentation et appareils de mesures et l'unité de commande (carte ARDUINO DUE et PC Debugger).

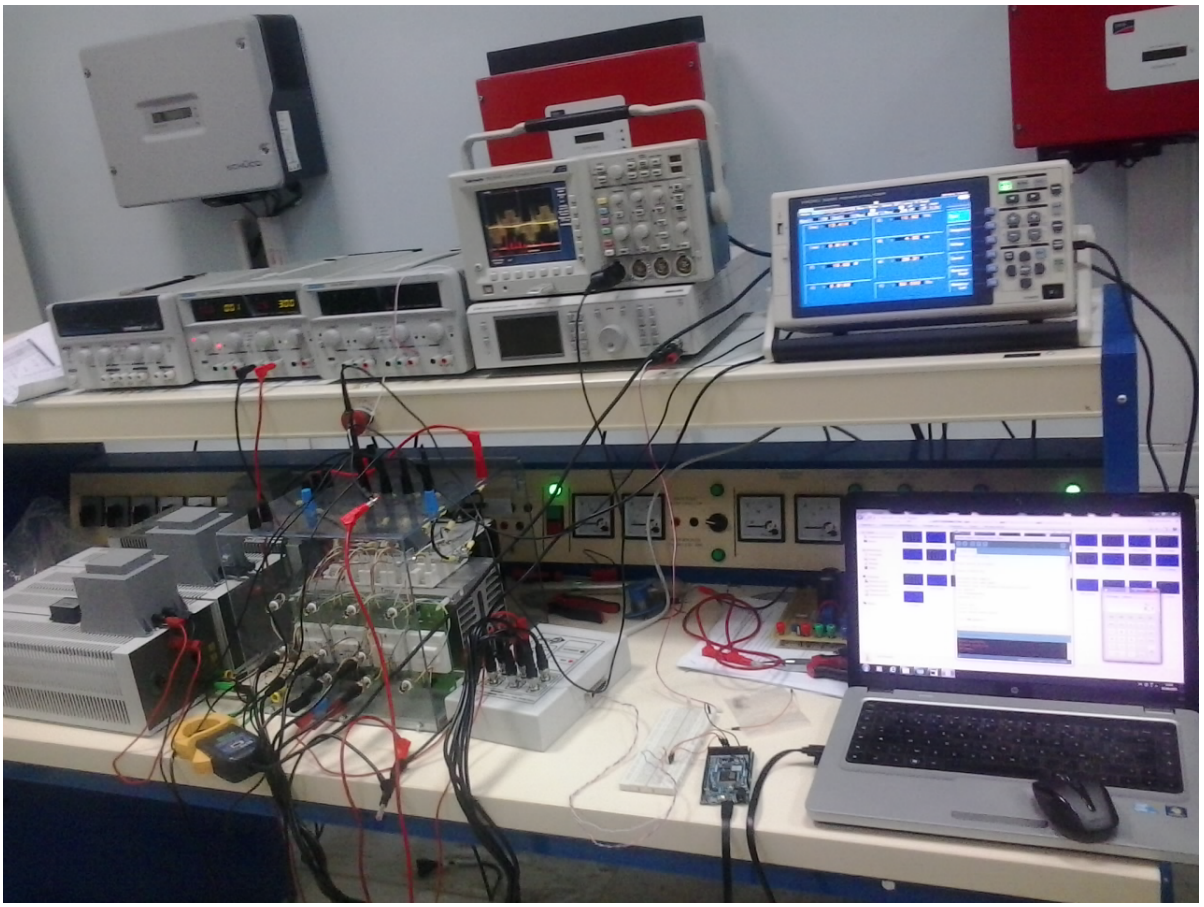


FIGURE 4.1: La plateforme expérimentale

La figure 4.1 illustre la plateforme expérimentale qui est constituée de :

1- Le convertisseur multi-fonctions IGBT du SEMIKRON :

La partie de puissance de la plateforme expérimentale est constituée d'un convertisseur *SEMISTACK–IGBT* du *SEMIKRON* présentée dans la figure 4.2. Cet dispositif didactique permet d'émuler la plupart des applications industrielles comme : Onduleur triphasé, Hacheur de freinage, Hacheur Buck ou Boost, Onduleur monophasé et redresseur mono ou triphasé. Il est conçu à partir des modules IGBT *SKM50GB123D* de la société *SEMIKRON*. Ces IGBTs sont pilotés par des drivers de référence *SKHI22A* distribués également par la société *SEMIKRON* [15].

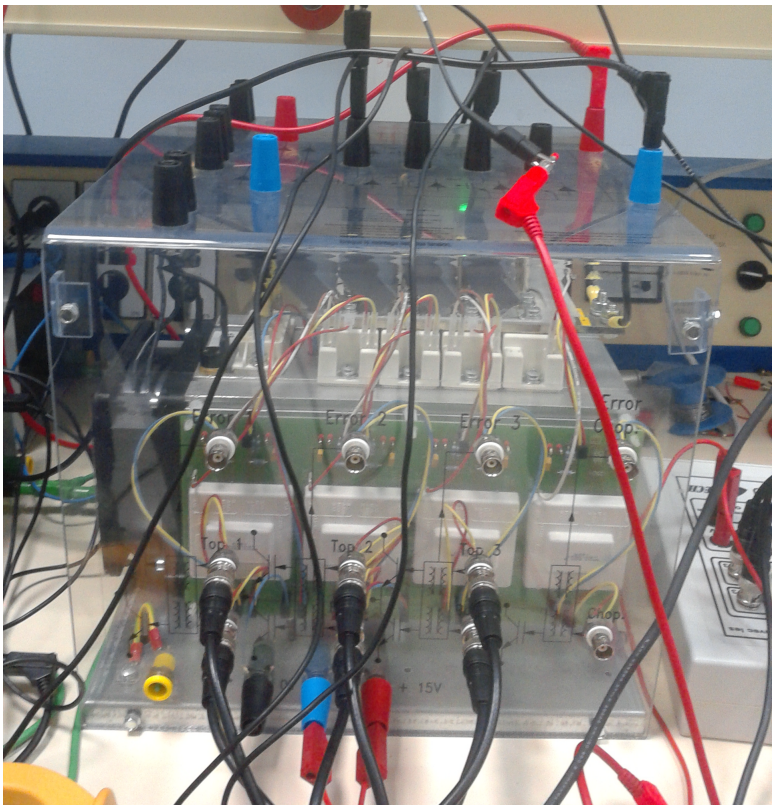


FIGURE 4.2: Onduleur à deux niveaux SEMIKRON

2- Module de génération des compléments des trois signaux de commande et des temps morts de l'onduleur :

Ce module est décrit sur la figure 4.3. l'entrée reçoit les trois impulsions de commande, générées par la carte ARDUINO DUE avec une tension de 0 V pour la logique 0, et 3.3 V pour la logique 1. Ces signaux de commande correspondants au pilotage des trois bras de l'onduleur de tension triphasé. Ce système joue le rôle d'un étage d'adaptation par l'amplification pour atteindre des signaux de sortie de 0 V et 15 V pour des signaux d'entrée de 0 V et 3.3 V respectivement, afin de piloter les commandes rapprochées basées sur les circuits *SKHI22* de

la société *SEMIKRON*. Ce système génère aussi la complémentarité des commandes pour chaque bras de l'onduleur avec une génération de temps mort, afin d'assurer la protection des bras contre les courts circuit.

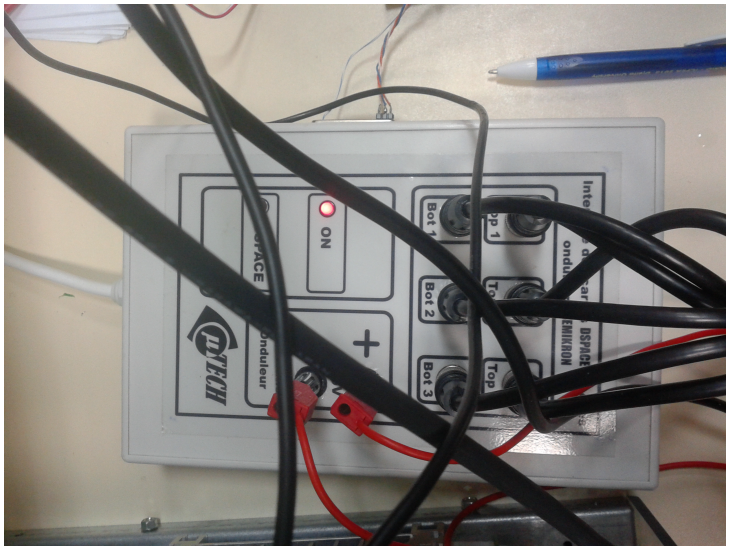


FIGURE 4.3: Interface d'adaptation

Une mesure pratique à l'aide de l'oscilloscope a permis d'évaluer réellement le temps mort entre les deux commandes complémentaires du même bras.

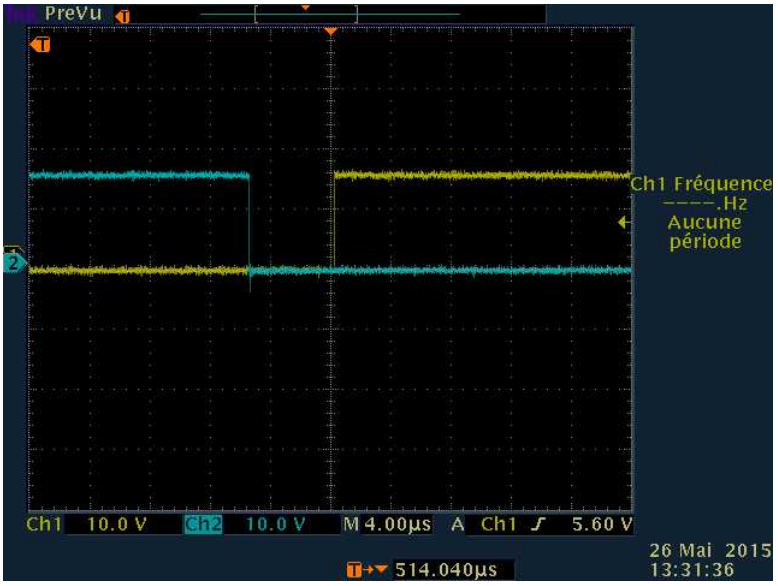


FIGURE 4.4: Retard généré par l'interface d'adaptation

La Figure 4.4 illustre le signal de commande de l'IGBT haut d'un bras et son complément de l'IGBT bas. On constate de cette figure que le system génère un temps mort de 5 us.

3- Alimentation stabilisé :

Elle permet d'alimenter l'entrée de l'onduleur par une tension continue (Figure 4.5).

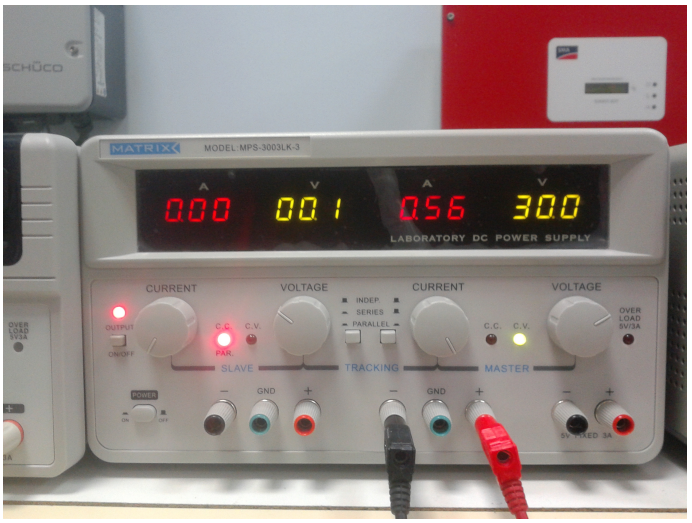


FIGURE 4.5: Alimentation stabilisée

4- Charge R-L :

Une charge triphasée R-L est branchée à la sortie de l'onduleur, et montée en étoile à neutre isolée. La charge R-L est composée par phase d'une inductance de 3 mH, connectée en série avec une résistance variable, de valeur maximale 33 Ohm.

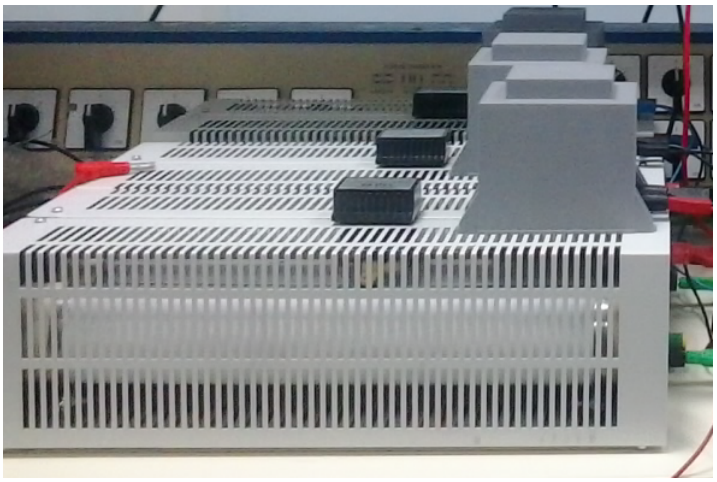


FIGURE 4.6: Charge R-L

5- Instrumentation pour la mesure des grandeurs électriques :

Un oscilloscope de 500 MHz est utilisé pour visualiser les formes des tensions et leurs spectres fréquentiel.

Un analyseur de spectre est utilisé aussi afin de mesurer et visualiser les tensions, et les courants qui circulent dans la charge R-L. Cette dispositif permet aussi de récupérer la valeur efficace, le fondamental, le taux de distorsion (THD%) et l'amplitude des harmoniques de rang donné (Figure 4.7).

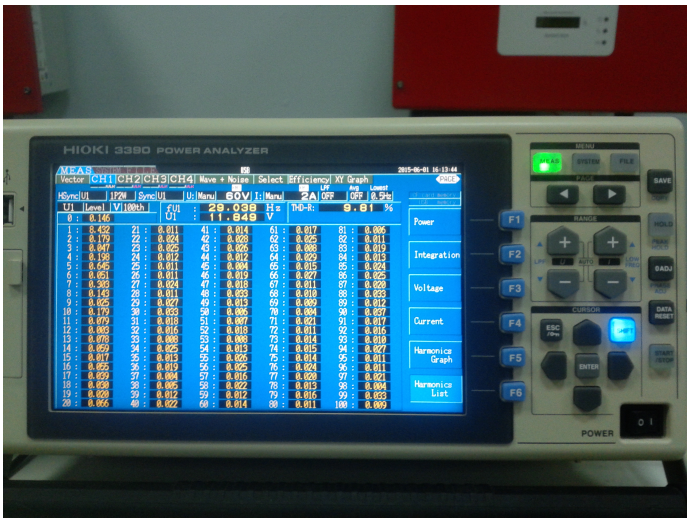


FIGURE 4.7: Analyseur de spectre

6- Unité de commande :

Constituée de la carte ARDUINO DUE, sur laquelle l'algorithme de la SVM à deux niveaux a était implémenté, et un PC debugger muni de l'environnement de développement intégré d'ARDUINO, afin de programmer, compiler et charger le programme dans la carte de commande.

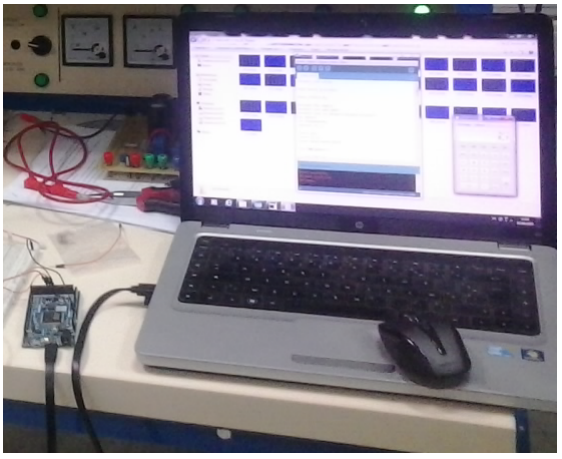


FIGURE 4.8: Unité de commande

4.3 Étude expérimentale de la SVM à deux niveaux

4.3.1 Implémentation de la SVM à deux niveaux sur la carte ARDUINO DUE

Arduino Due est un circuit de la famille *Arduino* réalisé autour du microcontrôleur *Atmel SAM3X8E ARM Cortex-M3 CPU*. C'est le premier *ARDUINO* réalisé autour d'un microcontrôleur 32-bit de core *ARM*. Pour implémenter l'Algorithme de la *SVPM* deux niveaux, on va utiliser le module *PWM* de *l'ARDUINO DUE*. Ce dernier, possède 8 chaines de modulation à largeur d'impulsion (PWM channel) de 16 bit, avec une sortie complémentaire[3].

4.3.2 Description du module PWM

Le module *PWM* contrôle 8 chaines. Chaque chaine contrôle deux sorties complémentaires. Les caractéristiques du signal de sortie (période, rapport cyclique ... etc.) sont configurés à travers l'interface d'utilisateur. Chaque chaine utilise un signal d'horloge généré par le générateur d'horloge (clock generator). L'accès au module *PWM* se fait à travers les registres liés au bus de périphéries. Les chaines *PWM* peuvent être synchronisées ensemble pour générer des signaux de sorties synchronisées avec des rapports cycliques différents. L'actualisation des rapports cycliques des chaines synchronisées peut être réalisée par le *Peripheral DMA Controller Channel (PDC)*, qui offre un transfert tampon sans la nécessité de l'intervention du processeur [1].
Le module *PWM* offre 8 unités de comparaison indépendantes capables de comparer une valeur programmée au compteur de la chaine synchrone (compteur de la chaine 0), avec la possibilité de générer des interruptions ou de basculer l'état du signal de sortie.

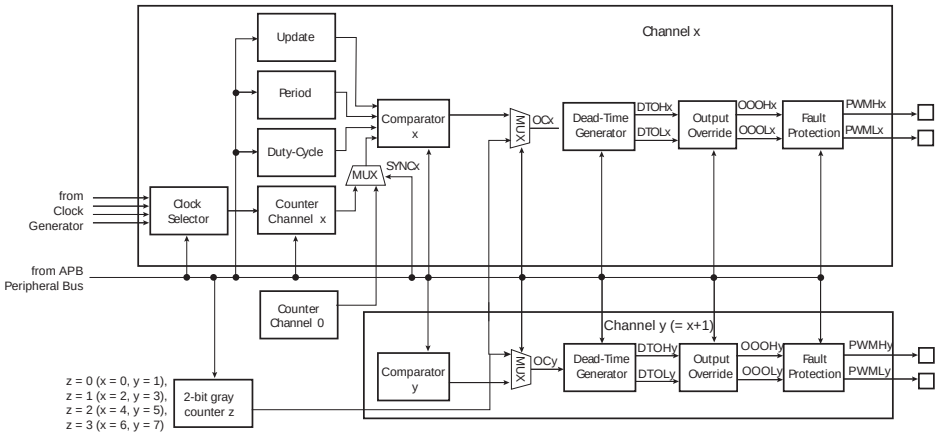


FIGURE 4.9: Module PWM de la carte ARDUINO DUE

Chaque chaine *x* du module *PWM* peut être configurée en deux modes différents :

1- Center Aligned Mode : ou le compteur de la chaine *x (PWM_CCNTx)* incrémente de *0* a la valeur du registre de période *(PWM_CPRDx)* puis il décrémente de cette valeur a *0*.
2- Left Aligned Mode : ou le compteur de la chaine *x (PWM_CCNTx)* incrémente de *0* a la valeur du registre de période *(PWM_CPRDx)* puis il se remet a *0*.

La sortie de chaque chaine *x* du module *PWM* peut être configurée en deux polarités différentes :

1- Signal au niveau bas au début du période *(CPOL(PWM_CMRx) = 0)*, et il passe au niveau haut lorsque la valeur du registre de compteur (PWM_CCNTx) devient supérieur a la valeur du registre de période (PWM_CPRDx), c'est-a-dire lorsque :
$PWM\_CCNTx >= PWM\_CPRDx$.

2- Signal au niveau haut au début du période *(CPOL(PWM_CMRx) = 1)*, et il passe au niveau bas lorsque la valeur du registre de compteur (PWM_CCNTx) devient supérieur a la valeur du registre de période (PWM_CPRDx), c'est-a-dire lorsque :
$PWM\_CCNTx >= PWM\_CPRDx$.

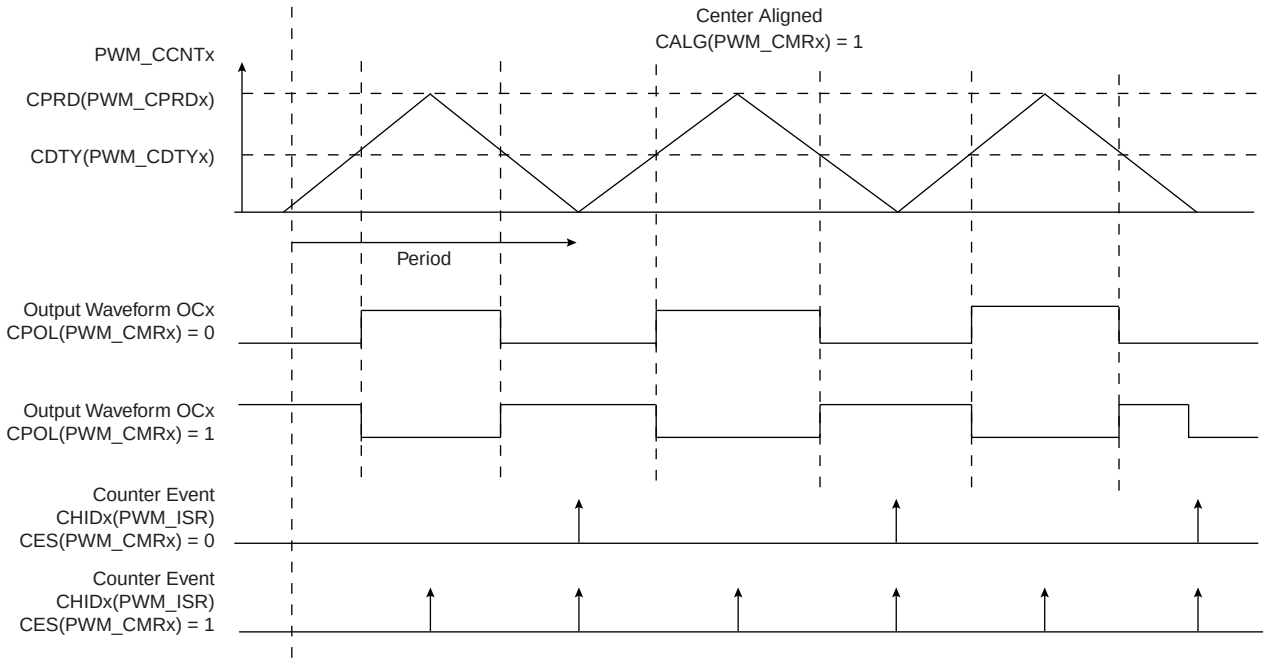


FIGURE 4.10: Module PWM de la carte ARDUINO DUE en mode Center Aligned

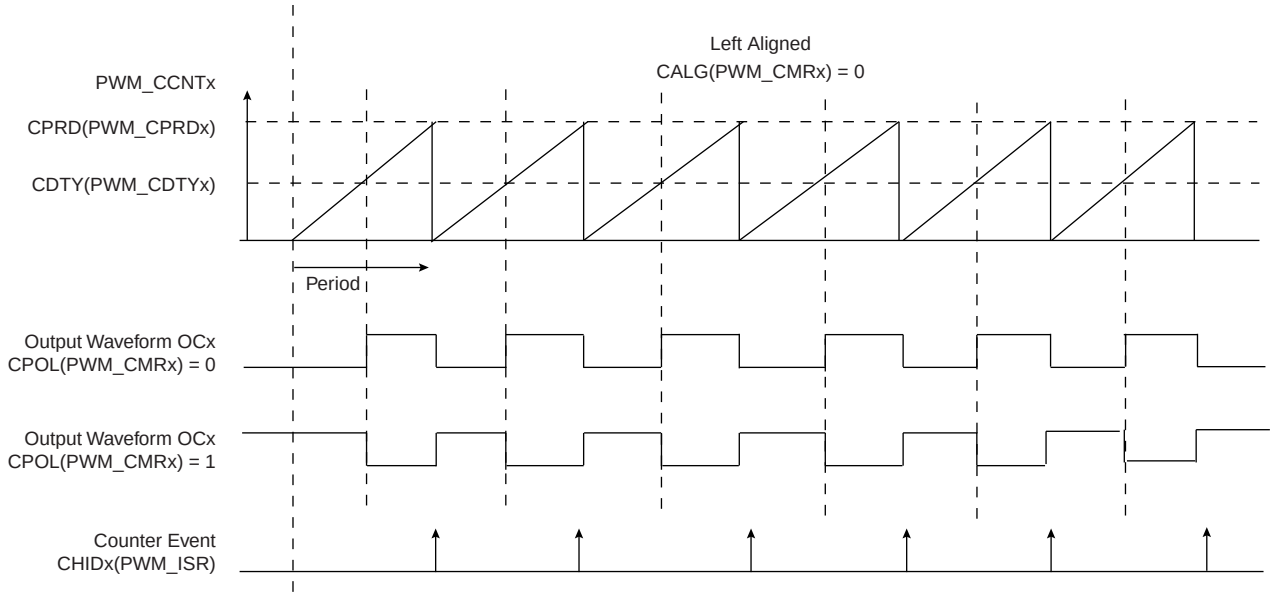


FIGURE 4.11: Module PWM de la carte ARDUINO DUE en mode Left Aligned

4.3.3 Initialisation du module PWM

Avant d'activer les chaines *PWM*, ils doivent être configurés par le programme :

- Déverrouiller l'interface d'utilisateur par l'écriture du champ *WPCMD* dans le registre *PWM_WPCR*.
- Configuration du générateur d'horloge.
- Configuration de l'alignement du signal de sortie.
- Configuration de la polarité du signal de sortie.
- Configuration de la période de chaque chaine par le registre *PWM_CPRDx*.
- Configuration du rapport cyclique de chaque chaine par le registre *PWM_CDTYx*.
- Sélection des chaines synchronisées (*SYNCx* dans le registre *PWM_SCM*).
- Activation des interruptions.
- Activation des chaines *PWM*.

4.3.4 Algorithme de la SVM a deux niveaux sur la carte ARDUINO DUE

On va utiliser les broches (*pins*) **35**, **37** et **39** de la carte *ARDUINO DUE* qui sont connectés aux chaines *PWMH0*, *PWMH1* et *PWMH2* respectivement, pour générer les signaux du bras **a**, **b** et **c** respectivement de l'onduleur a deux niveaux.

D'après les figures 2.6, 2.7, 2.8, 2.9, 2.10 et 2.11, on remarque que les impulsions des bras **a**,**b** et **c** sont symétriques par rapport à la demi-période d'échantillonnage, donc on va configurer les trois chaines de modulation a largeur d'impulsion en mode *alignement centré* (*Center Aligned*).
De plus, les impulsions des trois bras passent toujours du niveau bas au niveau haut, donc on va encore configurer les trois chaines de modulation a largeur d'impulsion avec une polarité positive ($CPOL(PWM\_CMRx) = 0$), c'est-a-dire que le signal du sortie est au niveau bas au début du période , et passe au niveau haut lorsque la valeur du registre de compteur (PWM_CCNTx) devient supérieur a la valeur du registre de période (PWM_CPRDx), c'est-a-dire lorsque : $PWM\_CCNTx >= PWM\_CPRDx$.

Cette méthode d'implémentation facilite l'exécution de l'algorithme et réduit la taille du programme.

Pour avoir une fréquence d'échantillonnage de *4 KHz*, et sachant que la fréquence d'horloge de la carte *ARDUINO DUE* est de *84 MHz*, et que les chaines sont configurées en alignement centré (*Center Aligned*), On doit charger les registres de période *PWM_CPRD0*, *PWM_CPRD1* et *PWM_CPRD2* par la valeur :

$$PWM\_CPRD0 = PWM\_CPRD1 = PWM\_CPRD2 = \frac{84000000}{2 \times 4000} = 10500 \qquad (4.1)$$

On va activer la génération automatique des interruptions à chaque début de période, c'est-à-dire a une fréquence de *4 KHz*, ce qui permet la non saturation du processeur et l'exécution de l'algorithme en une seule subroutine.
Au sous programme d'interruption on va :

- Calculer l'argument du vecteur tension de référence a l'aide d'un temporisateur (*timer*) de la carte ARDUINO DUE.
- Déterminer le secteur par une division en nombre entier de l'argument du vecteur tension de référence par $\pi/3$.
- Déterminer les vecteurs a appliqués en fonction du secteur. Ces vecteurs sont prédéfinies dans un tableau.
- Calculer les durées T_1 et T_2 a partir des équations *1.13* et *1.14*.
- Calculer la durée T_0 a partir de l'équation *1.15*.
- Calculer le rapport cyclique pour chaque chaine *PWMx (x=0,1 et 2)* en fonction des vecteurs a appliqué, et les durées T_0, T_1 et T_2.
- Chargement du registre tampon *PWM_CDTYUPDx (x=0,1 et 2)* pour actualiser le rapport cyclique de chaque chaine *PWMx (x=0,1 et 2)* dans le début du prochaine période d'échantillonnage. Cette manière d'implémentation permet d'avoir une durée d'impulsion très précis, et inaffecté par le temps d'exécution du programme.

Le diagramme de la figure 4.12 résume l'implémentation de la SVM a deux niveaux avec la carte ARDUINO DUE.

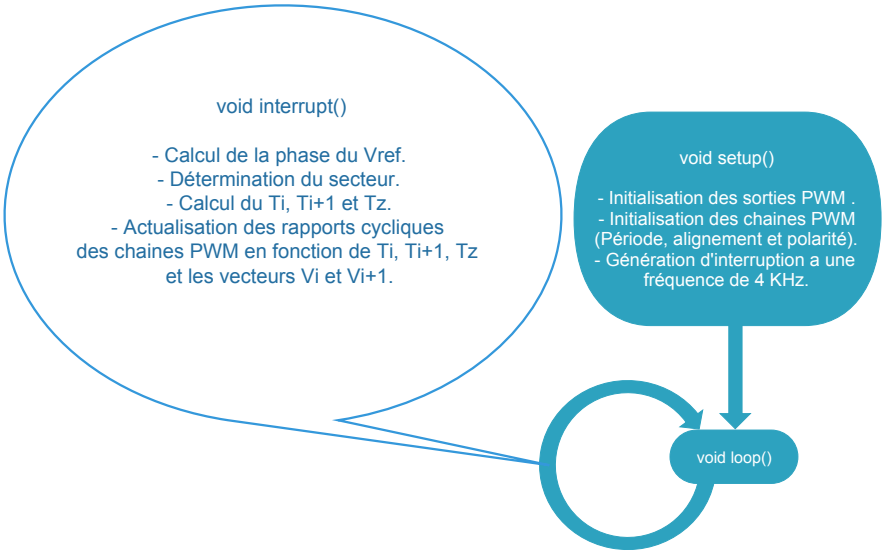
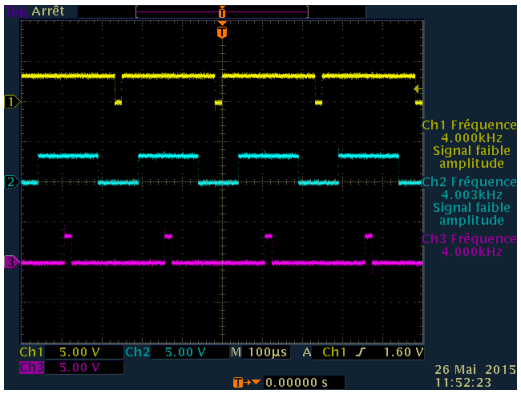


FIGURE 4.12: Diagramme SVM 2 niveaux

4.3.5 Signaux de commande

On va visualiser les signaux de commande de la carte ARDUINO DUE après l'implémentation de la SVM à deux niveaux.



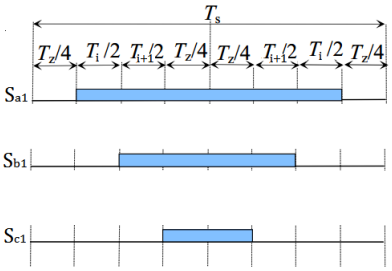


FIGURE 4.13: signaux de commande pour le secteur 1

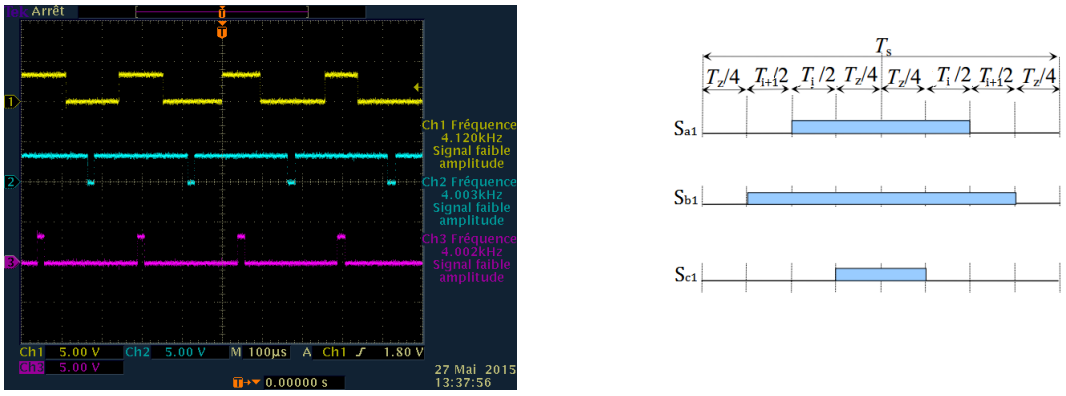


FIGURE 4.14: signaux de commande pour le secteur 2

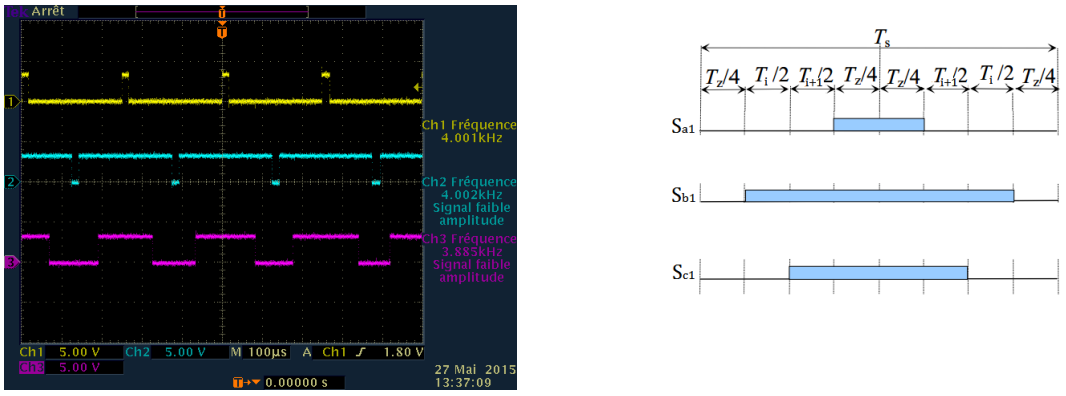


FIGURE 4.15: signaux de commande pour le secteur 3

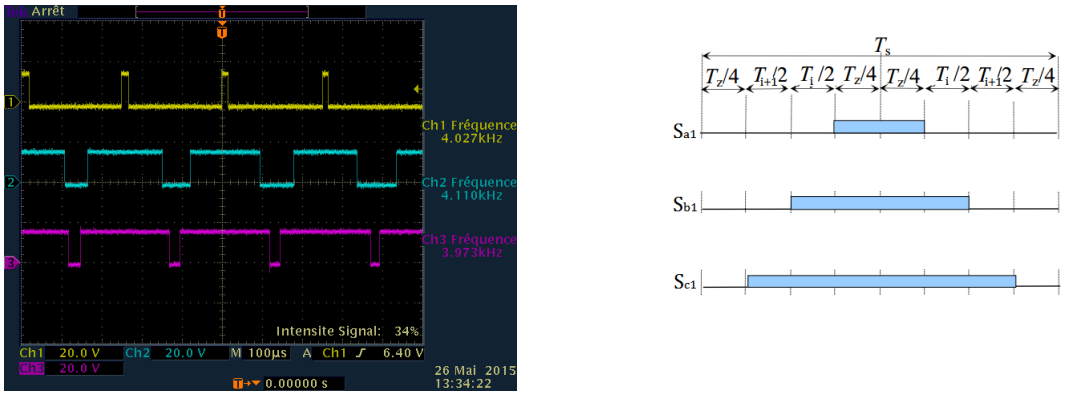


FIGURE 4.16: signaux de commande pour le secteur 4

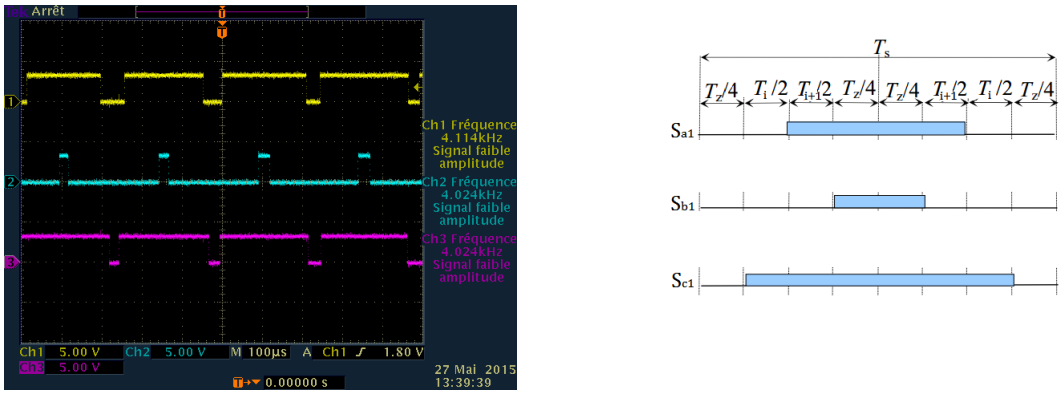


FIGURE 4.17: signaux de commande pour le secteur 5

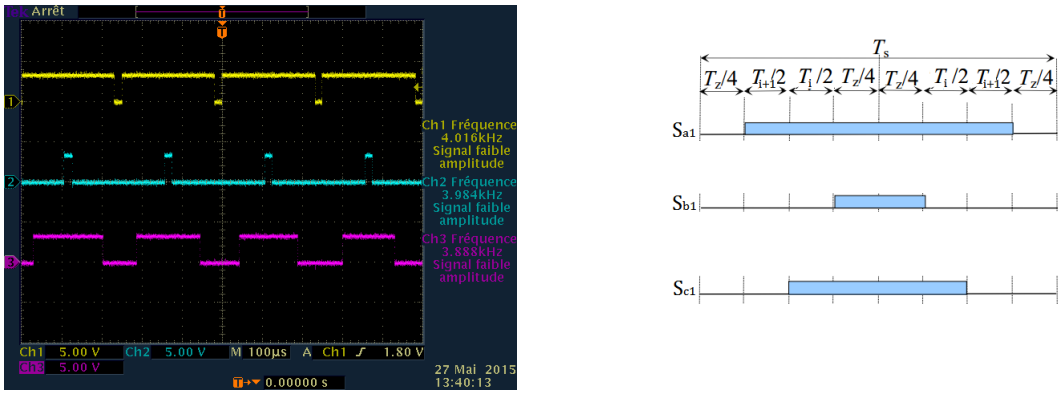


FIGURE 4.18: signaux de commande pour le secteur 6

On remarque que les impulsions générées par la carte ARDUINO DUE sont symétriques par rapport a la demi-période d'échantillonnage, avec une fréquence de 4 KHz.

4.3.6 Étude de l'onduleur avec une charge résistive R

Tout d'abord, on va commander l'onduleur Semikron associé a une charge résistive triphasée équilibrée (R=33 ohm), avec une tension d'alimentation de 30 V, une fréquence d'échantillonnage de 4 KHz, et un taux de réglage r=1.

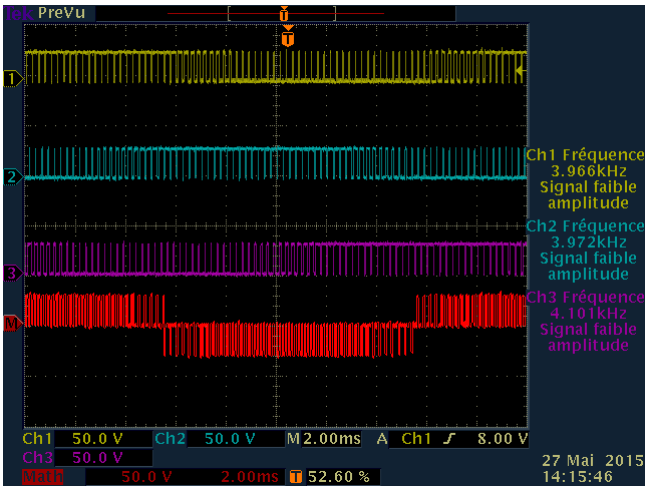


FIGURE 4.19: Tension V_{ao}, V_{bo} et V_{co} (r=1, $F_e = 4KHz$) : expérimentale

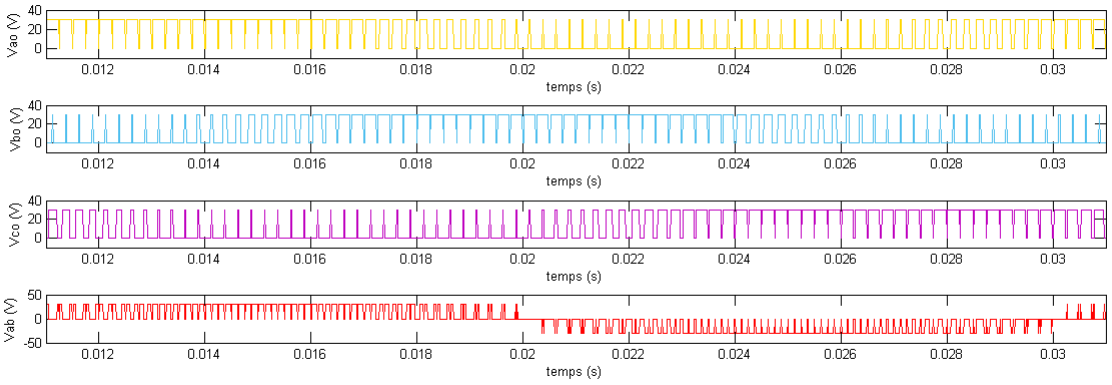


FIGURE 4.20: Tension V_{ao}, V_{bo} et V_{co} (r=1, $F_e = 4KHz$) : simulation

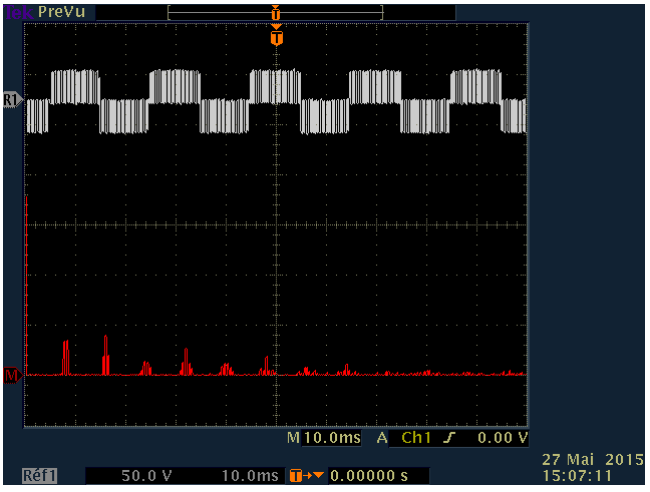


FIGURE 4.21: Tension V_{ab} avec son spectre (r=1, $F_e = 4KHz$) : expérimentale

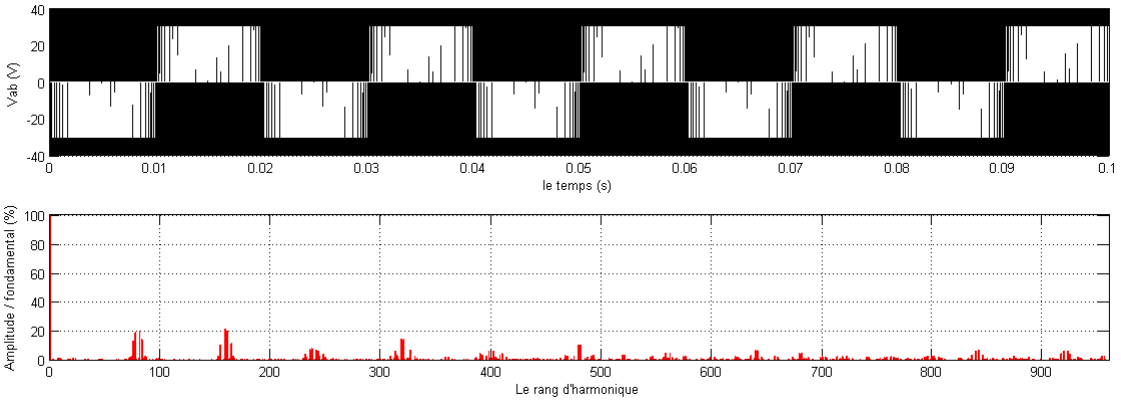


FIGURE 4.22: Tension V_{ab} avec son spectre (r=1, $F_e = 4KHz$) : simulation

La figure 4.19 montre les tensions des trois bras par rapport au neutre de la source d'alimentation **'o'**. On remarque que chaque tension prend la valeur 0 V ou 30 V (V_{dc}). De plus on remarque que la tension V_{bo} est retardé de $2\pi/3$ par rapport a V_{ao}, et V_{co} est retardé de $2\pi/3$ par rapport a V_{bo}.

La figure 4.21 montre la tension V_{ab} entre le bras **'a'** et le bras **'b'**. Cette tension prend la valeur 30 V (+ V_{dc}), ou - 30 V (- V_{dc}). L'analyse spectral montre que les harmoniques se regroupent en familles autour de la fréquence d'échantillonnage (4 KHz).

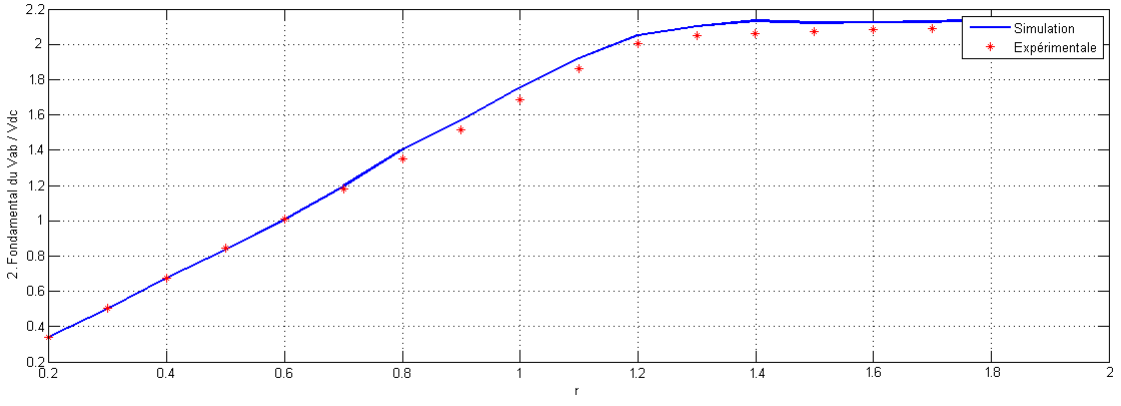


FIGURE 4.23: Variation du fondamental normalisé de V_{ab} en fonction de r

La figure 4.23 montre la courbe du fondamental normalisé de V_{ab} ($\frac{fondamental\ \ de\ \ V_{ab}}{V_{dc}/2}$) en fonction du taux de réglage r. On remarque que les résultats expérimentaux coïncide avec la courbe obtenue par simulation. De plus, on remarque une saturation pour $r > 1.4$.

4.3.7 Étude de l'onduleur avec une charge R-L

On va maintenant commander l'onduleur Semikron associé a une charge R-L (R=15 ohm, L=0.003 H) triphasée équilibrée, une tension d'alimentation de 30 V, une fréquence d'échantillonnage de 4 KHz, et un taux de réglage r=1.

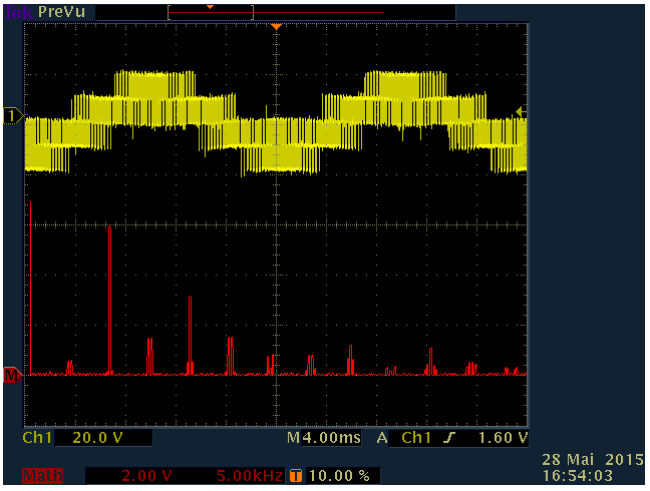


FIGURE 4.24: Tension V_{an} avec son spectre (r=1, $F_e = 4KHz$) : expérimentale

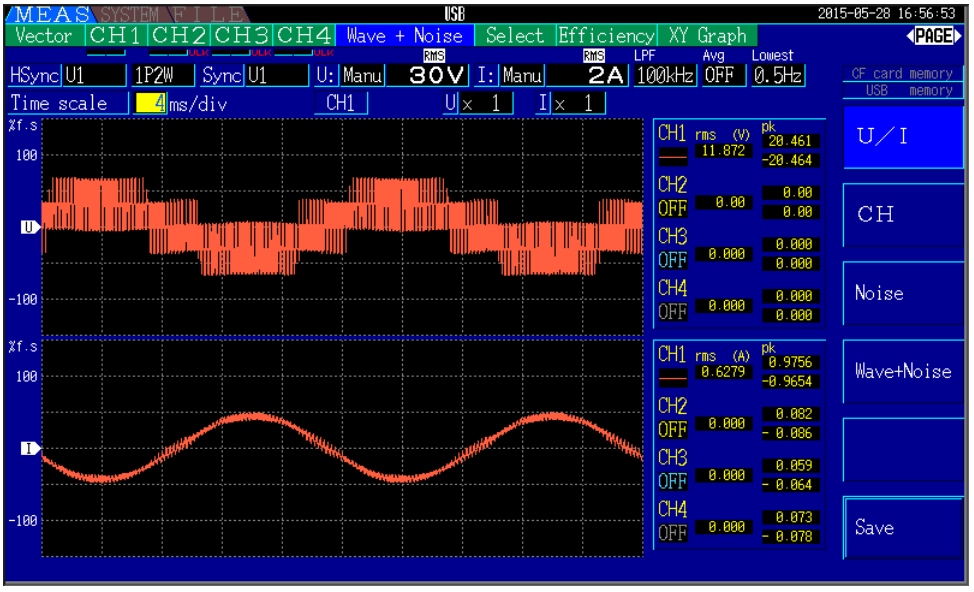


FIGURE 4.25: Tension V_{an}, Courant i_a (r=1, $F_e = 4KHz$) : expérimentale

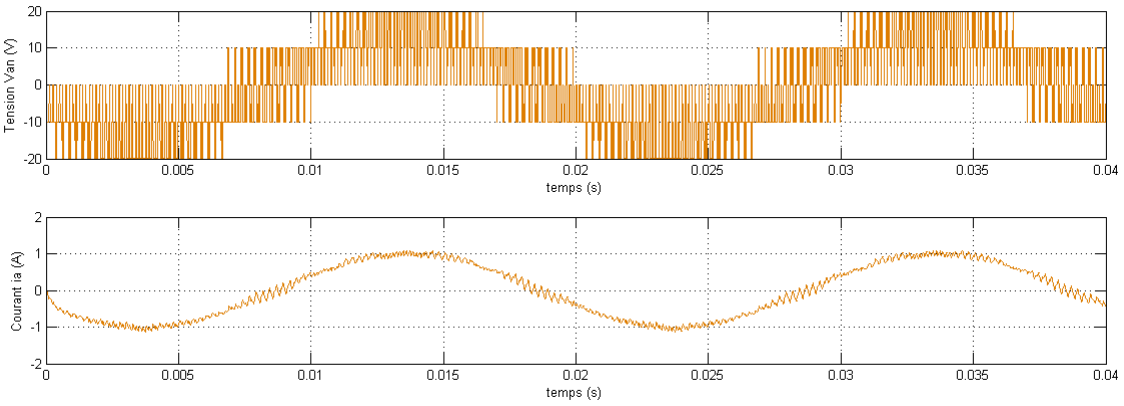


FIGURE 4.26: Tension V_{an}, Courant i_a (r=1, $F_e = 4KHz$) : simulation

On constate que la tension V_{an} entre le bras **'a'** et le neutre de la charge **'N'** prend cinq valeurs de tension différentes : + 20 V (+ $2V_{dc}/3$), + 10 V (+ $V_{dc}/3$), 0 V, - 10 V (- $V_{dc}/3$) et - 20 V (- $2V_{dc}/3$). Elle est périodique de période 0.02 ms (50 Hz).
Les harmoniques de tension se regroupent autour de la fréquence d'échantillonnage 4 KHz.

La figure 4.25 montre le courant i_a de la phase **'a'**. On remarque que le courant est filtré par l'inductance (L=0.003 H), et prend la forme d'une sinusoïde de fréquence 50 Hz.

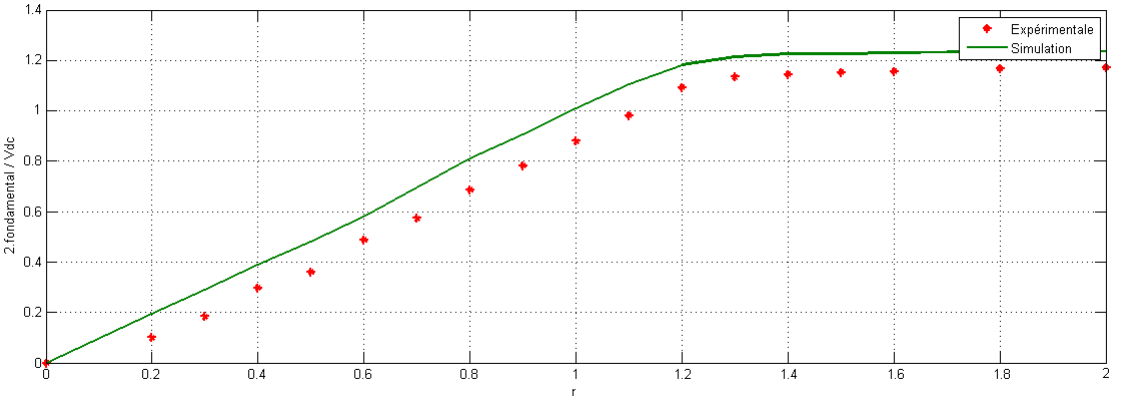


FIGURE 4.27: Variation du fondamental normalisé de V_{an} en fonction du taux de réglage r

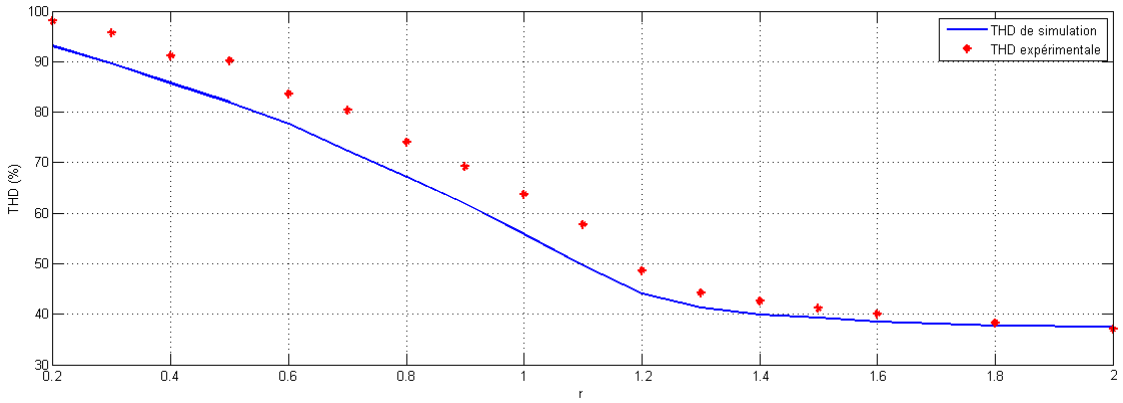


FIGURE 4.28: THD (%) = f (r)

On voit sur La figure 4.27 la courbe de réglage de l'onduleur associé a une charge R-L. On remarque que la courbe est linéaire de 0 jusqu'à la valeur r=$\frac{2\sqrt{3}}{3} \approx 1.15$, ou on remarque une saturation.

La figure 4.28 montre la variation du THD(%) en fonction de r. On constate que le taux de distorsion d'harmonique (THD(%)) diminue avec l'augmentation de r.

Pour les deux courbes précédents (4.27, 4.28), on constate un petit décalage entre les résultats expérimentaux et ceux obtenus par simulation. On peut expliquer ça par le fait que la simulation suppose que les composants et les mesures sont idéales, ce qui est erroné en pratique (charge non parfaitement équilibrée, non linéarité des composants, erreurs des appareils de mesures ...etc).

4.3.8 Influence de la fréquence d'échantillonnage

On va augmenter la fréquence d'échantillonnage pour un r=1, et une fréquence de tension de sortie de 50 Hz.

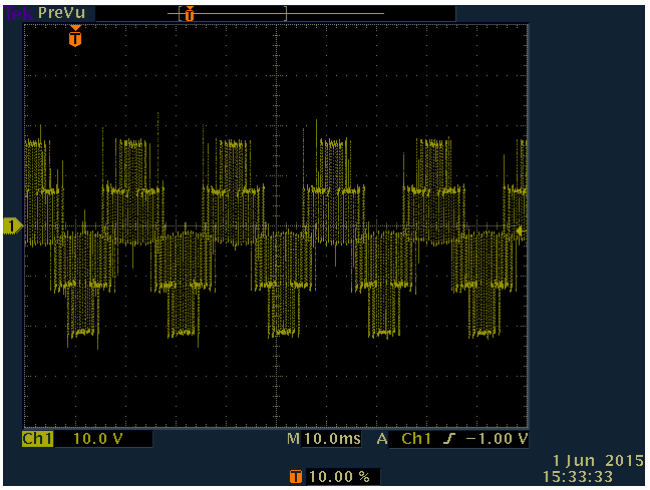


FIGURE 4.29: Tension V_{an} (r=1, F_e=1 KHz)

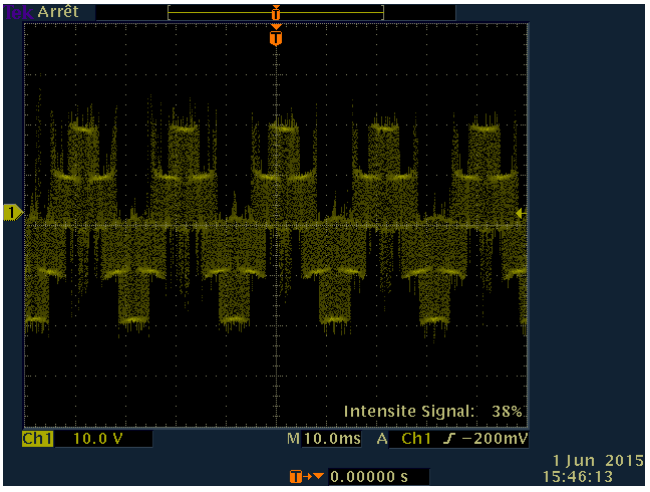


FIGURE 4.30: Tension V_{an} (r=1, F_e=5 KHz)

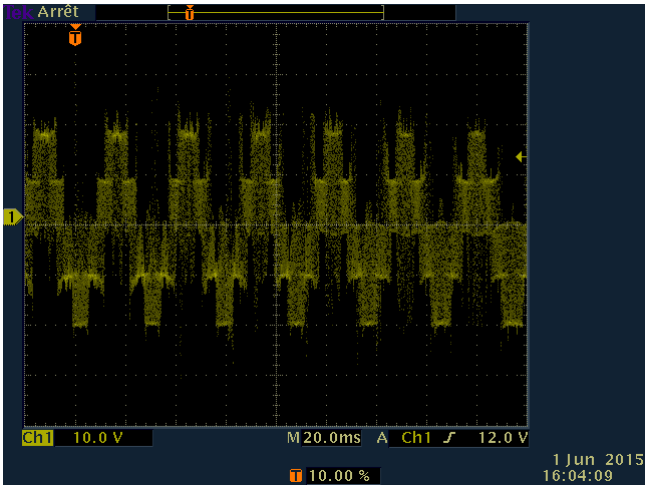


FIGURE 4.31: Tension V_{an} (r=1, F_e=10 KHz)

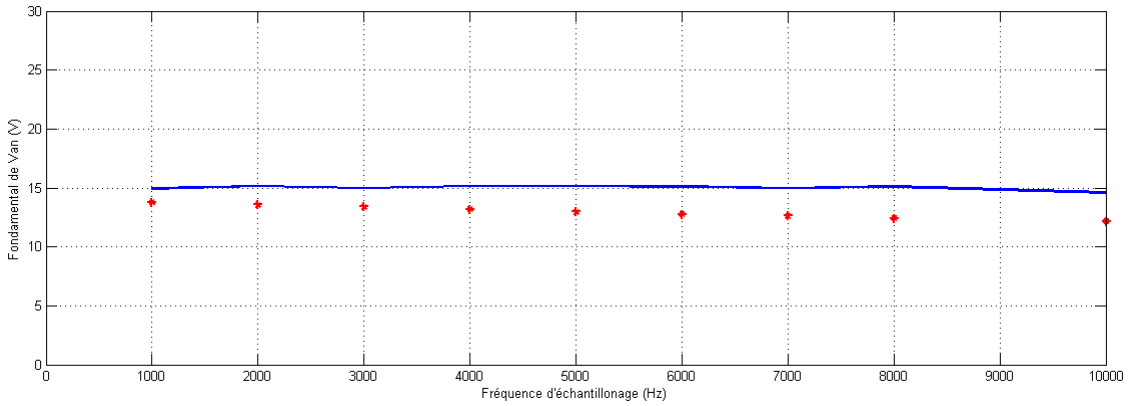


FIGURE 4.32: Fondamental de V_{an} en fonction de F_e (r=1)

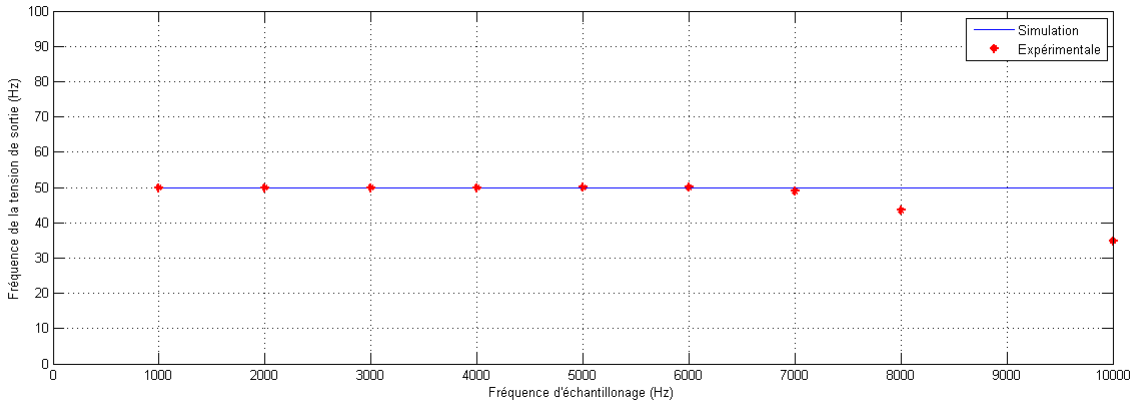


FIGURE 4.33: Fréquence de V_{an} en fonction de F_e (r=1)

Les figures 4.32 et 4.33 montrent respectivement la variation du fondamental de la tension de sortie et sa fréquence en fonction de la fréquence d'échantillonnage. On constate une convergence entre les résultats obtenus par simulation et ceux obtenue par expérimentation jusqu'à la fréquence F_e=7000 Hz. Donc on peut dire que la limitation de la SVM à deux niveaux implémentée sur la carte ARDUINO DUE se situe à une fréquence d'échantillonnage de 7 KHz.

4.4 Étude expérimentale de la SVM à trois niveaux

L'implémentation de l'algorithme de la SVM à trois niveaux sur la carte ARDUINO DUE passe par les étapes suivantes :

4.4.1 Détermination de la phase du vecteur tension de référence

Tout d'abord, on doit déterminer la phase θ du vecteur tension de référence $\vec{V}_{ref}$ à l'aide d'un temporisateur (*timer*) de la carte ARDUINO DUE.

4.4.2 Détermination du secteur et de la région

La détermination du secteur se fait par le résultat de la division en nombre entier de la phase du vecteur tension de référence (θ) par $\pi/3$.
En se référent sur le premier secteur par le reste de la division en nombre entier de la phase (θ) du vecteur tension de référence ($\vec{V}_{ref}$) par $\pi/3$ (α=$\theta[\pi/3]$), on détermine les composants $\hat{V}_d$ et $\hat{V}_q$ du vecteur tension de référence.
Ensuite, la détermination de la région se fait comme suit :

- Si ($\hat{V}_q + \sqrt{3}.\hat{V}_d$ - $\sqrt{3}/3 < 0$) alors région=1
- Sinon si ($\hat{V}_q$ - $\sqrt{3}.\hat{V}_d + \sqrt{3}/3 < 0$) alors région=3
- Sinon si ($\hat{V}_q$ - $\sqrt{3}/6 < 0$) alors région=2
- Sinon région=4.

4.4.3 Détermination des durées relatives

Du tableau 3.3, on remarque que l'équation de la durée relative de chaque vecteur X,Y et Z ne dépend que de la région, donc on détermine les durées relatives d_x, d_y et d_z comme suit :

- Si (région=1) alors :
  - $d_y = 6\hat{V}_q/\sqrt{(3)}$.
  - $d_x = 3\hat{V}_d$ - $d_y/2$.

  - d_z= 1 - d_x - d_y.
- Sinon si (région=2) alors :
  - d_y = 1 - $6\hat{V}_q/\sqrt{(3)}$.
  - d_x = 3/2 - $3\hat{V}_d$ - $d_y/2$.
  - d_z=1 - d_x - d_y.
- Sinon si (région=3) alors :
  - d_y = $6\hat{V}_q/sqrt(3)$.
  - d_x = $3\hat{V}_d$ - 1 - $d_y/2$.
  - d_z= 1 - d_x - d_y.
- Sinon :
  - d_y = $6\hat{V}_q/sqrt(3)$ - 1.
  - d_x = $3\hat{V}_d$ - 1/2 - $d_y/2$.
  - d_z= 1 - d_x - d_y.

4.4.4 Arrangement des vecteurs X, Y et Z

Du tableau 3.4, on remarque que les durées d'application T_x, T_y et T_z de chaque vecteur X,Y et Z sont arrangées en fonction de la région comme suit :

- **Pour la région 1 :** Les vecteurs sont appliqués dans l'ordre : $T_z/6$, $T_x/4$, $T_y/4$, $T_z/6$, $T_x/4$, $T_y/4$, $T_z/6$ dans la demi-période, et dans l'ordre inverse dans l'autre demi-période.
- **Pour la région 2 :** Les vecteurs sont appliqués dans l'ordre $T_y/4$, $T_x/4$, $T_z/2$, $T_y/4$, $T_x/4$ dans la demi-période, et dans l'ordre inverse dans l'autre demi-période.
- **Pour la région 3 :** Les vecteurs sont appliqués dans l'ordre $T_z/4$, $T_x/2$, $T_y/2$, $T_z/4$ dans la demi-période, et dans l'ordre inverse dans l'autre demi-période.
- **Pour la région 4 :** Les vecteurs sont appliqués dans l'ordre $T_z/4$, $T_x/2$, $T_y/2$, $T_z/4$ dans la demi-période, et dans l'ordre inverse dans l'autre demi-période.

Les tableaux 4.1, 4.2, 4.3 et 4.4 résument l'ordre d'application de chaque vecteur pour chaque région.

	Période T_s													
	Demi période ($T_s/2$)							Demi période ($T_s/2$)						
Région 1	$\frac{T_z}{6}$	$\frac{T_x}{4}$	$\frac{T_y}{4}$	$\frac{T_z}{6}$	$\frac{T_x}{4}$	$\frac{T_y}{4}$	$\frac{T_z}{6}$	$\frac{T_z}{6}$	$\frac{T_y}{4}$	$\frac{T_x}{4}$	$\frac{T_z}{6}$	$\frac{T_y}{4}$	$\frac{T_x}{4}$	$\frac{T_z}{6}$

TABLE 4.1: Arrangement des durées T_x, T_y et T_z pour la région 1

	Période T_s									
	Demi période $(T_s/2)$					Demi période $(T_s/2)$				
Région 2	$\frac{T_y}{4}$	$\frac{T_x}{4}$	$\frac{T_z}{2}$	$\frac{T_y}{4}$	$\frac{T_x}{4}$	$\frac{T_x}{4}$	$\frac{T_y}{4}$	$\frac{T_z}{2}$	$\frac{T_x}{4}$	$\frac{T_y}{4}$

TABLE 4.2: Arrangement des durées T_x, T_y et T_z pour la région 2

	Période T_s							
	Demi période $(T_s/2)$				Demi période $(T_s/2)$			
Région 3	$\frac{T_z}{4}$	$\frac{T_x}{2}$	$\frac{T_y}{2}$	$\frac{T_z}{4}$	$\frac{T_z}{4}$	$\frac{T_y}{2}$	$\frac{T_x}{2}$	$\frac{T_z}{4}$

TABLE 4.3: Arrangement des durées T_x, T_y et T_z pour la région 3

	Période T_s							
	Demi période $(T_s/2)$				Demi période $(T_s/2)$			
Région 4	$\frac{T_z}{4}$	$\frac{T_x}{2}$	$\frac{T_y}{2}$	$\frac{T_z}{4}$	$\frac{T_z}{4}$	$\frac{T_y}{2}$	$\frac{T_x}{2}$	$\frac{T_z}{4}$

TABLE 4.4: Arrangement des durées T_x, T_y et T_z pour la région 4

4.4.5 Codage des états à appliquer

Du tableau 3.4, on remarque que l'état de chaque interrupteur est symétrique par rapport à la demi période d'échantillonnage avec un seul front montant ou descendent, donc on va exploiter cette caractéristique pour implémenter l'algorithme de la SVM en utilisant le module PWM de la carte ARDUINO DUE.

Le codage du tableau des états se fait comme suit :

Pour chaque interrupteur, on code le numéro du segment ou le front se produit avec un signe positif pour un front montant, ce qui correspond à une polarité positive du module PWM, et un signe négatif pour un front descendent, ce qui correspond à une polarité négative du module PWM.
Par exemple, la Figure 4.34 indique la forme des fonctions logiques associées aux interrupteurs des demi-bras supérieurs, sur une période d'échantillonnage dans la première région du premier secteur du diagramme vectoriel. On remarque sur la figure que l'interrupteur S_{11} est activé après le segment 4, donc on code + 4. De même pour les autres interrupteurs on trouve :

– Pour l'interrupteur S_{21} : on code +1.
– Pour l'interrupteur S_{12} : on code +5.
– Pour l'interrupteur S_{22} : on code +2.
– Pour l'interrupteur S_{13} : on code +6.
– Pour l'interrupteur S_{23} : on code +3.

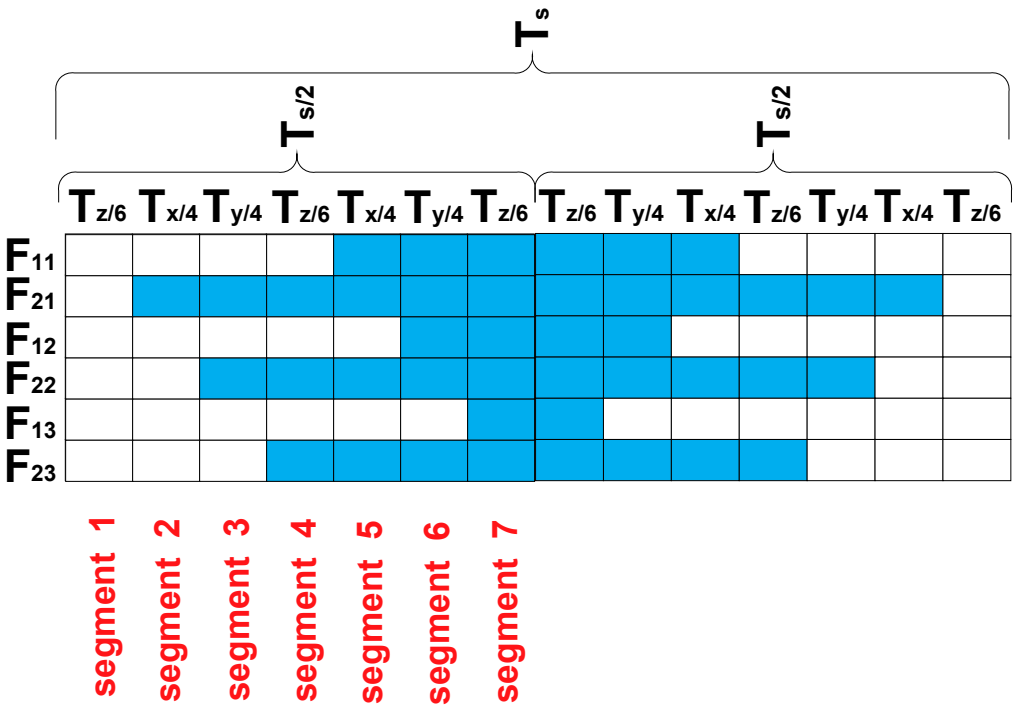


FIGURE 4.34: Signaux des interrupteurs pour la première région du premier secteur

La figure 4.35 montre les signaux des interrupteurs pour la quatrième région du sixième secteur. On remarque que l'interrupteur S_{11} passe de l'état activé a l'état désactivé *(front descendent)* après le segment 3, donc on code - 3. L'interrupteur S_{21} est activé sur toute la demi période, donc on suppose qu'il passe a l'état désactivé après le segment 4 et on code - 4. De même pour tous les autres interrupteurs on trouve :

– Pour l'interrupteur S_{11} : on code -3.
– Pour l'interrupteur S_{21} : on code -4.
– Pour l'interrupteur S_{12} : on code +4.
– Pour l'interrupteur S_{22} : on code -1.
– Pour l'interrupteur S_{13} : on code +4.
– Pour l'interrupteur S_{23} : on code -2.

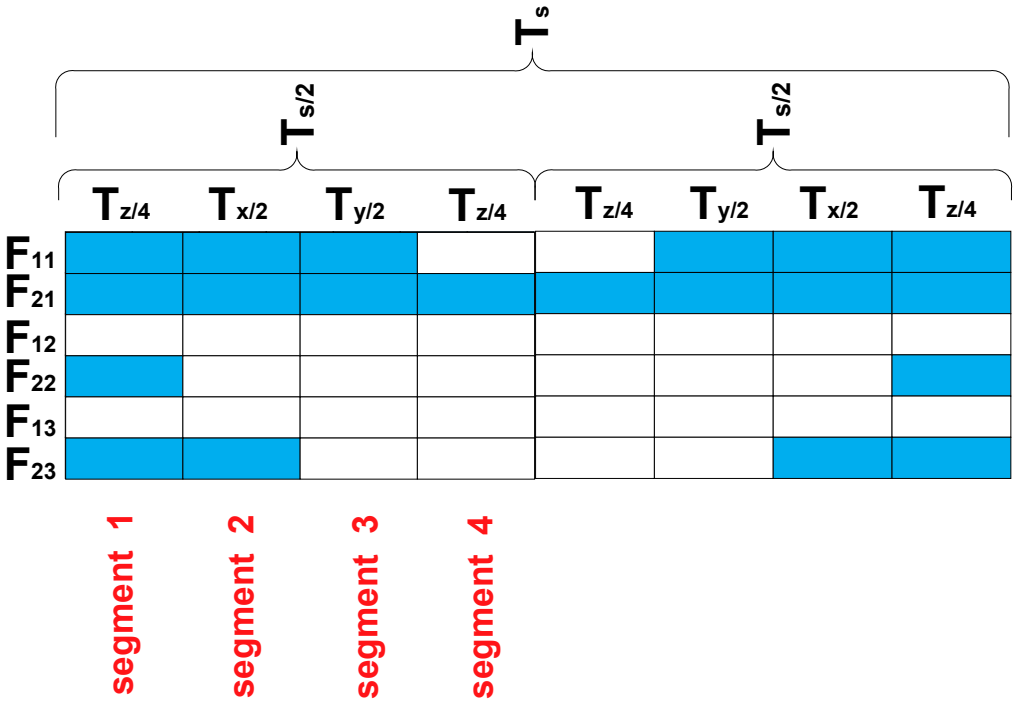


FIGURE 4.35: Signaux des interrupteurs pour la quatrième région du sixième secteur

Le codage complet est résumé dans les tableaux suivants :

	Région 1	Région 2	Région 3	Région 4
Secteur 1	4	2	1	1
Secteur 2	-2	-1	-2	4
Secteur 3	6	5	4	4
Secteur 4	-1	5	4	4
Secteur 5	5	4	4	2
Secteur 6	-3	-3	-3	-3

FIGURE 4.36: Tableau de codage pour l'interrupteur S_{11}

	Région 1	Région 2	Région 3	Région 4
Secteur 1	1	-5	-4	-4
Secteur 2	-5	-4	-4	-2
Secteur 3	3	3	3	3
Secteur 4	-4	-2	-1	-1
Secteur 5	2	1	2	-4
Secteur 6	6	-5	-4	-4

FIGURE 4.37: Tableau de codage pour l'interrupteur S_{21}

	Région 1	Région 2	Région 3	Région 4
Secteur 1	5	4	4	2
Secteur 2	-3	-3	-3	-3
Secteur 3	4	2	1	1
Secteur 4	-2	-1	-2	4
Secteur 5	6	5	4	4
Secteur 6	-1	5	4	4

FIGURE 4.38: Tableau de codage pour l'interrupteur S_{12}

	Région 1	Région 2	Région 3	Région 4
Secteur 1	2	1	2	-4
Secteur 2	-6	-5	-4	-4
Secteur 3	1	-5	-4	-4
Secteur 4	-5	-4	-4	-2
Secteur 5	3	3	3	3
Secteur 6	-4	-2	-1	-1

TABLE 4.5: Tableau de codage pour l'interrupteur S_{22}

	Région 1	Région 2	Région 3	Région 4
Secteur 1	6	5	4	4
Secteur 2	-1	5	4	4
Secteur 3	5	4	4	2
Secteur 4	-3	-3	-3	-3
Secteur 5	4	2	1	1
Secteur 6	-2	-1	-2	4

TABLE 4.6: Tableau de codage pour l'interrupteur S_{13}

	Région 1	Région 2	Région 3	Région 4
Secteur 1	3	3	3	3
Secteur 2	-4	-2	-1	-1
Secteur 3	2	1	2	-4
Secteur 4	-6	-5	-4	-4
Secteur 5	1	-5	-4	-4
Secteur 6	-5	-4	-4	-2

TABLE 4.7: Tableau de codage pour l'interrupteur S_{23}

En comparant le tableau de l'interrupteur S_{11} (tableau 4.36) et le tableau de l'interrupteur S_{21} (tableau 4.37), on trouve que ce dernier se déduit par une rotation circulaire des lignes du premier tableau (rotation circulaire de 3 lignes) avec une inversion de signe.
De plus, puisque les tensions engendrées par les deux autres bras **2** et **3** sont identiques a la tension engendré par le bras **1** avec un déphasage de $\frac{-2.\pi}{3}$ et $\frac{2.\pi}{3}$ respectivement, les tableaux des interrupteurs S_{12} et S_{13} *(respectivement S_{22} et S_{23})* se déduisent du tableau de l'interrupteur S_{11} *(respectivement S_{21})* par une simple rotation circulaire des lignes, par un pas de : $\frac{-2.\pi}{3} \times \frac{6}{2.\pi} =$ - 2 et $\frac{-2.\pi}{3} \times \frac{6}{2.\pi} =$ + 2 respectivement.

Donc on va exploiter cette redondance d'information pour coder le tableau en 24 emplacements mémoire.

4.4.6 Implémentation sur la carte ARDUINO DUE

On va utiliser les broches *(pins)* 35,37, 39, 41, 44 et 45 de la carte ARDUINO DUE, qui sont connectés aux chaines PWMH0, PWMH1, PWMH2, PWMH3, PWMH5 et PWMH6 respectivement, pour générer les signaux des interrupteurs S_{11}, S_{21}, S_{12}, S_{22}, S_{13} et S_{23} respectivement de l'onduleur à trois niveaux. Les chaines PWM de la carte ARDUINO DUE doivent être synchroniser par le même temporisateur *(timer)*.

Pour avoir une fréquence d'échantillonnage de *4 KHz*, et sachant que la fréquence d'horloge de la carte *ARDUINO DUE* est de *84 MHz*, et que les chaines sont configurées en alignement centré (*Center Aligned*), On doit charger les registres de période *PWM_CPRD0*, *PWM_CPRD1* et *PWM_CPRD2* par la valeur :

$$PWM\_CPRD0 = PWM\_CPRD1 = PWM\_CPRD2 = \frac{84000000}{2 \times 4000} = 10500 \qquad (4.2)$$

On va activer la génération automatique des interruptions à chaque début de période, c'est-à-dire a une fréquence de *4 KHz*, ce qui permet la non saturation du processeur et l'exécution de l'algorithme en une seule subroutine.

Au sous programme d'interruption on va :

- Calculer l'argument du vecteur tension de référence a l'aide d'un temporisateur (*timer*) de la carte ARDUINO DUE.
- Déterminer le secteur.
- Calculer les composants $\hat{V}_d$ et $\hat{V}_q$ du vecteur tension de référence.
- Déterminer la région.
- Déterminer les durées relatives d_x, d_y et d_z en fonction du région.
- Déterminer la polarité et le rapport cyclique de chaque chaine *PWMx (x=0,1,2,3,4 et 5)* a partir du tableau de codage.
- Chargement du registre tampon *PWM_CDTYUPDx (x=0,1,2,3,4 et 5)* pour actualiser le rapport cyclique de chaque chaine *PWMx (x=0,1,2,3,4 et 5)* dans le début du prochaine période d'échantillonnage. Cette manière d'implémentation permet d'avoir une durée d'impulsion très précis et d'exploiter toute la durée du période d'échantillonnage pour le calcul des paramètres du prochaine période d'échantillonnage.

Le diagramme 4.39 résume l'implémentation de la SVM à trois niveaux sur la carte ARDUINO DUE.

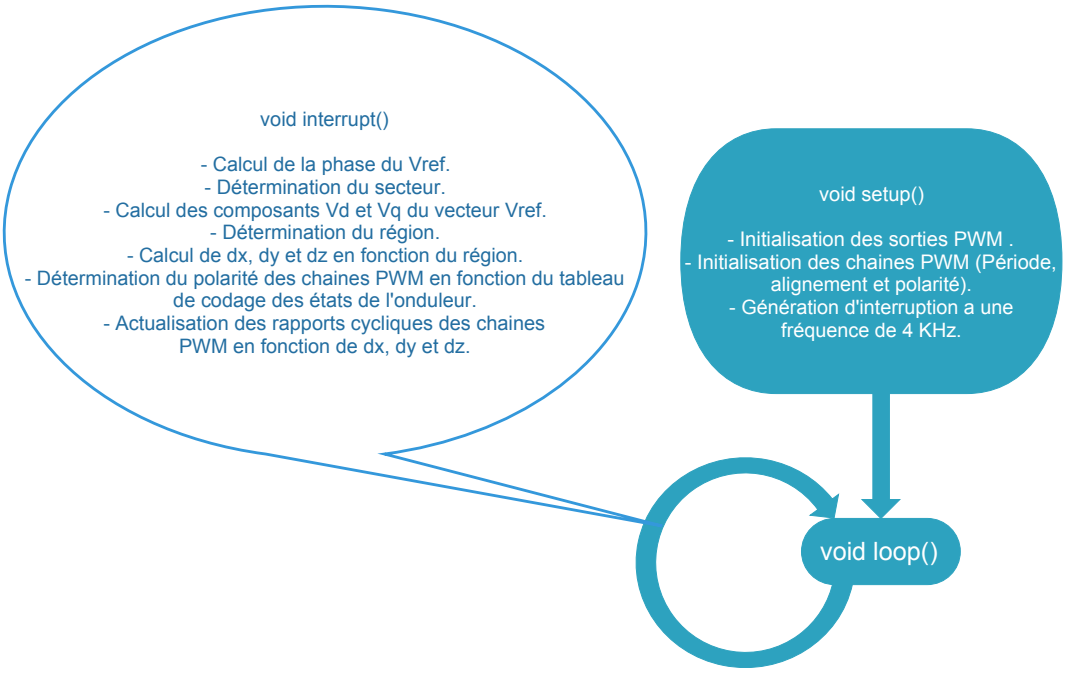


FIGURE 4.39: Diagramme SVM 3 niveaux

4.4.7 Signaux de commande

On va visualiser les signaux de commande de la carte ARDUINO DUE après l'implémentation de la SVM à trois niveaux.

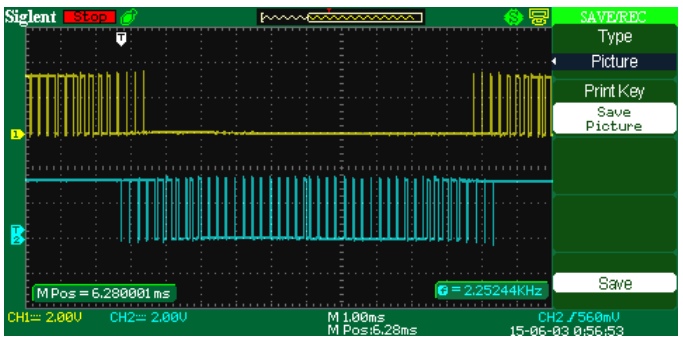


FIGURE 4.40: Signaux de commande des interrupteurs S_{11} et S_{21} : expérimentale

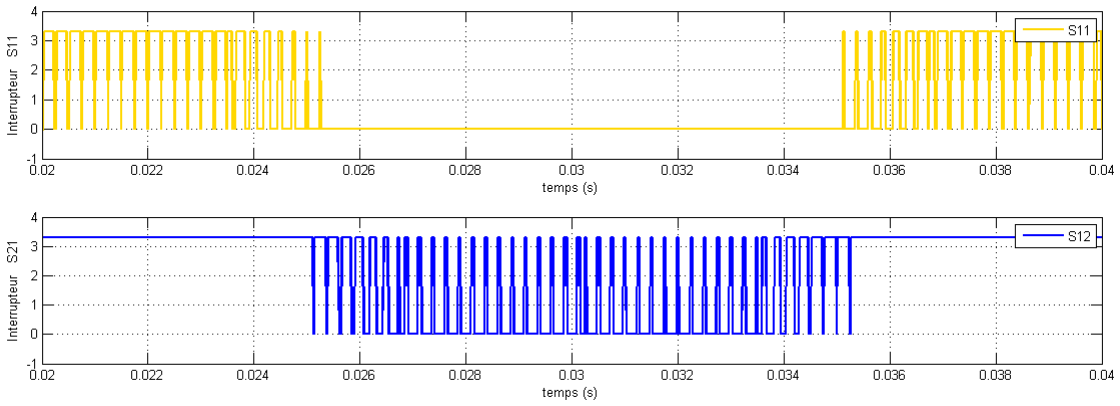


FIGURE 4.41: Signaux de commande des interrupteurs S_{11} et S_{21} : simulation

La figure 4.40 montre les signaux de commande des interrupteurs S_{11} *(chaine 1 de l'oscilloscope)* et S_{21} *(chaine 2 de l'oscilloscope)* du premier bras. On remarque que ces signaux peuvent prendre trois états différentes :

1- Interrupteur S_{11} a l'état 1 et interrupteur S_{21} a l'état 1, ce qui correspondre a l'état P du bras de l'onduleur (figure 4.42).
2- Interrupteur S_{11} a l'état 0 et interrupteur S_{21} a l'état 1, ce qui correspondre a l'état O du bras de l'onduleur (figure 4.43).
3- Interrupteur S_{11} a l'état 0 et interrupteur S_{21} a l'état 0, ce qui correspondre a l'état N du bras de l'onduleur (figure 4.44).

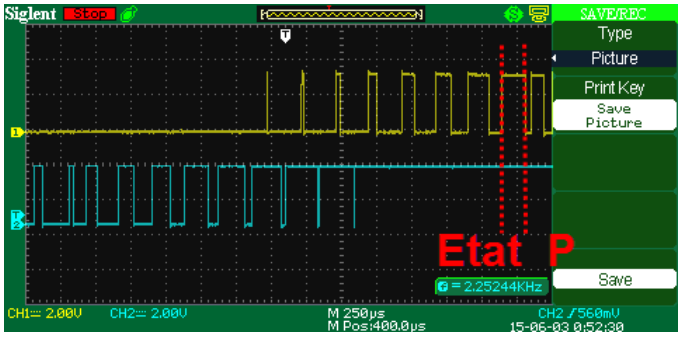


FIGURE 4.42: Etat P

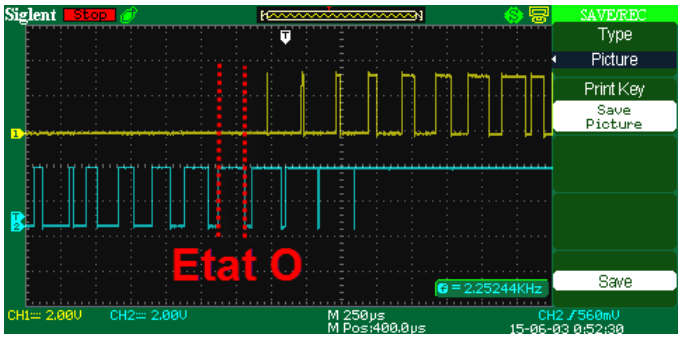


FIGURE 4.43: Etat O

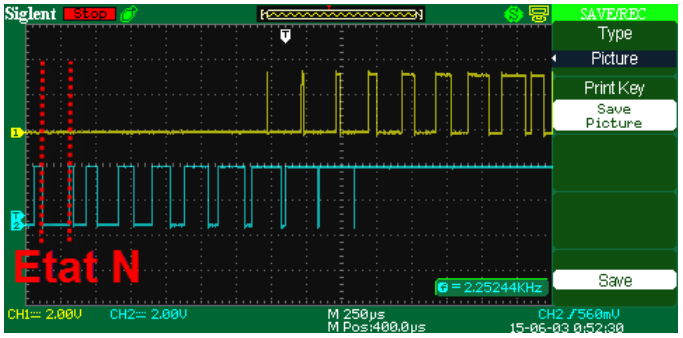


FIGURE 4.44: Etat N

4.4.8 Méthode de validation

On va récupérer les signaux de commande générées par la carte ARDUINO DUE, pour les interrupteurs S_{11} et S_{21} *(interrupteurs supérieurs du premier bras)* sous la forme d'un fichier d'extension *.csv*. l'état des deux autre bras se déduit par un décalage de phase de - $2\pi/3$ et + $2\pi/3$ respectivement.

Ces signaux de commande vont être utiliser sur le modèle d'un onduleur à trois niveaux à structure NPC idéale, sous l'environnement MATLAB. Le modèle utilisé suppose qu'on a une charge triphasée parfaitement équilibrée, et une tension d'alimentation de 700 V.

4.4.9 Résultats obtenus

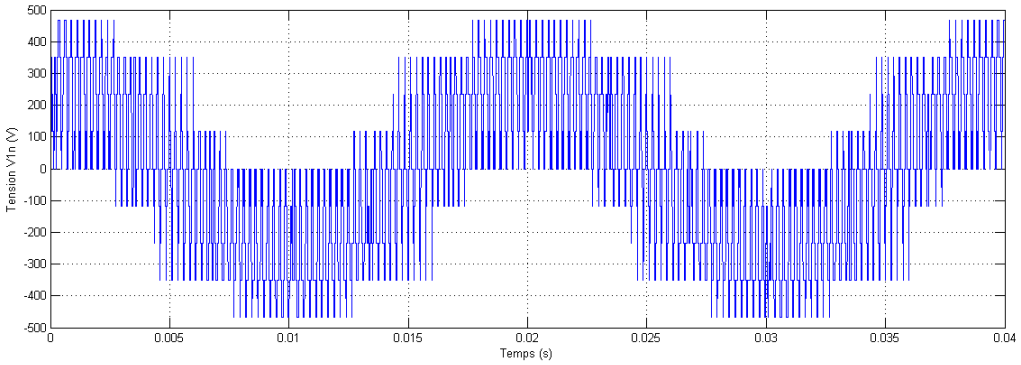


FIGURE 4.45: Tension V_{1n} (r=0.8, F_e=4 KHz)

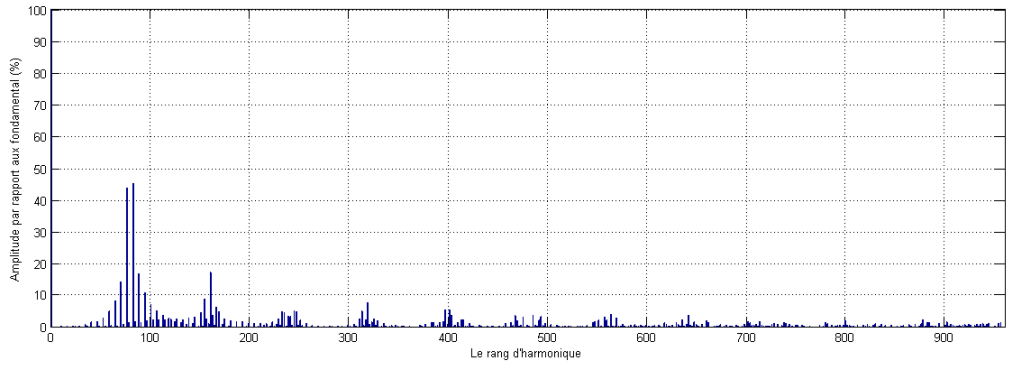


FIGURE 4.46: Spectre du tension V_{1n} (r=0.8, F_e=4 KHz)

La figure 4.45 montre la tension de sortie V_{1n} pour r=0.8. On remarque que cette tension suit une référence sinusoïdale de fréquence 50 Hz. De plus, elle a pris neuf valeurs de tensions différentes : 467 V ($2V_{dc}/3$), 350 V ($V_{dc}/2$), 234 V ($V_{dc}/3$),167 V ($V_{dc}/6$),0 V, -167 V (- $V_{dc}/6$), - 234 V (- $V_{dc}/3$), - 350 V (- $V_{dc}/2$) et - 467 V (- $2V_{dc}/3$).

La figure 4.46 montre l'analyse spectrale de la tension de sortie V_{1n} pour r=0.8. On voit bien que les harmoniques de tension se regroupent autour des multiples de la fréquence 4 KHz (La fréquence d'échantillonnage).

4.4.10 Courbe de réglage

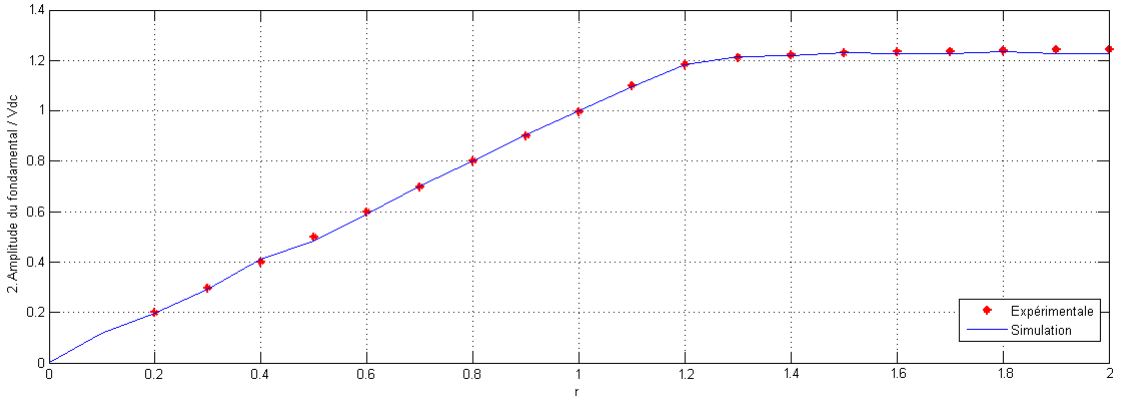


FIGURE 4.47: Courbe de réglage

La figure 4.47 montre que les résultats obtenus par expérimentation coïncide avec les résultats de simulation.
On remarque que la valeur relative du fondamental de la tension de sortie par rapport a la tension d'alimentation V_{dc} augmente linéairement avec l'augmentation du r jusqu'à la valeur $r = \frac{2\sqrt{3}}{3} \approx 1.15$.
Au delà de cette valeur, le vecteur tension de référence r se situe en dehors du cercle délimité par l'hexagone externe du diagramme vectoriel de l'onduleur à trois niveaux.

4.4.11 Courbe du taux de distorsion d'harmonique

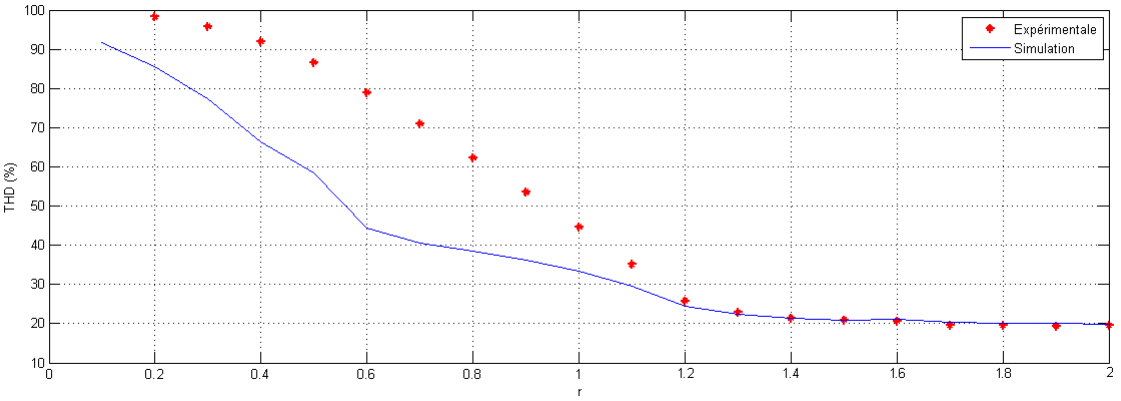


FIGURE 4.48: Courbe de THD(%)

La figure 4.48 montre la variation du THD(%) en fonction de r. On remarque que le taux de distorsion d'harmonique diminue avec l'augmentation de r. On constate que les valeurs expérimentaux du THD(%) sont supérieurs aux valeurs obtenus par simulation lorsque r<1.2. Au delà de cette valeur, les valeurs expérimentaux et théoriques sont identiques.

4.4.12 Influence de la fréquence d'échantillonnage

Comme pour le cas de la SVM à deux niveaux, on va augmenter la fréquence d'échantillonnage pour un r=1 et une fréquence de tension de sortie de 50 Hz.

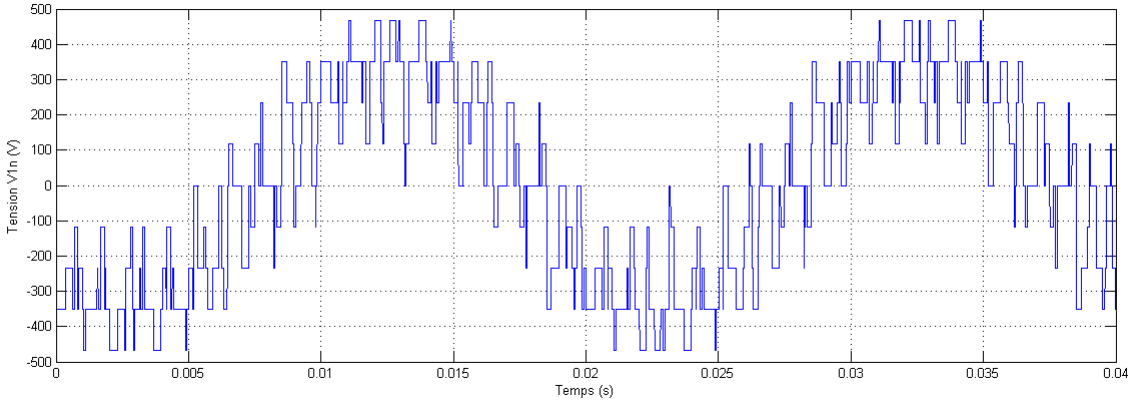


FIGURE 4.49: Tension V_{1n} (r=1, F_e=1 KHz)

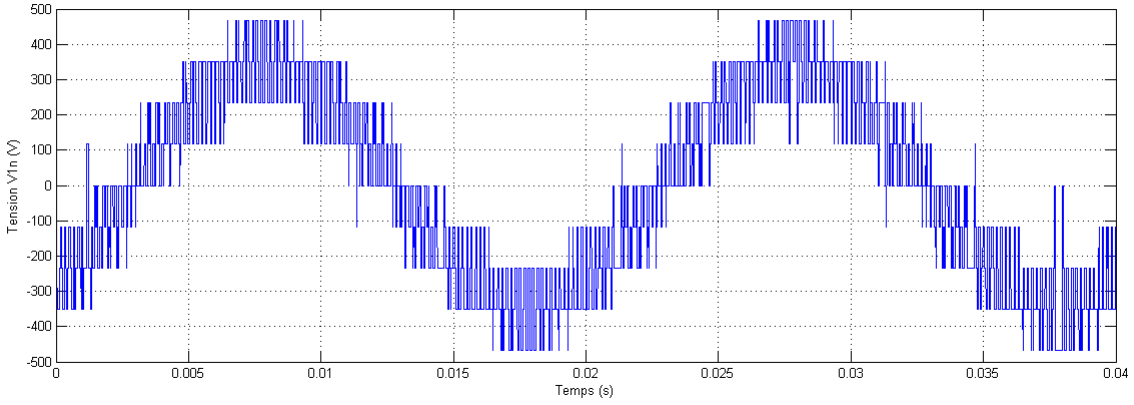


FIGURE 4.50: Tension V_{1n} (r=1, F_e=3 KHz)

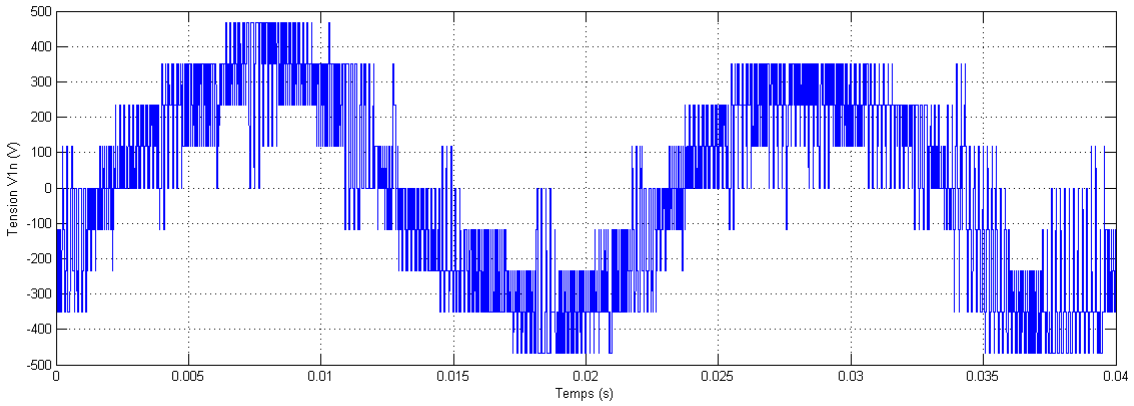


FIGURE 4.51: Tension V_{1n} (r=1, F_e=6 KHz)

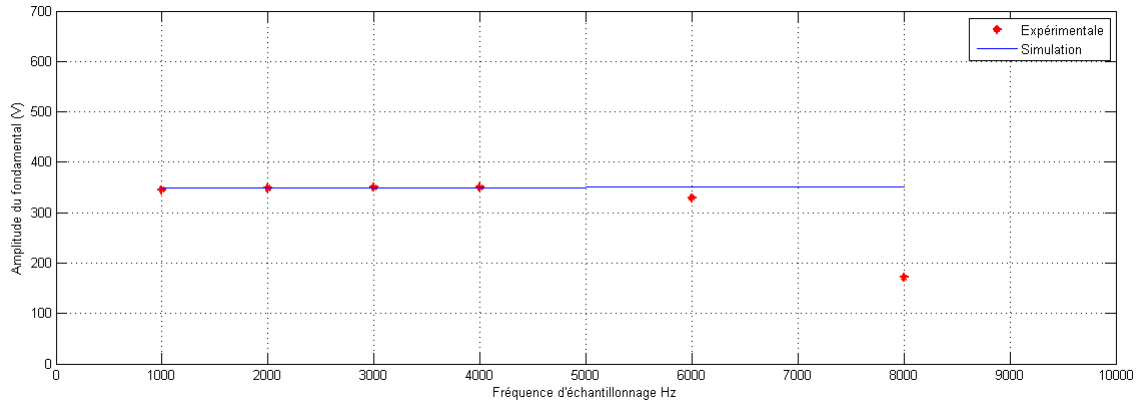


FIGURE 4.52: fondamental de la tension V_{1n} en fonction de F_e

La figure 4.52 montre la valeur de la fondamental du tension de sortie V_{1n} en fonction de la fréquence d'échantillonnage F_e. On remarque que les valeurs expérimentaux coïncident avec les résultats de simulation pour F_e<4 KHz.
A partir d'une fréquence d'échantillonnage de 6 KHz, Les valeurs du fondamental obtenus expérimentalement divergent des valeurs théoriques de simulation, avec une distorsion de la forme du tension V_{1n} (figure 4.51).

Donc on peut dire que la limitation de la SVM à trois niveaux implémentée sur la carte ARDUINO DUE se tourne autour d'une fréquence d'échantillonnage de 4 KHz, qui est plus bas que celle de la SVM à deux niveaux, car la complexité de l'algorithme est plus grande.

4.5 Conclusion

L'expérimentation montre qu'on a réussi à implémenter la SVM à deux niveaux en utilisant le module PWM de l'ARDUINO DUE.

On a exploité le même module de l'ARDUINO DUE pour implémenter la SVM à trois niveaux. Le codage des différents états de l'onduleur a permis la minimisation du nombre total de commutations des interrupteurs, ce qui réduit les pertes en puissance.

Pour les deux algorithmes, on a utilisé une seule interruption à chaque période d'échantillonnage, ce qui libère le processeur de la carte ARDUINO DUE pour réaliser d'autre tâches.

La visualisation des tensions de sortie, et le tracé de la courbe du réglage et du THD(%) montrent une convergence entre les résultats expérimentaux et ceux de simulation, dans la limite des erreurs expérimentaux et des hypothèses théoriques.

Finalement, on a trouvé que la fréquence d'échantillonnage limite était de 7 KHz pour la SVM à deux niveaux, et 4 KHz pour la SVM à trois niveaux.

Conclusion générale

L'objectif principal de ce mémoire consiste à implémenter l'algorithme de la modulation vectorielle pour les onduleurs triphasée à deux niveaux et à trois niveaux à structure NPC, en utilisant la carte ARDUINO DUE.

Nous avons donc commencé par la présentation de la carte utilisée dans notre travail. On a vu que cette carte est la première carte ARDUINO qui bénéfice d'un core ARM 32-bit, avec de très bonnes fonctionnalités matériels. L'environnement de développement intégré associé à leur utilisation est un environnement compact, qui combine entre la simplicité et l'efficacité.

Dans le deuxième chapitre, on a vu le modèle de connaissance et de commande de l'onduleur triphasé à deux niveaux. On a constaté que ce convertisseur génère une tension de sortie qui approxime sa référence sinusoïdale en forme par cinq niveaux de tensions différents.

Dans le troisième chapitre, on a vu le modèle de connaissance et de commande de l'onduleur triphasé à trois niveaux à structure NPC. On a constaté que ce convertisseur génère une tension de sortie qui approxime sa référence sinusoïdale par neuf niveaux de tensions différents. Une comparaison du taux de distorsion d'harmonique THD(%) montre que ces convertisseurs génèrent une tension de sortie de qualité meilleure que celle générer par les onduleurs à deux niveaux au prix de la complexité de l'algorithme de commande et du coût.

Le quatrième chapitre représente le cœur de notre travail. On a vu qu'on a exploiter le module PWM de la carte ARDUINO DUE pour implémenter la SVM à deux niveaux, toute en minimisant le temps de calcul, et les ressources matériels de la carte ARDUINO DUE par l'utilisation des interruptions, ce qui libère le processeur de la carte pour réaliser d'autre tâches, comme la communication avec le PC, le diagnostic et le debugging.
On a étendue notre travail précédent aux onduleurs à trois niveaux à structure NPC, car on a pu exploité le même module pour optimiser l'implémentation de la SVM à trois niveaux.
Pour les deux algorithmes développés (SVM à deux niveaux et SVM à trois niveaux), On a réussi à coder les différents états de l'onduleur de manière à minimiser le nombre total de commutations des interrupteurs, ce qui réduit les pertes en puissance dans les onduleurs. On a réussi aussi à générer des impulsions symétriques dans chaque période, ce qui réduit les harmoniques de la tension de sortie.
Les résultats expérimentaux obtenus montrent la validité de notre travail.

En résumé, on peut dire qu'on a réussi à implémenter l'algorithme de la modulation vectorielle à deux niveaux et à trois niveaux toute en réalisant les objectifs suivants :

- Optimisation des ressources matérielles utilisées (circuit de commande).
- Réduction des pertes en puissance par la minimisation du nombre de commutations des interrupteurs.
- Réduction des harmoniques par la génération des impulsions symétriques.

Le travail effectué ouvre les perspectives suivantes :

- Adapter les algorithmes développés sur d'autre microprocesseurs de CORE ARM.
- Exploiter le module de génération de temps mort (Dead-Time Generator) de la carte ARDUINO DUE pour réaliser le retard entre les interrupteurs du même bras, ce qui enlève la nécessité de l'utilisation du circuit de génération de temps morts (réduction du coût).
- Développer une interface d'utilisateurs qui permet la commande du taux de réglage.
- Réalisation d'un onduleur à trois niveaux à structure NPC.

Bibliographie

[1] AT91SAM ARM-based Flash MCU, SAM3X SAM3A Series datasheet.

[2] AMROUS HAMZA, ADJIR NABIL. "Réalisation de la commande triangulo-sinusoidale a une porteuse triangulaire unipolaire". Projet de fin d'étude, ENP, 2003.

[3] S.Niveditha, B.N.Kartheek, P.S.D.M.Chandana. "An optimized code for space vector pwm for a two level voltage source inverter". International Journal of Science and Modern Engineering (IJISME), Avril 2013.

[4] Weixing Feng. "Space vector modulation for three-level neutral point clamped inverter". Thèse de Master, Ryerson University, 2004.

[5] H.CHEKIREB. "La conversion continu-alternatif". Cours d'électronique de puissance, ENP, 2012.

[6] J.Holtz. "Pulsewidth modulation - a survey". Industrial Electronics, IEEE Transactions on, 39(5):410-420, Octobre 1992.

[7] R.Jobing, F.S.van der Merwe, and M.J.Kamper. "Digital implementation of bus clamped space vector modulation". Energy Conversion, IEEE Transactions on, 9(2):344-348, Juin 1994.

[8] AKEL Fethi, KERMADI Mostefa, BERKOUK EL Madjid. "Real time implementation of space vector pulse width modulation using arduino uno board". 9 éme conférence sur le Génie Electrique, Bordj El Bahri, Algérie, Avril 2015.

[9] Djaafer Lalili. "MLI Vectorielle et Commande Non Linéaire du Bus Continu des Onduleurs Multiniveaux". Thèse de Doctorat, ENP, 2009.

[10] Meng Yeong Lee. "Three-level Neutral-point-clamped Matrix Converter Topology". Thèse de Doctorat, University of Nottingham, 2009.

[11] Yakoub Saadi, Mohamed Tadjine, Fethi Akel, Ali Benachour, Islam Ziouani, Mohammed Faouzi Nadjoui, Mostefa Kermadi, El Madjid Berkouk. " Overview of arduino development platform and its applications in control systems". 2nd International Conference on Power Electronics and their Applications (ICPEA 2015), Djelfa, Algérie, Mars 2015.

[12] Charles Severance. " Massimo banzi: Building arduino" . Computer, Janvier, 2014.

[13] H.W.van der Broeck, H.C.Skudelny, and G.V.Stanke. " Analysis and realization of a pulsewidth modulator based on voltage space vectors" . Industriel Applications, IEEE Transactions on, 24(1):142-150, Janvier 1988.

[14] www.arduino.com.

[15] www.SEMIKRON.com.

[16] www.sparkfun.com.

[17] www.wikipidia.com.

Annexe : Atmel SAM3X8E ARM Cortex-M3 CPU DATASHEET

Features

- Core
  - ARM® Cortex®-M3 revision 2.0 running at up to 84 MHz
  - Memory Protection Unit (MPU)
  - Thumb®-2 instruction set
  - 24-bit SysTick Counter
  - Nested Vector Interrupt Controller
- Memories
  - From 256 to 512 Kbytes embedded Flash, 128-bit wide access, memory accelerator, dual bank
  - From 32 to 100 Kbytes embedded SRAM with dual banks
  - 16 Kbytes ROM with embedded bootloader routines (UART, USB) and IAP routines
  - Static Memory Controller (SMC): SRAM, NOR, NAND support. NAND Flash controller with 4-kbyte RAM buffer and ECC
- System
  - Embedded voltage regulator for single supply operation
  - POR, BOD and Watchdog for safe reset
  - Quartz or ceramic resonator oscillators: 3 to 20 MHz main and optional low power 32.768 kHz for RTC or device clock.
  - High precision 8/12 MHz factory trimmed internal RC oscillator with 4 MHz Default Frequency for fast device startup
  - Slow Clock Internal RC oscillator as permanent clock for device clock in low power mode
  - One PLL for device clock and one dedicated PLL for USB 2.0 High Speed Mini Host/Device
  - Temperature Sensor
  - Up to 17 peripheral DMA (PDC) channels and 6-channel central DMA plus dedicated DMA for High-Speed USB Mini Host/Device and Ethernet MAC
- Low Power Modes
  - Sleep and Backup modes, down to 2.5 µA in Backup mode.
  - Backup domain: VDDBU pin, RTC, eight 32-bit backup registers
  - Ultra Low-power RTC
- Peripherals
  - USB 2.0 Device/Mini Host: 480 Mbps, 4-kbyte FIFO, up to 10 bidirectional Endpoints, dedicated DMA
  - Up to 4 USARTs (ISO7816, IrDA®, Flow Control, SPI, Manchester and LIN support) and one UART
  - 2 TWI (I2C compatible), up to 6 SPIs, 1 SSC (I2S), 1 HSMCI (SDIO/SD/MMC) with up to 2 slots
  - 9-Channel 32-bit Timer/Counter (TC) for capture, compare and PWM mode, Quadrature Decoder Logic and 2-bit Gray Up/Down Counter for Stepper Motor
  - Up to 8-channel 16-bit PWM (PWMC) with Complementary Output, Fault Input, 12-bit Dead Time Generator Counter for Motor Control
  - 32-bit Real Time Timer (RTT) and RTC with calendar and alarm features
  - 16-channel 12-bit 1Msps ADC with differential input mode and programmable gain stage
  - One 2-channel 12-bit 1 MSPS DAC
  - One Ethernet MAC 10/100 (EMAC) with dedicated DMA
  - Two CAN Controller with eight Mailboxes
  - One True Random Number Generator (TRNG)
  - Write Protected Registers
- I/O
  - Up to 103 I/O lines with external interrupt capability (edge or level sensitivity), debouncing, glitch filtering and on-die Series Resistor Termination
  - Up to Six 32-bit Parallel Input/Outputs (PIO)
- Packages
  - 100-lead LQFP, 14 x 14 mm, pitch 0.5 mm
  - 100-ball LFBGA, 9 x 9 mm, pitch 0.8 mm
  - 144-lead LQFP, 20 x 20 mm, pitch 0.5 mm
  - 144-ball LFBGA, 10 x 10 mm, pitch 0.8 mm

AT91SAM ARM-based Flash MCU

SAM3X SAM3A Series

11057B–ATARM–28-May-12

1. SAM3X/A Description

Atmel's SAM3X/A series is a member of a family of Flash microcontrollers based on the high performance 32-bit ARM Cortex-M3 RISC processor. It operates at a maximum speed of 84 MHz and features up to 512 Kbytes of Flash and up to 100 Kbytes of SRAM. The peripheral set includes a High Speed USB Host and Device port with embedded transceiver, an Ethernet MAC, 2x CANs, a High Speed MCI for SDIO/SD/MMC, an External Bus Interface with NAND Flash controller, 5x UARTs, 2x TWIs, 4x SPIs, as well as 1 PWM timer, 9x general-purpose 32-bit timers, an RTC, a 12-bit ADC and a 12-bit DAC.

The SAM3X/A series is ready for capacitive touch thanks to the QTouch library, offering an easy way to implement buttons, wheels and sliders.

The SAM3X/A architecture is specifically designed to sustain high speed data transfers. It includes a multi-layer bus matrix as well as multiple SRAM banks, PDC and DMA channels that enable it to run tasks in parallel and maximize data throughput.

It operates from 1.62V to 3.6V and is available in 100- and 144-pin QFP and LFBGA packages.

The SAM3X/A devices are particularly well suited for networking applications: industrial and home/building automation, gateways.

1.1 Configuration Summary

The SAM3X/A series devices differ in memory sizes, package and features list. Table 1-1 below summarizes the configurations.

Table 1-1. Configuration Summary

Feature	SAM3X8E	SAM3X8C	SAM3X4E	SAM3X4C	SAM3A8C	SAM3A4C
Flash	2 x 256 Kbytes	2 x 256 Kbytes	2 x 128 Kbytes	2 x 128 Kbytes	2 x 256 Kbytes	2 x 128 Kbytes
SRAM	64 + 32 Kbytes	64 + 32 Kbytes	32 + 32 Kbytes	32 + 32 Kbytes	64 + 32 Kbytes	32 + 32 Kbytes
Nand Flash Controller (NFC)	Yes	-	Yes	-	-	-
NFC SRAM(1)	4K bytes	-	4K bytes	-	-	-
Package	LQFP144 LFBGA144	LQFP100 LFBGA100	LQFP144 LFBGA144	LQFP100 LFBGA100	LQFP100 LFBGA100	LQFP100 LFBGA100
Number of PIOs	103	63	103	63	63	63
SHDN Pin	Yes	No	Yes	No	No	No
EMAC	MII/RMII	RMII	MII/RMII	RMII	-	-
External Bus Interface	16-bit data, 8 chip selects, 23-bit address	-	16-bit data, 8 chip selects, 23-bit address	-	-	-
SDRAM Controller	-	-	-	-	-	-
Central DMA	6	4	6	4	4	4
12-bit ADC	16 ch.(2)	16 ch.(2)	16 ch.(2)	16 ch.(2)	16 ch.(2)	16 ch.(2)
12-bit DAC	2 ch.	2 ch.	2 ch.	2 ch.	2 ch.	2 ch.
32-bit Timer	9(4)	9(5)	9(4)	9(5)	9(4)	9(4)
PDC Channels	17	15	17	15	15	15

Table 1-1. Configuration Summary (Continued)

Feature	SAM3X8E	SAM3X8C	SAM3X4E	SAM3X4C	SAM3A8C	SAM3A4C
USART/ UART	3/2[6]	3/1	3/2[6]	3/1	3/1	3/1
SPI [3]	1/4 + 3	1/4 + 3	1/4 + 3	1/4 + 3	1/4 + 3	1/4 + 3
HSMCI	1 slot 8 bits	1 slot 4 bits	1 slot 8 bits	1 slot 4 bits	1 slot 4 bits	1 slot 4 bits

Notes:
1. 4 Kbytes RAM buffer of the NAND Flash Controller (NFC) which can be used by the core if not used by the NFC
2. One channel is reserved for internal temperature sensor
3. 2 / 8 + 4 = Number of SPI Controllers / Number of Chip Selects + Number of USART with SPI Mode
4. 6 TC channels are accessible through PIO
5. 3 TC channels are accessible through PIO
6. USART3 in UART mode (RXD3 and TXD3 available)

Note: The SAM3X-EK evaluation kit for the SAM3X and SAM3A series is mounted with a SAM3X8H in an LFBGA217 package. This device is not commercially available.

39. Pulse Width Modulation (PWM)

39.1 Description

The PWM macrocell controls 8 channels independently. Each channel controls two complementary square output waveforms. Characteristics of the output waveforms such as period, duty-cycle, polarity and dead-times (also called dead-bands or non-overlapping times) are configured through the user interface. Each channel selects and uses one of the clocks provided by the clock generator. The clock generator provides several clocks resulting from the division of the PWM master clock (MCK).

All PWM macrocell accesses are made through registers mapped on the peripheral bus. All channels integrate a double buffering system in order to prevent an unexpected output waveform while modifying the period, the duty-cycle or the dead-times.

Channels can be linked together as synchronous channels to be able to update their duty-cycle or dead-times at the same time.

The update of duty-cycles of synchronous channels can be performed by the Peripheral DMA Controller Channel (PDC) which offers buffer transfer without processor Intervention.

The PWM macrocell provides 8 independent comparison units capable of comparing a programmed value to the counter of the synchronous channels (counter of channel 0). These comparisons are intended to generate software interrupts, to trigger pulses on the 2 independent event lines (in order to synchronize ADC conversions with a lot of flexibility independently of the PWM outputs), and to trigger PDC transfer requests.

The PWM outputs can be overridden synchronously or asynchronously to their channel counter.

The PWM block provides a fault protection mechanism with 6 fault inputs**,** capable of detecting a fault condition and to override the PWM outputs asynchronously.

For safety usage, some control registers are write-protected.

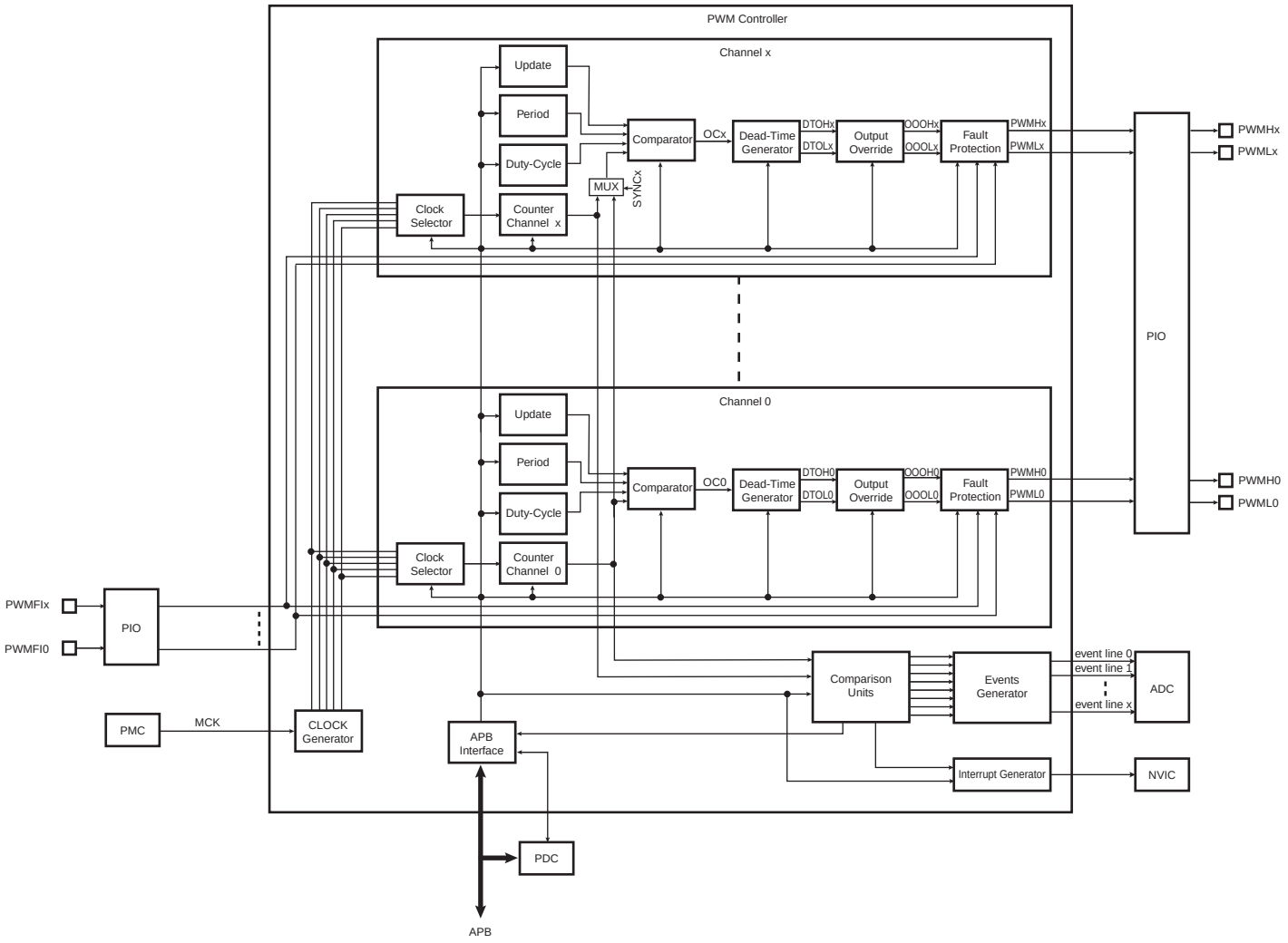
39.2 Embedded Characteristics

- 8 Channels
- Common Clock Generator Providing Thirteen Different Clocks
  - A Modulo n Counter Providing Eleven Clocks
  - Two Independent Linear Dividers Working on Modulo n Counter Outputs
- Independent Channels
  - Independent 16-bit Counter for Each Channel
  - Independent Complementary Outputs with 12-bit Dead-Time Generator (Also Called Dead-Band or Non-Overlapping Time) for Each Channel
  - Independent Enable Disable Command for Each Channel
  - Independent Clock Selection for Each Channel
  - Independent Period, Duty-Cycle and Dead-Time for Each Channel
  - Independent Double Buffering of Period, Duty-Cycle and Dead-Times for Each Channel
  - Independent Programmable Selection of The Output Waveform Polarity for Each Channel

  - Independent Programmable Center or Left Aligned Output Waveform for Each Channel
  - Independent Output Override for Each Channel
- 4 2-bit Gray Up/Down Channels for Stepper Motor Control
- Synchronous Channel Mode
  - Synchronous Channels Share the Same Counter
  - Mode to Update the Synchronous Channels Registers after a Programmable Number of Periods
  - Synchronous Channels Supports Connection of one Peripheral DMA Controller Channel (PDC) Which Offers Buffer Transfer Without Processor Intervention To Update Duty-Cycle Registers
- 2 Independent Events Lines Intended to Synchronize ADC Conversions
- 8 Comparison Units Intended to Generate Interrupts, Pulses on Event Lines and PDC Transfer Requests
- 6 Programmable Fault Inputs Providing an Asynchronous Protection of PWM Outputs
  - User Driven through PIO inputs
  - PMC Driven when Crystal Oscillator Clock Fails
  - ADC Controller Driven through Configurable Comparison Function
  - Timer/Counter Driven through Configurable Comparison Function
- Write-Protect Registers

39.3 Block Diagram

Figure 39-1. Pulse Width Modulation Controller Block Diagram



39.4 I/O Lines Description

Each channel outputs two complementary external I/O lines.

Table 39-1. I/O Line Description

Name	Description	Type
PWMHx	PWM Waveform Output High for channel x	Output
PWMLx	PWM Waveform Output Low for channel x	Output
PWMFIx	PWM Fault Input x	Input

39.5 Product Dependencies

39.5.1 I/O Lines

The pins used for interfacing the PWM are multiplexed with PIO lines. The programmer must first program the PIO controller to assign the desired PWM pins to their peripheral function. If I/O lines of the PWM are not used by the application, they can be used for other purposes by the PIO controller.

All of the PWM outputs may or may not be enabled. If an application requires only four channels, then only four PIO lines will be assigned to PWM outputs.

Table 39-2. I/O Lines

Instance	Signal	I/O Line	Peripheral
PWM	PWMFI0	PA5	B
PWM	PWMFI1	PA3	B
PWM	PWMFI2	PD6	B
PWM	PWMH0	PA8	B
PWM	PWMH0	PB12	B
PWM	PWMH0	PC3	B
PWM	PWMH0	PE15	A
PWM	PWMH1	PA19	B
PWM	PWMH1	PB13	B
PWM	PWMH1	PC5	B
PWM	PWMH1	PE16	A
PWM	PWMH2	PA13	B
PWM	PWMH2	PB14	B
PWM	PWMH2	PC7	B
PWM	PWMH3	PA9	B
PWM	PWMH3	PB15	B
PWM	PWMH3	PC9	B
PWM	PWMH3	PF3	A
PWM	PWMH4	PC20	B
PWM	PWMH4	PE20	A
PWM	PWMH5	PC19	B
PWM	PWMH5	PE22	A
PWM	PWMH6	PC18	B
PWM	PWMH6	PE24	A
PWM	PWMH7	PE26	A
PWM	PWML0	PA21	B
PWM	PWML0	PB16	B
PWM	PWML0	PC2	B

Table 39-2. I/O Lines

PWM	PWML0	PE18	A
PWM	PWML1	PA12	B
PWM	PWML1	PB17	B
PWM	PWML1	PC4	B
PWM	PWML2	PA20	B
PWM	PWML2	PB18	B
PWM	PWML2	PC6	B
PWM	PWML2	PE17	A
PWM	PWML3	PA0	B
PWM	PWML3	PB19	B
PWM	PWML3	PC8	B
PWM	PWML4	PC21	B
PWM	PWML4	PE19	A
PWM	PWML5	PC22	B
PWM	PWML5	PE21	A
PWM	PWML6	PC23	B
PWM	PWML6	PE23	A
PWM	PWML7	PC24	B
PWM	PWML7	PE25	A

39.5.2 Power Management

The PWM is not continuously clocked. The programmer must first enable the PWM clock in the Power Management Controller (PMC) before using the PWM. However, if the application does not require PWM operations, the PWM clock can be stopped when not needed and be restarted later. In this case, the PWM will resume its operations where it left off.

In the PWM description, Master Clock (MCK) is the clock of the peripheral bus to which the PWM is connected.

39.5.3 Interrupt Sources

The PWM interrupt line is connected on one of the internal sources of the Nested Vectored Interrupt Controller (NVIC). Using the PWM interrupt requires the NVIC to be programmed first.

Table 39-3. Peripheral IDs

Instance	ID
PWM	36

39.5.4 Fault Inputs

The PWM has the FAULT inputs connected to the different modules. Please refer to the implementation of these module within the product for detailed information about the fault generation procedure. The PWM receives faults from PIO inputs, PMC, ADC controller, and Timer/Counters

Table 39-4. Fault Inputs

Fault Inputs	External PWM Fault Input Number	Polarity Level[1]	Fault Input ID
PA5	PWMFI0	User Defined	0
PA3	PWMFI1	User Defined	1
PD6	PWMFI2	User Defined	2
MAIN OSC	–	1	3
ADC	–	1	4
Timer0	–	1	5

Note: 1. FPOL bit in PWMC_FMR.

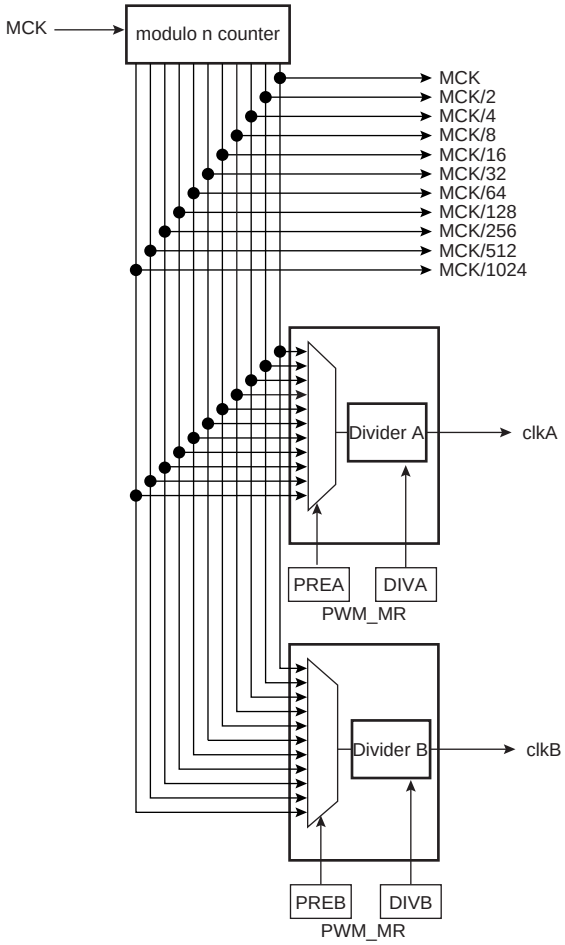
39.6 Functional Description

The PWM macrocell is primarily composed of a clock generator module and 8 channels.

- Clocked by the master clock (MCK), the clock generator module provides 13 clocks.
- Each channel can independently choose one of the clock generator outputs.
- Each channel generates an output waveform with attributes that can be defined independently for each channel through the user interface registers.

39.6.1 PWM Clock Generator

Figure 39-2. Functional View of the Clock Generator Block Diagram



The PWM master clock (MCK) is divided in the clock generator module to provide different clocks available for all channels. Each channel can independently select one of the divided clocks.

The clock generator is divided in three blocks:

- a modulo n counter which provides 11 clocks: F_{MCK}, $F_{MCK}/2$, $F_{MCK}/4$, $F_{MCK}/8$, $F_{MCK}/16$, $F_{MCK}/32$, $F_{MCK}/64$, $F_{MCK}/128$, $F_{MCK}/256$, $F_{MCK}/512$, $F_{MCK}/1024$
- two linear dividers (1, 1/2, 1/3, ... 1/255) that provide two separate clocks: clkA and clkB

Each linear divider can independently divide one of the clocks of the modulo n counter. The selection of the clock to be divided is made according to the PREA (PREB) field of the PWM Clock register (PWM_CLK). The resulting clock clkA (clkB) is the clock selected divided by DIVA (DIVB) field value.

After a reset of the PWM controller, DIVA (DIVB) and PREA (PREB) are set to 0. This implies that after reset clkA (clkB) are turned off.

At reset, all clocks provided by the modulo n counter are turned off except clock ”MCK”. This situation is also true when the PWM master clock is turned off through the Power Management Controller.

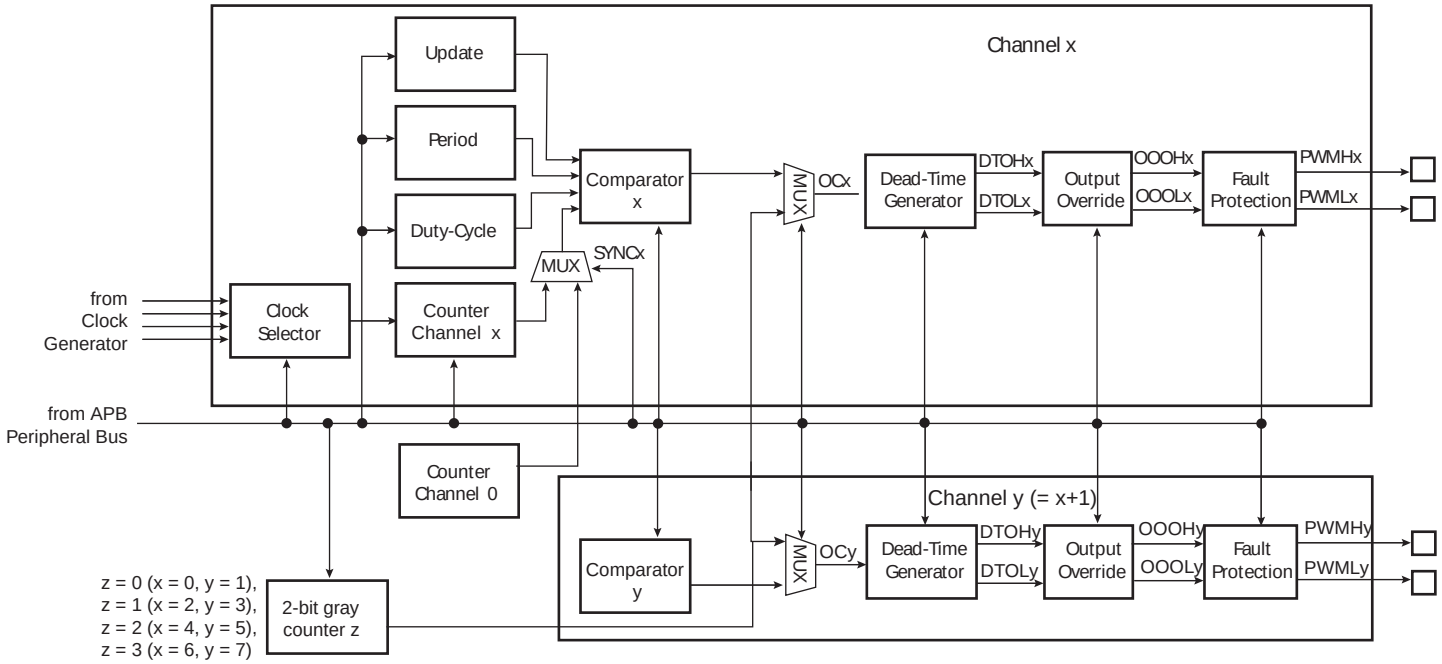
CAUTION:

- Before using the PWM macrocell, the programmer must first enable the PWM clock in the Power Management Controller (PMC).

39.6.2 PWM Channel

39.6.2.1 Block Diagram

Figure 39-3. Functional View of the Channel Block Diagram



Each of the 8 channels is composed of six blocks:

- A clock selector which selects one of the clocks provided by the clock generator (described in Section 39.6.1 on page 989).
- A counter clocked by the output of the clock selector. This counter is incremented or decremented according to the channel configuration and comparators matches. The size of the counter is 16 bits.
- A comparator used to compute the OCx output waveform according to the counter value and the configuration. The counter value can be the one of the channel counter or the one of the channel 0 counter according to SYNCx bit in the “PWM Sync Channels Mode Register” on page 1028 (PWM_SCM).
- A 2-bit configurable gray counter enables the stepper motor driver. One gray counter drives 2 channels.

- A dead-time generator providing two complementary outputs (DTOHx/DTOLx) which allows to drive external power control switches safely.
- An output override block that can force the two complementary outputs to a programmed value (OOOHx/OOOLx).
- An asynchronous fault protection mechanism that has the highest priority to override the two complementary outputs in case of fault detection (PWMHx/PWMLx).

39.6.2.2 Comparator

The comparator continuously compares its counter value with the channel period defined by CPRD in the "PWM Channel Period Register" on page 1062 (PWM_CPRDx) and the duty-cycle defined by CDTY in the "PWM Channel Duty Cycle Register" on page 1060 (PWM_CDTYx) to generate an output signal OCx accordingly.

The different properties of the waveform of the output OCx are:

- the ***clock selection***. The channel counter is clocked by one of the clocks provided by the clock generator described in the previous section. This channel parameter is defined in the CPRE field of the "PWM Channel Mode Register" on page 1058 (PWM_CMRx). This field is reset at 0.
- the ***waveform period***. This channel parameter is defined in the CPRD field of the PWM_CPRDx register.
  If the waveform is left aligned, then the output waveform period depends on the counter source clock and can be calculated:
  By using the PWM master clock (MCK) divided by an X given prescaler value (with X being 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, or 1024), the resulting period formula will be:

  $$\frac{(X \times CPRD)}{MCK}$$

  By using the PWM master clock (MCK) divided by one of both DIVA or DIVB divider, the formula becomes, respectively:

  $$\frac{(CRPD \times DIVA)}{MCK} \text{ or } \frac{(CRPD \times DIVB)}{MCK}$$

  If the waveform is center aligned then the output waveform period depends on the counter source clock and can be calculated:
  By using the PWM master clock (MCK) divided by an X given prescaler value
  (with X being 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, or 1024). The resulting period formula will be:

  $$\frac{(2 \times X \times CPRD)}{MCK}$$

  By using the PWM master clock (MCK) divided by one of both DIVA or DIVB divider, the formula becomes, respectively:

  $$\frac{(2 \times CPRD \times DIVA)}{MCK} \text{ or } \frac{(2 \times CPRD \times DIVB)}{MCK}$$

- the ***waveform duty-cycle***. This channel parameter is defined in the CDTY field of the PWM_CDTYx register.

If the waveform is left aligned then:

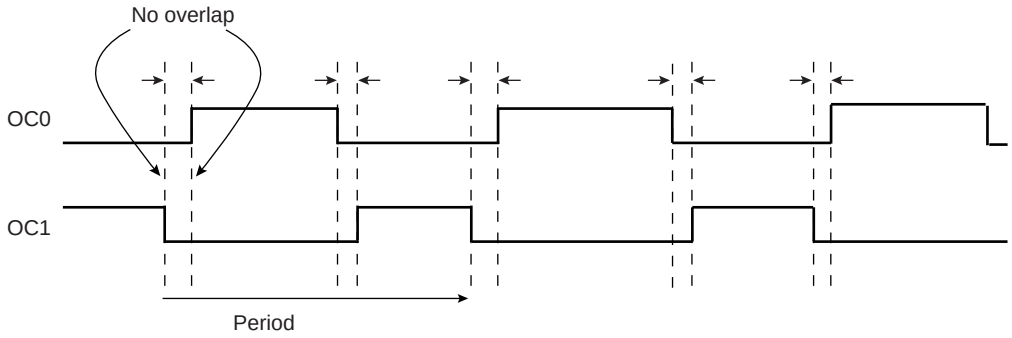
$$\text{duty cycle} = \frac{(period - 1 / \text{fchannel\_x\_clock} \times CDTY)}{period}$$

If the waveform is center aligned, then:

$$\text{duty cycle} = \frac{((period / 2) - 1 / \text{fchannel\_x\_clock} \times CDTY))}{(period / 2)}$$

- the ***waveform polarity.*** At the beginning of the period, the signal can be at high or low level. This property is defined in the CPOL field of the PWM_CMRx register. By default the signal starts by a low level.
- the ***waveform alignment***. The output waveform can be left or center aligned. Center aligned waveforms can be used to generate non overlapped waveforms. This property is defined in the CALG field of the PWM_CMRx register. The default mode is left aligned.

Figure 39-4. Non Overlapped Center Aligned Waveforms



Note: 1. See Figure 39-5 on page 994 for a detailed description of center aligned waveforms.

When center aligned, the channel counter increases up to CPRD and decreases down to 0. This ends the period.

When left aligned, the channel counter increases up to CPRD and is reset. This ends the period.

Thus, for the same CPRD value, the period for a center aligned channel is twice the period for a left aligned channel.

Waveforms are fixed at 0 when:

- CDTY = CPRD and CPOL = 0
- CDTY = 0 and CPOL = 1

Waveforms are fixed at 1 (once the channel is enabled) when:

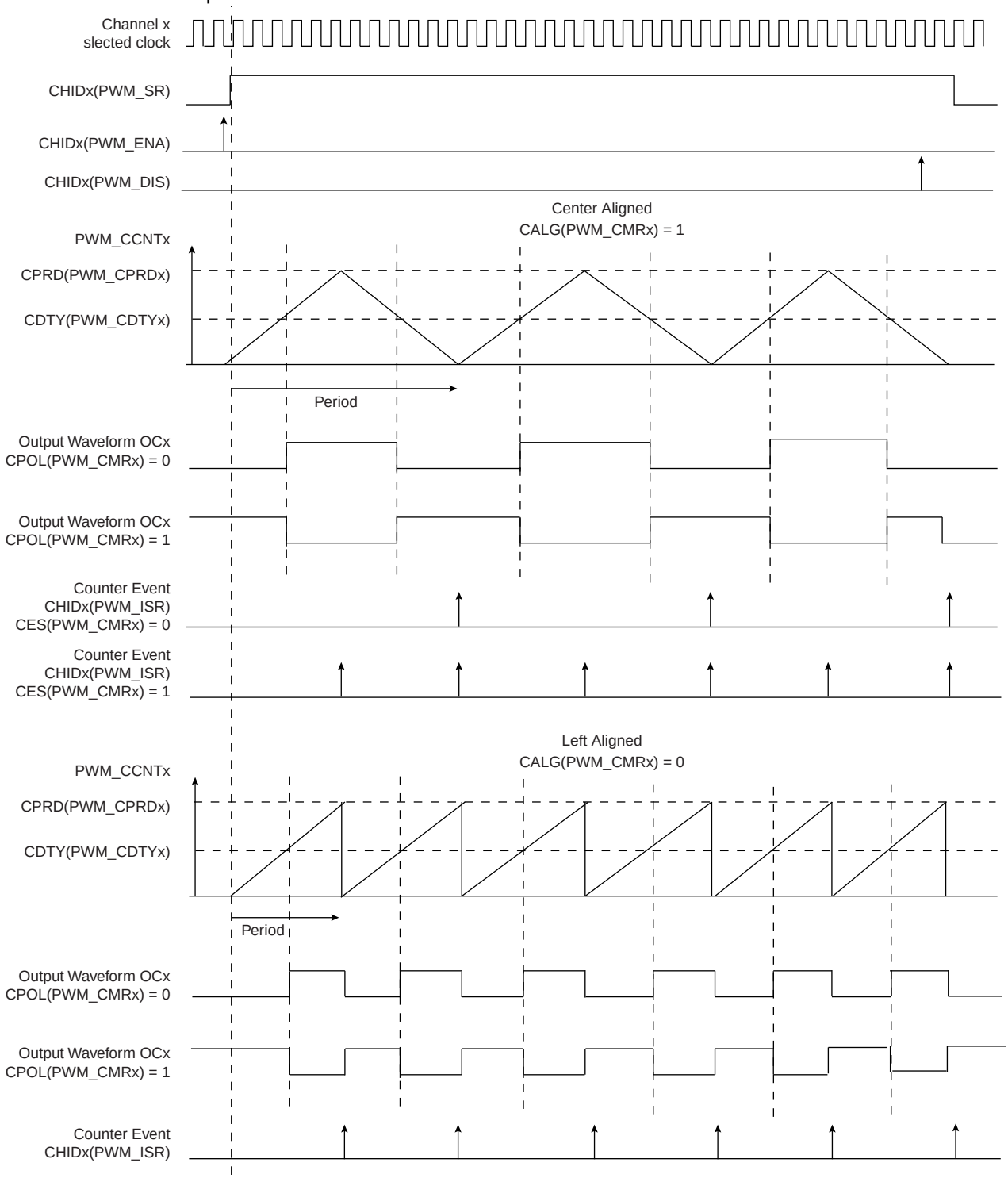
- CDTY = 0 and CPOL = 0
- CDTY = CPRD and CPOL = 1

The waveform polarity must be set before enabling the channel. This immediately affects the channel output level. Changes on channel polarity are not taken into account while the channel is enabled.

Besides generating output signals OCx, the comparator generates interrupts in function of the counter value. When the output waveform is left aligned, the interrupt occurs at the end of the counter period. When the output waveform is center aligned, the bit CES of the PWM_CMRx register defines when the channel counter interrupt occurs. If CES is set to 0, the interrupt occurs at the end of the counter period. If CES is set to 1, the interrupt occurs at the end of the counter period and at half of the counter period.

Figure 39-5 “Waveform Properties” illustrates the counter interrupts in function of the configuration.

Figure 39-5. Waveform Properties



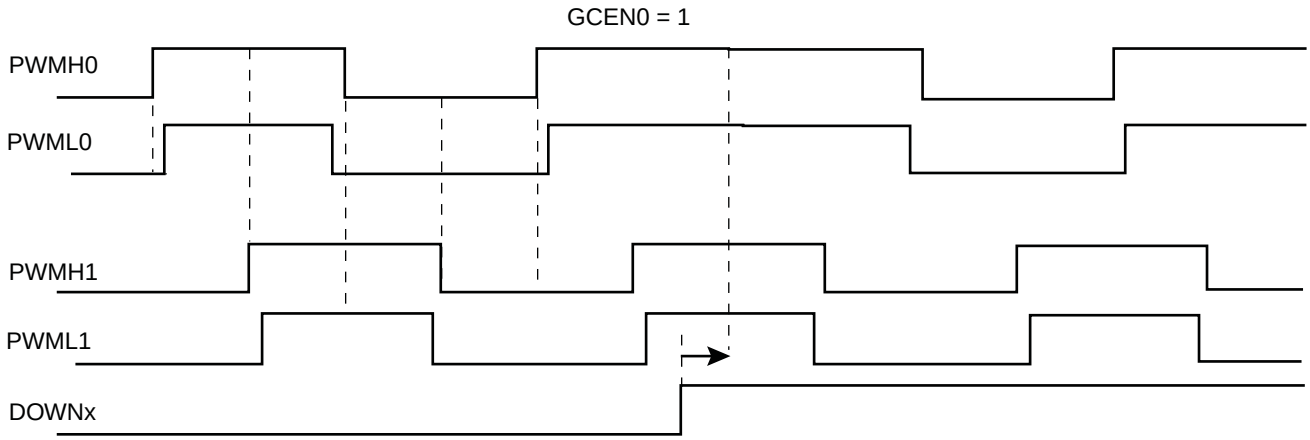
39.6.2.3 2-bit Gray Up/Down Counter for Stepper Motor

It is possible to configure a couple of channels to provide a 2-bit gray count waveform on 2 outputs. Dead-Time Generator and other downstream logic can be configured on these channels.

Up or down count mode can be configured on-the-fly by means of PWM_SMMR configuration registers.

When GCEN0 is set to 1, channels 0 and 1 outputs are driven with gray counter.

Figure 39-6. 2-bit Gray Up/Down Counter



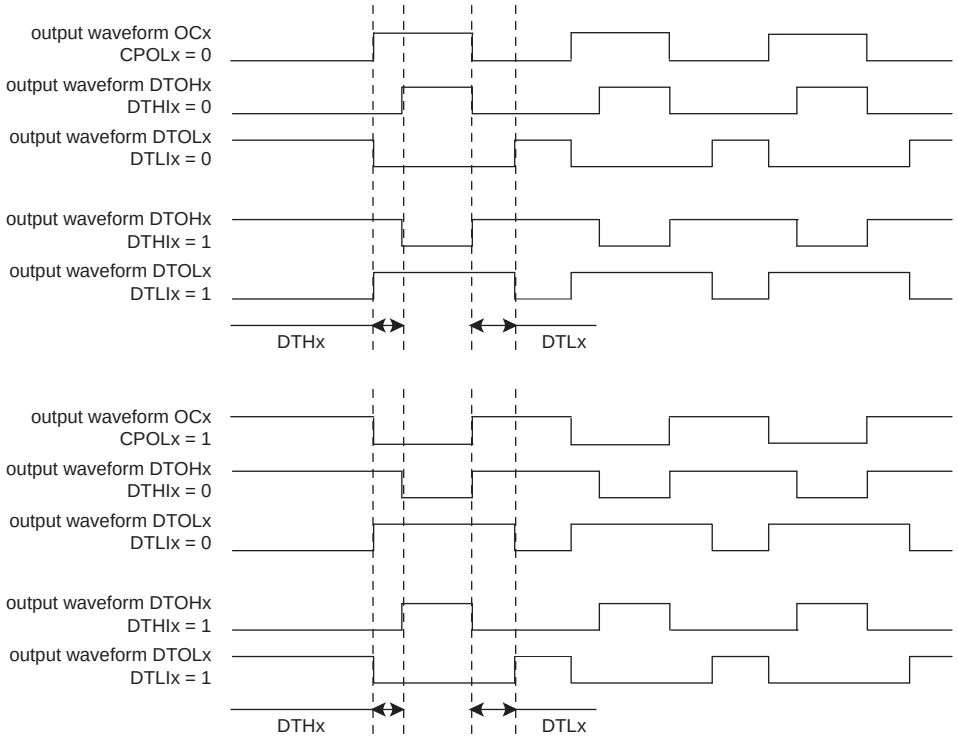
39.6.2.4 Dead-Time Generator

The dead-time generator uses the comparator output OCx to provide the two complementary outputs DTOHx and DTOLx, which allows the PWM macrocell to drive external power control switches safely. When the dead-time generator is enabled by setting the bit DTE to 1 or 0 in the "PWM Channel Mode Register" (PWM_CMRx), dead-times (also called dead-bands or non-overlapping times) are inserted between the edges of the two complementary outputs DTOHx and DTOLx. Note that enabling or disabling the dead-time generator is allowed only if the channel is disabled.

The dead-time is adjustable by the "PWM Channel Dead Time Register" (PWM_DTx). Both outputs of the dead-time generator can be adjusted separately by DTH and DTL. The dead-time values can be updated synchronously to the PWM period by using the "PWM Channel Dead Time Update Register" (PWM_DTUPDx).

The dead-time is based on a specific counter which uses the same selected clock that feeds the channel counter of the comparator. Depending on the edge and the configuration of the dead-time, DTOHx and DTOLx are delayed until the counter has reached the value defined by DTH or DTL. An inverted configuration bit (DTHI and DTLI bit in the PWM_CMRx register) is provided for each output to invert the dead-time outputs. The following figure shows the waveform of the dead-time generator.

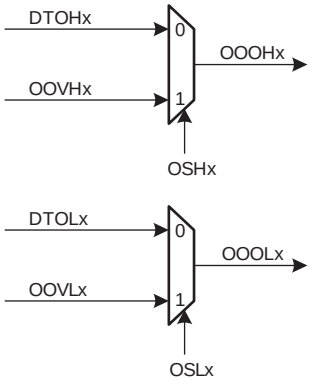
Figure 39-7. Complementary Output Waveforms



39.6.2.5 *Output Override*

The two complementary outputs DTOHx and DTOLx of the dead-time generator can be forced to a value defined by the software.

Figure 39-8. Override Output Selection



The fields OSHx and OSLx in the "PWM Output Selection Register" (PWM_OS) allow the outputs of the dead-time generator DTOHx and DTOLx to be overridden by the value defined in the fields OOVHx and OOVLx in the"PWM Output Override Value Register" (PWM_OOV).

The set registers "PWM Output Selection Set Register" and "PWM Output Selection Set Update Register" (PWM_OSS and PWM_OSSUPD) enable the override of the outputs of a channel regardless of other channels. In the same way, the clear registers "PWM Output Selection Clear Register" and "PWM Output Selection Clear Update Register" (PWM_OSC and PWM_OSCUPD) disable the override of the outputs of a channel regardless of other channels.

By using buffer registers PWM_OSSUPD and PWM_OSCUPD, the output selection of PWM outputs is done synchronously to the channel counter, at the beginning of the next PWM period.

By using registers PWM_OSS and PWM_OSC, the output selection of PWM outputs is done asynchronously to the channel counter, as soon as the register is written.

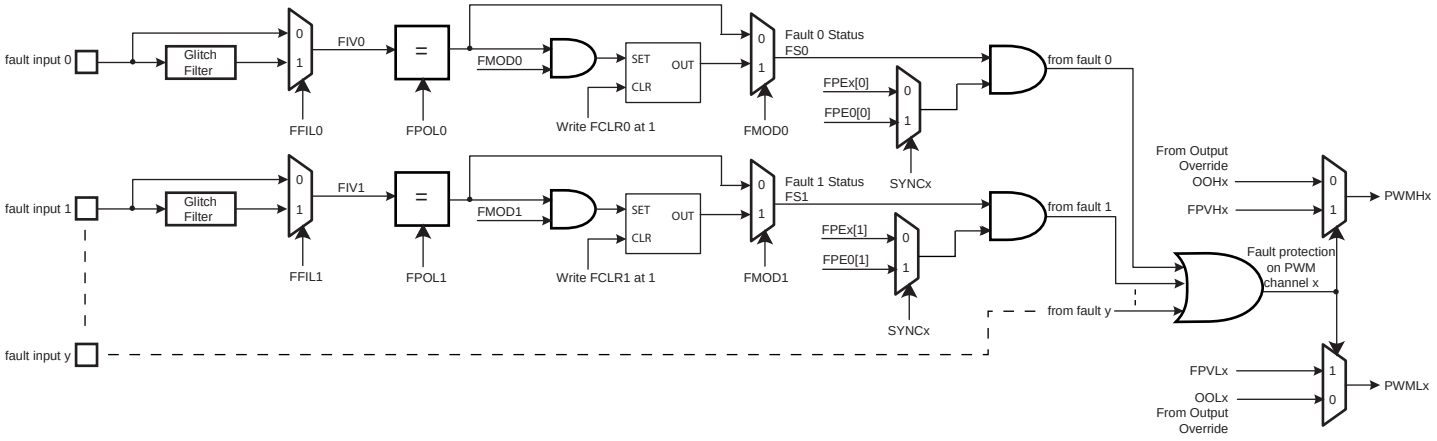
The value of the current output selection can be read in PWM_OS.

While overriding PWM outputs, the channel counters continue to run, only the PWM outputs are forced to user defined values.

39.6.2.6 Fault Protection

6 inputs provide fault protection which can force any of the PWM output pair to a programmable value. This mechanism has priority over output overriding.

Figure 39-9. Fault Protection



The polarity level of the fault inputs is configured by the FPOL field in the "PWM Fault Mode Register" (PWM_FMR). For fault inputs coming from internal peripherals such as ADC, Timer Counter, to name but a few, the polarity level must be FPOL = 1. For fault inputs coming from external GPIO pins the polarity level depends on the user's implementation.

The configuration of the Fault Activation Mode (FMOD bit in PWMC_FMR) depends on the peripheral generating the fault. If the corresponding peripheral does not have "Fault Clear" management, then the FMOD configuration to use must be FMOD = 1, to avoid spurious fault detection. Check the corresponding peripheral documentation for details on handling fault generation.

The fault inputs can be glitch filtered or not in function of the FFIL field in the PWM_FMR register. When the filter is activated, glitches on fault inputs with a width inferior to the PWM master clock (MCK) period are rejected.

A fault becomes active as soon as its corresponding fault input has a transition to the programmed polarity level. If the corresponding bit FMOD is set to 0 in the PWM_FMR register, the fault remains active as long as the fault input is at this polarity level. If the corresponding FMOD bit is set to 1, the fault remains active until the fault input is not at this polarity level anymore and until it is cleared by writing the corresponding bit FCLR in the "PWM Fault Clear Register" (PWM_FSCR). By reading the "PWM Fault Status Register" (PWM_FSR), the user can read the

current level of the fault inputs by means of the field FIV, and can know which fault is currently active thanks to the FS field.

Each fault can be taken into account or not by the fault protection mechanism in each channel. To be taken into account in the channel x, the fault y must be enabled by the bit FPEx[y] in the “PWM Fault Protection Enable Registers” (PWM_FPE1/2). However the synchronous channels (see Section 39.6.2.7 “Synchronous Channels”) do not use their own fault enable bits, but those of the channel 0 (bits FPE0[y]).

The fault protection on a channel is triggered when this channel is enabled and when any one of the faults that are enabled for this channel is active. It can be triggered even if the PWM master clock (MCK) is not running but only by a fault input that is not glitch filtered.

When the fault protection is triggered on a channel, the fault protection mechanism forces the channel outputs to the values defined by the fields FPVHx and FPVLx in the “PWM Fault Protection Value Register” (PWM_FPV) and leads to a reset of the counter of this channel. The output forcing is made asynchronously to the channel counter.

CAUTION:

- To prevent an unexpected activation of the status flag FSy in the PWM_FSR register, the FMODy bit can be set to “1” only if the FPOLy bit has been previously configured to its final value.
- To prevent an unexpected activation of the Fault Protection on the channel x, the bit FPEx[y] can be set to “1” only if the FPOLy bit has been previously configured to its final value.

If a comparison unit is enabled (see Section 39.6.3 “PWM Comparison Units”) and if a fault is triggered in the channel 0, in this case the comparison cannot match.

As soon as the fault protection is triggered on a channel, an interrupt (different from the interrupt generated at the end of the PWM period) can be generated but only if it is enabled and not masked. The interrupt is reset by reading the interrupt status register, even if the fault which has caused the trigger of the fault protection is kept active.

39.6.2.7 Synchronous Channels

Some channels can be linked together as synchronous channels. They have the same source clock, the same period, the same alignment and are started together. In this way, their counters are synchronized together.

The synchronous channels are defined by the SYNCx bits in the "PWM Sync Channels Mode Register" (PWM_SCM). Only one group of synchronous channels is allowed.

When a channel is defined as a synchronous channel, the channel 0 is automatically defined as a synchronous channel too, because the channel 0 counter configuration is used by all the synchronous channels.

If a channel x is defined as a synchronous channel, it uses the following configuration fields of the channel 0 instead of its own:

- CPRE0 field in PWM_CMR0 register instead of CPREx field in PWM_CMRx register (same source clock)
- CPRD0 field in PWM_CMR0 register instead of CPRDx field in PWM_CMRx register (same period)
- CALG0 field in PWM_CMR0 register instead of CALGx field in PWM_CMRx register (same alignment)

Thus writing these fields of a synchronous channel has no effect on the output waveform of this channel (except channel 0 of course).

Because counters of synchronous channels must start at the same time, they are all enabled together by enabling the channel 0 (by the CHID0 bit in PWM_ENA register). In the same way, they are all disabled together by disabling channel 0 (by the CHID0 bit in PWM_DIS register). However, a synchronous channel x different from channel 0 can be enabled or disabled independently from others (by the CHIDx bit in PWM_ENA and PWM_DIS registers).

Defining a channel as a synchronous channel while it is an asynchronous channel (by writing the bit SYNCx to 1 while it was at 0) is allowed only if the channel is disabled at this time (CHIDx = 0 in PWM_SR register). In the same way, defining a channel as an asynchronous channel while it is a synchronous channel (by writing the SYNCx bit to 0 while it was 1) is allowed only if the channel is disabled at this time.

The field UPDM (Update Mode) in the PWM_SCM register allow to select one of the three methods to update the registers of the synchronous channels:

- Method 1 (UPDM = 0): the period value, the duty-cycle values and the dead-time values must be written by the CPU in their respective update registers (respectively PWM_CPRDUPDx, PWM_CDTYUPDx and PWM_DTUPDx).The update is triggered at the next PWM period as soon as the bit UPDULOCK in the "PWM Sync Channels Update Control Register" (PWM_SCUC) is set to 1 (see "Method 1: Manual write of duty-cycle values and manual trigger of the update" on page 1001).
- Method 2 (UPDM = 1): the period value, the duty-cycle values, the dead-time values and the update period value must be written by the CPU in their respective update registers (respectively PWM_CPRDUPDx, PWM_CDTYUPDx and PWM_DTUPD). The update of the period value and of the dead-time values is triggered at the next PWM period as soon as the bit UPDULOCK in the "PWM Sync Channels Update Control Register" (PWM_SCUC) is set to 1. The update of the duty-cycle values and the update period value is triggered automatically after an update period defined by the field UPR in the "PWM Sync Channels

Update Period Register" (PWM_SCUP) (see "Method 2: Manual write of duty-cycle values and automatic trigger of the update" on page 1002).

- Method 3 (UPDM = 2): same as Method 2 apart from the fact that the duty-cycle values of ALL synchronous channels are written by the Peripheral DMA Controller (PDC) (see "Method 3: Automatic write of duty-cycle values and automatic trigger of the update" on page 1004). The user can choose to synchronize the PDC transfer request with a comparison match (see Section 39.6.3 "PWM Comparison Units"), by the fields PTRM and PTRCS in the PWM_SCM register.

Table 39-5. Summary of the Update of Registers of Synchronous Channels

	UPDM=0	UPDM=1	UPDM=2
Period Value (PWM_CPRDUPDx)	Write by the CPU		
	Update is triggered at the next PWM period as soon as the bit UPDULOCK is set to 1		
Dead-Time Values (PWM_DTUPDx)	Write by the CPU		
	Update is triggered at the next PWM period as soon as the bit UPDULOCK is set to 1		
Duty-Cycle Values (PWM_CDTYUPDx)	Write by the CPU	Write by the CPU	Write by the PDC
	Update is triggered at the next PWM period as soon as the bit UPDULOCK is set to 1	Update is triggered at the next PWM period as soon as the update period counter has reached the value UPR	
Update Period Value (PWM_SCUPUPD)	Not applicable	Write by the CPU	
	Not applicable	Update is triggered at the next PWM period as soon as the update period counter has reached the value UPR	

Method 1: Manual write of duty-cycle values and manual trigger of the update

In this mode, the update of the period value, the duty-cycle values and the dead-time values must be done by writing in their respective update registers with the CPU (respectively PWM_CPRDUPDx, PWM_CDTYUPDx and PWM_DTUPDx).

To trigger the update, the user must use the bit UPDULOCK of the "PWM Sync Channels Update Control Register" (PWM_SCUC) which allows to update synchronously (at the same PWM period) the synchronous channels:

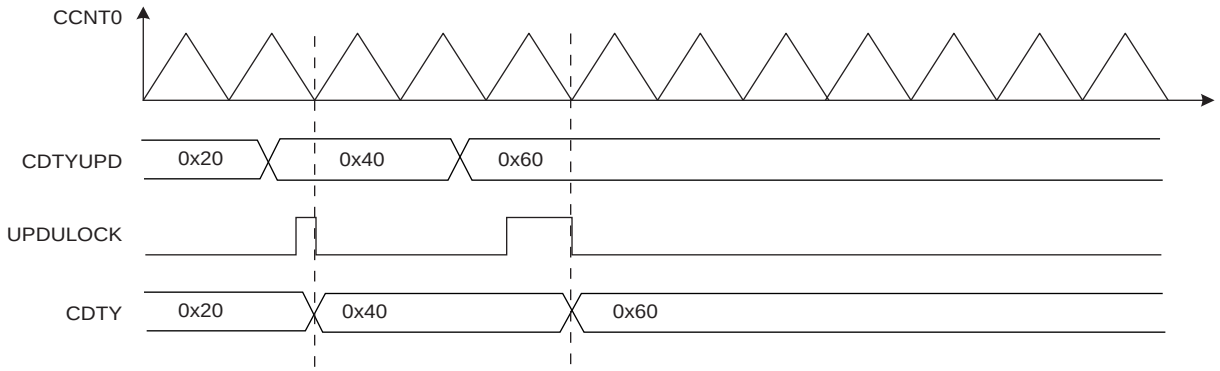
- If the bit UPDULOCK is set to 1, the update is done at the next PWM period of the synchronous channels.
- If the UPDULOCK bit is not set to 1, the update is locked and cannot be performed.

After writing the UPDULOCK bit to 1, it is held at this value until the update occurs, then it is read 0.

Sequence for Method 1:

1. Select the manual write of duty-cycle values and the manual update by setting the UPDM field to 0 in the PWM_SCM register
2. Define the synchronous channels by the SYNCx bits in the PWM_SCM register.
3. Enable the synchronous channels by writing CHID0 in the PWM_ENA register.
4. If an update of the period value and/or the duty-cycle values and/or the dead-time values is required, write registers that need to be updated (PWM_CPRDUPDx, PWM_CDTYUPDx and PWM_DTUPDx).
5. Set UPDULOCK to 1 in PWM_SCUC.
6. The update of the registers will occur at the beginning of the next PWM period. At this moment the UPDULOCK bit is reset, go to Step 4.) for new values.

Figure 39-10. Method 1 (UPDM = 0)



Method 2: Manual write of duty-cycle values and automatic trigger of the update

In this mode, the update of the period value, the duty-cycle values, the dead-time values and the update period value must be done by writing in their respective update registers with the CPU (respectively PWM_CPRDUPDx, PWM_CDTYUPDx, PWM_DTUPDx and PWM_SCUPUPD).

To trigger the update of the period value and the dead-time values, the user must use the bit UPDULOCK of the "PWM Sync Channels Update Control Register" (PWM_SCUC) which allows to update synchronously (at the same PWM period) the synchronous channels:

- If the bit UPDULOCK is set to 1, the update is done at the next PWM period of the synchronous channels.
- If the UPDULOCK bit is not set to 1, the update is locked and cannot be performed.

After writing the UPDULOCK bit to 1, it is held at this value until the update occurs, then it is read 0.

The update of the duty-cycle values and the update period is triggered automatically after an update period.

To configure the automatic update, the user must define a value for the Update Period by the UPR field in the "PWM Sync Channels Update Period Register" (PWM_SCUP). The PWM controller waits UPR+1 period of synchronous channels before updating automatically the duty values and the update period value.

The status of the duty-cycle value write is reported in the "PWM Interrupt Status Register 2" (PWM_ISR2) by the following flags:

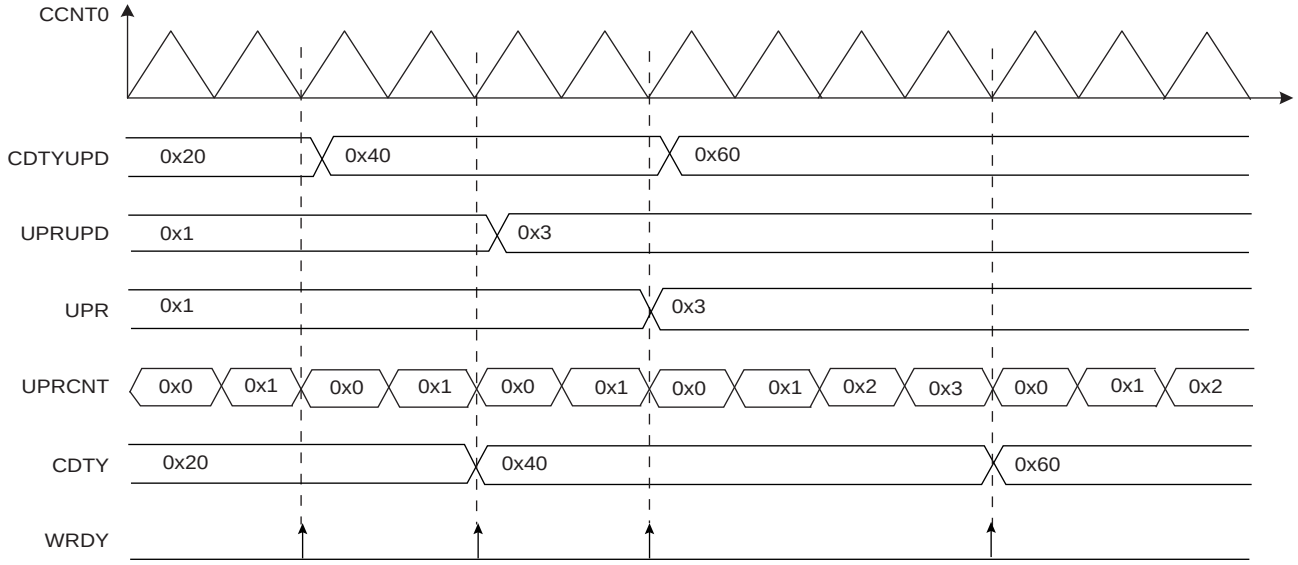
- WRDY: this flag is set to 1 when the PWM Controller is ready to receive new duty-cycle values and a new update period value. It is reset to 0 when the PWM_ISR2 register is read.

Depending on the interrupt mask in the PWM_IMR2 register, an interrupt can be generated by these flags.

Sequence for Method 2:

1. Select the manual write of duty-cycle values and the automatic update by setting the field UPDM to 1 in the PWM_SCM register
2. Define the synchronous channels by the bits SYNCx in the PWM_SCM register.
3. Define the update period by the field UPR in the PWM_SCUP register.
4. Enable the synchronous channels by writing CHID0 in the PWM_ENA register.
5. If an update of the period value and/or of the dead-time values is required, write registers that need to be updated (PWM_CPRDUPDx, PWM_DTUPDx), else go to Step 8.
6. Set UPDULOCK to 1 in PWM_SCUC.
7. The update of these registers will occur at the beginning of the next PWM period. At this moment the bit UPDULOCK is reset, go to Step 5. for new values.
8. If an update of the duty-cycle values and/or the update period is required, check first that write of new update values is possible by polling the flag WRDY (or by waiting for the corresponding interrupt) in the PWM_ISR2 register.
9. Write registers that need to be updated (PWM_CDTYUPDx, PWM_SCUPUPD).
10. The update of these registers will occur at the next PWM period of the synchronous channels when the Update Period is elapsed. Go to Step 8. for new values.

Figure 39-11. Method 2 (UPDM=1)



Method 3: Automatic write of duty-cycle values and automatic trigger of the update

In this mode, the update of the duty cycle values is made automatically by the Peripheral DMA Controller (PDC). The update of the period value, the dead-time values and the update period value must be done by writing in their respective update registers with the CPU (respectively PWM_CPRDUPDx, PWM_DTUPDx and PWM_SCUPUPD).

To trigger the update of the period value and the dead-time values, the user must use the bit UPDULOCK which allows to update synchronously (at the same PWM period) the synchronous channels:

- If the bit UPDULOCK is set to 1, the update is done at the next PWM period of the synchronous channels.
- If the UPDULOCK bit is not set to 1, the update is locked and cannot be performed.

After writing the UPDULOCK bit to 1, it is held at this value until the update occurs, then it is read 0.

The update of the duty-cycle values and the update period value is triggered automatically after an update period.

To configure the automatic update, the user must define a value for the Update Period by the field UPR in the "PWM Sync Channels Update Period Register" (PWM_SCUP). The PWM controller waits UPR+1 periods of synchronous channels before updating automatically the duty values and the update period value.

Using the PDC removes processor overhead by reducing its intervention during the transfer. This significantly reduces the number of clock cycles required for a data transfer, which improves microcontroller performance.

The PDC must write the duty-cycle values in the synchronous channels index order. For example if the channels 0, 1 and 3 are synchronous channels, the PDC must write the duty-cycle of the channel 0 first, then the duty-cycle of the channel 1, and finally the duty-cycle of the channel 3.

The status of the PDC transfer is reported in the "PWM Interrupt Status Register 2" (PWM_ISR2) by the following flags:

- WRDY: this flag is set to 1 when the PWM Controller is ready to receive new duty-cycle values and a new update period value. It is reset to 0 when the PWM_ISR2 register is read. The user can choose to synchronize the WRDY flag and the PDC transfer request with a comparison match (see Section 39.6.3 "PWM Comparison Units"), by the fields PTRM and PTRCS in the PWM_SCM register.
- ENDTX: this flag is set to 1 when a PDC transfer is completed
- TXBUFE: this flag is set to 1 when the PDC buffer is empty (no pending PDC transfers)
- UNRE: this flag is set to 1 when the update period defined by the UPR field has elapsed while the whole data has not been written by the PDC. It is reset to 0 when the PWM_ISR2 register is read.

Depending on the interrupt mask in the PWM_IMR2 register, an interrupt can be generated by these flags.

Sequence for Method 3:

1. Select the automatic write of duty-cycle values and automatic update by setting the field UPDM to 2 in the PWM_SCM register.
2. Define the synchronous channels by the bits SYNCx in the PWM_SCM register.
3. Define the update period by the field UPR in the PWM_SCUP register.
4. Define when the WRDY flag and the corresponding PDC transfer request must be set in the update period by the PTRM bit and the PTRCS field in the PWM_SCM register (at the end of the update period or when a comparison matches).
5. Define the PDC transfer settings for the duty-cycle values and enable it in the PDC registers
6. Enable the synchronous channels by writing CHID0 in the PWM_ENA register.
7. If an update of the period value and/or of the dead-time values is required, write registers that need to be updated (PWM_CPRDUPDx, PWM_DTUPDx), else go to Step 10.
8. Set UPDULOCK to 1 in PWM_SCUC.
9. The update of these registers will occur at the beginning of the next PWM period. At this moment the bit UPDULOCK is reset, go to Step 7. for new values.
10. If an update of the update period value is required, check first that write of a new update value is possible by polling the flag WRDY (or by waiting for the corresponding interrupt) in the PWM_ISR2 register, else go to Step 13.
11. Write the register that needs to be updated (PWM_SCUPUPD).
12. The update of this register will occur at the next PWM period of the synchronous channels when the Update Period is elapsed. Go to Step 10. for new values.
13. Check the end of the PDC transfer by the flag ENDTX. If the transfer has ended, define a new PDC transfer in the PDC registers for new duty-cycle values. Go to Step 5.

Figure 39-12. Method 3 (UPDM=2 and PTRM=0)

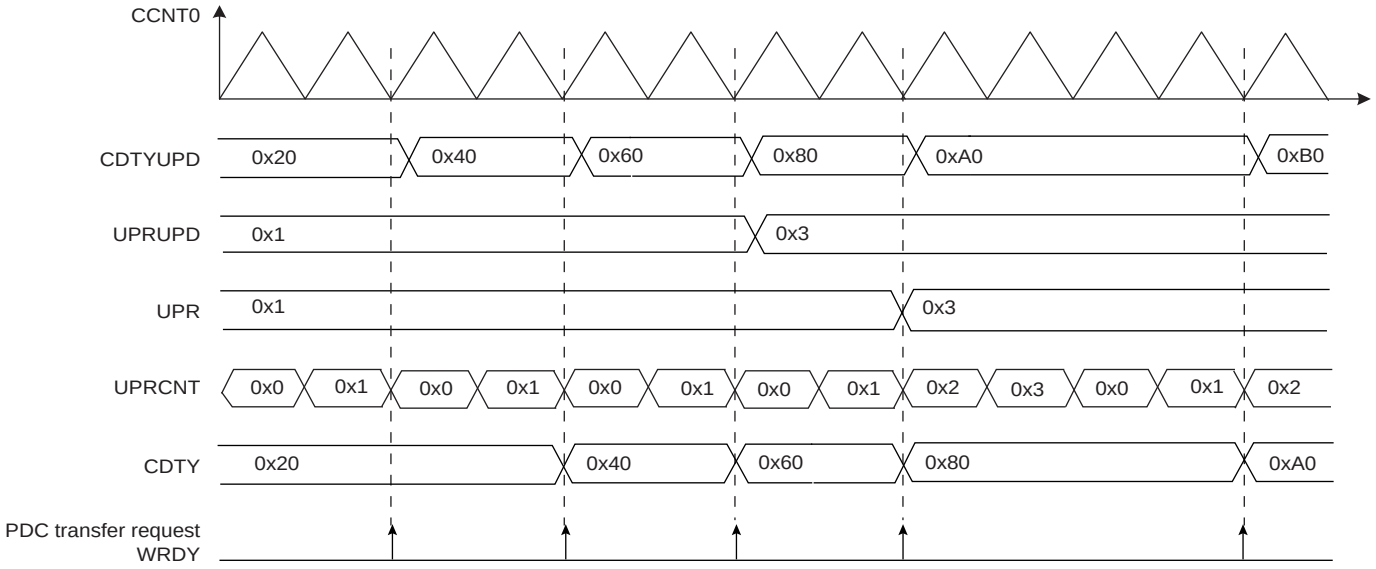
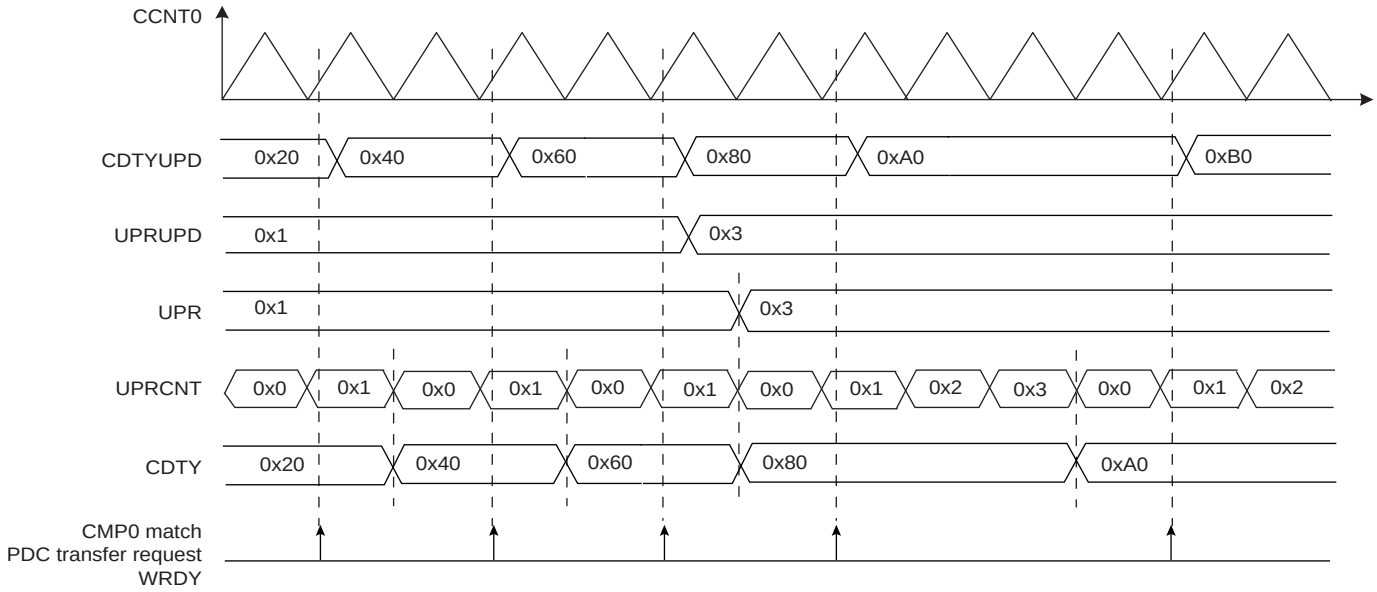


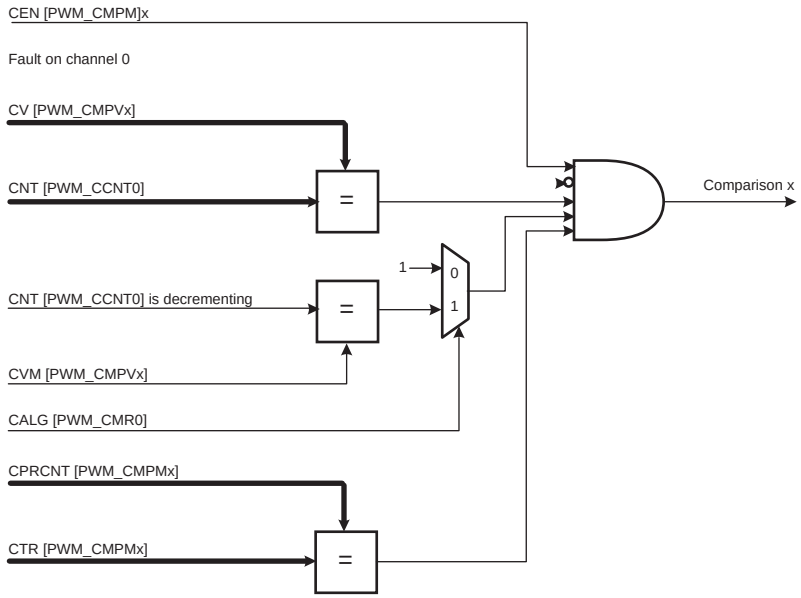
Figure 39-13. Method 3 (UPDM=2 and PTRM=1 and PTRCS=0)



39.6.3 PWM Comparison Units

The PWM provides 8 independent comparison units able to compare a programmed value with the current value of the channel 0 counter (which is the channel counter of all synchronous channels, Section 39.6.2.7 “Synchronous Channels”). These comparisons are intended to generate pulses on the event lines (used to synchronize ADC, see Section 39.6.4 “PWM Event Lines”), to generate software interrupts and to trigger PDC transfer requests for the synchronous channels (see “Method 3: Automatic write of duty-cycle values and automatic trigger of the update” on page 1004).

Figure 39-14. Comparison Unit Block Diagram



The comparison x matches when it is enabled by the bit CEN in the “PWM Comparison x Mode Register” (PWM_CMPMx for the comparison x) and when the counter of the channel 0 reaches the comparison value defined by the field CV in “PWM Comparison x Value Register” (PWM_CMPVx for the comparison x). If the counter of the channel 0 is center aligned (CALG = 1 in “PWM Channel Mode Register”), the bit CVM (in PWM_CMPVx) defines if the comparison is made when the counter is counting up or counting down (in left alignment mode CALG=0, this bit is useless).

If a fault is active on the channel 0, the comparison is disabled and cannot match (see Section 39.6.2.6 “Fault Protection”).

The user can define the periodicity of the comparison x by the fields CTR and CPR (in PWM_CMPVx). The comparison is performed periodically once every CPR+1 periods of the counter of the channel 0, when the value of the comparison period counter CPRCNT (in PWM_CMPMx) reaches the value defined by CTR. CPR is the maximum value of the comparison period counter CPRCNT. If CPR=CTR=0, the comparison is performed at each period of the counter of the channel 0.

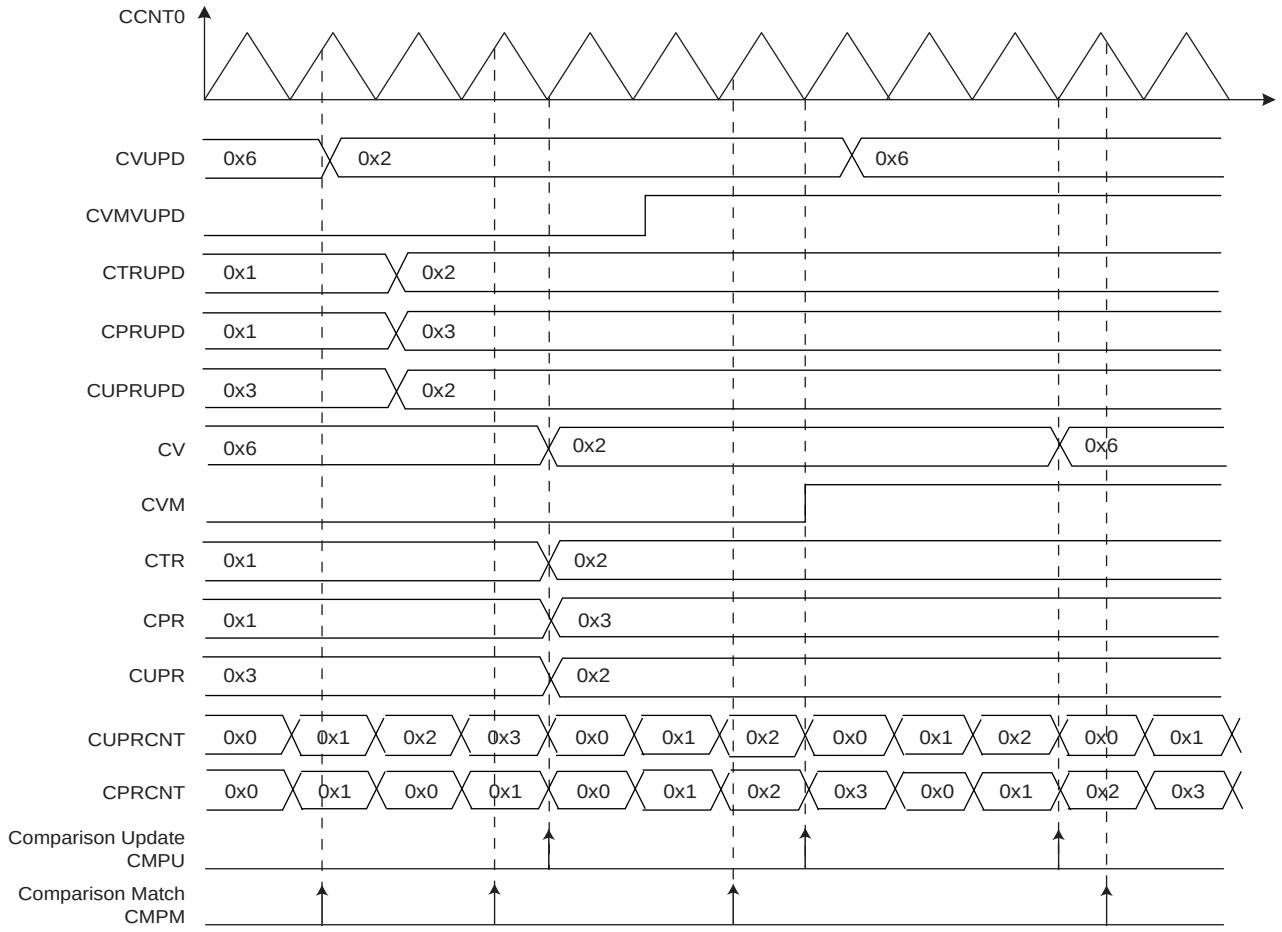
The comparison x configuration can be modified while the channel 0 is enabled by using the “PWM Comparison x Mode Update Register” (PWM_CMPMUPDx registers for the comparison x). In the same way, the comparison x value can be modified while the channel 0 is enabled by using the “PWM Comparison x Value Update Register” (PWM_CMPVUPDx registers for the comparison x).

The update of the comparison x configuration and the comparison x value is triggered periodically after the comparison x update period. It is defined by the field CUPR in the PWM_CMPMx. The comparison unit has an update period counter independent from the period counter to trigger this update. When the value of the comparison update period counter CUPRCNT (in PWM_CMPMx) reaches the value defined by CUPR, the update is triggered. The comparison x update period CUPR itself can be updated while the channel 0 is enabled by using the PWM_CMPMUPDx register.

CAUTION: to be taken into account, the write of the register PWM_CMPVUPDx must be followed by a write of the register PWM_CMPMUPDx.

The comparison match and the comparison update can be source of an interrupt, but only if it is enabled and not masked. These interrupts can be enabled by the "PWM Interrupt Enable Register 2" and disabled by the "PWM Interrupt Disable Register 2" . The comparison match interrupt and the comparison update interrupt are reset by reading the "PWM Interrupt Status Register 2"

Figure 39-15. Comparison Waveform

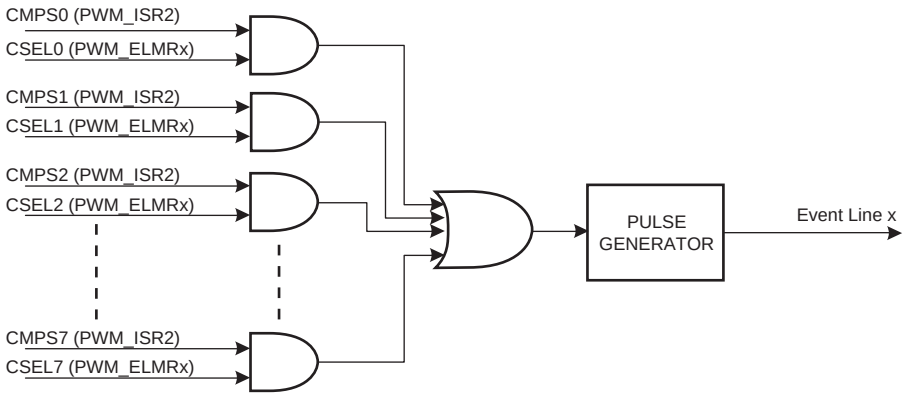


39.6.4 PWM Event Lines

The PWM provides 2 independent event lines intended to trigger actions in other peripherals (in particular for ADC (Analog-to-Digital Converter)).

A pulse (one cycle of the master clock (MCK)) is generated on an event line, when at least one of the selected comparisons is matching. The comparisons can be selected or unselected independently by the CSEL bits in the "PWM Event Line x Register" (PWM_ELMRx for the Event Line x).

Figure 39-16. Event Line Block Diagram



39.6.5 PWM Controller Operations

39.6.5.1 Initialization

Before enabling the channels, they must have been configured by the software application:

- Unlock User Interface by writing the WPCMD field in the PWM_WPCR Register.
- Configuration of the clock generator (DIVA, PREA, DIVB, PREB in the PWM_CLK register if required).
- Selection of the clock for each channel (CPRE field in the PWM_CMRx register)
- Configuration of the waveform alignment for each channel (CALG field in the PWM_CMRx register)
- Selection of the counter event selection (if CALG = 1) for each channel (CES field in the PWM_CMRx register)
- Configuration of the output waveform polarity for each channel (CPOL in the PWM_CMRx register)
- Configuration of the period for each channel (CPRD in the PWM_CPRDx register). Writing in PWM_CPRDx register is possible while the channel is disabled. After validation of the channel, the user must use PWM_CPRDUPDx register to update PWM_CPRDx as explained below.
- Configuration of the duty-cycle for each channel (CDTY in the PWM_CDTYx register). Writing in PWM_CDTYx register is possible while the channel is disabled. After validation of the channel, the user must use PWM_CDTYUPDx register to update PWM_CDTYx as explained below.
- Configuration of the dead-time generator for each channel (DTH and DTL in PWM_DTx) if enabled (DTE bit in the PWM_CMRx register). Writing in the PWM_DTx register is possible while the channel is disabled. After validation of the channel, the user must use PWM_DTUPDx register to update PWM_DTx
- Selection of the synchronous channels (SYNCx in the PWM_SCM register)
- Selection of the moment when the WRDY flag and the corresponding PDC transfer request are set (PTRM and PTRCS in the PWM_SCM register)
- Configuration of the update mode (UPDM in the PWM_SCM register)
- Configuration of the update period (UPR in the PWM_SCUP register) if needed.
- Configuration of the comparisons (PWM_CMPVx and PWM_CMPMx).
- Configuration of the event lines (PWM_ELMRx).
- Configuration of the fault inputs polarity (FPOL in PWM_FMR)
- Configuration of the fault protection (FMOD and FFIL in PWM_FMR, PWM_FPV and PWM_FPE/2)
- Enable of the Interrupts (writing CHIDx and FCHIDx in PWM_IER1 register, and writing WRDYE, ENDTXE, TXBUFE, UNRE, CMPMx and CMPUx in PWM_IER2 register)
- Enable of the PWM channels (writing CHIDx in the PWM_ENA register)

39.6.5.2 Source Clock Selection Criteria

The large number of source clocks can make selection difficult. The relationship between the value in the "PWM Channel Period Register" (PWM_CPRDx) and the "PWM Channel Duty Cycle Register" (PWM_CDTYx) can help the user in choosing. The event number written in the Period Register gives the PWM accuracy. The Duty-Cycle quantum cannot be lower than *1/CPRDx* value. The higher the value of PWM_CPRDx, the greater the PWM accuracy.

For example, if the user sets 15 (in decimal) in PWM_CPRDx, the user is able to set a value from between 1 up to 14 in PWM_CDTYx Register. The resulting duty-cycle quantum cannot be lower than 1/15 of the PWM period.

39.6.5.3 Changing the Duty-Cycle, the Period and the Dead-Times

It is possible to modulate the output waveform duty-cycle, period and dead-times.

To prevent unexpected output waveform, the user must use the "PWM Channel Duty Cycle Update Register" , the "PWM Channel Period Update Register" and the "PWM Channel Dead Time Update Register" (PWM_CDTYUPDx, PWM_CPRDUPDx and PWM_DTUPDx) to change waveform parameters while the channel is still enabled.

- If the channel is an asynchronous channel (SYNCx = 0 in "PWM Sync Channels Mode Register" (PWM_SCM)), these registers hold the new period, duty-cycle and dead-times values until the end of the current PWM period and update the values for the next period.
- If the channel is a synchronous channel and update method 0 is selected (SYNCx = 1 and UPDM = 0 in PWM_SCM register), these registers hold the new period, duty-cycle and dead-times values until the bit UPDULOCK is written at "1" (in "PWM Sync Channels Update Control Register" (PWM_SCUC)) and the end of the current PWM period, then update the values for the next period.
- If the channel is a synchronous channel and update method 1 or 2 is selected (SYNCx=1 and UPDM=1 or 2 in PWM_SCM register):
  - registers PWM_CPRDUPDx and PWM_DTUPDx hold the new period and dead-times values until the bit UPDULOCK is written at "1" (in PWM_SCUC register) and the end of the current PWM period, then update the values for the next period.
  - register PWM_CDTYUPDx holds the new duty-cycle value until the end of the update period of synchronous channels (when UPRCNT is equal to UPR in "PWM Sync Channels Update Period Register" (PWM_SCUP)) and the end of the current PWM period, then updates the value for the next period

Note: If the update registers PWM_CDTYUPDx, PWM_CPRDUPDx and PWM_DTUPDx are written several times between two updates, only the last written value is taken into account.