

Synchro - Synchronisation de générateur

Projet Conception numérique



Orientation : Énergie et techniques environnementales (ETE)
Cours : Conception numérique (Cnum)
Auteur : Christophe Bianchi, François Corthay, Silvan Zahno, Axel Amand
Date : 12 avril 2023
Version : v2.1

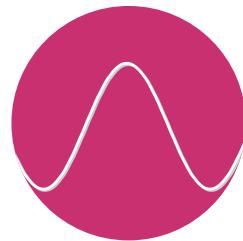
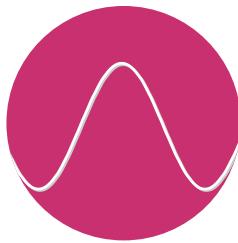


Table des matières

1	Introduction	2
2	Spécification	3
2.1	Fonctions	3
2.2	Circuit	3
2.2.1	Signaux	3
2.2.2	Todo	3
2.3	Projet HDL-Designer	4
3	Composants	5
3.1	Chariot du synchro	5
3.2	Carte FPGA	6
3.3	Boutons et LEDs	7
4	Evaluation	8
5	Premières étapes	9
5.1	Tips	9
	Références	10
	Acronymes	10



1 Introduction

Le but du projet est d'appliquer directement les connaissances acquises à un exemple pratique en fin de semestre. Il s'agit de figure 2

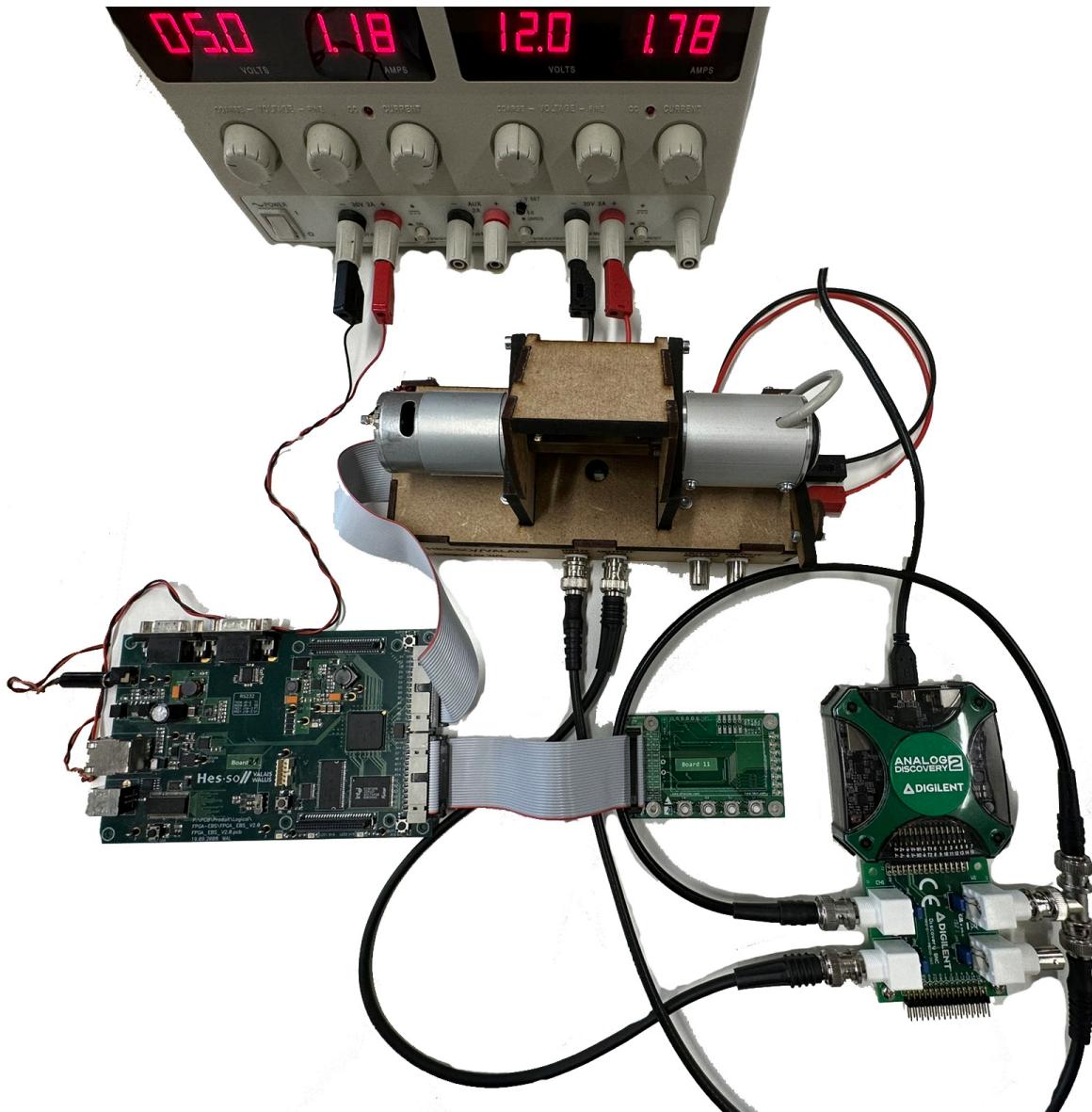
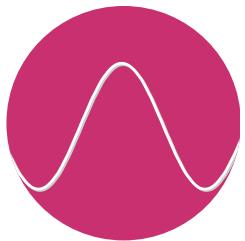


FIGURE 1 – Équipement du synchro

Le but est de réaliser la [Spécification](#) minimale définie au (chapitre 2). Les étudiants peuvent, en option, ajouter des fonctions supplémentaires. Il n'y a pas de limites aux idées, par exemple l'écran [LCD](#) peut être utilisé pour afficher certaines informations.



Les fonctions supplémentaires permettent d'obtenir quelques points supplémentaires



2 Spécification

2.1 Fonctions

Le signal de référence et la sortie du générateur sont numérisés sur 1 bit à l'aide de deux comparateurs. Le circuit numérique reçoit ces signaux et commande un moteur à courant continu couplé au générateur.

2.2 Circuit

Le circuit peut être contrôlé par 4 boutons. Il peut afficher des informations sur une rangée de 8 LEDs. Il fonctionne comme suit :

- La différence entre la période du secteur et la période du générateur est calculée.
- Si le générateur est trop lent, la tension du moteur CC est augmentée ; si le générateur est trop rapide, la tension du moteur CC est diminuée.

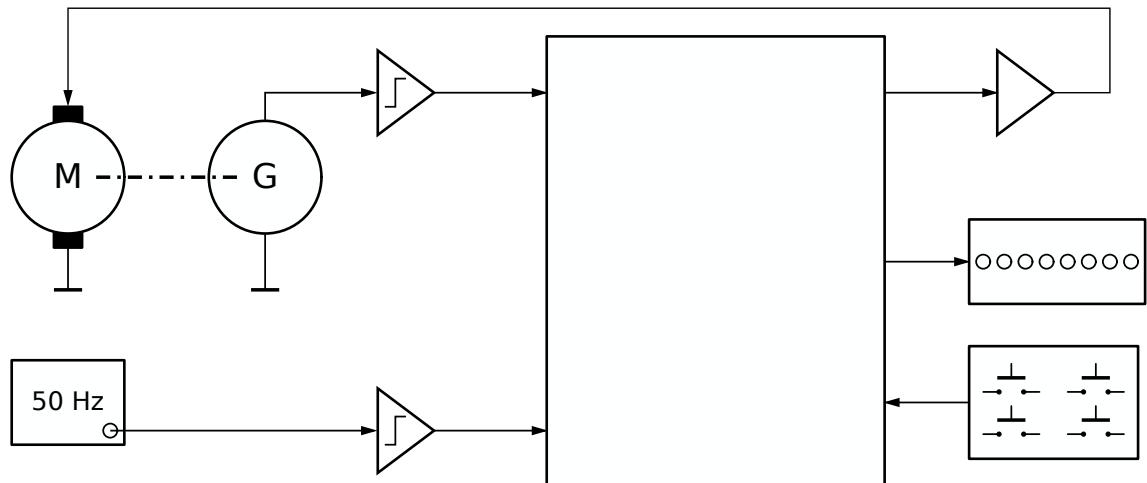


FIGURE 2 – Schéma du système

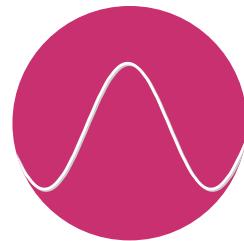
2.2.1 Signaux

Les signaux du système sont les suivants :

- **mains** et **generator** créent une impulsion à '1' à chaque tour du moteur, respectivement de la génératrice. Ils permettent par comptage et connaissance préalable de la fréquence d'horloge de déterminer les deux fréquences de rotation.
- Le vecteur **buttons** représente les boutons de la carte [Boutons et LEDs](#).
- Le vecteur **leds** représente les [LEDs](#) de la carte [Boutons et LEDs](#).
- Le signal **PWM** contrôle le moteur. La fréquence **PWM** maximale est de 100kHz.
- Le vecteur **testOut** permet de sortir des signaux de debug afin de les mesurer.

2.2.2 Todo

Le système peut être décomposé en 4 blocs principaux :



- Deux compteurs de période pour le moteur et la génératrice
- Un comparateur déterminant si le moteur doit être freiné, accéléré, ou maintenu tel quel
- Un régulateur permettant d'ajuster le duty-cycle de la **PWM**
 - Si la génératrice est plus lente, augmenter D+
 - Si le moteur est plus lent, augmenter D-
 - Sinon, ne pas changer le duty-cycle
- Un modulateur **PWM** qui génère un signal de sortie de maximum 100 kHz avec un duty-cycle selon l'amplitude donnée

2.3 Projet HDL-Designer

Un projet HDL-Designer prédéfini peut être téléchargé ou cloné dans [Cyberlearn](#). La structure de fichier du projet se présente comme suit :

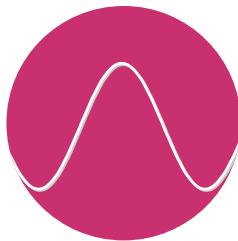
```
eln synchro
+--Board/                                # Project and files for programming the FPGA
|   +-concat/                            # Complete VHDL file including PIN-UCF file
|   +-ise/                               # Xilinx ISE project
+--GeneratorControl/                      # Library for the components of the student solution
+--GeneratorControl_test/                 # Library for the simulation testbenches
+--img/                                   # Pictures
+--Libs/                                  # External libraries which can be used e.g. gates, io, sequenti
+--Prefs/                                 # HDL-Designer settings
+--Scripts/                              # HDL-Designer scripts
+--Simulation/                           # Modelsim simulation files
```



Le chemin d'accès au dossier du projet ne doit pas contenir d'espaces



Dans le dossier de projet *doc/*, on peut trouver de nombreuses informations importantes : fiches techniques, évaluation de projet et documents d'aide pour HDL-Designer, pour n'en citer que quelques-unes



3 Composants

Le système se compose de 3 platines matérielles différentes, visibles dans la figure 2.

- Un assemblage avec un moteur couplé à un générateur, voir figure 3
- Une carte de développement **FPGA**, voir figure 6
- Une carte de contrôle à 4 boutons et 8 **LEDs**, voir figure 7

3.1 Chariot du synchro

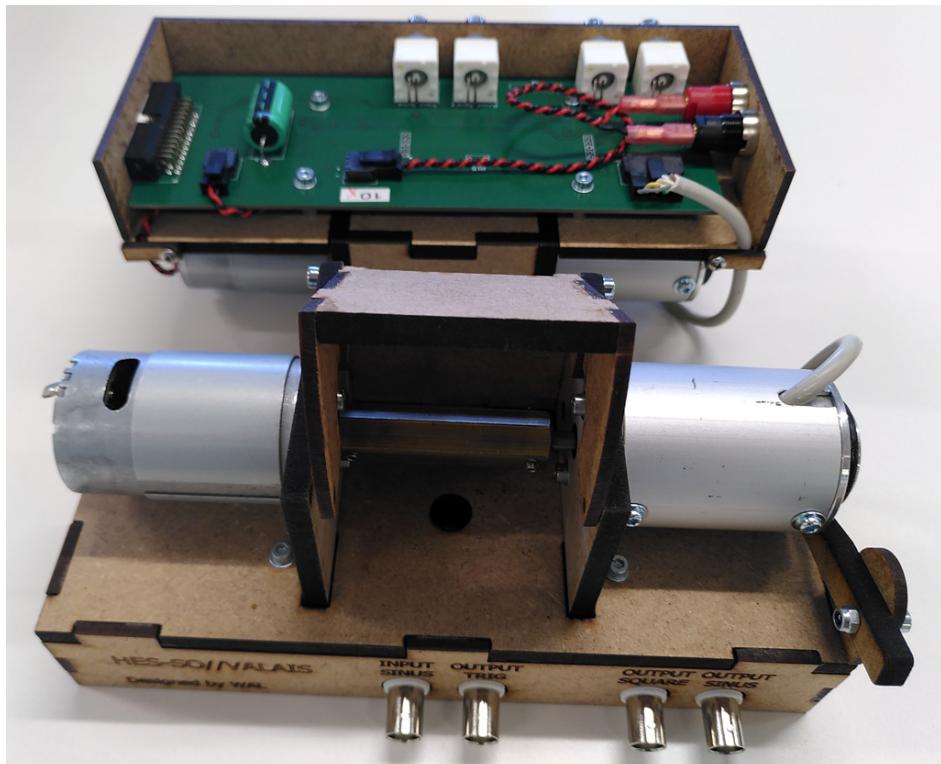


FIGURE 3 – Chariot du synchro

Afin d'observer une correcte régulation du système, il est possible d'injecter un sinus 50Hz sur l'entrée **Input sinusoïdale** du système. Cette dernière contrôle la **LED** située sous l'axe de couplage moteur-génératrice.

En oscillant à 50Hz, avec un système réglé à la même fréquence, l'axe semblera statique.

Il est possible d'utiliser l'oscilloscope numérique **Analog Discovery 2** avec sa sortie **W1** pour injecter ce signal. Il suffit d'ouvrir le fichier **Board/synchro.dwf3work** et de cliquer sur **Run** sous l'onglet **Wavegen 1** :

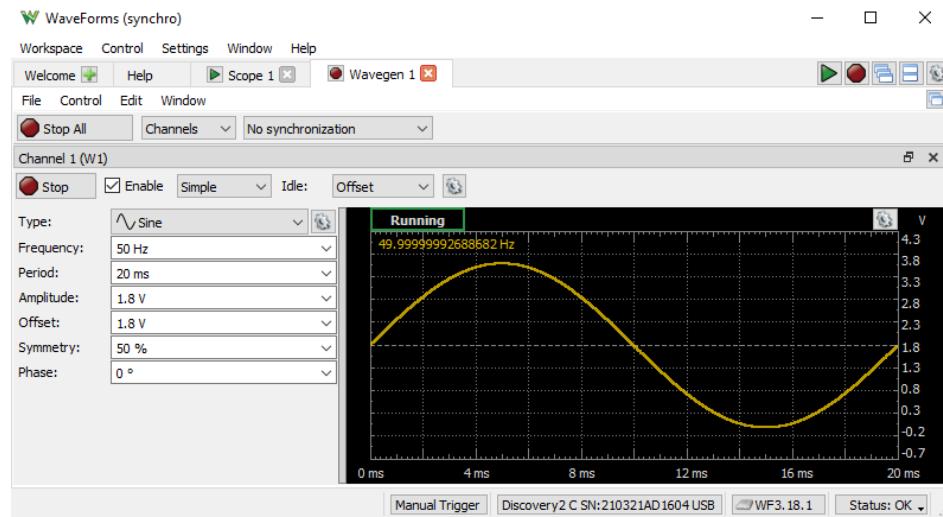
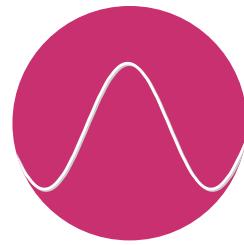


FIGURE 4 – Réglages sinus - Analog Discovery 2

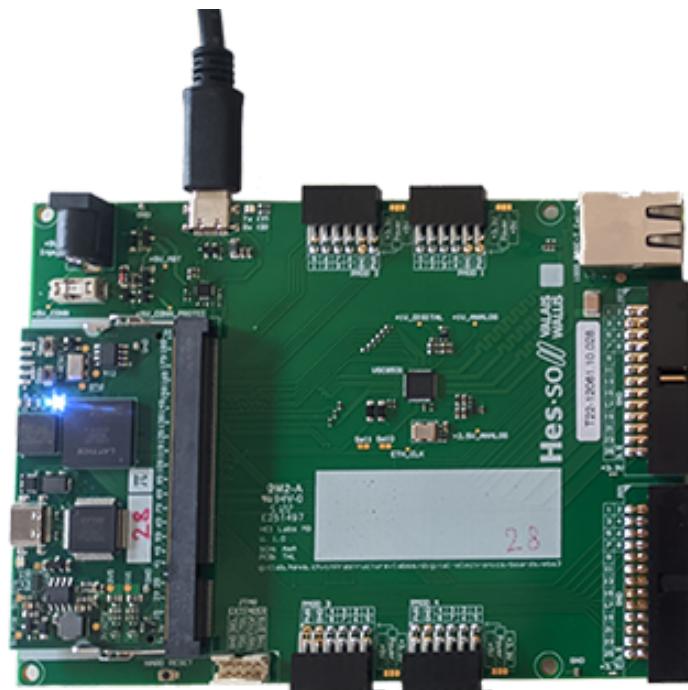
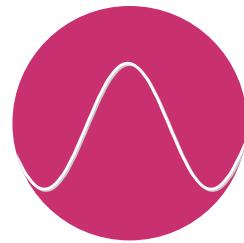
3.2 Carte FPGA

La carte principale est la carte de développement de laboratoire FPGA-EBS 2 de l'école [9]. Elle héberge une puce [Xilinx Spartan xc3s500e FPGA \[Spartan3FPGAFamily\]](#) [13] et dispose de nombreuses interfaces différentes ([Universal Asynchronous Receiver Transmitter \(UART\)](#), [Universal Serial Bus \(USB\)](#), Ethernet, etc.). L'oscillateur utilisé produit un signal d'horloge (*clock*) avec une fréquence de $f_{clk} = 66MHz$ [4].



FIGURE 5 – Carte électronique FPGA [9]

Sur la carte EBS3, l'oscillateur utilisé produit un signal d'horloge (*clock*) avec une fréquence de $f_{clk} = 100MHz$, réduit par PLL à $f_{clk} = 60MHz$.

FIGURE 6 – Carte électronique **FPGA** EBS3 [2]

Les simulateurs sont réglés par défaut pour les boards EBS3. Pour les modifier, ouvrez un bloc de testbench **xxx_tb** et double-cliquez sur les déclarations **Pre-User** (en haut à gauche de la page) pour modifier la variable **clockFrequency** selon la valeur de clock souhaitée.

3.3 Boutons et LEDs

La platine avec les boutons et les **LEDs** [10] est connectée à la platine **FPGA**. Elle a 4 boutons et 8 **LEDs** qui peuvent être utilisés dans le design. Si on le souhaite, cette platine peut être équipée d'un affichage **LCD** [11] [5].

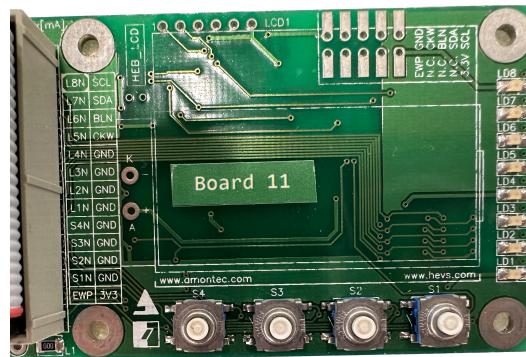
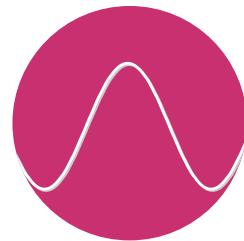


FIGURE 7 – Carte électronique boutons-LED-LCD [10]



4 Evaluation

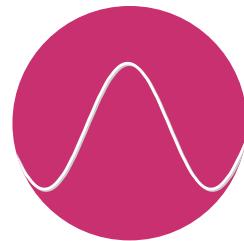
La note finale contient le rapport, le code ainsi qu'une présentation de votre système.

Aspects évalués	points
Rapport	54
Introduction	2
Cahier des charges	4
Conception	
Architecture du circuit (schéma-bloc)	8
Compteurs de période	8
Comparaison	4
Régulateur	8
Modulateur PWM	4
Vérification par simulation	2
Intégration sur plaque de test	2
Vérification sur banc de test	8
Conclusion	2
Aspects formels du rapport	2
Fonctionnalité du circuit	46
Démarrage, réglage	40
Bouton, LEDs	6
Options	0
Total	100

TABLE 1 – Grille d'évaluation



La grille d'évaluation donne des indications sur la structure du rapport. Pour un bon rapport, consultez le document "Comment rédiger un rapport de projet" [3]



5 Premières étapes

Pour commencer le projet, on peut procéder de la manière suivante :

- Lisez attentivement les spécifications et les informations ci-dessus.
- Examinez le matériel et testez le programme préprogrammé.
- Parcourez les documents dans le dossier *doc/* de votre projet.
- Dessinez sur papier le schéma bloc de votre système. Donnez des noms aux différents blocs et signaux avec leur nombre de bits.
- Développez un schéma fonctionnel détaillé. Vous devriez pouvoir expliquer les signaux et leurs fonctions.
- Implémenter et simuler les différents blocs.
- Testez la solution sur le circuit imprimé et trouvez les éventuelles erreurs .

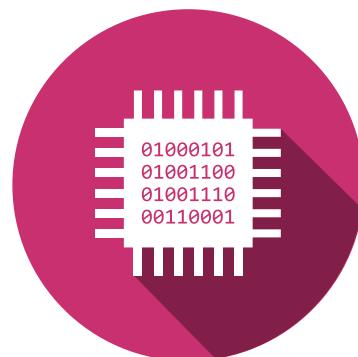
5.1 Tips

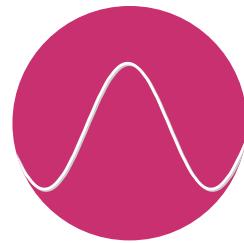
Ci-joint quelques conseils supplémentaires pour éviter les problèmes et les pertes de temps :

- Divisez le problème en différents blocs, utilisez pour cela le document Toplevel vide (*synchro-toplevel-empty.pdf*). Il est recommandé d'avoir un mélange équilibré entre le nombre de composants et la taille/complexité des composants.
- Analyser les différents signaux d'entrée et de sortie, pour cela il est conseillé d'utiliser en partie les fiches techniques.
- Respecter le chapitre DiD "Méthodologie de conception de circuits numériques (MET)" lors de la création du système. [6].



| N'oubliez pas de vous amuser 😊.





Références

- [1] AGILENT TECHNOLOGIES. *Datasheet Agilent AEDB-9140 Series Three Channel Optical Incremental Encoder Modules with Codewheel, 100 CPR to 500 CPR*. 2005.
- [2] AMAND AXEL. *Schematic : FPGA-EBS3 v1.0*. 2023.
- [3] CHRISTOPHE BIANCHI, FRANÇOIS CORTHAY et SILVAN ZAHNO. *Comment Rédiger Un Rapport de Projet ?* 2021.
- [4] CTS. *Datasheet CTS Model CB3 & CB3LV HCMOS/TTL Clock Oscillator*. 2006.
- [5] ELECTRONIC ASSEMBLY. *Datasheet : DOGM Graphics Series 132x32 Dots*. 2005.
- [6] FRANÇOIS CORTHAY, SILVAN ZAHNO et CHRISTOPHE BIANCHI. *Méthodologie de Conception de Circuits Numériques*. 2021.
- [7] OLIVIER WALPEN. *Schematic : Cursor Chariot Power Circuit*. 2009.
- [8] *Rotary Encoder*. In : Wikipedia. 23 août 2021. URL : https://en.wikipedia.org/w/index.php?title=Rotary_encoder&oldid=1040238329 (visité le 20/11/2021).
- [9] SILVAN ZAHNO. *Schematic : FPGA-EBS v2.2*. 2014.
- [10] SILVAN ZAHNO. *Schematic : Parallelport HEB LCD V2*. 2014.
- [11] SITRONIX. *Datasheet Sitronix ST7565R 65x1232 Dot Matrix LCD Controller/Driver*. 2006.
- [12] STMICROELECTRONICS. *Datasheet : DMOS Dual Full Bridge Driver with PWM Current Controller*. 2003.
- [13] XILINX. *Datasheet Spartan-3E FPGA Family*. 2008.

Acronymes

FPGA Field Programmable Gates Array. 1, 5–7

LCD Liquid Crystal Display. 2, 7

LED Light Emitting Diodes. 1, 3, 5, 7

PWM Pulse Width Modulation. 3, 4

UART Universal Asynchronous Receiver Transmitter. 6

USB Universal Serial Bus. 6