

Rapport de pré-étude

Ecole supérieure
Électronique

Laboratoire PROJ
Salle R110

Emetteur DCF

Réalisé par :

Culand Julie

A l'attention de :

M.Moreno & M.Castoldi

Dates :

Début du laboratoire : 21 novembre 2018
Fin du laboratoire : 20 juin 2019

Table des matières :

Emetteur DCF	1
1 Pré-étude	7
1.1 Cahier des charges	7
1.2 Schéma bloc du système	7
1.3 Schéma bloc du hardware	7
1.4 Choix technologiques	7
1.4.1 Microcontrôleur.....	7
1.4.2 Oscillateur	8
1.4.3 Antenne DCF.....	8
1.4.4 Affichage LCD	9
1.4.5 Amplificateur.....	9
1.4.6 Régulateur 5V à 3.3V	9
1.4.7 Connecteur RJ45	9
1.4.8 Connecteur micro-USB	9
1.4.9 Switchs	10
1.4.9.1 Choix mécaniques	10
1.4.9.1.1 Affichage LCD	10
1.4.9.2 Fixation de l'antenne	10
1.4.9.2.1 Dimensions PCB	11
1.4.9.2.2 Boîtier	11
1.5 Consommation	12
1.5.1 PIC32MX795F512H	12
1.5.2 Affichage LCD - DEM 20488 SYH-PY	12
1.5.2.1 Rétro-éclairage	12
1.5.2.2 Ecran	12
1.6 Interaction du système avec l'extérieur	13
1.7 Evaluation des coûts	13
1.8 Evaluation des coûts	15
1.9 Conclusion et perspectives.....	17
2 Design	19
2.1 Description du produit voulu	19
2.2 Choix technologiques du système.....	21
2.2.1 Microcontrôleur.....	21
2.2.2 Affichage LCD	21
2.2.3 Antenne DCF.....	21
2.2.4 Analog switch.....	21
2.2.5 Amplificateur.....	21
2.2.6 Régulateur 5V à 3V3	21
2.2.7 Connecteur RJ45	21
2.2.8 Connecteur micro-USB	22
2.2.9 Switchs	22
2.3 Choix des composants	22
2.3.1 Taille des boîtiers	22
2.3.2 Coûts.....	22
2.4 Consommation	25
2.5 Dimensionnement du hardware	26
2.5.1 Alimentation uUSB	26
2.5.2 Touches menu.....	26

2.5.3 Régulateur 5V à 3V3	27
2.5.4 Affichage LCD & uC	28
2.5.5 Signaux DCF	30
2.5.6 RESET du uC & Connecteur de programmation	33
2.5.7 Contrôleur Ethernet	34
2.6 Concept du logiciel	35
2.6.1 Partie DCF	35
2.6.2 Mise à jour de l'heure	35
2.6.3 Menu	35
2.7 Conclusion	36
3 Réalisation	37
3.1 Modifications apportées au schéma électrique	37
3.1.1 Affichage	37
3.1.2 Touches menu et RESET	37
3.1.3 Signaux DCF	38
3.2 PCB	39
3.2.1 Routage	39
3.2.2 Placement des composants	41
3.2.3 Sérigraphie	43
3.2.4 Eléments particuliers du PCB	43
3.3 Mécanique	44
3.3.1 Choix du boîtier	44
3.3.2 Emplacement de l'antenne	44
3.4 Montage	45
3.5 Test du PCB - Alimentations	46
3.6 Réalisation du software	46
3.6.1 Détails du protocole	46
3.6.2 Programmation sous MPLAB	47
3.6.2.1 Paramétrage de MPLAB Harmony	47
3.6.2.1.1 Pin Settings	47
3.6.2.1.2 Timer 1	48
3.6.2.1.3 Timer 2	49
3.6.2.1.4 Timer 3	50
3.6.2.1.5 OC3	51
3.6.2.1.6 OC4	51
3.6.3 Affichage LCD	52
3.6.3.1 Problèmes rencontrés	53
3.6.4 Ethernet	54
3.6.4.1 Programme de test	54
3.6.5 Tests	58
3.6.6 Gestion des menus	60
3.6.7 Signaux DCF	63
3.6.7.1 Porteuse	63
3.6.7.2 Explications	63
3.6.7.3 Analog switch	64
3.6.7.4 Test de la partie analogique	65
3.6.7.4.1 Possibilité 1	65
3.6.7.4.2 Possibilité 2	66
3.6.7.5 Mesure partie analogique	66
3.6.7.5.1 Mesure	67
3.6.7.6 Dépannage	68

3.6.7.6.1 Test de l'amplificateur OPA2347UA	69
3.6.7.6.1.1 Mesure @ 10kHz.....	69
3.6.7.6.1.2 Mesure @ 77.5kHz.....	70
3.6.7.6.1.3 Explication des mesures	70
3.7 Codage de l'heure.....	71
3.7.1.1 Machine d'état.....	73
3.7.1.2 Conversion des minutes et de l'heure en BCD	75
3.7.1.3 Extraction des bits	75
3.7.1.4 Vérification de la parité.....	77
3.7.2 Envoi des données	79
3.8 Travail restant à faire	80
3.9 Conclusion	80
3.10 Annexes	81

1 Pré-étude

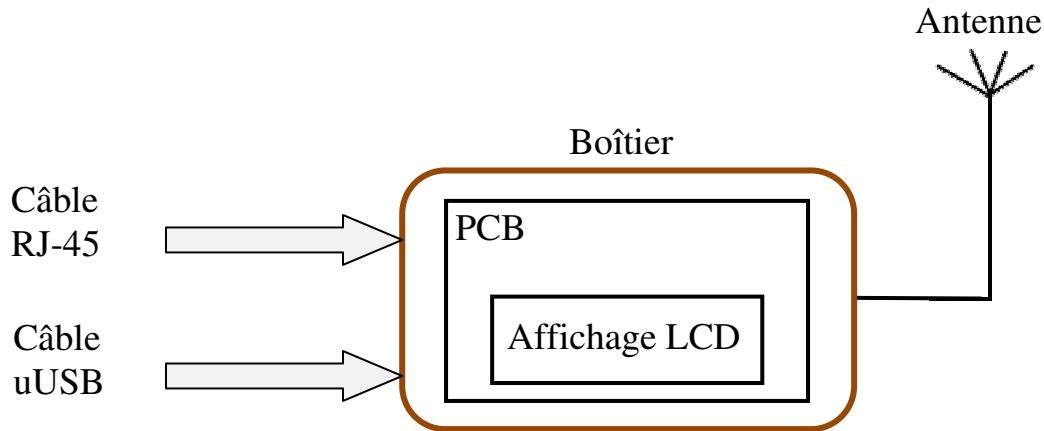
1.1 Cahier des charges

Voir annexe I : 1819_emetteurDcf-CDC-v1.doc

1.2 Schéma bloc du système

Voir annexe I : 1819_emetteurDcf-CDC-v1.doc

1.3 Schéma bloc du hardware



1.4 Choix technologiques

1.4.1 Microcontrôleur

Comme l'ES travaille avec la famille de microcontrôleur PIC32MX, j'ai décidé de reprendre le même modèle utilisé pour la carte de développement de l'ES, mais la taille en dessous, soit un 64pins (PIC32MX795F512H) au lieu d'un 100pins, car cela représente un nombre de patte beaucoup plus important que ce dont j'ai besoin.

Ce uC a également été choisi car il est muni d'une interface Ethernet.

Et de plus, l'environnement de programmation utilisé (MPLAB X IDE) m'est familier, et donc plus facilement utilisable et compréhensible.

Le PIC32MX795F512H a donc la signification suivante :

32	Le uC travaille sur 32bits
MX	Type d'architecture utilisée pour le uC
795	Numéro d'identification
F	Indique la famille de mémoire, ici Flash
512	Taille en [KB] de la mémoire programmable
L	Correspond au nombre de patte, ici 64 pins

1.4.2 Oscillateur

L'oscillateur d'une fréquence de 77.5kHz, sera généré par un timer du microcontrôleur.

Les Timers du PIC32MX795F512H, ont une résolution de 2^{16} , soit 65'536.

Le clock du microcontrôleur étant de 80MHz, celui-ci sera donc largement suffisant pour générer la fréquence de la porteuse

Features:

- 80MHz/105DMIPS, 32-bit MIPS M4K® Core

Peripheral Features (Continued):

- Internal 8 MHz and 32 kHz oscillators
- Six UART modules with:
 - RS-232, RS-485 and LIN support
 - IrDA® with on-chip hardware encoder and decoder
- Up to four SPI modules
- Up to five I²C™ modules
- Separate PLLs for CPU and USB clocks
- Parallel Master and Slave Port (PMP/PSP) with 8-bit and 16-bit data, and up to 16 address lines
- Hardware Real-Time Clock and Calendar (RTCC)
- Five 16-bit Timers/Counters (two 16-bit pairs combine to create two 32-bit timers)

$$f_{\text{porteuse}} = 77.5 \text{kHz}$$

$$f_{\text{clock}} = 80 \text{MHz}$$

$$T_{\text{porteuse}} = \frac{1}{f_{\text{porteuse}}} = \frac{1}{77.5 \times 10^3} = 12.90 \mu\text{s}$$

$$T_{\text{clock}} = \frac{1}{f_{\text{clock}}} = \frac{1}{80 \times 10^6} = 12.5 \text{ns}$$

$$\text{NbTics} = \frac{T_{\text{porteuse}}}{T_{\text{clock}}} = \frac{12.90 \times 10^{-6}}{12.5 \times 10^{-9}} = 1032$$

Après calcul, on voit que le nombre de tick devra être de 1032, afin de pouvoir générer une fréquence de 77.5kHz, à partir d'un Timer.

Comme l'échelle de travail de ceux-ci est de 0 à 65'535, la valeur est donc largement dans cette gamme.

1.4.3 Antenne DCF

L'antenne pour émettre le signal, sera reprise directement d'une platine de réception DCF 2.5-15V/DC - 3mA, car il est difficile de trouver une antenne seule, sans module.

De plus, l'antenne choisie possède déjà un condensateur de filtrage.

1.4.4 Affichage LCD

Pour l'affichage LCD, j'ai choisi la famille DEM 20XXX, car c'est celui utilisé sur le PIC32 de l'ETML-ES, son environnement de travail m'est donc connu, au niveau de la programmation et de son utilisation.

Le modèle DEM 20488 SYH-PY 4x20 caractères, me semble adapté pour mon application, de par sa taille (77x47), qui permettra à l'utilisera d'avoir un affichage ni trop grand, ni trop petit, et qui sera donc confortable à regarder.

De plus, la tension doit être comprise entre 4.5V et 5.5V, il pourra ainsi être alimenté par l'alimentation principale du circuit (micro-USB +5V).

1.4.5 Amplificateur

Pour définir quel type d'amplificateur opérationnel sera utilisé dans l'étage de sortie, nous devons tenir compte des caractéristiques suivantes :

- Transmission de données numériques : besoin de gérer des flancs raides => Faire attention au Slew Rate.
- Circuit alimenté en 5V : choisir un AOP compatible avec une alimentation 0-5V asymétrique.

1.4.6 Régulateur 5V à 3.3V

Afin d'alimenter le microcontrôleur en +3.3V, j'ai décidé d'utiliser un régulateur à découpage, car ceux-ci ont un meilleur rendement que les régulateurs linéaires.

J'ai ainsi choisi le modèle TPS54228DR, qui a une valeur d'entrée entre 4.5 et 18V, donc qui pourra être alimenté par l'alimentation principale, qui est de +5V.

La tension de sortie est ajustable entre 0.76V et 7V, et le courant est de 2A.

Il a été choisi en fonction du courant de consommation du circuit (voir point 5). Nous avons un courant de consommation d'approximativement 150mA. Mais prenons une marge à 2A car à ce stade nous ne connaissons pas encore la puissance d'émission du signal.

1.4.7 Connecteur RJ45

Pour le connecteur RJ45, pour le câble Ethernet, j'ai repris celui déjà présent sur le Kit PIC32MX, comme il correspond à la connexion standard utilisée par l'école. C'est donc le modèle SI-60062-F.

1.4.8 Connecteur micro-USB

Le connecteur micro-USB choisi est de type B, car actuellement c'est le type qui est le plus couramment utilisé (chargeur smartphone, etc...).

1.4.9 Switchs

Pour les switchs, j'ai décidé de prendre ceux de chez Würth Elektronik, car ils sont simples d'utilisation et de plus, la gamme possède de nombreux switchs avec des hauteurs différentes.

Cela me sera très pratique du fait que, comme le PCB sera placé dans un boîtier, cela signifie que les touches devront être accessible par l'utilisateur, et donc présenter une certaine hauteur pour permettre cela.

1.4.9.1 Choix mécaniques

1.4.9.1.1 Affichage LCD

Au début, je pensais placer l'affichage LCD sur le circuit avec une barrette à broche pour réaliser les connexions, ainsi que des colonnettes pour fixer celui-ci.

Cela engendre que les switchs, qui permettront de régler les menus du LCD, doivent avoir une certaine hauteur, c'est-à-dire être plus élevé que l'affichage, afin d'être accessible par l'utilisateur, lorsque la carte sera dans le boîtier.

Or, il s'est avéré difficile de trouver des touches suffisamment hautes. J'ai donc pensé à la solution des touches encastrées, soit à 90°. Ce qui signifie que celles-ci seraient placées sur l'un des côtés du boîtier, au lieu de la face avant.

Après réflexion, cette solution ne m'a pas parue optimale, de par le fait que leurs emplacements ne seraient pas agréables à l'utilisateur, et que de plus, celles-ci ont un coût relativement élevé.

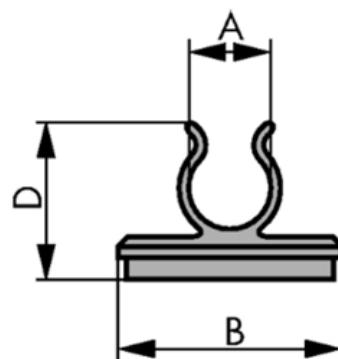
Pour remédier à ces problèmes, j'ai décidé que mon affichage LCD serait placé, non pas sur la face TOP de mon circuit, mais sur la face BOTTOM:

Cela veut dire qu'il y aurait une ouverture faite pour que seulement l'affichage puisse dépasser et la barrette à broche, ainsi que les colonnettes seraient également sur l'autre face.

1.4.9.2 Fixation de l'antenne

Comme l'antenne représente une surface importante, celle-ci ne sera pas directement placée dans le boîtier.

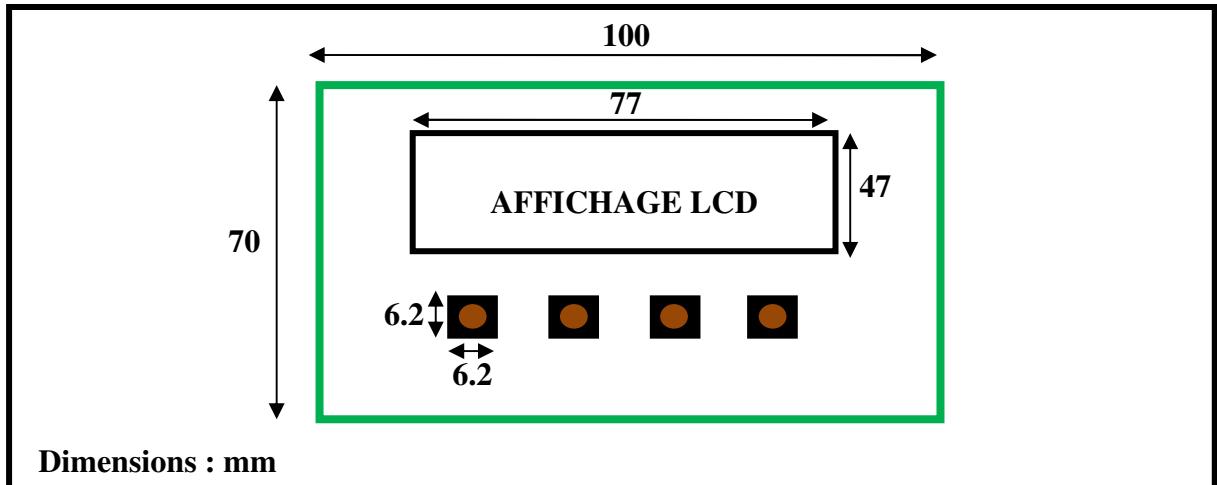
J'ai donc pensé à la placer dans un support de fixation, qui serait vissé sur l'un des côtés du boîtier, aux deux extrémités de l'antenne.



1.4.9.2.1 Dimensions PCB

Comme le composant le plus encombrant est l'affichage (77x47mm), et que le côté TOP n'aura certainement que l'affichage et les touches, j'ai donc estimé approximativement la grandeur du PCB.

J'ai donc, dans la longueur, et dans la largeur, laissé une marge d'environ 20mm, ce qui donnerait les dimensions suivantes : 100mm x 70mm



1.4.9.2.2 Boîtier

Concernant le boîtier, plusieurs idées sont possibles, j'ai donc pensé aux propositions suivantes :

- Boîtier plastique avec face avant transparente
- Boîtier plastique couleur uniforme
- Boîtier plastique « pupitre »

Plusieurs critères seront donc pris en compte lors du choix du boîtier :

- Agréable d'utilisation pour l'utilisateur (boîtier pas trop grand)
- Hauteur (PCB avec connecteur RJ-45 et micro-USB + Affichage et switchs)

1.5 Consommation

COMPOSANT			CONSOMMATION		
PIC32MX795F512H			120mA (pire cas)		
Affichage LCD - DEM 20488 SYH-PY			Rétro-éclairage 30mA - Ecran 0.5mA => 30.5mA		

1.5.1 PIC32MX795F512H

DC23	85	98	mA	Code executing from Flash	-40°C, +25°C, +85°C	—	80 MHz
	DC23b	90			+105°C		
	DC23a	85		Code executing from SRAM	—		

Fréquence = 80MHz Pire cas +105° soit 120mA

1.5.2 Affichage LCD - DEM 20488 SYH-PY

1.5.2.1 Rétro-éclairage

7.2 Electrical-Optical Characteristics

ITEM	SYMBOL	MIN.	TYP.	MAX	UNIT	CONDITION
Forward Current	If	---	12x2	15x2	mA	Vf=4.5V
Reverse Current	Ir	---	15x2	---	uA	Vr=4V
Peak Wave length	λ_P	569	572	575	nm	If=20*2mA
Spectral Line Half Width	$\Delta\lambda$	---	30	---	nm	If=20*2mA
Luminance	Lv	---	---	---	cd/m ²	Vf=4.5V
Backlight color	Yellow-Green					

Forward Current : If = 2 * 15mA = 30mA

1.5.2.2 Ecran

10. ELECTRICAL CHARACTERISTICS

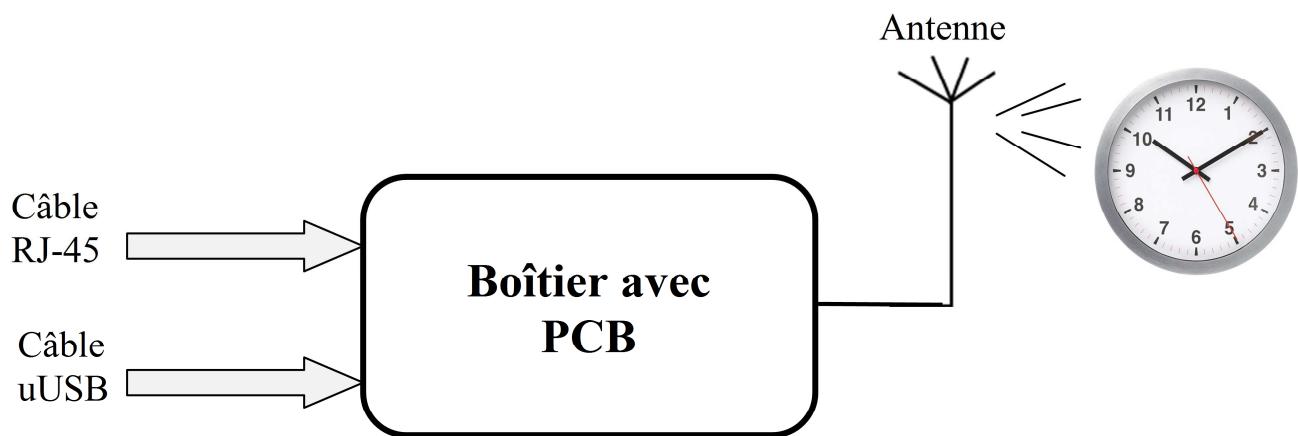
10-1. DC Characteristics (VDD = 4.5V ~ 5.5V, Ta = -20 ~ +70°C)

Item	Symbol	Standard Value			Test Condition	Unit
		MIN	TYP	MAX		
Operating Voltage	V _{DD}	4.5	5.0	5.5	-----	V
LCD Voltage	V _{LCD}	3.0	4.5	10.0	V _{DD} - V5	V
Supply Current	I _{DD}	---	0.2	0.5	V _{DD} =5V, fosc=270kHz	mA

Supply Current : I_{DD} (MAX) = 0.5mA

Consommation totale : If + I_{DD} (MAX) = 30*10⁻³ + 0.5*10⁻³ = 30.5mA

1.6 Interaction du système avec l'extérieur



Le système va interagir avec les éléments extérieurs suivants :

- Le connecteur Ethernet recevra l'heure via un câble RJ45
- Le circuit est alimenté par un câble micro-USB
- L'antenne à l'extérieur du boîtier va permettre de transmettre le signal qui aura les données de l'heure, pour régler automatiquement les horloges de l'école

1.7 Evaluation des coûts

D'après la taille que j'ai estimée pour le PCB, j'ai réalisé, sous Euro-circuit, une "simulation" afin d'estimer le prix du circuit.

Price summary		
Service	PCB proto	
Delivery term	7 working days	
Estimated shipment date	04-01-2019	
Quantity	1 PCBs	
Board surface / Order surface	0.70 dm ² / 0.70 dm ²	
Prices		
Single PCB	Net € 44.44	Gross* € 47.86
Total boards	€ 44.44	€ 47.86
Express transport	€ 9.96	€ 10.73
Total	€ 54.40	€ 58.59

1 euro = 1.13 Franc suisse

Le PCB coûterait donc : $58.59 * 1.13 = 66.20.-$

1.8 Evaluation des coûts

Pour l'évaluation des coûts, j'ai, dans le tableau ci-dessous, mis tous les éléments principaux du système, comme le microcontrôleur, l'affichage, etc...

Le choix de certains composants n'est pas encore définitif, mais cela permet d'avoir un ordre d'idée des prix, comme pour le PCB et le boîtier, qui seront définis dans la phase suivante de design.

Quantités	Description	Tension Tolérance	Type	Fabriquant	Fournisseur	N° commande	Prix unité	Prix total
1	PIC32MX795F512H-80I/PT	3.6V max	32 bit TQFP 64	Microchip	Distrelec	173-87-440	10.90.-	10.90.-
1	Affichage LCD	---	DEM 20488 SYH-PY	Display Elektronik	Distrelec	301-02-071	22.00.-	22.00.-
2	Antenne DCF	2.5-15V/DC 3mA	Module récepteur DCF avec antenne Ferrite	---	Conrad	641138-62	16.95.-	33.90.-
1	Régulateur de tension	0.76 - 7V / 2A	TPS54228 SOIC-8	Texas Instruments	MOUSER	595-TPS54228DR	1.58.-	1.58.-
1	Connecteur RJ45 Femelle	---	SI-60062-F	STEWART CONNECTOR	MOUSER	530-SI-60062-F	4.33.-	4.33.-
1	Connecteur micro USB 2.0 5	---	47491-0001	MOLEX	Distrelec	300-76-400	1.06.-	1.06.-
4	Switch SMD	12V(DC)	WS-TASV SMD Tact Switch	Würth Elektronik	Würth Elektronik	430152095836	---	---
1	PCB	---	PCB proto - 2 layers	Euro-corcuit	Euro-corcuit	---	66.20.-	66.20.-
1	Boîtier plastique uniforme	---	125x85x55 Blanc polycarbonate IP65	Hammond	Distrelec	300-34-275	14.30.-	14.30.-
								Prix total : 154.27.-

1.9 Conclusion et perspectives

Cette première phase du projet m'a permis de me familiariser avec, et d'étudier de nouvelles notions, comme le DCF.

Celle-ci s'est bien passée, et est à mon avis importante, non seulement pour étudier les divers aspects du système, mais aussi les diverses technologies qui composeront celui-ci.

Pour la suite du projet, il va falloir choisir les composants définitifs pour les diverses parties du montage. Ce choix se fera en fonction de certaines contraintes (alimentation, consommation, etc...), mais aussi en fonction des autres projets.

C'est-à-dire que comme il y a plusieurs projets en cours, et que nous sommes plusieurs à avoir besoin de certains composants similaires, comme un affichage LCD, par exemple, le choix sera donc regroupé. Ceci pour non seulement simplifier les commandes, mais également pour se mettre dans une situation de travail en entreprise.

Ensuite, une fois les composants choisis, il faudra réaliser le dimensionnement de toutes les parties qui vont composer le système, pour la réalisation du schéma électrique.

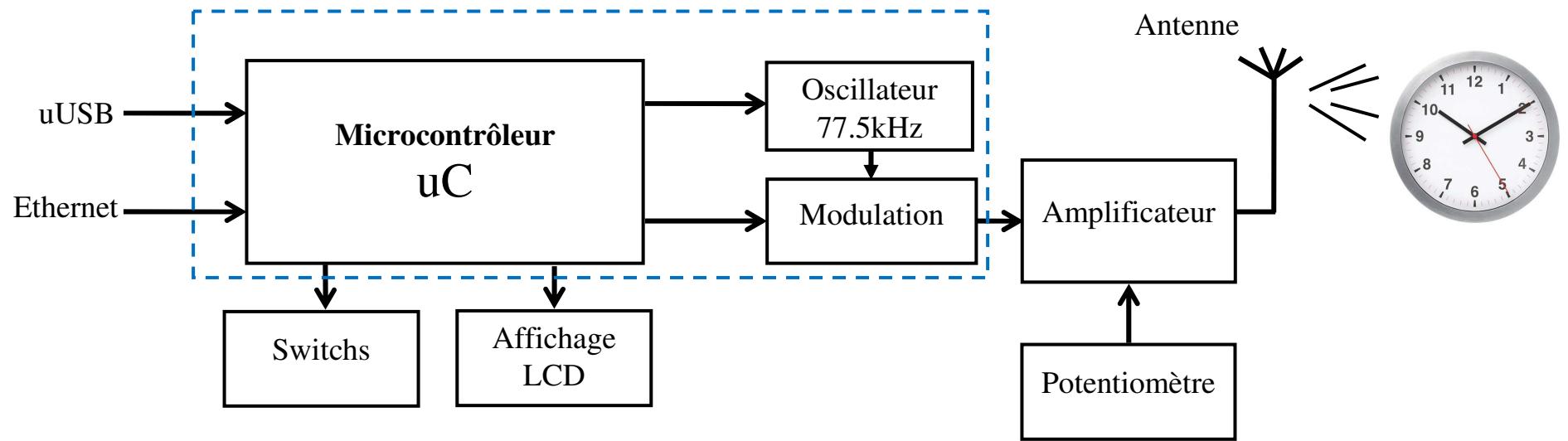
Concernant la partie mécanique, il faudra prêter attention au choix du boîtier, car il faut que les connecteurs micro-USB et RJ45 soient accessible, et également les touches.

Date : 13 décembre 2018

Signature : Julie Culand

2 Design

2.1 Description du produit voulu



Le système a pour but de créer un signal de remplacement, de l'émetteur DCF77, qui va permettre de régler les horloges de l'école ES.

Le signal qui est créé, doit comporter une porteuse d'une fréquence de 77.5kHz, qui sera générée par l'un des Timer du microcontrôleur. De même pour la modulation, celle-ci sera réalisée par le microcontrôleur.

Le signal modulé sera ensuite, amplifié et transmis via une antenne à l'extérieur du circuit.

La puissance d'émission pourra être réglée à l'aide d'un potentiomètre.

2.2 Choix technologiques du système

2.2.1 Microcontrôleur

Comme il m'a été imposé de travailler avec la famille PIC32, j'ai choisi de prendre le modèle PIC32MX795F512H (64 pins), car il possède les connexions nécessaires pour l'Ethernet. Et de plus, un précédent projet a été réalisé avec ce microcontrôleur et une partie Ethernet, que je vais donc reprendre de ce projet, comme c'est une partie du système qui n'est pas étudiée à l'ES, et qui ne m'est ainsi pas connue.

2.2.2 Affichage LCD

Concernant l'affichage LCD, le choix c'est porter sur une décision de groupe, comme nous sommes plusieurs à en avoir besoin. L'écran choisi a une tension d'entrée entre 3V et 3.6V, il pourra donc être alimenté par le régulateur 3V3. Il est doté d'une interface parallèle et de 4x20 caractères. Le modèle est le LCD MOD CHAR 4X20 GRY TRANSF STN.

2.2.3 Antenne DCF

L'antenne pour émettre le signal sera reprise directement d'une platine de réception DCF 2.5-15VDC - 3mA, car il est difficile de trouver une antenne seule, sans module.

De plus, l'antenne choisie possède déjà un condensateur de filtrage.

2.2.4 Analog switch

L'analog switch choisi est le modèle TS12A4515. Il est alimenté entre 2.7V et 12V, donc il pourra être alimenté par le uUSB (+5V).

Celui-ci est d'une extrême simplicité, puisque il ne possède qu'un switch, et de plus, il est peu coûteux.

2.2.5 Amplificateur

L'amplificateur opérationnel choisi, possède une tension d'alimentation asymétrique entre 2.3V à 5.5V, ce qui permet de l'alimenter avec celle principale (+5V).

De plus, il est rail-to-rail, cela implique qu'il peut atteindre les niveaux de tensions d'alimentation en sortie (pas tension de déchet).

l'IC possède également deux amplis, c'est le modèle OPA2347UA

2.2.6 Régulateur 5V à 3V3

Pour avoir une tension de 3V3, pour alimenter le microcontrôleur, j'ai repris le régulateur linéaire utilisé sur le Kit PIC32 (Projet 1102x_SK32MX775F512L), qui est un MAX1793, car il est en stock à l'ES.

Celui-ci a une tension d'entrée entre 2.5V et 5.5V, ce qui va permettre de l'alimenter avec l'alimentation principale du circuit (micro-USB +5V). De plus, la tension de sortie est ajustable entre 1.25V à 5.5V/3.3V. Le courant en sortie est de 1A, ce qui suffira largement pour alimenter le circuit. (Voir Annexe IV : Consommation)

2.2.7 Connecteur RJ45

Comme l'ES utilise un connecteur RJ45 sur la carte de développement Kit PIC32 (Projet 1102x_SK32MX775F512L), j'ai décidé de reprendre le même, car celui-ci correspond à la connexion standard utilisée par l'école. C'est donc le modèle SI-60062-F.

2.2.8 Connecteur micro-USB

Le connecteur micro-USB utilisé pour alimenter le circuit, est de type B, car actuellement, c'est le modèle le plus utilisé (chargeur de smartphone, etc...), il est donc facile de se procurer un câble d'alimentation, et à moindre coût.

Le modèle choisi est le 47346-0001 de chez MOLEX.

2.2.9 Switchs

Comme les touches doivent être accessibles par l'utilisateur à l'extérieur du boîtier, j'ai choisi de prendre les switchs de chez Würth Elektronik, car ceux-ci présentent un choix au niveau des hauteurs. J'ai donc pris le modèle 430152095836, qui a une hauteur de 9.5mm.

2.3 Choix des composants

2.3.1 Taille des boîtiers

Pour les composants passifs (résistances et condensateurs), j'ai décidé de prendre une taille de boîtiers de 0805. Ce choix c'est en premier lieu porter sur le fait que l'ETML-ES possède la majeure partie de son stock de composants en cette taille, ce qui va permettre de limiter les frais. Et deuxièmement, ce sont des composants que l'on peut aisément braser à la main.

2.3.2 Coûts

En ce qui concerne directement les composants mêmes, j'ai privilégié le stock de l'ETML-ES, afin de me mettre dans des conditions réelles de travail, comme dans une entreprise, où il est important d'utiliser d'abord le stock, puis de commander ce qui n'est pas disponible.

Pour l'estimation des coûts, j'ai regroupé les principaux composants du montage et j'ai réalisé une estimation pour le PCB, le boîtier, ainsi que pour les petits composants (résistances, condensateurs, etc..).

Concernant les choix des fournisseurs, je me suis axée sur Digi-key, qui est souvent moins cher que Distrelec et Mouser. Autrement, pour les composants non disponible chez Digi-key, je me suis retournée auprès des autres fournisseurs.

Estimation des coûts

Pour l'évaluation des coûts, j'ai dans le tableau ci-dessous, mis tous les éléments principaux du système, comme le microcontrôleur, l'affichage, l'antenne, etc...

Concernant le prix des petits composants (résistances, condensateurs, ...), j'ai réalisé une estimation d'environ 10.-.

Pour le PCB et le boîtier, j'ai réalisé une approximation comme leurs caractéristiques finales ne sont pas encore totalement connues (tailles, matières, etc..).

Quantités	Description	Tension Tolérance	Type	Fabriquant	Fournisseur	Nº commande	Prix unité	Prix total
1	PIC32MX795F512H-80I/PT	3.6V max	32 bit TQFP 64	Microchip	Digi-key	PIC32MX795F512H-80I/PT-ND	8.92.-	8.92.-
1	Affichage LCD	3V - 3.6V	LCD MOD CHAR 4x20 GRY TRANSF STN	Newhaven Display Intl	Digi-key	NHD-0420AZ-FSW-GBW-33V3-ND	19.10.-	19.10.-
2	Antenne DCF	2.5-15V/DC 3mA	Module récepteur DCF avec antenne Ferrite	---	Conrad	641138-62	16.95.-	33.90.-
1	Régulateur MAX1793EUE33+T	3.3V / 1A	TSSOP-16	Maxim Integrated	MOUSER	700-MAX1793EUE33T	3.73.-	3.73.-
1	Connecteur RJ45 Femelle	---	SI-60062-F	STEWART CONNECTOR	Digi-key	380-1106-ND	4.53.-	4.53.-
1	Connecteur micro USB 2.0	---	47346-0001	MOLEX	MOUSER	538-47346-0001	0.89.-	0.89.-
4	Switchs	12V(DC)	430152095836	Würth elektronik	Würth elektronik	430152095836	---	---
1	Switchs - RESET	12V(DC)	430152043826	Würth elektronik	Würth elektronik	430152043826	---	---
2	Résistance ajustable 10kΩ	---	Through Hole	Bourns	MOUSER	652-3362P-1-103LF	1.02.-	2.04.-
1	Amplificateur opérationnel	5.5V / 17mA	SOIC-8	Texas Instruments	Digi-key	OPA2347UA-ND	1.36.-	1.36.-
1	MOSFET IRLML2402TRPBF	20V / 1.2A	SOT23-3	Infineon / IR	Digi-key	IRLML2402PBFCT-ND	0.43.-	0.43.-
1	CI Ethernet	3.3V / 92mA	HLQFP-48	Texas Instruments	Digi-key	DP83848VYB/NOPB-ND	23.88.-	23.88.-
1	IC Analog Switch SPST TS12A4515	2.7V - 12V	8-SOIC	Texas Instruments	Digi-key	296-21909-1-ND	0.77.-	0.77.-
1	Quartz crystal 8MHz	---	HC-49/US	ECS	MOUSER	520-CSM800-18-X	0.46.-	0.46.-
1	XTAL OSC XO 50MHz CMOS SMD	3.3V	4-SMD	TXC CORPORATION	Digi-key	887-1379-1-ND	2.25.-	2.25.-
1	Bornier 2 pôles	---	691214110002	Würth elektronik	Würth elektronik	691214110002	---	---
1	PCB 100x70	---	PCB proto - 2 layers	Euro-circuit	Euro-circuit	---	66.20.-	66.20.-
1	Boîtier plastique uniforme	---	125x85x55 Blanc polycarbonate IP65	Hammond	Distrelec	300-34-275	14.30.-	14.30.-
1	Petits composants (résistances, condensateur, etc...)	---	---	---	---	---	---	10.00.-
Coût total : 192.76.-								

2.4 Consommation

COMPOSANT	CONSOMMATION
PIC32MX795F512H	120mA (pire cas)
Affichage LCD - NHD-0420AZ-FSW-GBW	Rétro-éclairage 30mA Ecran 2mA => 32mA
Contrôleur Ethernet - DP83848VYB	~100mA
Consommation totale :	~ 252mA

PIC32MX795F512H

DC23	85	98	mA	Code executing from Flash	-40°C, +25°C, +85°C	—	80 MHz
DC23b	90	120			+105°C		
DC23a	85	—		Code executing from SRAM	—		

Consommation => Fréquence = 80MHz Pire cas +105° soit 120mA

Affichage LCD - DEM 20488 SYH-PY

Electrical Characteristics

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
Operating Temperature Range	T_{OP}	Absolute Max	-20	-	+70	°C
Storage Temperature Range	T_{ST}	Absolute Max	-30	-	+80	°C
Supply Voltage	V_{DD}	-	3.0	3.3	3.6	V
Supply Current	I_{DD}	$V_{DD} = 3.3V$	0.5	1.0	2.0	mA
Supply for LCD (contrast)	V_{LCD}	$T_{OP} = 25°C$	3.1	3.3	3.5	V
"H" Level input	V_{IH}	-	$0.7*V_{DD}$	-	V_{DD}	V
"L" Level input	V_{IL}	-	V_{SS}	-	0.6	V
"H" Level output	V_{OH}	-	$0.75*V_{DD}$	-	V_{DD}	V
"L" Level output	V_{OL}	-	V_{SS}	-	$0.2*V_{DD}$	V
Backlight Supply Voltage	V_{LED}	-	2.8	3.0	3.2	V
Backlight Supply Current	I_{LED}	$V_{LED} = 3.0V$	5	20	30	mA

Consommation => Rétro-éclairage + Ecran : ~ 32mA

Contrôleur Ethernet

5.5 DC Specifications

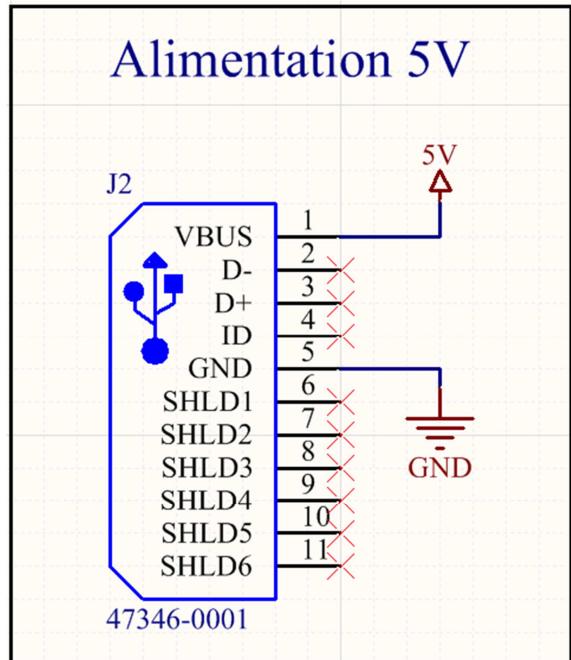
V_{TH1}	10BASE-T Receive Threshold		PMD Input Pair	585	mV
I_{dd100}	100BASE-TX		Supply	81	mA
	(Full Duplex)				
I_{dd10}	100BASE-TX		Supply	92	mA
	(Full Duplex)				
I_{dd}	Power Down Mode	CLK2MAC disabled	Supply	14	mA

Consommation => ~ 100mA

2.5 Dimensionnement du hardware

2.5.1 Alimentation uUSB

Comme le connecteur micro-USB est utilisé seulement pour alimenter le circuit, seules les pattes VBUS (+5V) et GND (0V) sont utiles.



2.5.2 Touches menu

Lorsque l'on regarde dans le datasheet du microcontrôleur (PIC32MX795F512H), on peut voir que l'entrée d'une pin consomme maximum 25mA, et la tension est de 3.3V.

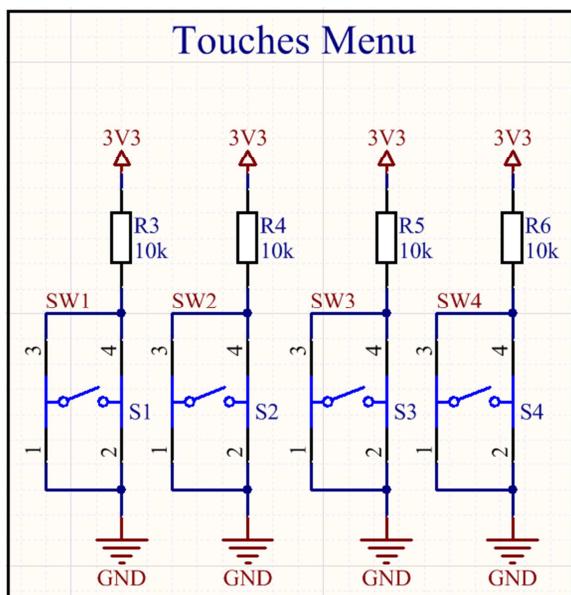
Maximum output current sunk by any I/O pin.....	25 mA
Maximum output current sourced by any I/O pin	25 mA

On peut donc en déduire la valeur de la résistance :

$$R = \frac{V_{cc} - V_{pin}}{I} = \frac{3.3 - 0}{25 \cdot 10^{-3}} = 132\Omega$$

Or, une consommation de 25mA sur une pin est relativement élevé, c'est pourquoi j'ai placé des résistances de 10kΩ, qui est un choix arbitraire et un standard. Cela permet notamment, de réduire la consommation.

$$I = \frac{U}{R} = \frac{3.3}{10 \cdot 10^3} = 330\mu A$$



2.5.3 Régulateur 5V à 3V3

Lorsque l'on regarde dans le datasheet du MAX1793, on peut voir que le type choisi (MAX1793EUE33+), à une sortie qui est déjà par défaut fixée à 3.3V, il n'y a ainsi pas besoin de l'ajuster.

Ordering Information

PART	TEMP RANGE	PIN-PACKAGE	V _{OUT} [†] (V)
MAX1793EUE-50	-40°C to +85°C	16 TSSOP-EP*	5.0 or Adj
MAX1793EUE-33	-40°C to +85°C	16 TSSOP-EP*	3.3 or Adj
MAX1793EUE-33/V	-40°C to +85°C	16 TSSOP-EP*	3.3 or Adj

Les connexions sont donc les suivantes :

VIN1 - VIN2 - VIN3 - VIN4

Correspondent à la tension d'alimentation +5V

VOUT1 - VOUT2 - VOUT3 - VOUT4
/SHDN

Correspond à la tension fixée en sortie +3.3V
Connecté à la tension d'alimentation pour un fonctionnement normal

DIE

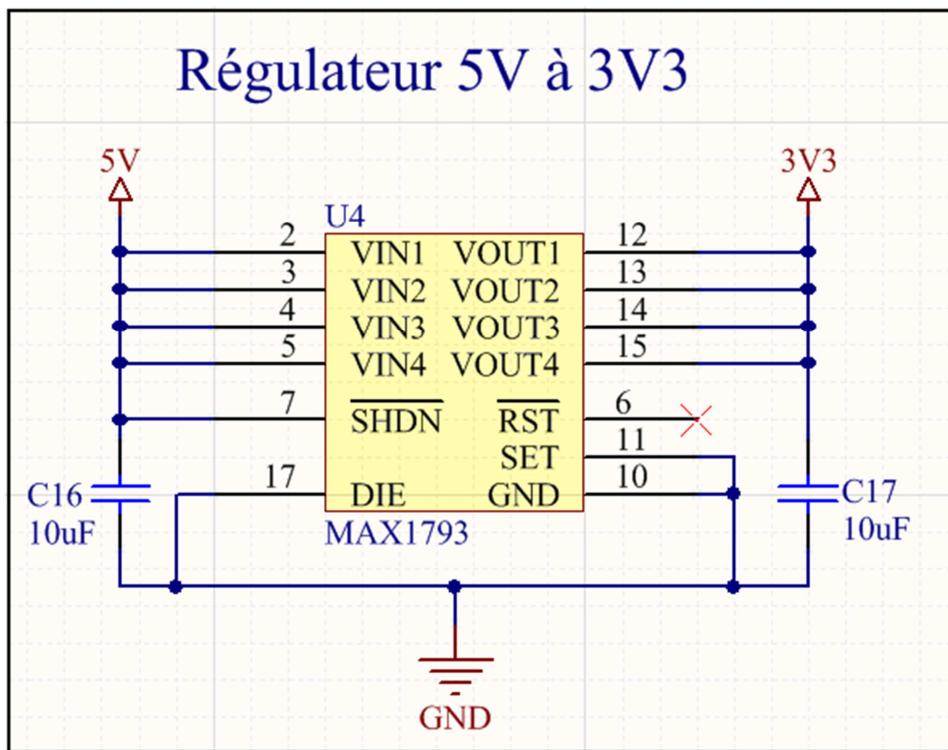
Patte située sous le circuit, pour la dissipation thermique, donc relié à la masse

SET
/RST

Connecté au GND pour avoir la tension de sortie fixe
Permet de réinitialiser la sortie, mais pas utilisé

PIN	NAME	FUNCTION
1, 8, 9, 16	N.C.	No Connection. Not internally connected.
2-5	IN	Regulator Input. Supply voltage ranges from +2.5V to +5.5V. Bypass with a 4.7µF capacitor to GND (see the <i>Capacitor Selection and Regulator Stability</i>). These inputs are internally connected, but they also must be externally connected for proper operation.
6	<u>RST</u>	Reset Output. Open-drain output is low when V _{OUT} is 6% below its nominal value. <u>RST</u> remains low while the output voltage (V _{OUT}) is below the reset threshold and for at least 4ms after V _{OUT} rises above the reset threshold. Connect a 100kΩ pullup resistor to OUT to obtain an output voltage.
7	<u>SHDN</u>	Active-Low Shutdown Input. A logic-low disables the output and reduces the supply current to 0.1µA. In shutdown, the <u>RST</u> output is low and OUT is pulled low through an internal 5kΩ resistance. Connect SHDN to IN for normal operation.
10	GND	Ground. This pin and the exposed pad also function as a heatsink. Solder both to a large pad or to the circuit-board ground plane to maximize power dissipation.
11	SET	Voltage-Setting Input. Connect to GND to select the factory-present output voltage. Connect SET to an external resistor-divider for adjustable-output operation.
12-15	OUT	Regulator Output. Bypass with a 6.8µF, low-ESR capacitor to GND. Connect all OUT pins together at the IC.
—	EP	Exposed Pad. Connect to the ground plane. EP also functions as a heatsink. Solder to the circuit-board ground plane to maximize thermal dissipation.

Le MAX1793 est donc connecté de la manière suivante :



Le régulateur MAX1793 est de type linéaire. Il a l'avantage d'engendrer moins de bruit sur l'alimentation, par rapport à une configuration à découpage.
De plus, il présente une moins grande complexité, donc un coût peu élevé.

2.5.4 Affichage LCD & uC

Lorsque l'on regarde dans le datasheet de l'affichage LCD, concernant le backlight, on peut que la Led doit avoir une chute de tension entre 2.8V et 3.2V et que le courant doit être compris entre 5mA et 30mA.

Backlight Supply Voltage	V_{LED}	-	2.8	3.0	3.2	V
Backlight Supply Current	I_{LED}	$V_{LED} = 3.0V$	5	20	30	mA

J'ai décidé de prendre les valeurs typiques, afin de ne pas être, d'un côté à la limite du fonctionnement, et dans l'autre dans un état critique.

$$V_{cc} = 5V$$

$$I_{LED} = 20mA$$

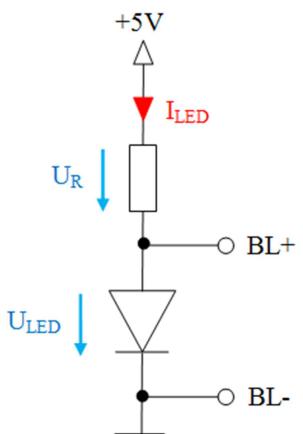
$$U_{LED} = 3.0V$$

$$U_R = V_{cc} - U_{LED} = 5 - 3 = 2V$$

$$R = \frac{U_R}{I_{LED}} = \frac{2}{20 \cdot 10^{-3}} = 100\Omega$$

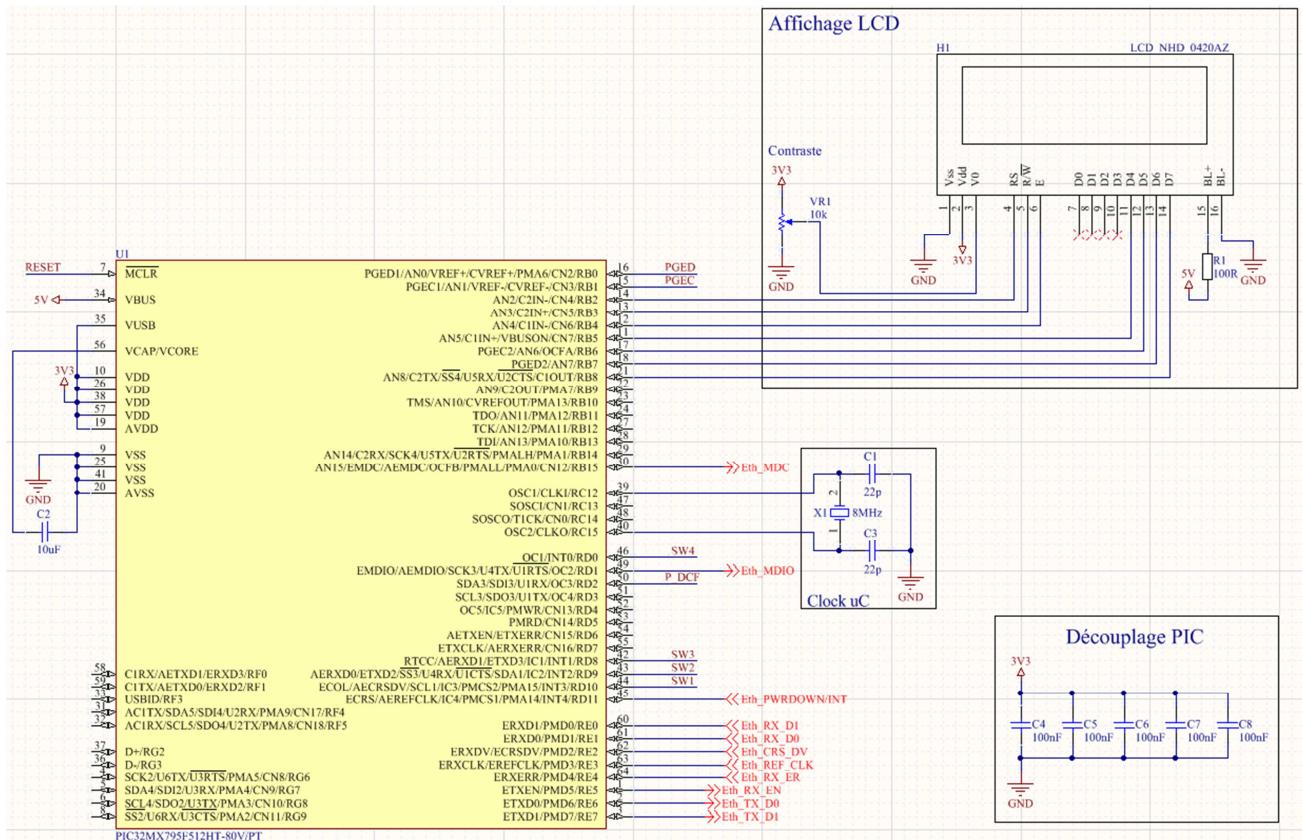
$$P_R = R * I_{LED}^2 = 100 * (20 * 10^{-3})^2 = 40mW$$

Ce qui donne la configuration suivante :



L'affichage LCD travaille avec une interface parallèle, il est donc connecté sur le port B du microcontrôleur.

Le schéma de l'affichage, avec le microcontrôleur, est le suivant :

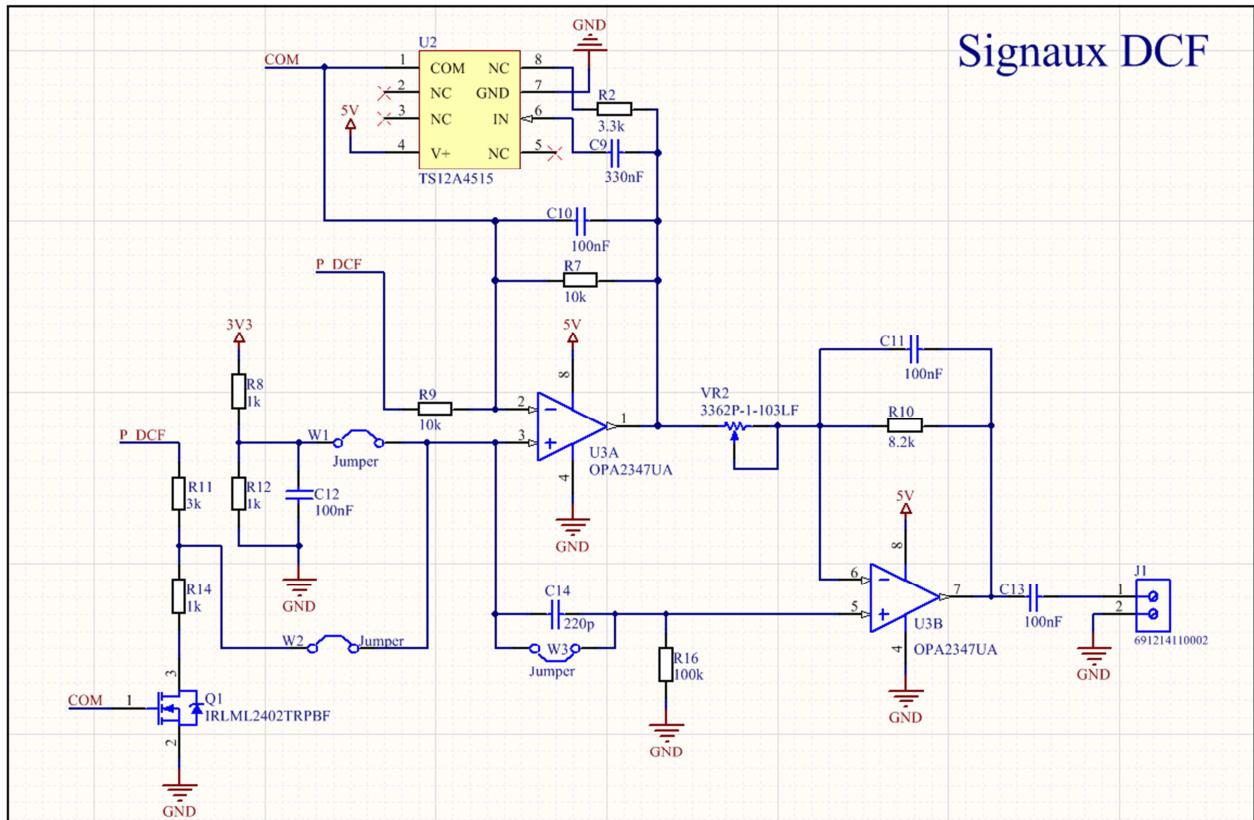


2.5.5 Signaux DCF

Cette partie du schéma va recréer le signal DCF. C'est-à-dire que la porteuse est modulée par des impulsions, au rythme d'une par seconde. Sa fréquence sera de 77.5kHz.

Ces impulsions se traduisent chaque seconde par une diminution de 25% de l'amplitude du signal reçu.

Voici le schéma complet du système :



Sur le schéma, j'ai décidé de mettre deux possibilités quant à la création du signal DCF.

1^{er} possibilité

Le pont diviseur permet d'obtenir la tension d'alimentation (3.3V), divisée par deux, comme les résistances possèdent la même valeur.

Le condensateur en sortie permet de lisser le signal.

$$U = \frac{V_{DD}}{2} = \frac{3.3}{2} = 1.65V$$

$$U = \frac{R_2}{R_8+R_{11}} * V_{DD} = \frac{1*10^3}{1*10^3 + 1*10^3} * 3.3 = 1.65V$$

Cette partie permet, lorsque le signal sera abaissé à 25%, d'avoir un signal centré (Figure 1), car dans le cas contraire, celui-ci restera à 0V (Figure 2).

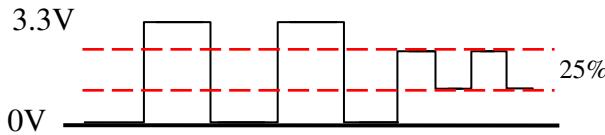


Figure 1.

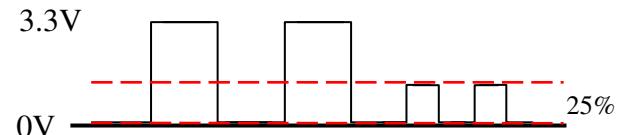


Figure 2.

Le premier amplificateur (UA3) permet d'abaisser l'amplitude de 25% à l'aide d'un analog switch.

Lorsque l'analog switch sera ouvert, seul le premier filtre sera actif (C10 et R7), ainsi le signal sera donc à 100%. La constante de temps est donc la suivante :

$$C_{10} = 100\text{nF} \quad R_7 = 10\text{k}\Omega \quad \text{Tau1} = R_7 * C_{10} = 10 * 10^3 * 100 * 10^{-9} = 1[\text{ms}]$$

Lorsque l'analog switch sera fermé, cette fois-ci, les deux filtres sont actifs, ce qui permet d'abaisser la tension du signal à 25%, soit $\frac{1}{4}$ de celle-ci.

Que l'analog switch soit ouvert ou fermé, la valeur de Tau doit rester la même, c'est pourquoi, il faut multiplier la valeur du condensateur par trois, et diviser celle de la résistance par la même valeur.

$$C_9 = 3 * C_{10} = 3 * 100 * 10^{-9} = 300\text{nF} \quad \text{Série E24 : } 330\text{nF}$$

$$R_2 = R_7 / 3 = 10 * 10^3 / 3 = 3.3\text{k}\Omega \quad \text{Série E24 : } 3.3\text{k}\Omega$$

$$\text{Tau2} = \frac{R}{4} * 4C = R * C \Rightarrow \text{Ce qui permet d'éliminer le facteur 4 et donc d'avoir un Tau de même valeur que précédemment}$$

Le second amplificateur (U3B) permet d'amplifier le signal, et aussi de pouvoir faire varier la puissance d'émission en sortie, grâce à un potentiomètre (VR2).

Le condensateur en sortie de l'amplificateur permet d'éliminer la composante continue. En sortie, l'antenne possède déjà un condensateur de filtrage, ce qui forme un circuit bouchon. On peut donc estimer que l'impédance du circuit est d'environ $200-300\Omega$.

Si le condensateur en sortie a une valeur de 100nF , la réactance capacitive X_C vaudra :

$$X_C = \frac{1}{2\pi f C} = \frac{1}{2\pi * 77.5 * 10^3 * 100 * 10^{-9}} = 20.53\Omega$$

On voit donc que l'impédance du condensateur est très faible comparé à celle du circuit bouchon, la valeur du condensateur est donc correcte et acceptable, et la réactance de celui-ci est très minime.

Concernant les gains des amplificateurs, et comme il y a deux étages, ceux-ci vont donc être multiplié, et doivent ainsi être inférieurs à 1, afin de limiter la puissance d'émission, qui est de l'ordre du [mW].

Comme la configuration est une rétroaction négative, l'amplificateur travaille en mode opérationnel.

$$\text{Ampli 1 : } A_{V1} = -\frac{R7}{R9} = -\frac{10 \times 10^3}{10 \times 10^3} = -1$$

$$\text{Ampli 2 : } A_{V2} = -\frac{R10}{VR1} = \frac{8.2 \times 10^3}{10 \times 10^3} = -0.82$$

$$\text{Gain}_{\text{tot}} = A_{V1} * A_{V2} = -1 * -0.82 = 0.82$$

2^{ème} possibilité

La seconde partie consiste à placer un pont diviseur (R11 et R14) de façon à avoir l'amplitude abaissée à 25%.

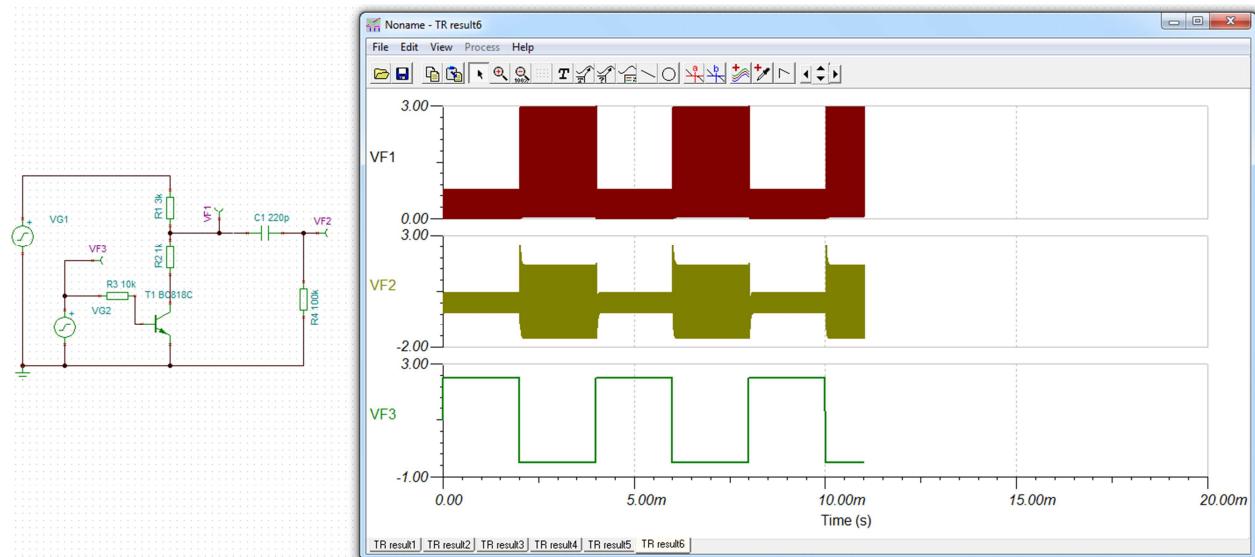
$$R_{\text{tot}} = 4k \quad R14 \Rightarrow \frac{1}{4} \text{ de la résistance totale}$$

$$\frac{1}{4} \text{ de } 3.3V = 825mV$$

On place un transistor MOSFET (Q1), en bas du pont diviseur, et à l'entrée, le signal de la porteuse à 77.5kHz.

Ensuite, pour centrer le signal, on place un filtre passe haut (C14 et R14).

Puis le signal sera amplifié, comme pour la première possibilité, via l'amplificateur (U3B).



La simulation, ci-dessus, montre la situation avec la 2^{ème} possibilité, où l'on voit bien que sur VF1, le signal est abaissé à 25%, mais il n'est pas centré.

L'ajout du filtre passe haut permet ainsi, de conserver l'amplitude abaissée, tout en centrant le signal.

2.5.6 RESET du uC & Connecteur de programmation

Lorsque l'on regarde dans le datasheet du PIC32MX795F512H, sous la catégorie Master clear (/MCLR) Pin, on trouve le schéma avec la configuration suivante :

La broche /MCLR fournit la fonction qui permet de réinitialiser l'appareil. Or, il est important de respecter certaines contraintes lors du dimensionnement du RESET, qui est donné dans la même rubrique.

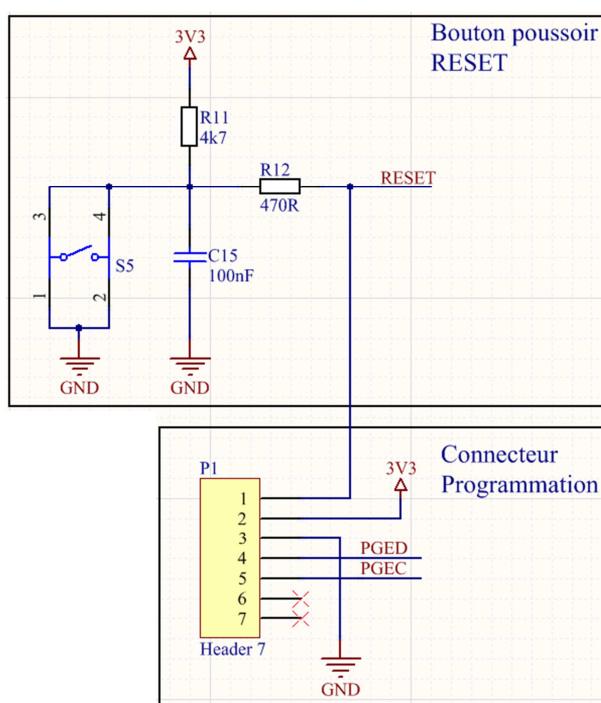
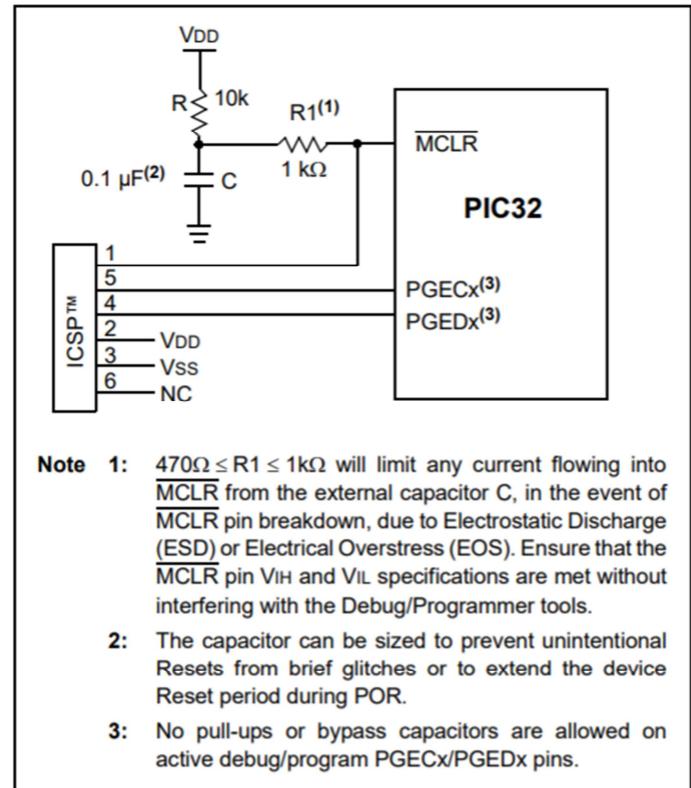
La résistance R1 doit être comprise entre 470Ω et $1k\Omega$, ce qui permet de limiter le courant entrant dans la pin /MCLR du condensateur externe C, en cas de rupture de la broche, due à une décharge électrostatique ou d'une surcharge électrique.

On voit que la résistance R, a une valeur dix fois supérieure à celle de R1. Et que la valeur du condensateur est de $0.1\mu F$.

J'ai donc décidé de reprendre la configuration du Kit PIC32, où R11 vaut $4.7k$, R12 $470R$ et C15 $100nF$. Ce qui répond aux critères donnés.

Concernant le connecteur de programmation, on voit qu'il faut simplement connecter les pattes de programmation sur celles PGECx et PGEDx du microcontrôleur.

Cela donne la connectivité suivante :



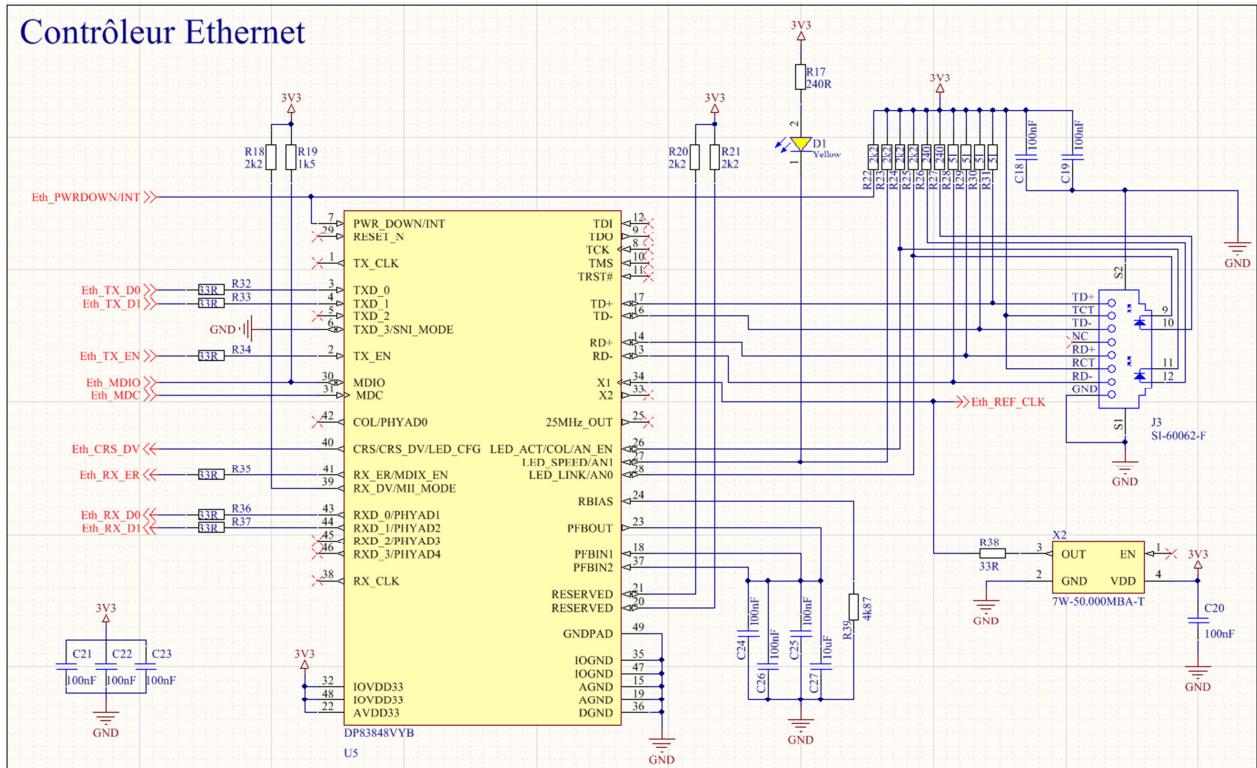
2.5.7 Contrôleur Ethernet

La partie Ethernet a été reprise d'un précédent projet (1630x_TimbreuseBadgeRFID). Aucune modification n'a été apportée au schéma, comme celui-ci est fonctionnel.

Néanmoins, l'élément principal est le composant DP83848VYB, qui permet de faire le pont entre le microcontrôleur et le RJ45.

Ce dernier communique via une interface RMII.

Le contrôleur Ethernet est donc le suivant :



2.6 Concept du logiciel

2.6.1 Partie DCF

La partie principale consiste à la reconstitution du signal DCF.

Un Timer du microcontrôleur va permettre de recréer le signal de la porteuse, d'une fréquence de 77.5kHz, et d'une amplitude de 0V à 3.3V.

L'onde porteuse est modulée par des impulsions, au rythme d'une par seconde, les informations sont donc transmises sous forme binaire, à raison d'un bit par seconde, contenant ainsi les données de l'heure.

Ces impulsions se traduisent chaque seconde par une diminution de 25% de l'amplitude du signal, qui est réalisé à l'aide d'un analog switch et à une disposition de condensateurs et de résistances, de par le choix des valeurs.

L'impulsion qui est émise au début de chaque seconde dure 100ms pour un '0' logique et 200ms pour un '1' logique. Seule la 59^{ème} seconde n'est pas modulée et permet d'annoncer le début d'une nouvelle trame.

2.6.2 Mise à jour de l'heure

L'heure sera mise à jour via le protocole SNTP (Simple Network Time Protocol).

Ce protocole permet de synchroniser, via un réseau informatique, l'horloge locale d'ordinateurs sur une référence d'heure.

2.6.3 Menu

Les quatre switch vont permettre de régler un menu, qui va gérer les ajustements suivants :

- Réglage du fuseau horaire
- Réglage de l'heure été/hiver
- Possibilité de désactiver le changement d'heure
- Emission du signal (5min par jour à midi)

2.7 Conclusion

La partie Design m'a permis d'établir le schéma électrique complet, et ainsi de fixer les technologies qui seront utilisées, ainsi que les différents types de communications.

Lors du choix des technologies, de nombreux composants ont été choisis par leurs caractéristiques électriques, mais certains aussi par choix de groupe, afin de limiter les coûts de certains éléments, comme l'affichage.

Lors de cette étape, je n'ai pas eu de problèmes majeurs, mise à part la réalisation de la partie DCF, qui posait problème, de par le centrage des signaux, lorsque l'on a l'abaissement de l'amplitude. Or ce problème a pu être résolu avec plusieurs solutions, et c'est notamment pour cela, que j'ai décidé de mettre en place deux systèmes différents sur ma partie DCF.

La deuxième solution nécessite l'ajout de peu de composants, donc un coût très bas.

La suite est donc positive, puisque le PCB va pouvoir être réalisé, ainsi que la fin du projet.

Lausanne, le 6 février 2019

Signature : Julie Culand

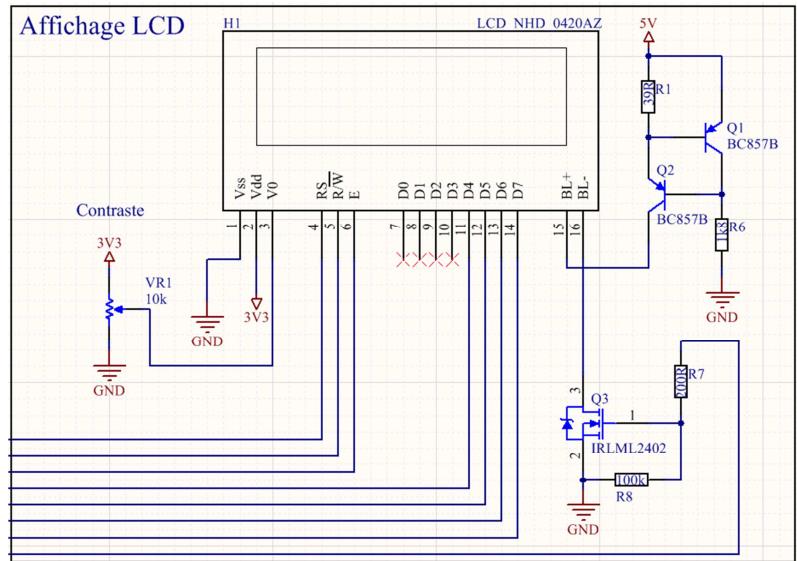
3 Réalisation

3.1 Modifications apportées au schéma électrique

Suite à la partie précédente réalisée, donc du Design, les éléments suivants ont été relevés pour être modifié. (Voir Annexe II : Schéma électrique)

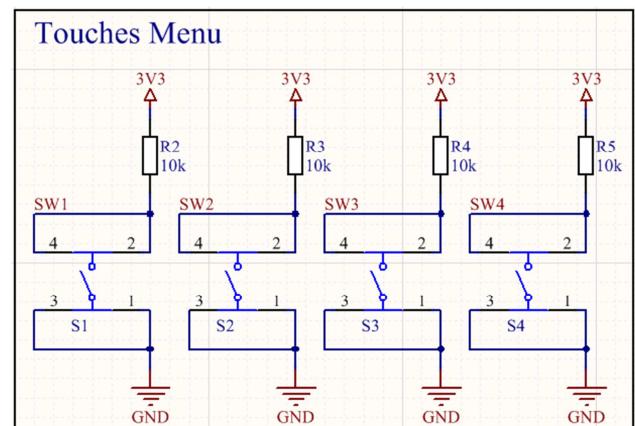
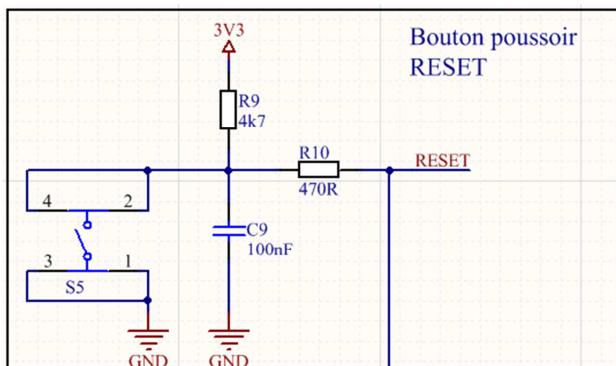
3.1.1 Affichage

Ajout d'une source de courant pour le backlight, et commutation on-off



3.1.2 Touches menu et RESET

Rotation de 90° des switchs

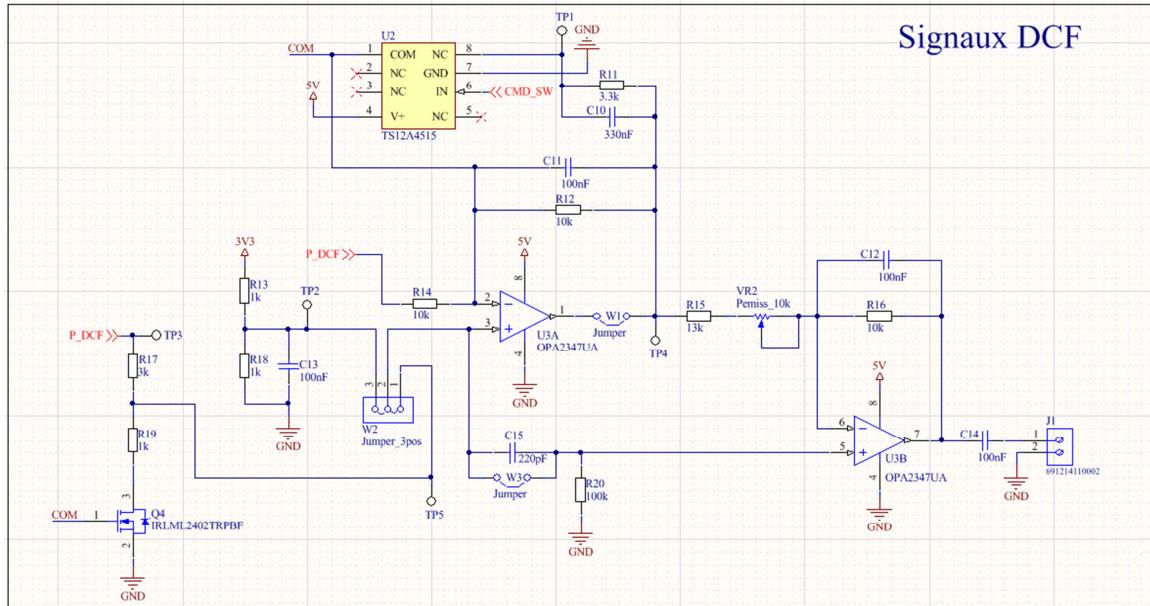


3.1.3 Signaux DCF

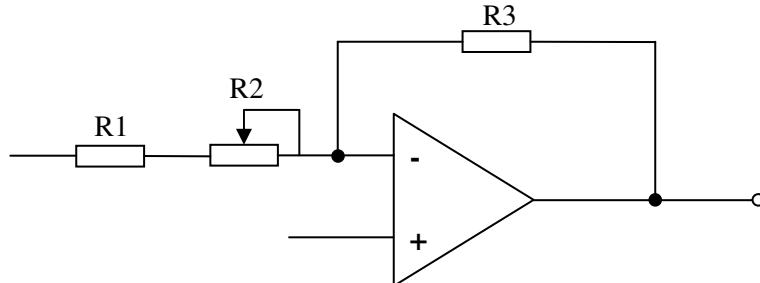
Remplacer les deux jumpers à une position, par un seul à deux positions.

Ajouter un jumper en sortie de l'amplificateur U3A.

Adapter le gain de l'amplificateur U3B, en fonction du potentiomètre ($A_{V_{min}} = 0.8$).



Ajustement du gain



$$R2 = 10k\Omega$$

$$\text{Gain min : } 0.8$$

$$R3 = 10k\Omega$$

Gain maximum si R2 est au minimum (0Ω) :

$$Av = - \frac{R3}{R1} = 0.8 \quad \Rightarrow \quad R1 = \frac{R3}{Av} = \frac{10 \cdot 10^3}{0.8} = 12.5k\Omega \quad \text{Série E24 : } 13k\Omega$$

Gain minimum si R2 est au maximum (10kΩ) :

$$Av = - \frac{R3}{R1+R2} = - \frac{10 \cdot 10^3}{13 \cdot 10^3 + 10 \cdot 10^3} = 0.43$$

Vérification du gain après normalisation

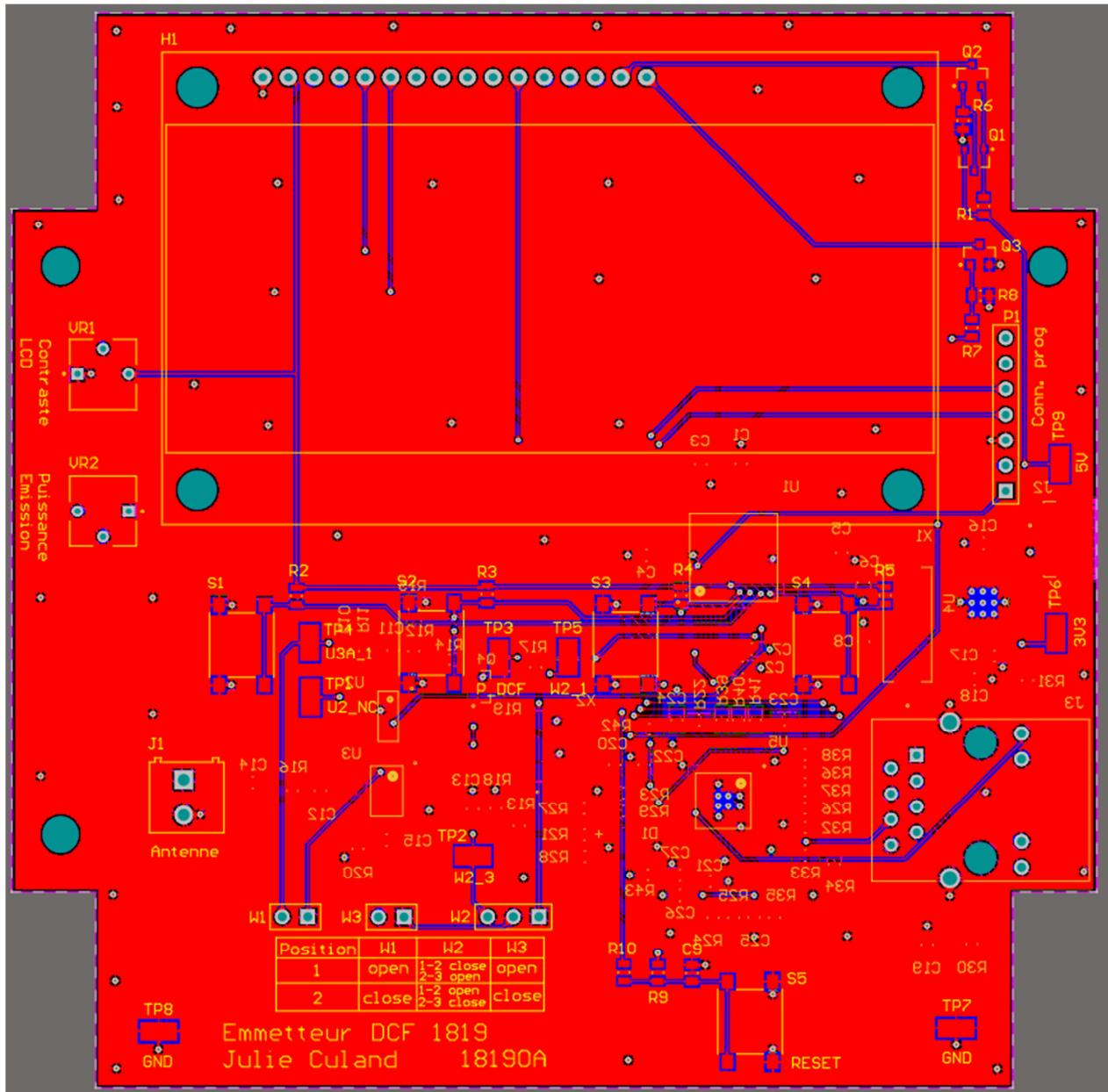
$$Av_{max} = - \frac{R3}{R1} = - \frac{10 \cdot 10^3}{13 \cdot 10^3} = -0.77$$

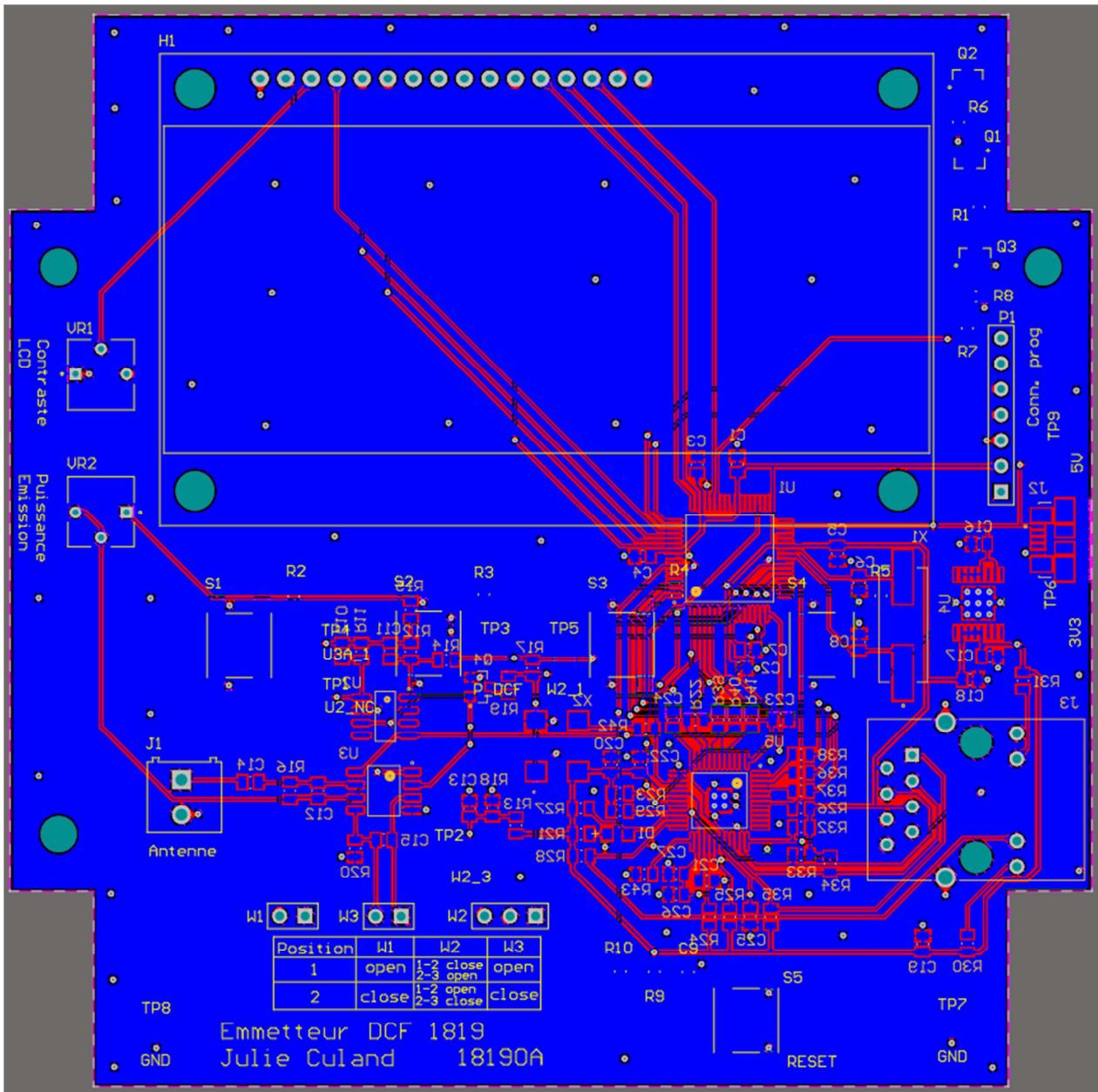
3.2 PCB

3.2.1 Routage

Pour le routage, les tailles des pistes et via ont été paramétré en fonction des normes euro-circuit donnée. (Voir Annexe VIII : Normes euro-circuit)

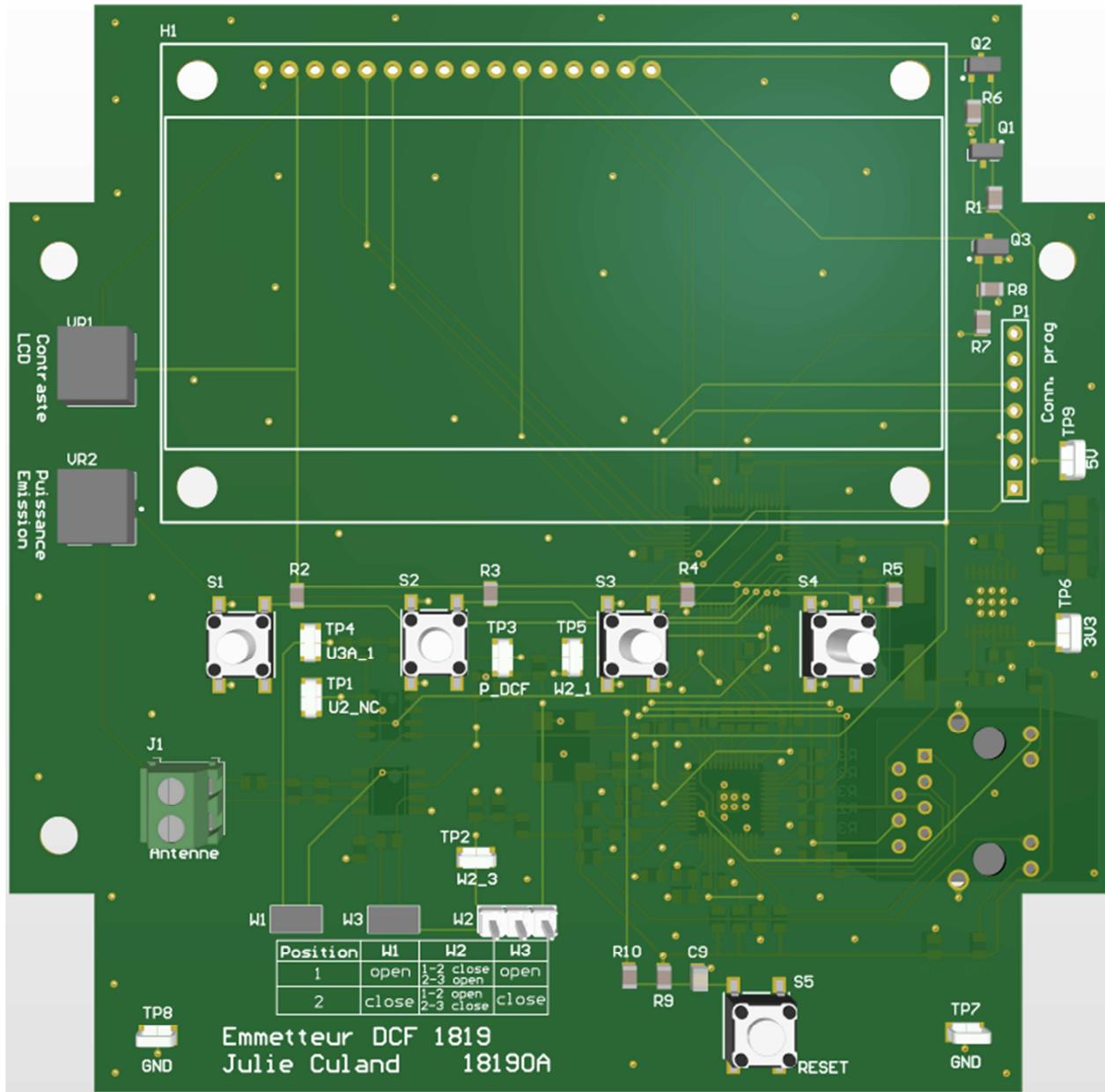
Des plans de masse ont été placés sur le TOP, ainsi que sur la face BOTTOM.





3.2.2 Placement des composants

Concernant le placement des composants, sur le layer TOP, on était placé tous les éléments dont il faut avoir accès, qui permettent de réaliser l'interface homme-machine, soit : L'affichage LCD, switchs, potentiomètres, jumpers, Reset, points de test, connecteur de programmation, bornier pour l'antenne.

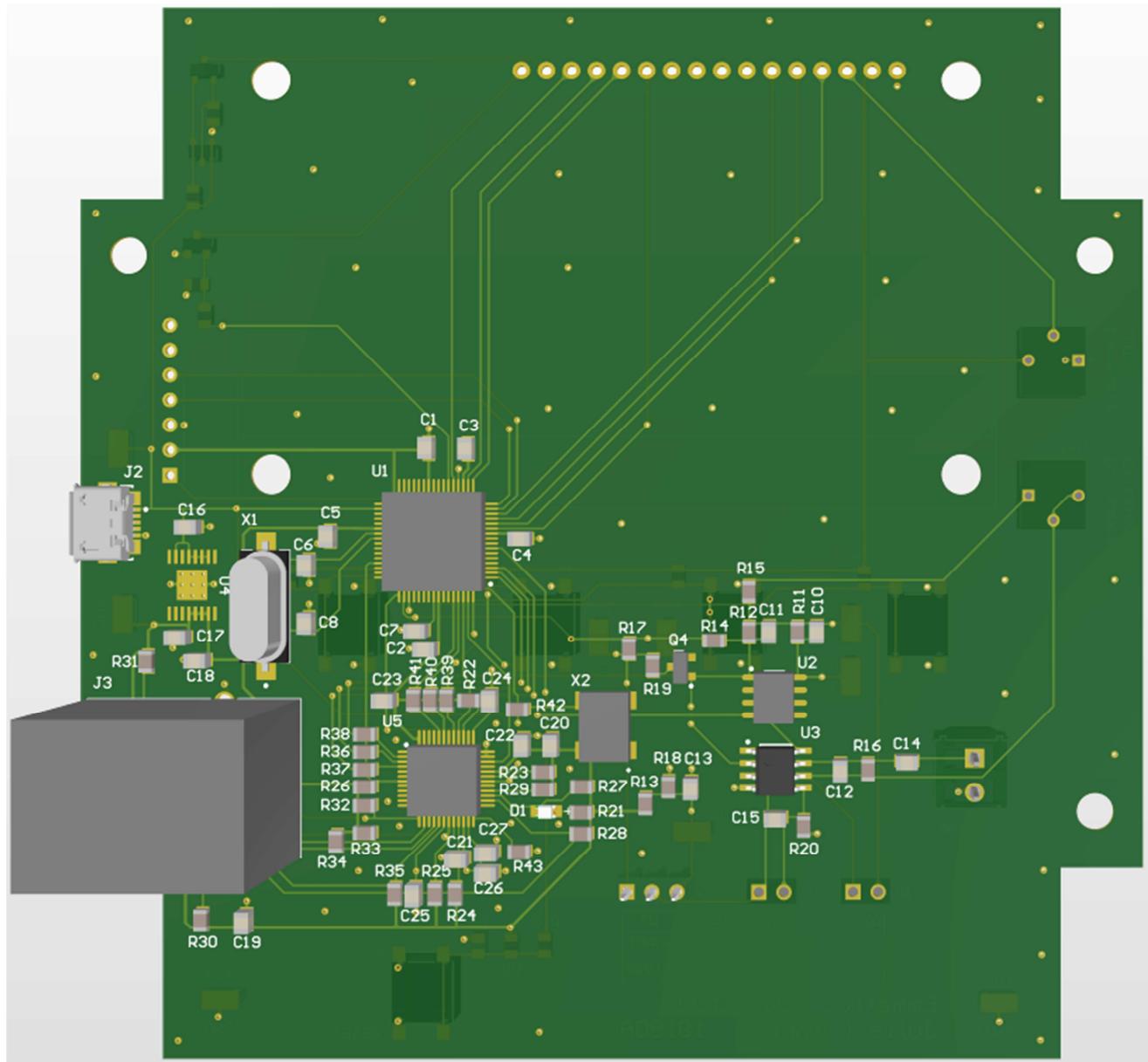


Pour faciliter l'utilisation des divers éléments, j'ai décidé de mettre une description à ceux-ci. J'ai également fait pareil pour le positionnement des jumpers, selon le circuit DCF que l'on souhaite utiliser.

Sur le layer BOTTOM, tous les autres composants on était placé, et notamment les deux connecteurs (micro-USB et Ethernet).

Comme le PCB sera surélevé dans le boîtier, et que le connecteur Ethernet a une hauteur plus ou moins importante, cela permettra de ne pas gêner l'affichage sur le TOP.

C'est donc pour cela, que ceux-ci ont été placés au bord du PCB (J2 et J3).

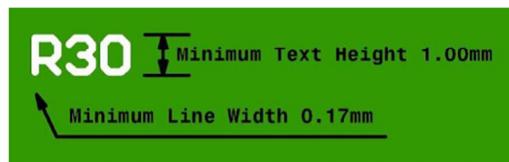


Concernant les fixations pour le PCB, l'une d'entre elles n'a pas été faite, car la place est prise par le connecteur RJ-45. Malgré tout, cela n'est pas gênant car il y a les trois autres trous de fixations qui sont utilisables.

3.2.3 Sérigraphie

Pour la sérigraphie, les normes Euro-circuits ont également été reprises (dimensions texte).

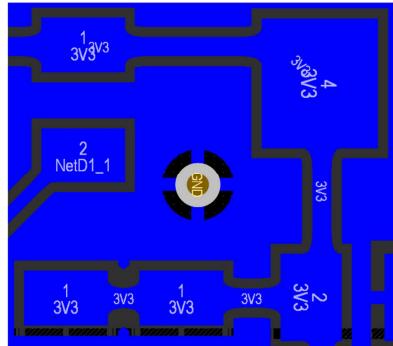
- ✓ Minimum Legend Line Width: 0.17mm (7mil)
- ✓ Minimum Text height for good readability: 1.00mm (39.5mil).



3.2.4 Eléments particuliers du PCB

Via sur plan de masse :

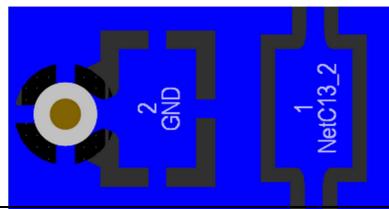
Le plan de masse peut à certain endroit être relativement fin, on place donc des vias un peu partout, afin d'assurer une bonne connexion entre le plan de masse du TOP et du BOTTOM, ceci pour éviter tout bruit ou parasites.



PAD du plan de masse :

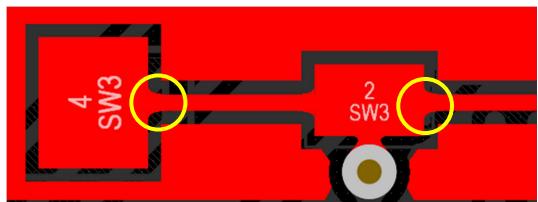
Afin d'assurer une bonne connexion entre le plan des layers, tous les PADs concernés (GND), sont reliés par un via.

La liaison entre le PAD et le via doit être le plus court possible (idem pour les alimentations).



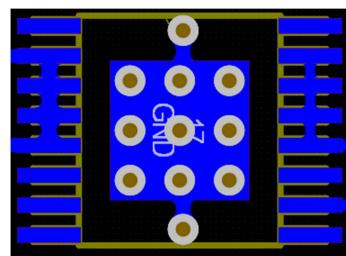
Teardrops :

Consiste en l'ajout de matière, afin que les connexions soient plus solides entre les pistes. L'objectif principal est d'améliorer l'intégrité de la structure en présence de contraintes thermiques ou mécanique, mais surtout d'éviter des défauts de gravure, notamment aux endroits où il y a des angles droits.



Dissipation de chaleur :

Des zones avec plusieurs vias on était placé notamment sous le régulateur 3V3, ainsi que le CI Ethernet, afin de mieux dissiper la chaleur.



3.3 Mécanique

3.3.1 Choix du boîtier

Pour le choix du boîtier, j'ai pris en compte les contraintes suivantes :

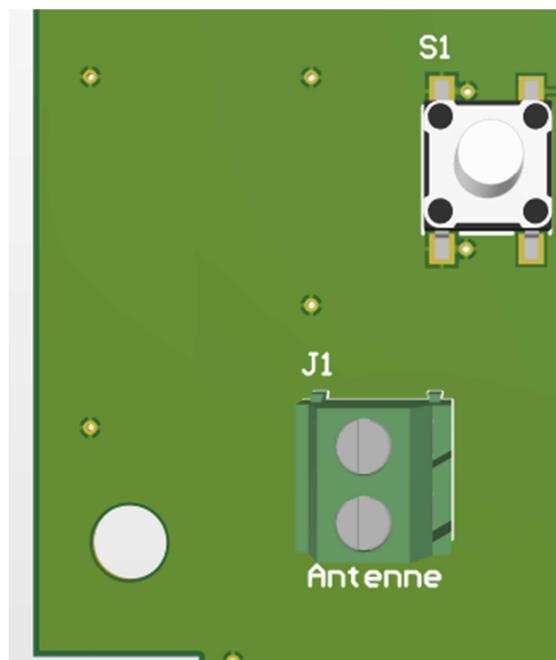
- Hauteur du PCB avec l'affichage et le connecteur Ethernet
- Facile à transporter et déplacer

J'ai ainsi choisi un boîtier de chez Hammond, avec un couvercle transparent afin de ne pas avoir besoin d'usiner une fenêtre pour l'affichage LCD.



3.3.2 Emplacement de l'antenne

Dans la partie de l'étude du projet, j'avais comme première idée de placer l'antenne à l'extérieur du boîtier. Or comme celle-ci est relativement fragile et ne présente pas une importante surface, j'ai décidé de la positionner directement sur le PCB, comme la place me le permet.



3.4 Montage

Pour le montage, j'ai décidé de réaliser un plan, afin d'indiquer l'ordre dans lequel les composants vont être montés. Ceci afin de commencer par les composants les plus bas au plus haut (hauteur).

En revanche, les deux premiers composants qui seront montés sont le PIC32 ainsi que l'IC Ethernet, car ce sont ceux les plus difficile et risquer à braser.

PLAN DE MONTAGE						
Quantity	Comment	Description	Boîtier	Designator	SMD / THT	Ordre montage
19	100nF	Capacitor	0805	C1, C2, C3, C4, C5, C9, C11, C12, C13, C14, C18, C19, C20, C21, C22, C23, C24, C25, C26		3
2	22pF	Capacitor	0805	C6, C8		3
4	10uF	Capacitor	0805	C7, C16, C17, C27		3
1	330nF	Capacitor	0805	C10		3
1	220pF	Capacitor	0805	C15		3
1	Yellow	SMT Chip LED	0805	D1		5
1	LCD_NHD_0420AZ	Affichage LCD	H1		23
1	691214110002S	Bornier 2 pôles	J1		21
1	47346-0001	Micro-USB B	J2		13
1	SI-60062-F	RJ45 Connector	J3		22
1	Header 7	Header, 7-Pin	P1		20
2	BC857B	PNP Transistor	SOT23	Q1, Q2		6
2	IRLML2402TRPBFB	MOSFET, N-Channel	SOT23-3	Q3, Q4		7
1	39R	Resistor	0805	R1		4
7	10k	Resistor	0805	R2, R3, R4, R5, R12, R14, R16		4
1	1k8	Resistor	0805	R6		4
1	200R	Resistor	0805	R7		4
2	100k	Resistor	0805	R8, R20		4
1	4k7	Resistor	0805	R9		4
1	470R	Resistor	0805	R10		4
1	3.3k	Resistor	0805	R11		4
3	1k	Resistor	0805	R13, R18, R19		4
1	13k	Resistor	0805	R15		4
1	3k	Resistor	0805	R17		4
1	240R	Resistor	0805	R21		4
7	2k2	Resistor	0805	R22, R24, R25, R26, R27, R28, R29		4
1	1k5	Resistor	0805	R23		4
2	240	Resistor	0805	R30, R31		4
4	51	Resistor	0805	R32, R33, R34, R35		4
7	33R	Resistor	0805	R36, R37, R38, R39, R40, R41, R42		4
1	4k87	Resistor	0805	R43		4
4	430152095836	SMD Tact Switch	S1, S2, S3, S4		14
1	430152043836	SMD Tact Switch	S5		15
9	5019	Test Point SMD	TP1, TP2, TP3, TP4, TP5, TP6, TP7, TP8, TP9		16
1	PIC32MX795F512HT	32-Bit Microcontroller	U1			1
1	TS12A4515D	CMOS Analog Switch	U2			8
1	OPA2347UA	Operational Amplifier	SOIC8	U3		9
1	MAX1793	Régulateur 3V3	TSSOP-16	U4		10
1	DP83848VYB	CI Ethernet	U5		2
2	3362P-1-103LF	Potentiometer, 10 K	VR1, VR2		17
2	Jumper_2pos	Jumper Wire	W1, W3		18
1	Jumper_3pos	Jumper Wire	W2		19
1	ECS-80-18-5PX-TR	Quartz 8.0 MHz	HC-49/US	X1		12
1	7w-50.000MBA-T	OSC, 50.000 MHZ	4-SMD	X2		11

3.5 Test du PCB - Alimentations

Le but de cette étape est de tester que le circuit fonctionne sans le software, c'est-à-dire que les alimentations de la carte soient toutes fonctionnelles, ainsi que sur les différents ICs.

	Valeur attendue	Valeur mesurée	Erreur absolue
Alim 5V - TP9	5V	5.08V	1.6%
Alim 3.3V - TP6	3.3V	3.292V	-0.24%
uC - VDD (patte 10)	3.3V	3.290V	-0.30%
uC - VBUS (patte 34)	5V	5.08V	1.6%
U2 - V+ (patte 4)	5V	5.08V	1.6%
U5 - IOVDD33 (patte 22)	3.3V	3.284V	0.48%
LCD - Vdd (patte 2)	3.3V	3.290V	-0.30%

On peut voir que les alimentations du circuit sont toutes fonctionnelles.

3.6 Réalisation du software

3.6.1 Détails du protocole

La fréquence porteuse est de 77.5kHz. Le principal défaut de cette fréquence est sa sensibilité aux parasites. Il est donc important de vérifier la cohérence des données reçues, par le biais de bits de parité.

Les informations sont transmises sous forme binaire à raison d'un bit à chaque seconde. Celles-ci sont codées en BCD (décimal codé en binaire), leur décodage fournit au récepteur les éléments comme la date et l'heure.

Les impulsions se traduisent chaque seconde par une diminution de 25% de l'amplitude du signal reçu. La durée d'impulsion détermine le niveau du bit reçu, à savoir qu'une impulsion de 100ms représente un '0' et qu'une impulsion de 200ms, représente un '1'.

Les informations horaires sont émises par trame d'une minute. Chaque trame est divisée en soixante secondes, chacun d'entre elles débutant par le front de l'impulsion.

Il faut noter que, pour la 59^e seconde, il n'y a pas d'impulsion afin de permettre au décodeur de repérer le début d'une trame.

Ainsi, l'impulsion suivante détermine le début de la trame suivante. Toute absence d'impulsion plus grande que 999ms doit donc être considérée comme le début d'une nouvelle trame.

La synchronisation des récepteurs se fait sur le premier bit (bit n°0). L'apparition de la première modulation marque alors le début d'une nouvelle minute.

L'heure sera reçue via Ethernet et sera mise à jour via le protocole SNTP (Simple Network Time Protocol).

Ce protocole permet de synchroniser, via un réseau information, l'horloge locale d'ordinateurs sur une référence d'heure.

En revanche, ce protocole ne s'occupe pas du changement de l'heure dû au fuseau horaire, ainsi que du passage à l'heure d'été et d'hiver.

3.6.2 Programmation sous MPLAB

3.6.2.1 Paramétrage de MPLAB Harmony

3.6.2.1.1 Pin Settings

Avant de commencer à programmer, il faut paramétrer les pins settings, soit toutes les pins qui seront utilisées à la programmation.

Voici donc la configuration réalisée sous "Harmony Configurator > Pin settings" :

Pin Number	Pin ID	Voltage Tolerance	Name	Function	Direction (TRIS)	Latch (LAT)	Open Drain (ODC)	Mode (ADPCFG)	Change Notification (CNEN)	Pull Up (CNPUE)
1	RE5	5V	ETH_TX_EN	Available	In	n/a	<input type="checkbox"/>	Digital		<input type="checkbox"/>
2	RE6	5V	ETH_TX_D0	Available	In	n/a	<input type="checkbox"/>	Digital		<input type="checkbox"/>
3	RE7	5V	ETH_TX_D1	Available	In	n/a	<input type="checkbox"/>	Digital		<input type="checkbox"/>
4	RG6	5V		Available	In	n/a	<input type="checkbox"/>	Digital	<input type="checkbox"/>	<input type="checkbox"/>
5	RG7	5V		Available	In	n/a	<input type="checkbox"/>	Digital	<input type="checkbox"/>	<input type="checkbox"/>
6	RG8	5V		Available	In	n/a	<input type="checkbox"/>	Digital	<input type="checkbox"/>	<input type="checkbox"/>
7	MCLR	5V			In	n/a	<input type="checkbox"/>	Digital		<input type="checkbox"/>
8	RG9	5V		Available	In	n/a	<input type="checkbox"/>	Digital	<input type="checkbox"/>	<input type="checkbox"/>
9	VSS				In	n/a	<input type="checkbox"/>	Digital		<input type="checkbox"/>
10	VDD				In	n/a	<input type="checkbox"/>	Digital		<input type="checkbox"/>
11	RB5		LCD_DB4	Available	Out	Low	<input type="checkbox"/>	Digital	<input type="checkbox"/>	<input type="checkbox"/>
12	RB4		LCD_E	Available	Out	Low	<input type="checkbox"/>	Digital	<input type="checkbox"/>	<input type="checkbox"/>
13	RB3		LCD_R/W	Available	Out	Low	<input type="checkbox"/>	Digital	<input type="checkbox"/>	<input type="checkbox"/>
14	RB2		LCD_RS	Available	Out	Low	<input type="checkbox"/>	Digital	<input type="checkbox"/>	<input type="checkbox"/>
15	RB1		PGENC	Available	In	n/a	<input type="checkbox"/>	Analog	<input type="checkbox"/>	<input type="checkbox"/>
16	RB0		PGEND	Available	In	n/a	<input type="checkbox"/>	Analog	<input type="checkbox"/>	<input type="checkbox"/>
17	RB6		LCD_DB5	Available	Out	Low	<input type="checkbox"/>	Digital		<input type="checkbox"/>
18	RB7		LCD_DB6	Available	Out	Low	<input type="checkbox"/>	Digital		<input type="checkbox"/>
19	AVDD				In	n/a	<input type="checkbox"/>	Digital		<input type="checkbox"/>
20	AVSS				In	n/a	<input type="checkbox"/>	Digital		<input type="checkbox"/>
21	RB8		LCD_DB7	Available	Out	Low	<input type="checkbox"/>	Digital		<input type="checkbox"/>
22	RB9		LCD_BL	Available	Out	Low	<input type="checkbox"/>	Digital		<input type="checkbox"/>
23	RB10			Available	In	n/a	<input type="checkbox"/>	Analog		<input type="checkbox"/>
24	RB11			Available	In	n/a	<input type="checkbox"/>	Analog		<input type="checkbox"/>
25	VSS				In	n/a	<input type="checkbox"/>	Digital		<input type="checkbox"/>
26	VDD				In	n/a	<input type="checkbox"/>	Digital		<input type="checkbox"/>
27	RB12			Available	In	n/a	<input type="checkbox"/>	Analog		<input type="checkbox"/>
28	RB13			Available	In	n/a	<input type="checkbox"/>	Analog		<input type="checkbox"/>
29	RB14			Available	In	n/a	<input type="checkbox"/>	Analog		<input type="checkbox"/>
30	RB15		ETH_MDC	Available	In	n/a	<input type="checkbox"/>	Digital	<input type="checkbox"/>	<input type="checkbox"/>
31	RF4	5V		Available	In	n/a	<input type="checkbox"/>	Digital	<input type="checkbox"/>	<input type="checkbox"/>
32	RF5	5V		Available	In	n/a	<input type="checkbox"/>	Digital	<input type="checkbox"/>	<input type="checkbox"/>
33	RF3	5V		Available	In	n/a	<input type="checkbox"/>	Digital		<input type="checkbox"/>
34	VBUS	5V			In	n/a	<input type="checkbox"/>	Digital		<input type="checkbox"/>
35	VUSB3V3				In	n/a	<input type="checkbox"/>	Digital		<input type="checkbox"/>
36	RG3			Available	In	n/a	<input type="checkbox"/>	Analog		<input type="checkbox"/>
37	RG2			Available	In	n/a	<input type="checkbox"/>	Analog		<input type="checkbox"/>
38	VDD				In	n/a	<input type="checkbox"/>	Digital		<input type="checkbox"/>
39	RC12			Available	In	n/a	<input type="checkbox"/>	Analog		<input type="checkbox"/>
40	RC15			Available	In	n/a	<input type="checkbox"/>	Analog		<input type="checkbox"/>
41	VSS				In	n/a	<input type="checkbox"/>	Digital		<input type="checkbox"/>
42	RD8	5V	SW_3	Available	In	n/a	<input type="checkbox"/>	Digital		<input type="checkbox"/>
43	RD9	5V	SW_2	Available	In	n/a	<input type="checkbox"/>	Digital		<input type="checkbox"/>
44	RD10	5V	SW_1	Available	In	n/a	<input type="checkbox"/>	Digital		<input type="checkbox"/>
45	RD11	5V	ETH_PWRDOWN_INT	Available	In	n/a	<input type="checkbox"/>	Digital		<input type="checkbox"/>
46	RD0	5V	SW_4	Available	In	n/a	<input type="checkbox"/>	Digital		<input type="checkbox"/>
47	RC13			Available	In	n/a	<input type="checkbox"/>	Analog	<input type="checkbox"/>	<input type="checkbox"/>
48	RC14			Available	In	n/a	<input type="checkbox"/>	Analog	<input type="checkbox"/>	<input type="checkbox"/>
49	RD1	5V	ETH_MDIO	Available	In	n/a	<input type="checkbox"/>	Digital		<input type="checkbox"/>
50	RD2	5V	P_DCF	Available	Out	Low	<input type="checkbox"/>	Digital		<input type="checkbox"/>
51	RD3	5V	CMD_SW	Available	Out	Low	<input type="checkbox"/>	Digital		<input type="checkbox"/>
52	RD4	5V		Available	In	n/a	<input type="checkbox"/>	Digital	<input type="checkbox"/>	<input type="checkbox"/>
53	RD5	5V		Available	In	n/a	<input type="checkbox"/>	Digital	<input type="checkbox"/>	<input type="checkbox"/>
54	RD6	5V		Available	In	n/a	<input type="checkbox"/>	Digital	<input type="checkbox"/>	<input type="checkbox"/>
55	RD7	5V		Available	In	n/a	<input type="checkbox"/>	Digital	<input type="checkbox"/>	<input type="checkbox"/>
56	VCAP				In	n/a	<input type="checkbox"/>	Digital		<input type="checkbox"/>
57	VDD				In	n/a	<input type="checkbox"/>	Digital		<input type="checkbox"/>
58	RF0	5V		Available	In	n/a	<input type="checkbox"/>	Digital		<input type="checkbox"/>
59	RF1	5V		Available	In	n/a	<input type="checkbox"/>	Digital		<input type="checkbox"/>
60	RE0	5V	ETH_RX_D1	Available	In	n/a	<input type="checkbox"/>	Digital		<input type="checkbox"/>
61	RE1	5V	ETH_RX_D0	Available	In	n/a	<input type="checkbox"/>	Digital		<input type="checkbox"/>
62	RE2	5V	ETH_CRS_DV	Available	In	n/a	<input type="checkbox"/>	Digital		<input type="checkbox"/>
63	RE3	5V	ETH_REF_CLK	Available	In	n/a	<input type="checkbox"/>	Digital		<input type="checkbox"/>
64	RE4	5V	ETH_RX_ER	Available	In	n/a	<input type="checkbox"/>	Digital		<input type="checkbox"/>

3.6.2.1.2 Timer 1

Le Timer 1 est paramétré pour avoir une période de 1ms avec un niveau de priorité à 3. Celui-ci va permettre de gérer les différents menus.

$$f_{osc} = 80\text{MHz} \quad T_{\text{TIMER}} = 1\text{ms}$$

$$T_{osc} = \frac{1}{f_{osc}} = \frac{1}{80*10^6} = 12.5\text{ns}$$

$$\text{NbCoup} = \frac{T_{\text{TIMER}}}{T_{osc}} = \frac{1*10^{-3}}{12.5*10^{-9}} = 80'000$$

Comme le Timer a 2^{16} pas, soit 0 à 65'535, la valeur calculée est trop élevée vu qu'elle dépasse la valeur max.

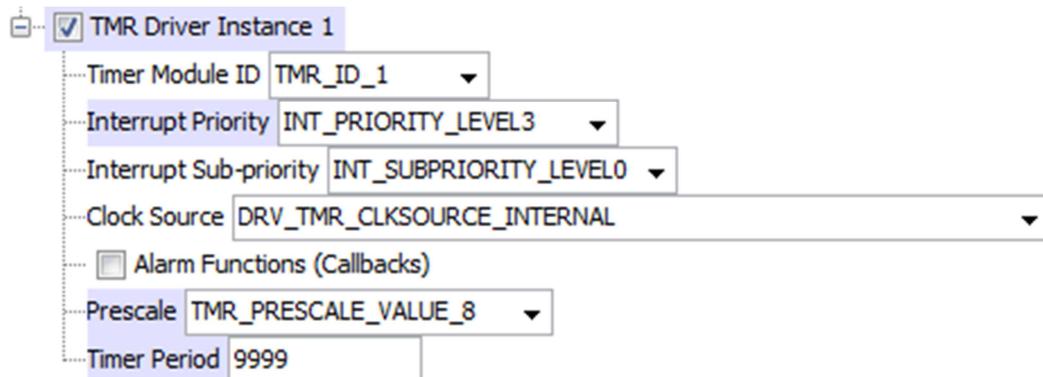
Pour remédier à ça, il faut utiliser un "prescale", soit diviser notre valeur, afin qu'elle soit dans l'échelle de travail du Timer.

$$\text{NbCoup} = \frac{\text{NbCoup}}{\text{prescale}} = \frac{80'000}{8} = 10'000$$

On peut voir qu'avec un "prescale" à 8, la nouvelle valeur est de 10'000, ce qui est correct comme nous sommes dans la tranche de valeur du Timer.

Attention : Comme l'on commence à compter à 0, la valeur finale sera donc NbCoup -1, soit 9999.

La configuration du Timer 1 est donc la suivante :



3.6.2.1.3 Timer 2

Il faut créer un Timer de façon à générer la fréquence de 77.5kHz pour la porteuse du signal DCF.

$$f_{osc} = 80\text{MHz}$$

$$f_{porteuse} = 77.5\text{kHz}$$

$$T_{osc} = \frac{1}{f_{osc}} = \frac{1}{80*10^6} = 12.5\text{ns}$$

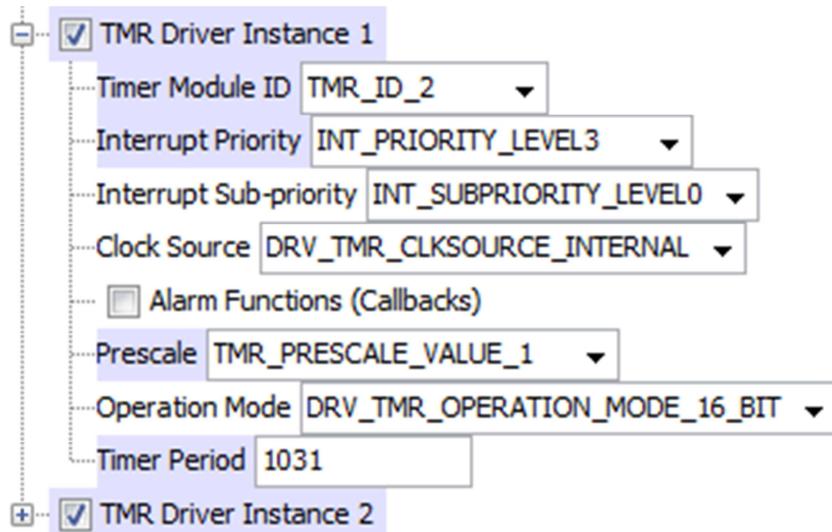
$$T_{porteuse} = \frac{1}{f_{porteuse}} = \frac{1}{77.5*10^3} = 12.90\mu\text{s}$$

$$NbCoup = \frac{T_{porteuse}}{T_{osc}} = \frac{12.9*10^{-6}}{12.5*10^{-9}} = 1032.258 = 1032$$

Comme le Timer a une échelle de valeur de 2^{16} (0 à 65'535), on peut voir que la valeur calculée se trouve dans la tranche de valeur, et qu'il n'y a ainsi pas besoin de prescale.

Attention : Comme l'on commence à compter à 0, la valeur finale sera de NbCoup -1, soit 1031.

La configuration du Timer 2 est donc la suivante :



3.6.2.1.4 Timer 3

Le Timer 3, d'une période de 200ms, va permettre de gérer la modulation.

$$f_{osc} = 80\text{MHz} \quad T_{\text{TIMER}} = 200\text{ms}$$

$$T_{osc} = \frac{1}{f_{osc}} = \frac{1}{80*10^6} = 12.5\text{ns}$$

$$NbCoup = \frac{T_{\text{TIMER}}}{T_{osc}} = \frac{200*10^{-3}}{12.5*10^{-9}} = 16'000'000$$

Comme le Timer a 2^{16} pas, soit 0 à 65'535, la valeur calculée est trop élevée vu qu'elle dépasse la valeur max.

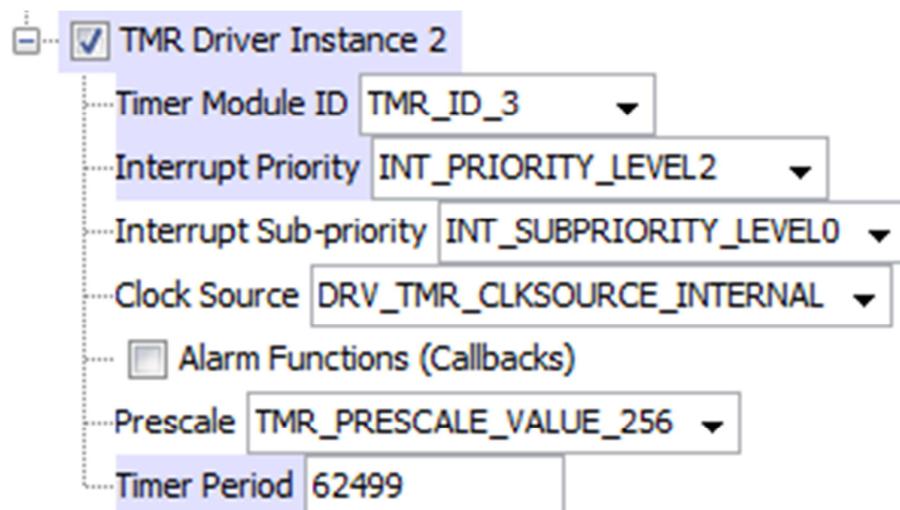
Pour remédier à ça, il faut utiliser un "prescale", soit diviser notre valeur, afin qu'elle soit dans l'échelle de travail du Timer.

$$NbCoup = \frac{NbCoup}{\text{prescale}} = \frac{16'000'000}{256} = 62'500$$

On peut voir qu'avec un "prescale" à 256, la nouvelle valeur est de 31'250, ce qui est correct comme nous sommes dans la tranche de valeur du Timer.

Attention : Comme l'on commence à compter à 0, la valeur finale sera donc NbCoup - 1, soit 62'499.

La configuration du Timer 3 est donc la suivante :

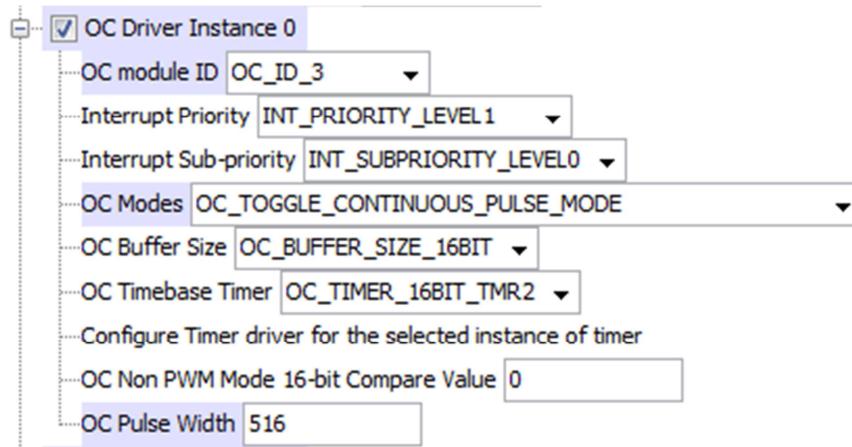


3.6.2.1.5 OC3

L'OC3 va permettre d'obtenir le signal pour la porteuse DCF à 77.5kHz, qui sera donc en relation avec le Timer2. Cet OC a été choisi car le signal DCF est directement connecté sur la pin OC3 (patte 50).

Celui-ci est en mode "PWM", est et paramétré pour fonctionner à 50% de la valeur du Timer 2, soit : $1032 / 2 = 516$

La configuration de l'OC3 est donc la suivante :

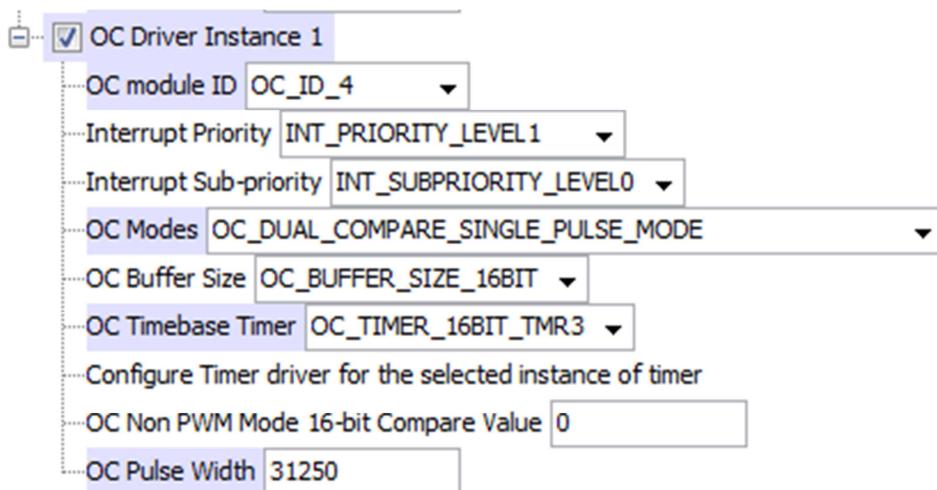


3.6.2.1.6 OC4

L'OC4 va permettre de gérer la durée des impulsions pour la modulation, en relation avec le Timer 3, qui a une période de 200ms.

Celui-ci va donc être réglé, pour être à une valeur de 50%, soit $62'500 / 2 = 31'250$, qui correspond à 100ms. L'OC Modes doit être paramétré en "oneshot", soit en mode "OC_DUAL_COMPARE_SINGLE_PULSE_MODE".

La configuration de l'OC4 est donc la suivante :



3.6.3 Affichage LCD

Pour la programmation de l'affichage LCD, j'ai repris les fichiers "Mc32DriverLcd.c" et "Mc32DriverLcd.h", comme l'affichage que j'utilise et similaire à celui du kit PIC32.

Pour l'adapter, j'ai modifier les numéros des ports pour chaque pins et j'ai repris la configuration donnée dans le datasheet de l'affichage NHD-0420AZ-FSW-GBW-33V3, concernant l'initialisation¹.

```
/*
*****void init()
{
    P1 = 0;
    P3 = 0;
    Delay(100);                                //Wait >40 msec after power is applied
    P1 = 0x30;                                  //put 0x30 on the output port
    Delay(30);                                 //must wait 5ms, busy flag not available
    Nybble();                                    //command 0x30 = Wake up
    Delay(10);                                 //must wait 160us, busy flag not available
    Nybble();                                    //command 0x30 = Wake up #2
    Delay(10);                                 //must wait 160us, busy flag not available
    Nybble();                                    //command 0x30 = Wake up #3
    Delay(10);                                 //can check busy flag now instead of delay
    P1= 0x20;                                  //put 0x20 on the output port
    Nybble();                                    //Function set: 4-bit interface
    command(0x28);                            //Function set: 4-bit/2-line
    command(0x10);                            //Set cursor
    command(0x0F);                            //Display ON; Blinking cursor
    command(0x06);                            //Entry Mode set
}
*****
```

Ce qui donne le code suivant :

```
114 void lcd_init(void)
115 {
116     // on va effectuer exactement ce que demande le ST0066U
117     // on repositionne LCD_E tout pour un démarrage correct
118     LCD_E_W = 0;
119     delay_usCt(10); // si LCD_E était à 1, on attend
120     LCD_RS_W = 0; // demandé pour une commande
121     LCD_RW_W = 0;
122
123     delay_msCt(50);
124
125     lcd_send_nibble(0x03); // correspond à 0x30, interface 4 bits
126     delay_msCt(50);
127     lcd_send_nibble(0x03); // correspond à 0x30, interface 4 bits
128     delay_msCt(50);
129     lcd_send_nibble(0x03); // correspond à 0x30, interface 4 bits
130     delay_msCt(50);
131
132     lcd_send_nibble(0x02); // correspond à 0x02, interface 4 bits
133     delay_msCt(50);
134
135
136     lcd_send_byte(0,0x28); //rs=0, 2 lines mode, display off
137     delay_usCt(400); //ds0066 demande >390us
138     lcd_send_byte(0,0x10); //rs=0, display on, cursor off, blink off
139     delay_usCt(400); //ds0066 demande >390us
140     lcd_send_byte(0,0x0F); //rs=0, display clear //0F
141     delay_msCt(200); //ds0066 demande >199ms
142     lcd_send_byte(0,0x06); //rs=0, increment mode, entire shift off
143     delay_usCt(400); //ds0066 demande >390us
144     lcd_send_byte(0,0x0C); //cursor off
145     delay_usCt(400);
146     lcd_send_byte(0,0x01); //clear display
147 }
```

¹ <http://www.newhavendisplay.com/specs/NHD-0420AZ-FSW-GBW-33V3.pdf>

3.6.3.1 Problèmes rencontrés

Lorsque l'affichage LCD a été testé, le backlight s'allumait bien, or aucun caractère ne s'affichait. L'erreur provenait du fait que les durées de temps paramétrées correspondaient à un clock de 40MHz du microcontrôleur, alors que celui de la carte est de 80MHz.

Cela signifie que le clock était beaucoup trop rapide pour les durées réglées et les caractères n'avaient ainsi pas le temps de s'afficher. Pour identifier le problème, j'ai réalisé un debug en mode pas à pas, afin de regarder où le programme restait bloqué.

J'ai ainsi pu observer que le programme restait bloqué dans la fonction "lcd_send_byte", à la ligne 105, soit en lien avec la fonction "lcd_read_byte()".

```
102     void lcd_send_byte( BYTE address, BYTE n )
103     {
104         LCD_RS_W = 0;
105         while ( (lcd_read_byte() & 0x80) == 0x80 ) ;
106         LCD_RS_W = address;
107         LCD_RW_W = 0;
108         //LCD_E déjà à 0
109         lcd_send_nibble(n >> 4);
110         lcd_send_nibble(n & 0xf);
111     }
```

J'ai donc multiplié les durées par un facteur 4, grâce à un "for", afin de rendre le code plus lisible, comme de base le temps est de 500ns, ce qui est beaucoup trop court par rapport au clock de 80MHz.

Ce qui donne la modification suivante dans la fonction "lcd_read_byte" :

```
37     BYTE lcd_read_byte( void )
38     {
39         UINT8_BITS lcd_read_byte;
40         uint8_t i;
41         LCD_DB4_T = 1; // 1=input
42         LCD_DB5_T = 1;
43         LCD_DB6_T = 1;
44         LCD_DB7_T = 1;
45         LCD_RW_W = 1;
46         for(i = 0; i < 4 ; i++)
47         {
48             delay500nsCt(); //ds0066 demande 0.5us
49         }
50         LCD_E_W = 1;
51         for(i = 0; i < 4 ; i++)
52         {
53             delay500nsCt(); //ds0066 demande 0.5us
54         }
55         lcd_read_byte.bits.b7 = LCD_DB7_R;
56         lcd_read_byte.bits.b6 = LCD_DB6_R;
57         lcd_read_byte.bits.b5 = LCD_DB5_R;
58         lcd_read_byte.bits.b4 = LCD_DB4_R;
59         LCD_E_W = 0; // attention e pulse min = 500ns à 1 et autant à 0
```

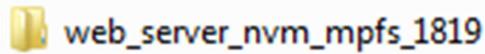
3.6.4 Ethernet

3.6.4.1 Programme de test

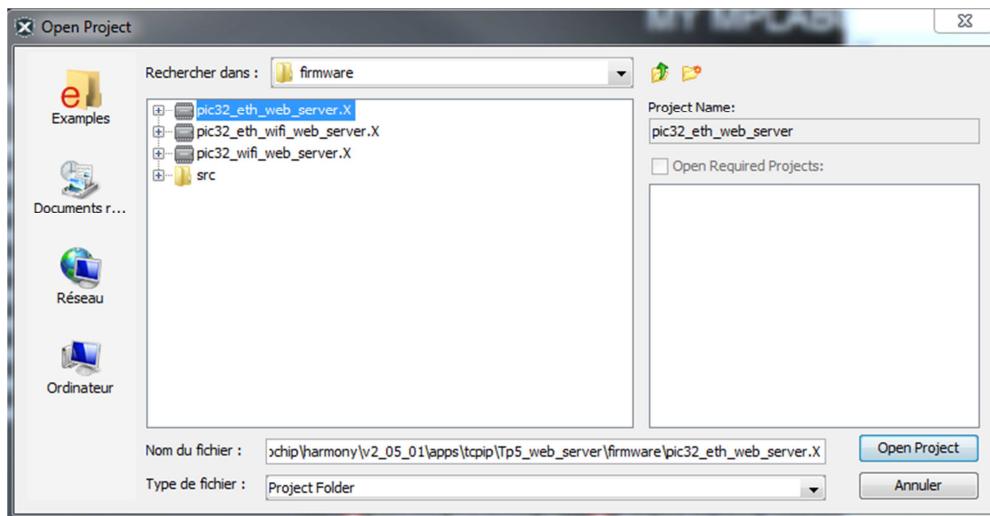
Pour tester la partie Ethernet, j'ai repris le projet "web_server_nvm_mpfs", qui se trouve sous :

C:\microchip\harmony\v2_05_01\apps\tcpip\web_server_nvm_mpfs

Afin de ne pas modifier le contenu de l'exemple, j'ai recopié celui-ci et est renommer le répertoire.

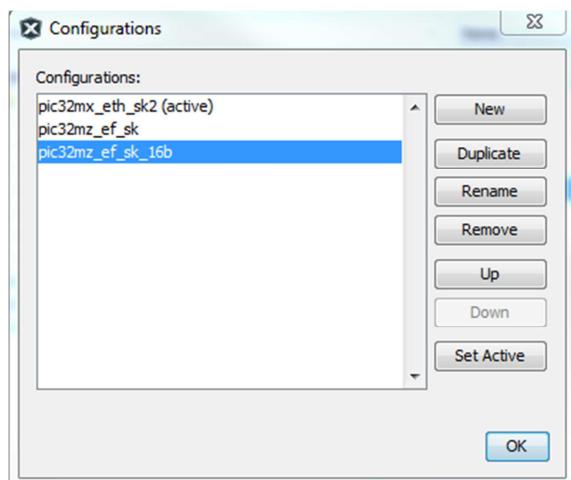


Ensute, j'ai ouvert "MPLAB X IDE v4.15", puis suis allé chercher le projet que l'on vient de renommer, et sélectionner le projet "pic32_eth_web_server.X".

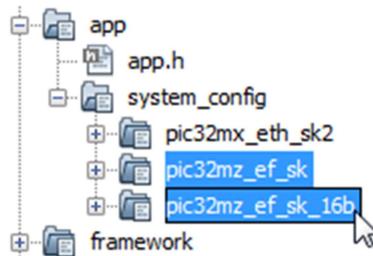


Ensute, on va supprimer les configurations dont nous n'avons pas besoin. Pour ce faire, il faut refaire un clic droit sur le projet -> Properties -> Manage Configuration ...

On retire les configurations qui ne nous intéressent pas, soit celle en "mz", que l'on "Remove".

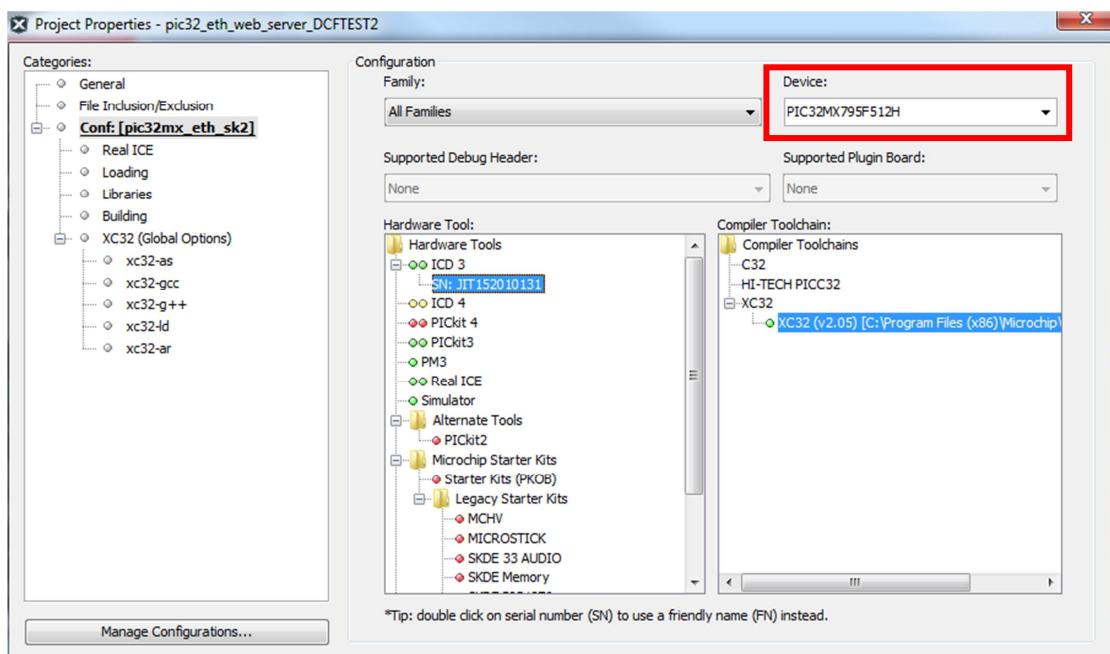


On réalise la même opération pour les fichiers sources "headers files" qui ne sont pas utiles (mz), que l'on "Remove" du projet.

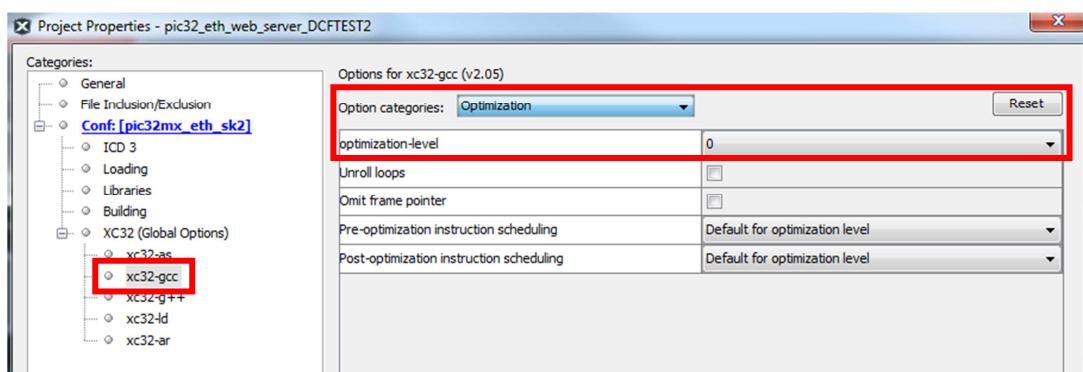


Dans les propriétés du projet, il faut se rendre dans les propriétés :
Clic droit sur le projet -> Properties

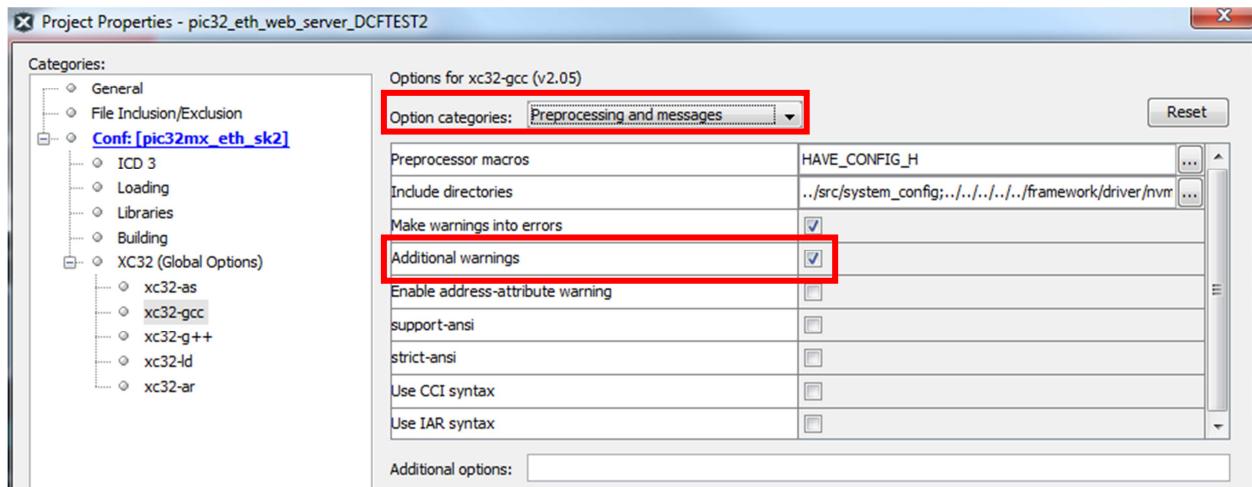
Dans "Device", il faut sélectionner le bon microcontrôleur : PIC32MX795F512H



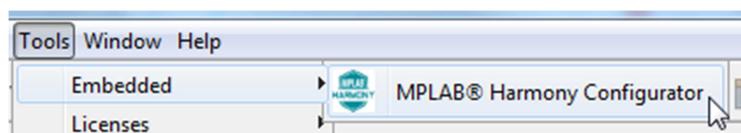
Dans "xc32-gcc", dans l'option "Option categories", il faut sélectionner "Optimization" puis mettre "optimization-level" à 0.



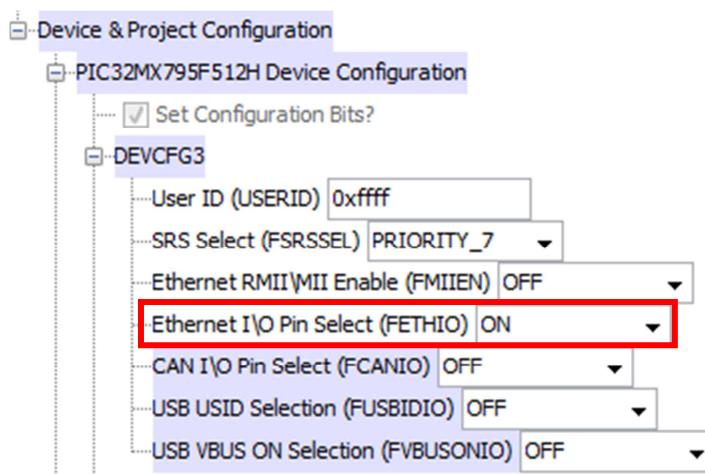
Toujours dans le "xc32-gcc", il faut sélectionner l'option "Preprocessing and messages", et cochez "Additional warnings".



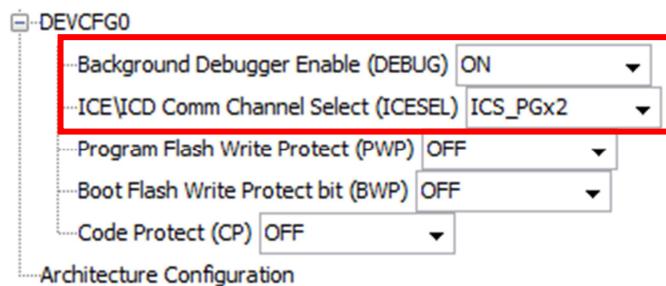
Ensuite, il faut se rendre dans "MPLAB Harmony Configurator".



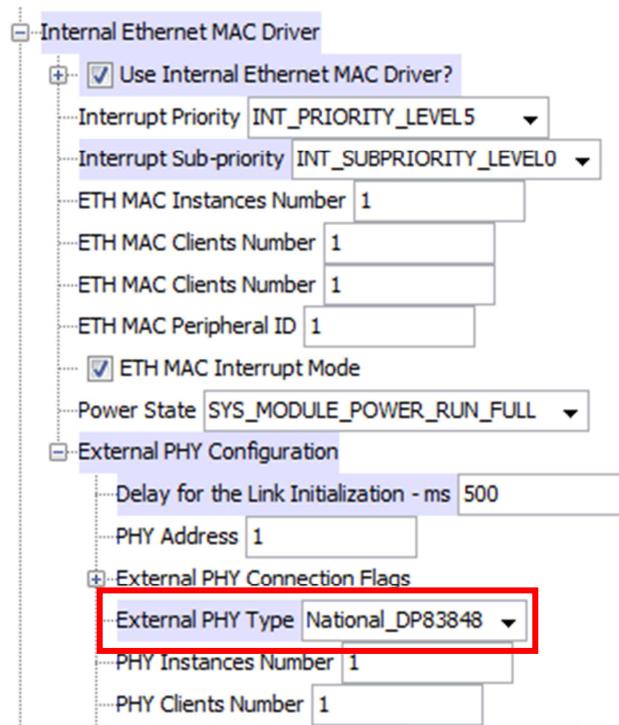
Dans les "Device & Project Configuration", il faut mettre à "ON" l'Ethernet I/O Pin Select (FETHIO), pour utiliser le port Ethernet par défaut.



On active le "DEBUG" et on sélectionne également "ICS_PGx2".



On sélectionner le bon cheap pour l'Ethernet "DP83848".



Lorsque les différents réglages sont terminés, j'ai compilé le projet et plusieurs erreurs sont survenues, qu'il faut corriger de la manière suivante :

Comme l'usb n'est pas utilisé dans le projet, les éléments suivants ont été mis en commentaire

Fichier "bsp.c"

```

391 void BSP_Initialize(void)
392 {
393     /* Setup the USB VBUS Switch Control Pin */
394     //BSP_USBVBUSSwitchStateSet(BSP_USB_VBUS_SWITCH_STATE_DISABLE);
395
396     /* Switch off LEDs */
397     BSP_LEDOff(BSP_LED_1);
398     BSP_LEDOff(BSP_LED_2);
399     BSP_LEDOff(BSP_LED_3);
400 }

321 //void BSP_USBVBUSSwitchStateSet(BSP_USB_VBUS_SWITCH_STATE state)
322 //{
323     /* Enable the VBUS switch */
324
325     // PLIB_PORTS_PinWrite( PORTS_ID_0, PORT_CHANNEL_None, PORTS_BIT_POS_-1, state );
326 }

361 ///void BSP_USBVBUSSPowerEnable(uint8_t port, bool enable)
362 //{
363     /* Enable the VBUS switch */
364
365     //PLIB_PORTS_PinWrite( PORTS_ID_0, PORT_CHANNEL_None, PORTS_BIT_POS_-1, enable );
366 }

```

Le port A n'existe pas sur le microcontrôleur utilisé, également le mettre en commentaire.

Fichier "sys_ports_static.c"

```

81  :     /* PORT A Initialization */
82  :     // PLIB_PORTS_OpenDrainEnable(PORTS_ID_0, PORT_CHANNEL_A, SYS_PORT_A_ODC);
83  :     // PLIB_PORTS_Write( PORTS_ID_0, PORT_CHANNEL_A,   SYS_PORT_A_LAT);
84  :     // PLIB_PORTS_DirectionOutputSet( PORTS_ID_0, PORT_CHANNEL_A,   SYS_PORT_A_TRIS ^ 0xFFFF);

```

3.6.5 Tests

Afin de voir si la partie Ethernet est fonctionnelle, il faut en premier lieu, simplement connecter la prise réseau dans le connecteur RJ-45.

Malheureusement le premier test n'a pas été concluant, car les Leds du connecteur ne réagissaient pas, donc ne s'allumait jamais.

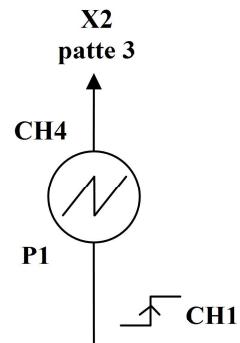
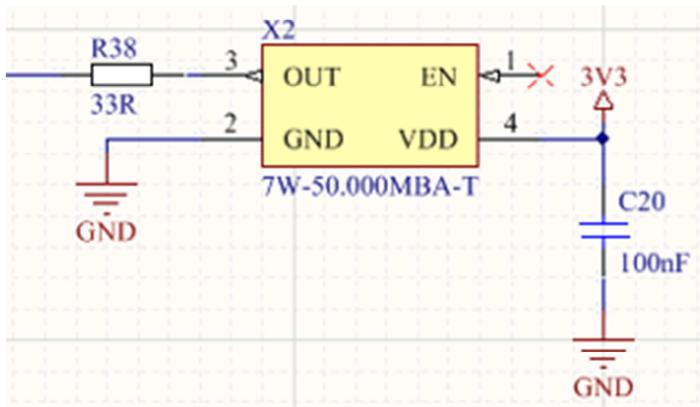
En premier lieu, j'ai refait un contrôle des brasures de la partie concernée, et je me suis aperçue qu'il y a avait deux pattes du chip Ethernet qui n'étaient pas brasée correctement. Malgré cette correction, le fonctionnement était toujours le même.

J'ai donc décidé de vérifier toutes les alimentations de cette partie, et j'en suis arrivé à la conclusion que celles-ci étaient toutes correctes, soit une valeur de 3.3V, comme on peut le voir dans le tableau ci-dessous.

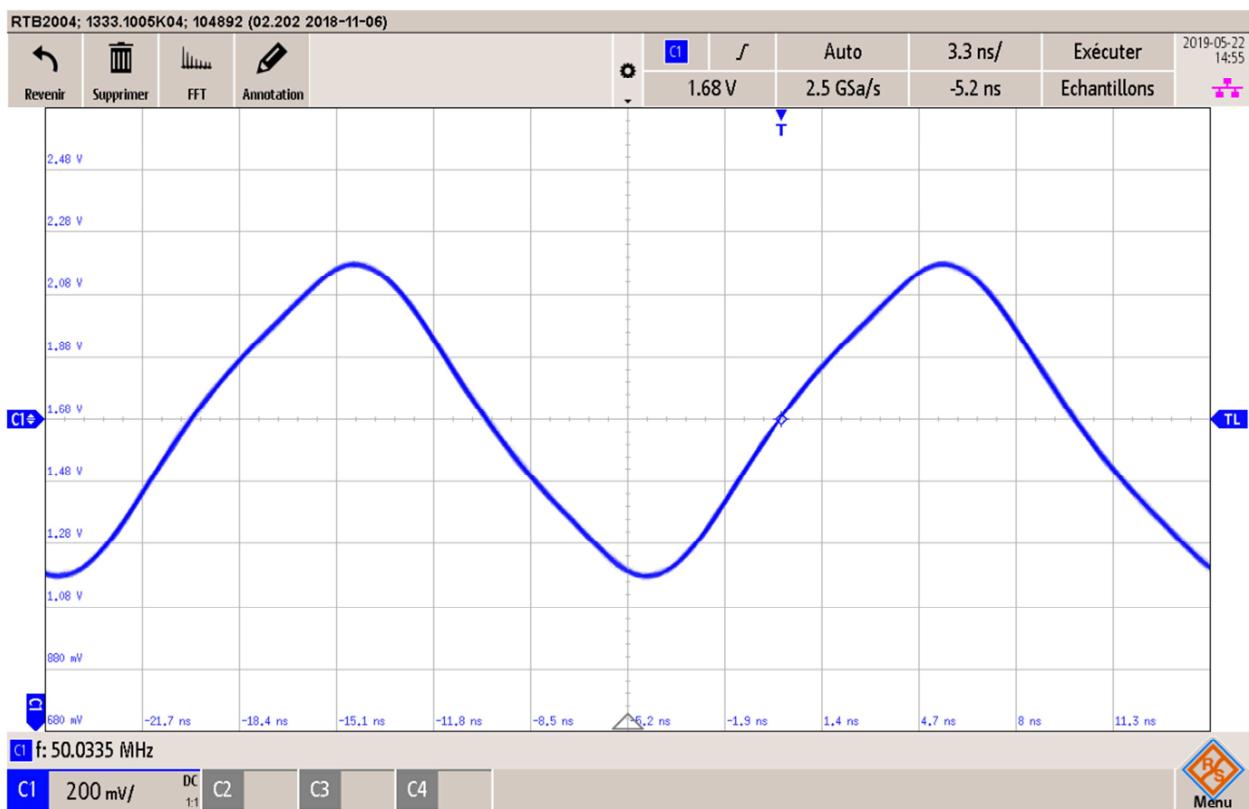
	Valeur théorique	Valeur mesurée	Erreur absolue
Résistance R22, R23	3.3V	3.292V	-0.24%
Résistance R24, R25, R21	3.3V	3.292V	-0.24%
Résistance R26, R27, R28, R29, R30, R31, R32, R33, R34, R35	3.3V	3.293V	-0.21%
Condensateur C21, C22, C23	3.3V	3.293V	-0.21%
IOVDD33 - patte 32	3.3V	3.292V	-0.24%
IOVDD33 - patte 48	3.3V	3.293V	-0.21%
AVDD33 - patte 22	3.3V	3.293V	-0.21%
X2 VDD - patte 4	3.3V	3.286V	-0.42%

$$\text{Erreur absolue} = \frac{\text{Valeur mesurée} - \text{Valeur théorique}}{\text{Valeur théorique}} * 100$$

Schéma de mesure



Comme deuxième vérification, j'ai mesuré le signal en sortie de l'oscillateur, qui est la référence du clock à 50MHz. On observe ainsi que la fréquence de sortie est correcte.



Pour finir, j'ai effectué un contrôle de la valeur de tous les composants, et est revérifié le schéma, que j'ai également fait contrôler à des tierces personnes.

Après ces deux vérifications, je n'ai détecté aucun problème.

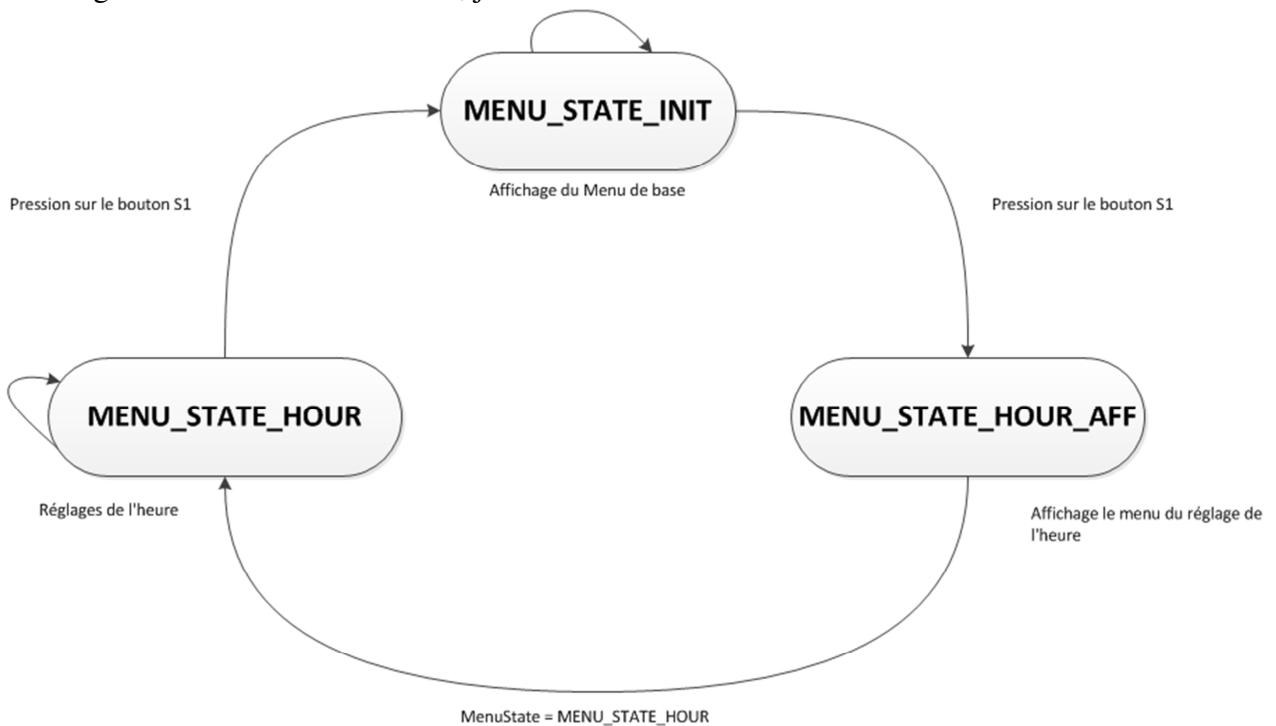
Afin de contrôler que le problème ne venait pas du software, j'ai repris le programme et changé le microcontrôleur, pour tester avec la carte du kit PIC32, qui c'est en effet avéré fonctionnel.

Je n'ai ainsi pas pu déterminer d'où venait le problème, car cette partie est relativement complexe.

Comme cette partie permettait de recevoir l'heure, celle-ci sera donc réglée à l'aide des boutons.

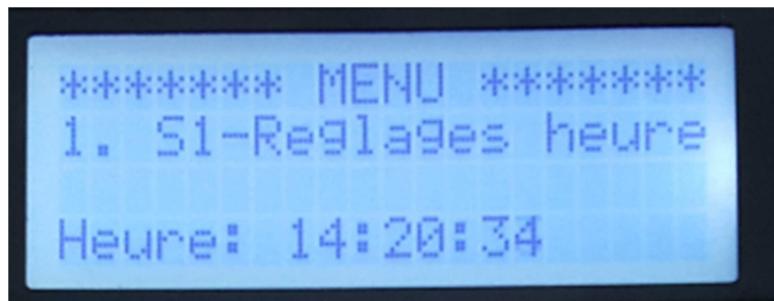
3.6.6 Gestion des menus

Pour la gestion des différents menus, j'ai décidé de réaliser une machine d'état.



MENU_STATE_INIT :

Affiche le menu initial, ainsi que l'heure avec les options de réglages possibles.



```

234 //Affichage initial des menus
235 case MENU_STATE_INIT :
236 {
237     lcd_gotoxy(1, 1);
238     printf_lcd("***** MENU *****");
239     lcd_gotoxy(1, 2);
240     printf_lcd("1. S1-Reglages heure");
241     lcd_ClearLine(3);
242     //Affichage de l'heure réglée
243     lcd_gotoxy(1, 4);
244     printf_lcd("Heure: %2d:%2d:%2d ",valHour, valMinute, valSeconde);
245     break;
246 }
  
```

MENU_STATE_HOUR_AFF :

Affiche le menu du réglage de l'heure.



```
247 //Affichage du menu du réglage de l'heure
248 case MENU_STATE_HOUR_AFF :
249 {
250     //Etat '0' pour bloquer l'incrémentation de l'heure
251     etatReglHour = 0;
252     //Affichage LCD
253     lcd_gotoxy(1, 1);
254     printf_lcd("REGLAGE HEURE **");
255     lcd_gotoxy(1, 2);
256     printf_lcd("Heure Minute Seconde");
257     lcd_gotoxy(1, 3);
258     printf_lcd("%2d:%2d:%2d", valHour, valMinute, valSeconde);
259
260     lcd_gotoxy(1, 4);
261     printf_lcd("S1 pour valider");
262     //Etat suivant
263     MenuState = MENU_STATE_HOUR;
264     break;
265 }
```

MENU_STATE_HOUR :

Permet de régler les paramètres de l'heure, à savoir les heures, minutes et secondes grâce aux boutons.

Pour les trois paramètres réglables, des fonctions de reboulement ont été réalisées, afin de rester dans des plages de valeurs correctes.

```

247 //Affichage du menu du réglage de l'heure
248 case MENU_STATE_HOUR_AFF :
249 {
250     //Etat '0' pour bloquer l'incrémentation de l'heure
251     etatReglHour = 0;
252     //Affichage LCD
253     lcd_gotoxy(1, 1);
254     printf_lcd("/** REGLAGE HEURE **");
255     lcd_gotoxy(1, 2);
256     printf_lcd("Heure Minute Seconde");
257     lcd_gotoxy(1, 3);
258     printf_lcd("      %2d:%2d:%2d      ", valHour, valMinute, valSeconde);
259
260     lcd_gotoxy(1, 4);
261     printf_lcd("S1 pour valider");
262     //Etat suivant
263     MenuState = MENU_STATE_HOUR;
264     break;
265 }
266 //Réglages de l'heure
267 case MENU_STATE_HOUR :
268 {
269     //Bouton "SW2" incrémente les heures
270     if(DebounceIsPressed(&DescrSW2))
271     {
272         DebounceClearPressed(&DescrSW2);
273         valHour += 1;
274         FlagChange = true;
275     }
276     reboulementHour (&valHour);
277
278     //Bouton "SW3" incrémente les minutes
279     if(DebounceIsPressed(&DescrSW3))
280     {
281         DebounceClearPressed(&DescrSW3);
282         valMinute += 1;
283         FlagChange = true;
284     }
285     reboulementMinute (&valMinute);
286
287     //Bouton "SW4" incrémente les secondes
288     if(DebounceIsPressed(&DescrSW4))
289     {
290         DebounceClearPressed(&DescrSW4);
291         valSeconde += 1;
292         FlagChange = true;
293     }
294     reboulementSeconde (&valSeconde);
295
296     //Affichage des réglages
297     if(FlagChange == true)
298     {
299         lcd_gotoxy(1, 3);
300         printf_lcd("      %2d:%2d:%2d      ", valHour, valMinute, valSeconde);
301         lcd_gotoxy(1, 4);
302         printf_lcd("S1 pour valider");
303     }
304
305     //Bouton "SW1" valide les réglages de l'heure
306     if(DebounceIsPressed(&DescrSW1))
307     {
308         DebounceClearPressed(&DescrSW1);
309         //Retour au menu initial
310         MenuState = MENU_STATE_INIT;
311         etatReglHour = 1;
312     }
313     break;
314 }
315 }
316 }
```

```
//Fonction de reboulement pour l'heure
void reboulementHour (int *valHour)
{
    if(*valHour > 23)
    {
        *valHour = 0;
    }
    else if (*valHour <= 0)
    {
        *valHour = 23;
    }
}
```

```
//Fonction de reboulement pour les minutes
void reboulementMinute (int *valMinute)
{
    if(*valMinute == 59)
    {
        *valMinute = 0;
    }
    else if(*valMinute <= 0)
    {
        *valMinute = 59;
    }
}
```

```
//Fonction de reboulement pour les secondes
void reboulementSeconde (int *valSeconde)
{
    if(*valSeconde == 60)
    {
        *valSeconde = 0;
    }
    else if(*valSeconde <= 0)
    {
        *valSeconde = 59;
    }
}
```

3.6.7 Signaux DCF

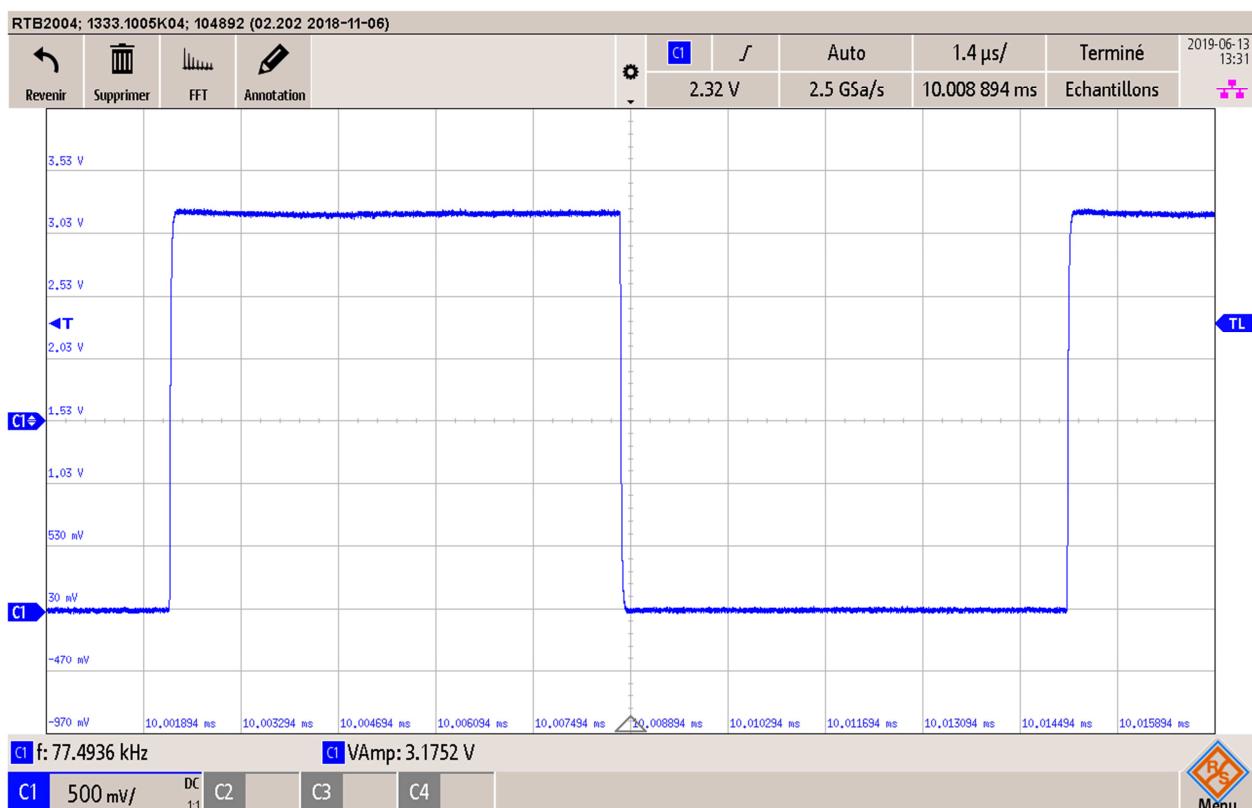
Cette partie permet de recréer le signal DCF, c'est-à-dire qu'une porteuse est modulée par des impulsions, au rythme d'une par seconde, avec une fréquence de 77.5kHz.

Ces impulsions se traduisent chaque seconde par une diminution de 25% de l'amplitude du signal reçu.

3.6.7.1 Porteuse

Le signal pour la porteuse a été créé sur la base d'un Timer, soit le Timer 2 et de l'OC3, qui permet de directement générer le signal à 77.5kHz, avec une amplitude de 3.3V. (**Voir point 3.6.2.1.3 Timer 2 & point 3.6.2.1.5 OC3**).

Comme le signal est directement sur la pin de l'OC3, celui-ci sera directement mis en sortie de celle-ci.



Fréquence : 77.4936kHz

Amplitude : 3.1752V

3.6.7.2 Explications

	Valeur théorique	Valeur mesurée	Erreur absolue
Fréquence porteuse	77.5kHz	77.4936kHz	-0.008%
Amplitude	3.3V	3.1752V	-3.78%

$$\text{Erreur absolue} = \frac{\text{Valeur mesurée} - \text{Valeur théorique}}{\text{Valeur théorique}} * 100$$

On peut ainsi voir que la porteuse a bien les valeurs souhaitées.

3.6.7.3 Analog switch

Comme on peut le voir dans le datasheet, l'état initial du composant est "NC", donc fermé.

The TS12A4514/TS12A4515 are single pole/single throw (SPST), low-voltage, single-supply CMOS analog switches, with very low switch ON-state resistance. The TS12A4514 is normally open (NO). The TS12A4515 is normally closed (NC).

N.C. – Not internally connected

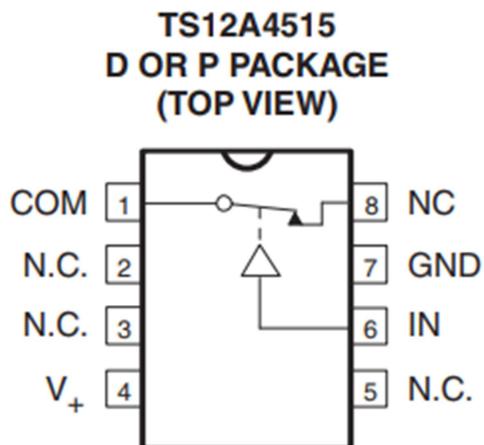
NO – Normally open

NC – Normally closed

Dans le tableau ci-dessous, l'analog switch (TS12A4515) sera ouvert ou fermé selon l'état appliqué sur l'entrée IN.

Etat haut sur IN : Fermé → Actif

Etat bas sur IN : Ouvert → Repos



INPUT	SWITCH STATE	
	TS12A4514	TS12A4515
LOW	OFF	ON
HIGH	ON	OFF

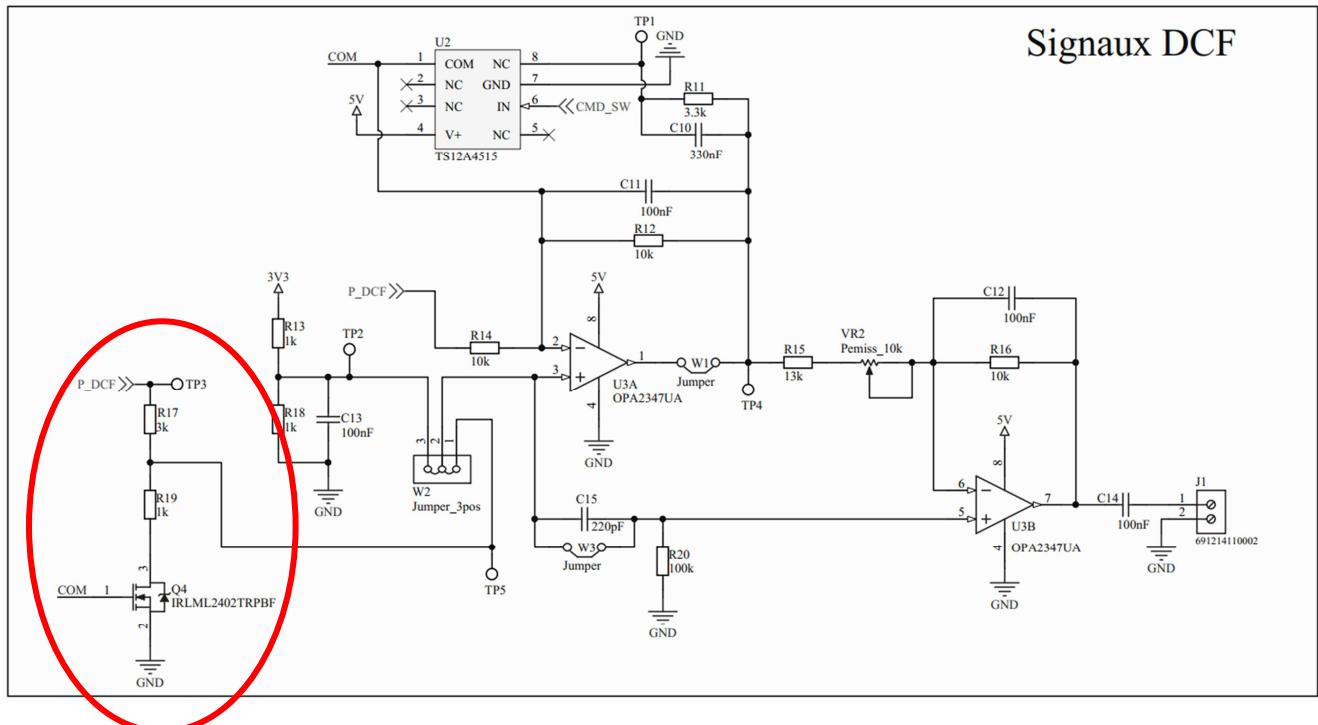
L'analog switch, lorsqu'il sera ouvert, va activer seulement le premier filtre (C10 et R7), ainsi le signal sera à 100%.

Dans le cas contraire, lorsqu'il sera fermé, cette fois-ci les deux filtres seront actifs, ce qui va permettre d'abaisser la tension du signal à 25%, soit $\frac{1}{4}$ de celle-ci.

3.6.7.4 Test de la partie analogique

Cette partie a pour but de réaliser un abaissement de 25% de l'amplitude du signal.

3.6.7.4.1 Possibilité 1



La première possibilité utilise la partie avec les composants suivants : Q4, R17, R19

Cela signifie que, pour que celle-ci fonctionne, il faut avoir les jumpers dans les états suivants :

W1 : Ouvert

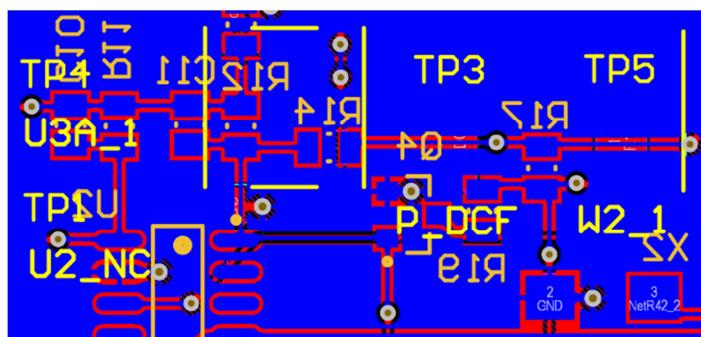
W2 : 1-2 Fermé

W3 : Ouvert

Or, après avoir réétudié le schéma, je me suis rendu compte que cette partie ne pourrait pas fonctionner, car elle présente un défaut au niveau du Hardware.

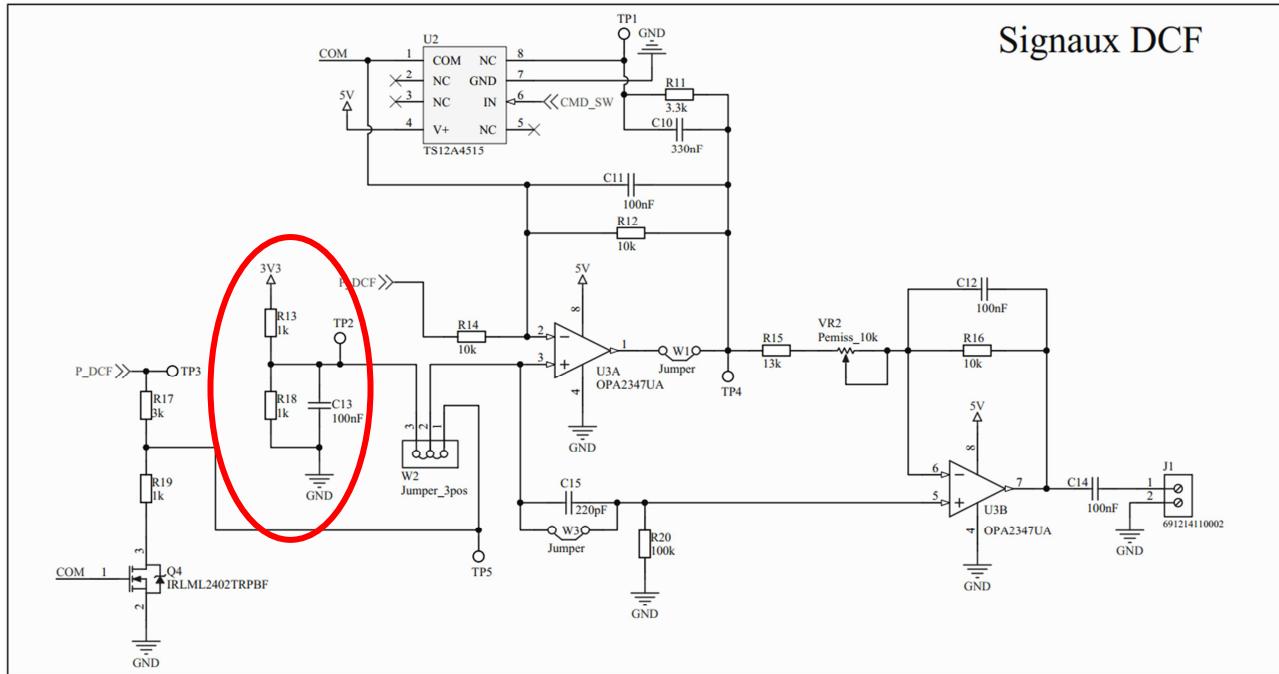
En effet, si le jumper W1 est ouvert, cela signifie qu'il n'y aura pas le signal voulu sur l'entrée de la partie concerné, soit 'COM'.

Afin que cette partie n'influence pas la seconde, j'ai décidé d'ôter les composants Q4 et R17.



(Voir Annexe VI : Fichier des modifications)

3.6.7.4.2 Possibilité 2



La seconde possibilité utilise la partie avec les composants suivants : R13, R18, C13
 Cela signifie que, pour que celle-ci fonctionne, il faut avoir les jumpers dans les états suivants :

W1 : Fermé

W2 : 2-3 Fermé

W3 : Fermé

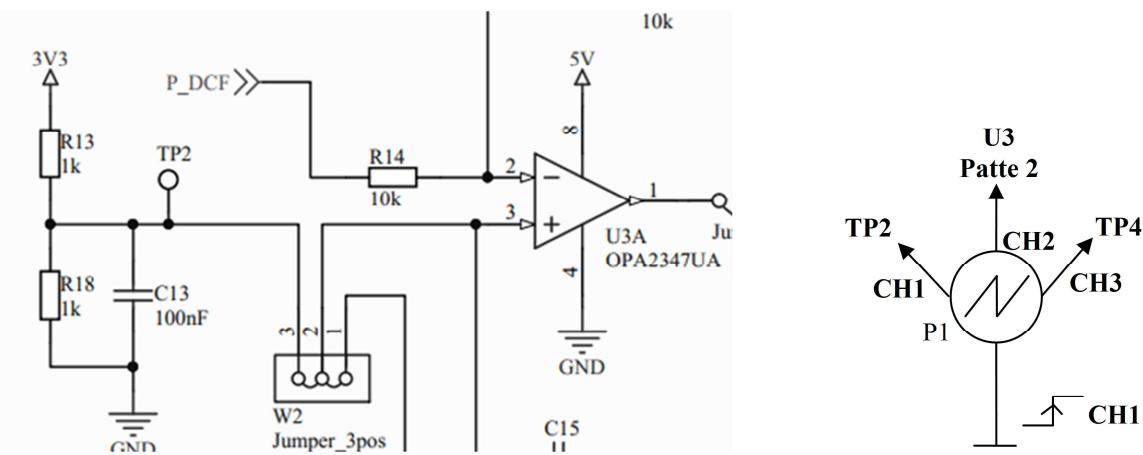
3.6.7.5 Mesure partie analogique

Conditions : Analog switch à l'état "open" donc un état bas à l'entrée.

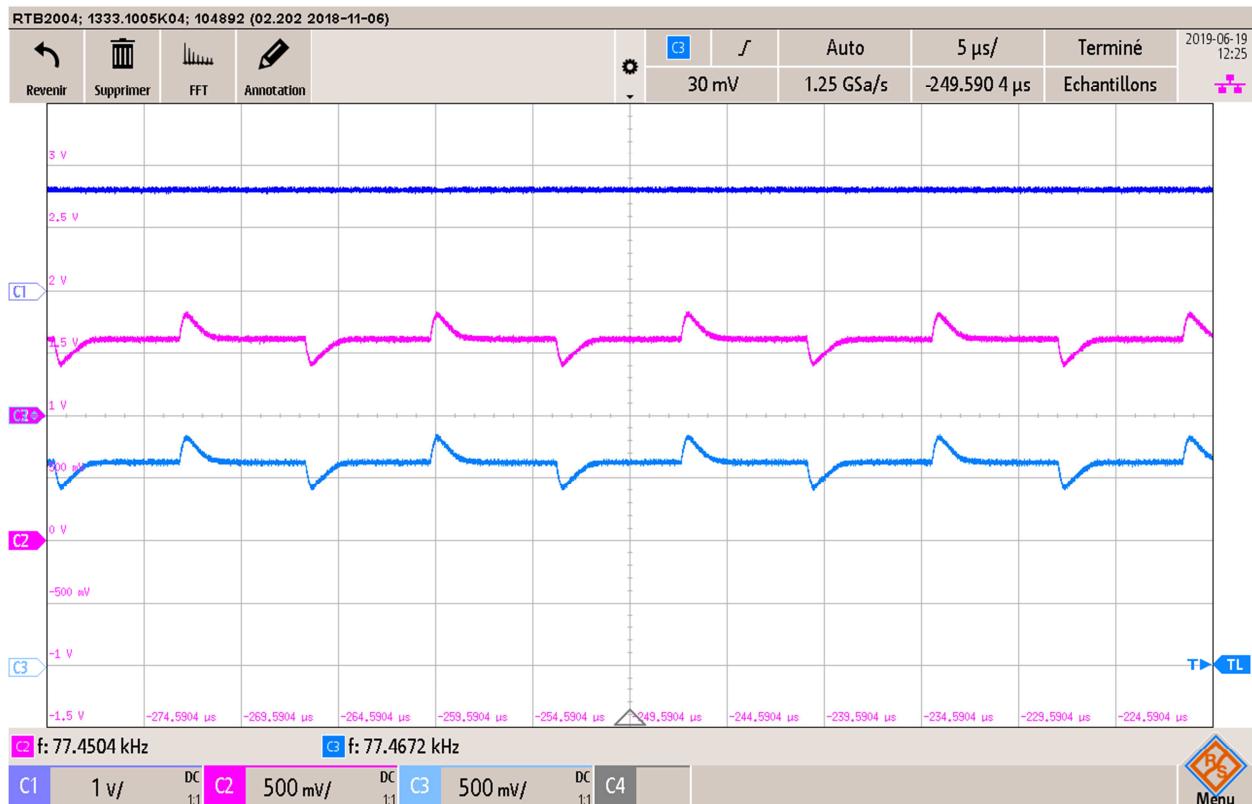
```

127 // **** TIMER 3 - Modulation ****
128 void __ISR(_TIMER_3_VECTOR, ipl2AUTO) IntHandlerDrvTmrInstance2(void)
129 {
130     PLIB_INT_SourceFlagClear(INT_ID_0, INT_SOURCE_TIMER_3);
131     CMD_SW_R = 0;
132 }
```

Schéma de mesure



3.6.7.5.1 Mesure



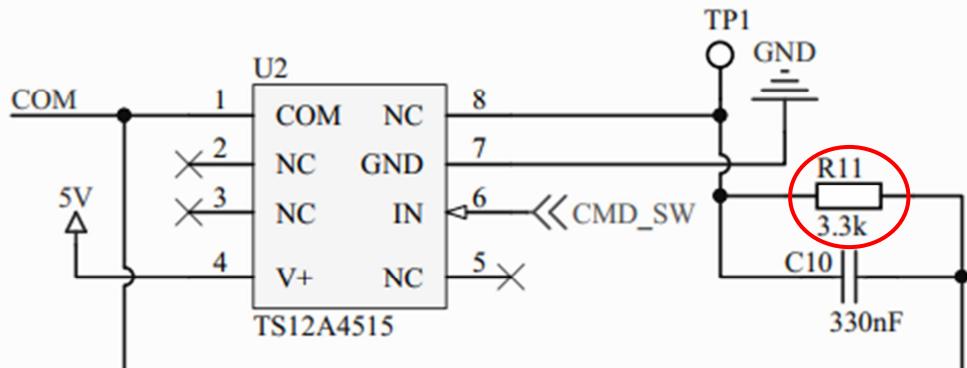
Sur le canal 1, on retrouve bien la tension de 1.65V du pont diviseur (R13 et R18). En revanche, on remarque que le signal sur la patte (-) de l'amplificateur, et identique à celui en sortie, ce qui n'est pas correct.

Cela permet d'émettre les hypothèses suivantes :

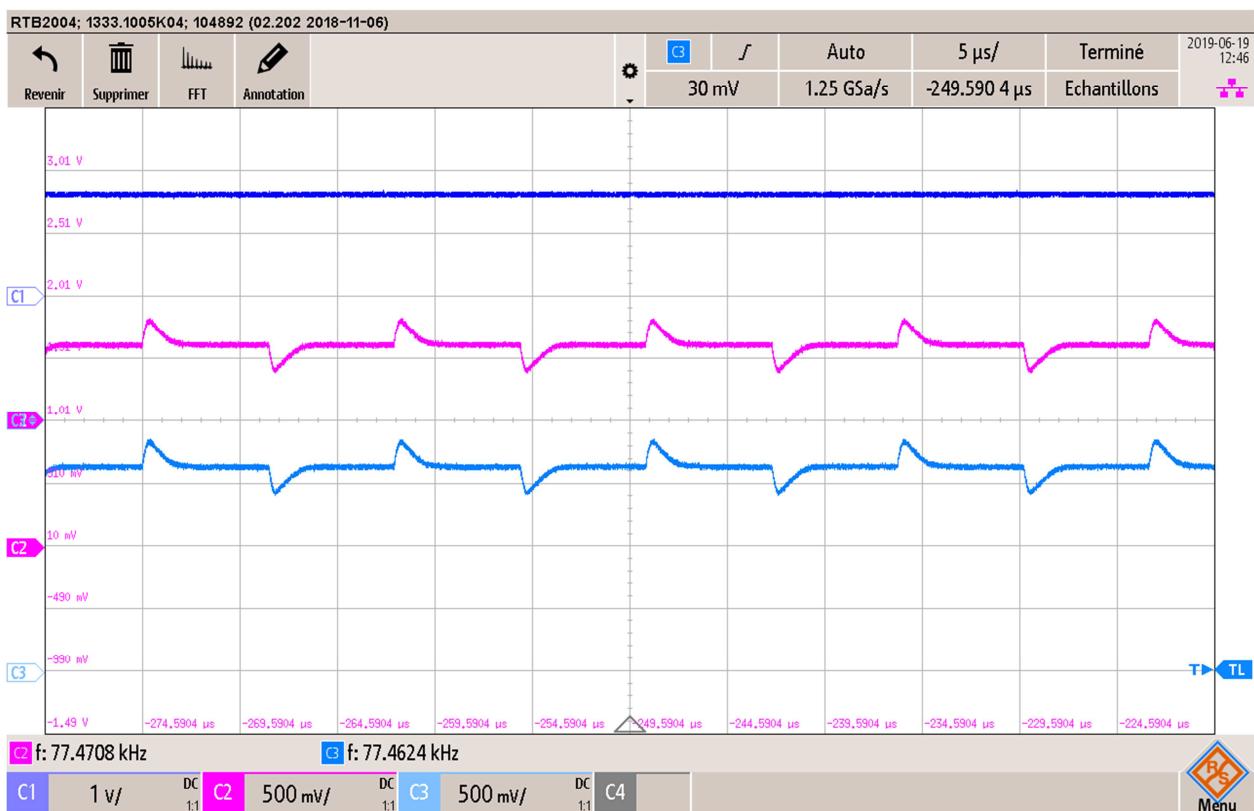
1. Il y a un court-circuit entre la sortie et l'entrée, ce qui créer un suiveur.
2. L'amplificateur U3 est KO ou n'est pas un choix judicieux.

3.6.7.6 Dépannage

En premier lieu, j'ai décidé d'ôter la résistance R11, afin que l'analog switch n'ait plus d'influence sur le système.



J'ai ainsi refait la même mesure que précédemment.



On observe que les signaux n'ont pas changé, et que l'analog switch était bien à l'état ouvert, et qu'il n'a pas d'influence sur le système.

Deuxièmement, j'ai recontrôlé toutes les brasures de la partie DCF, afin de vérifier qu'il n'y ait pas de court-circuit ou de pattes mal brasées, et là encore, tout est OK.

Troisièmement, j'ai vérifié toutes les valeurs des composants, et là tout était correct.

Comme dernière possibilité, je me suis penchée sur l'amplificateur U3. Je me suis ainsi demandé si la vitesse de balayage n'était pas trop importante.

3.6.7.6.1 Test de l'amplificateur OPA2347UA

Afin de vérifier que l'amplificateur utilisé supporte une fréquence de 77.5kHz, sans perturbation, j'ai décidé de prendre celui que j'avais en stock et de réaliser la configuration ci-dessous :

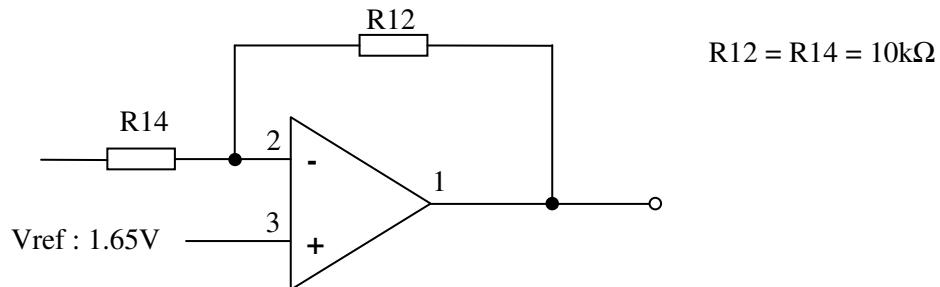
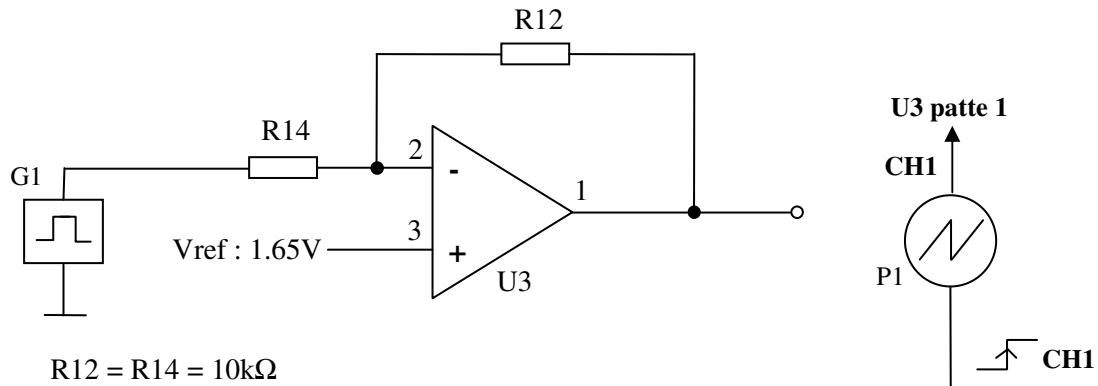
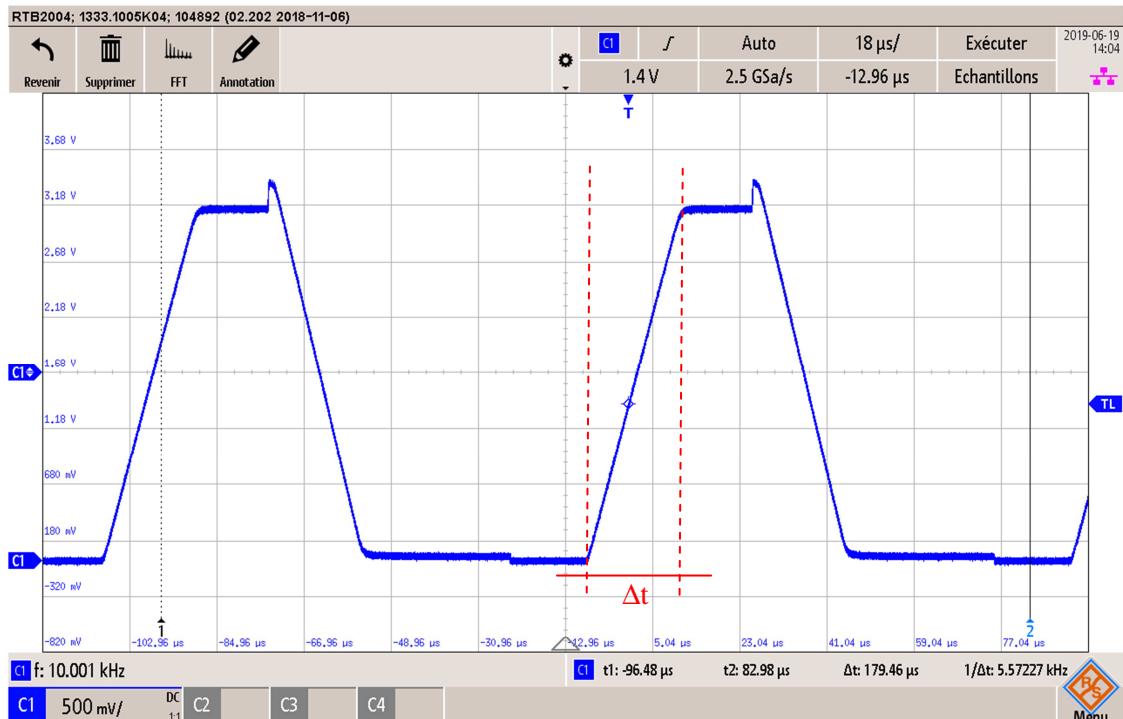


Schéma de mesure



Le générateur est paramétré pour sortir un signal carré d'une amplitude de 3.3V, avec pour la première mesure une fréquence de 10kHz, et pour la seconde une fréquence de 77.5kHz.

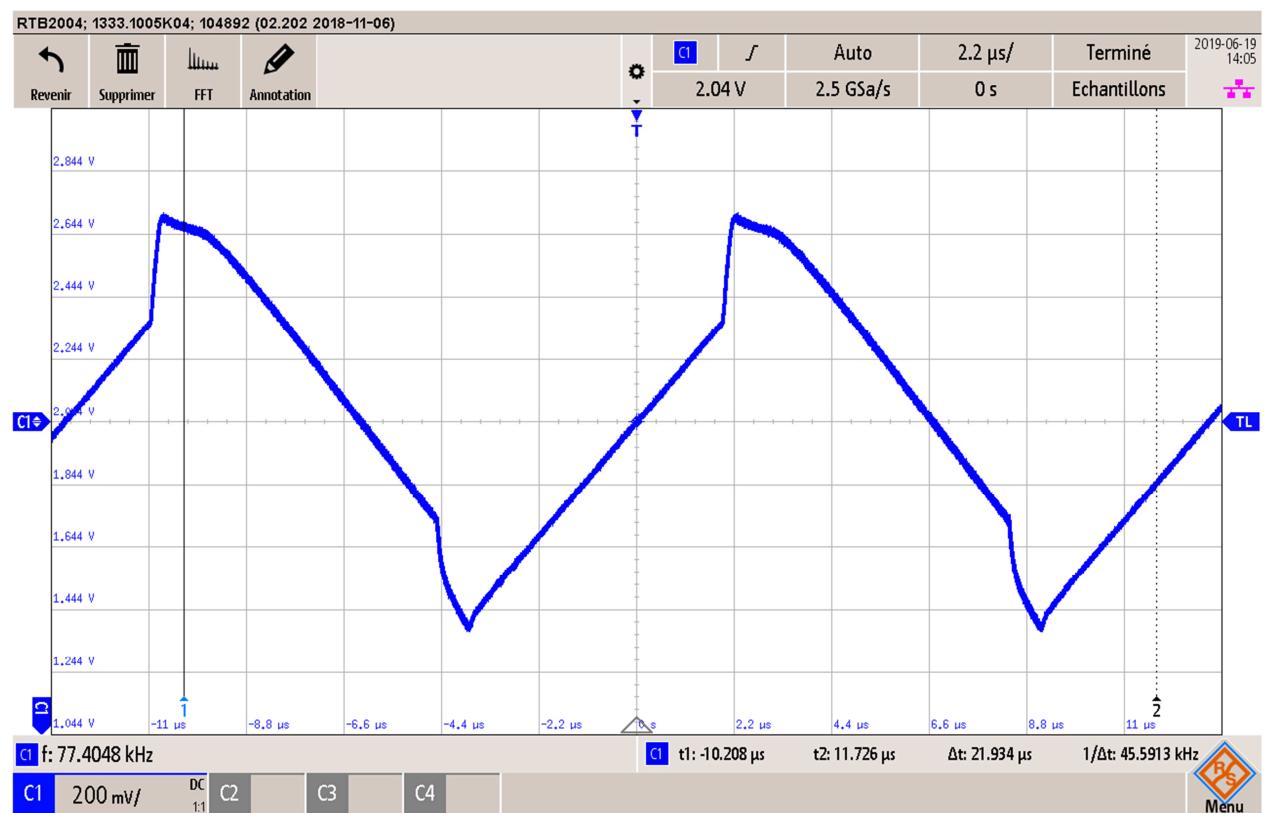
3.6.7.6.1.1 Mesure @ 10kHz



$$\Delta t = 18\text{us}$$

$$\Delta V = 3.18V$$

3.6.7.6.1.2 Mesure @ 77.5kHz



3.6.7.6.1.3 Explication des mesures

Sur la première mesure, à 10kHz, on peut voir que le delta en tension est de 3.18V et au niveau de la durée, celui-ci est de 18us.

On peut donc, à partir de ces valeurs, déterminer le slew rate : $\frac{\Delta V}{\Delta t} = \frac{3.18}{18 \times 10^{-6}} = 0.176 \text{V/us}$

Lorsque l'on consulte le datasheet de l'OPA2347UA, pour un gain de 1, la valeur de slew rate est la suivante, 0,17V/us :

FREQUENCY RESPONSE	GBW	$C_L = 100\text{pF}$					kHz
Gain-Bandwidth Product	SR	$G = +1$				0.17	V/us
Slew Rate	t_s	$V_S = 5\text{V}, 2\text{V Step}, G = +1$				21	μs
Settling Time, 0.1%		$V_S = 5\text{V}, 2\text{V Step}, G = +1$				27	μs
0.01%		$V_{IN} \times \text{Gain} = V_S$				23	μs
Overload Recovery Time							

On voit donc que la valeur mesurée, correspond à celle donnée par le fabricant.

Sur la seconde mesure, on peut voir qu'à une fréquence de 77.5kHz, le signal n'a pas le temps de réagir correctement. C'est-à-dire que lorsqu'il a atteint sa pente maximum, il n'a pas assez de temps pour réagir et redescendre.

L'amplificateur choisi n'est donc pas adapté à ce niveau-là, ce qui explique le dysfonctionnement de la partie analogique.

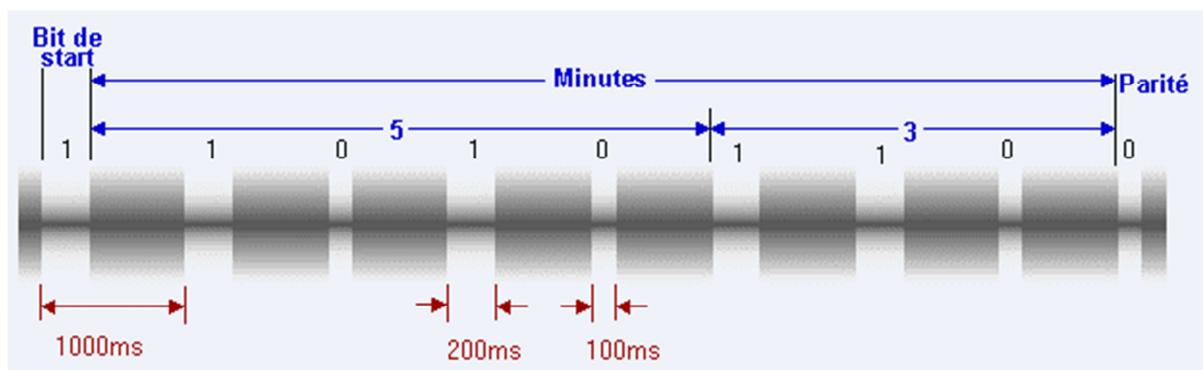
3.7 Codage de l'heure

Comme l'heure est réglée manuellement, il faut la coder, afin que les informations envoyées puissent être reçues et lues par l'horloge.

Les informations sont donc codées comme ci-dessous² :

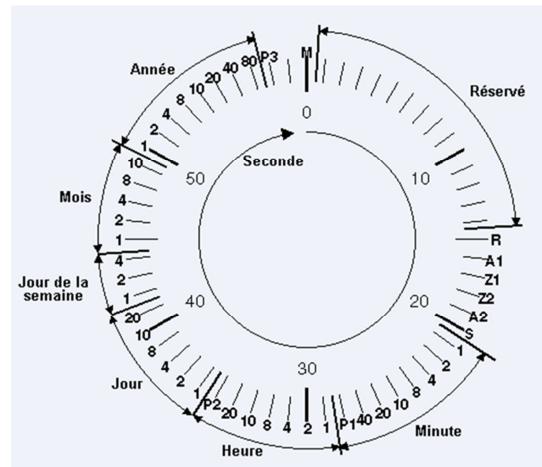
Bit	Poids	Signification	Bit	Poids	Signification	Bit	Poids	Signification
:00	1	Début de minute, toujours 0.	:20	S	Début du codage du temps, toujours 1.	:40	10	Jour du mois (suite)
:01	1		:21	1		:41	20	
:02	1		:22	2		:42	1	
:03	1		:23	4		:43	2	Jour de la semaine Lundi=1, Dimanche=7
:04	1		:24	8		:44	4	
:05	1		:25	10		:45	1	
:06	1	Témoin d'alerte civile ⁵ fourni par le <i>Bundesamt für Bevölkerungsschutz und Katastrophenwarnung.</i>	:26	20		:46	2	
:07	1	Peut contenir également des données de prévision météo ⁶	:27	40		:47	4	N° du mois 01-12
:08	1		:28	p. 1	Bit de parité paire sur les bits 21-27.	:48	8	
:09	1		:29	1		:49	10	
:10	0		:30	2		:50	1	
:11	0		:31	4		:51	2	
:12	0		:32	8		:52	4	
:13	0		:33	10		:53	8	
:14	0		:34	20		:54	10	
:15	R	Bit d'appel, permet d'alerter les employés de PTB à Braunschweig, responsables du DCF77, ou Émetteur de réserve	:35	p. 2	Bit de parité paire sur les bits 29-34.	:55	20	Année dans le siècle 00-99
:16	A1	1 = Annonce (pendant 1h) un basculement heure d'hiver/hiver au début de la prochaine heure	:36	1		:56	40	
:17	Z1	Z1 et Z2 indiquent le décalage horaire de l'heure émise par rapport au temps UTC.	:37	2		:57	80	
:18	Z2	si Z1Z2 = 01 : UTC+1h = CET = heure d'hiver si Z1Z2 = 10 : UTC+2h = CEST = heure d'été	:38	4	Jour du mois: 01-31	:58	p. 3	Bit de parité paire sur les bits 36-57.
:19	A2	1 = Annonce (pendant 1h) l'ajout d'une seconde intercalaire à la fin de l'heure (il y aura une 60 ^e impulsion supplémentaire. Le silence est reporté à la 61 ^e et dernière seconde. Puis, débute l'heure suivante.)	:39	8		:59	0	Marque de la minute : aucune modulation.

Exemple : codage de la minute 35



² <https://fr.wikipedia.org/wiki/DCF77>

On peut voir les informations contenues dans le signal d'horloge, sous une forme schématique, où l'on voit la signification des 59 bits durant une minute.



0 (M) : Début de trame (bit à 0)

1 – 14 : Réservé pour une utilisation future

15(R) : L'émetteur de réserve est actif lorsque ce bit est à 1

16 (A1) : Annonce de l'heure d'hiver

17, 18 (Z1, Z2) : Ces deux bits codent le fuseau horaire actuel

Z1	Z2	Fuseau horaire
0	1	CET (Central European Time) = UTC + 1h
1	0	CEST (Central European Sommer Time) = UTC + 2h

19 (A2) : Indique qu'une seconde va être supprimée pour corriger les irrégularités de la rotation de la terre

20 (S) : Bit de début de codage des informations horaire (toujours à 1)

21 – 27 : Minutes codées en BCD, bit de poids faible en premier

N° bit	21	22	23	24	25	26	27
Valeur	1	2	4	8	10	20	40

28(P1) : Bit de parité (parité paire) des minutes (bits 21 à 27)

29 – 34 : Heures codées en BCD, bit de poids faible en premier

N° bit	29	30	31	32	33	34
Valeur	1	2	4	8	10	20

35 (P2) : Bit de parité (parité paire) des heures (bits 29 à 34)

36 – 41 : Jour codé en BCD, bit de poids faible en premier

N° bit	36	37	38	39	40	41
Valeur	1	2	4	8	10	20

42 – 44 : Jour de la semaine codée en BCD, bit de poids faible en premier

N° bit	42	43	44
Valeur	1	2	4

45 – 49 : Mois codés en BCD, bit de poids faible en premier

N° bit	45	46	47	48	49
Valeur	1	2	4	8	10

50 -57 : Année (sur deux chiffres) codées en BCD, bit de poids faible en premier

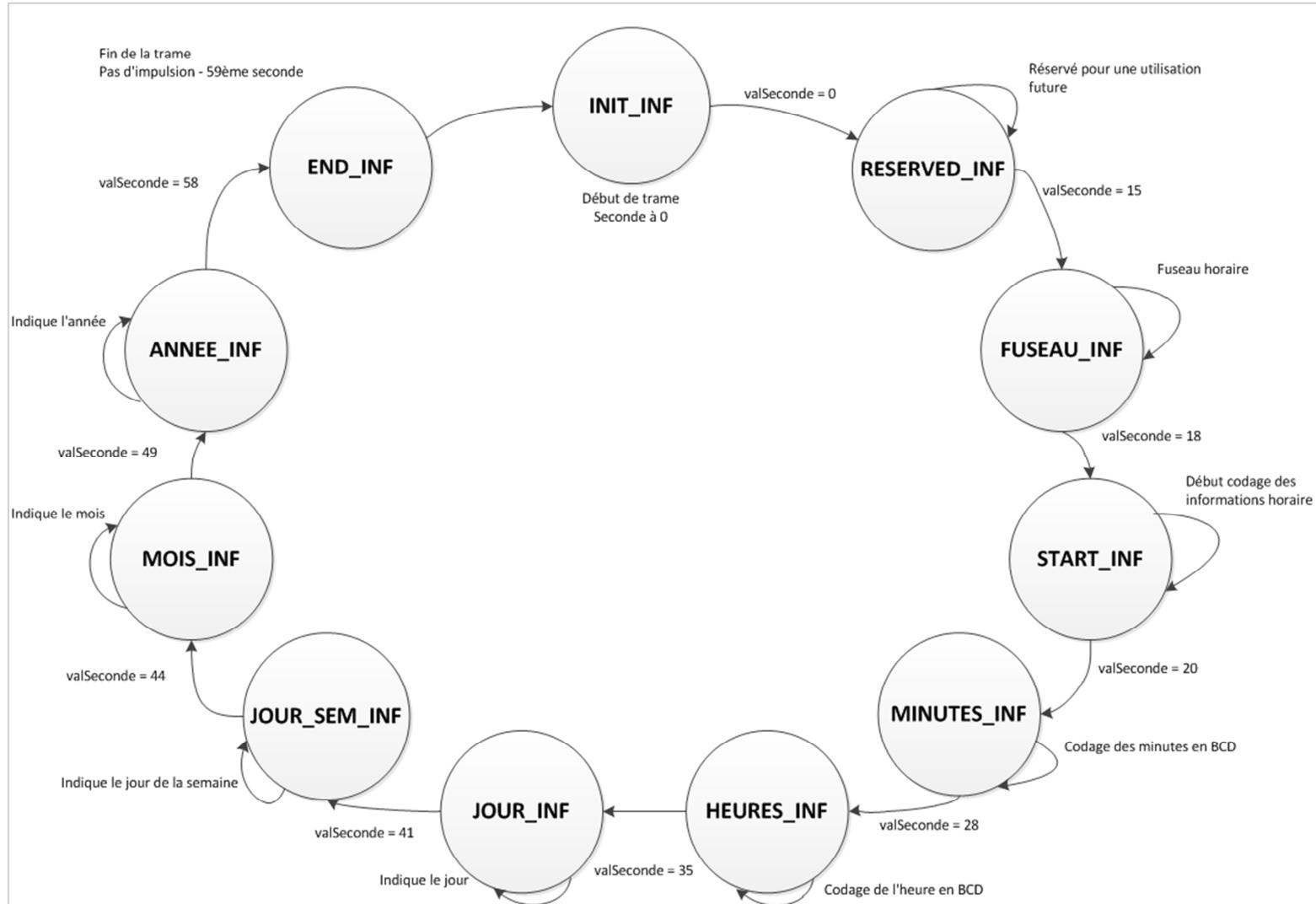
N° bit	50	51	52	53	54	55	56	57
Valeur	1	2	4	8	10	20	40	80

58 (P3) : Bit de parité (parité paire) de la date (bits 36 à 57)

59 : Pas d'impulsion

3.7.1.1 Machine d'état

Comme la trame est composée de plusieurs éléments, j'ai décidé de réaliser une machine d'état avec chacun de ceux-ci, qui sera cadencée à la seconde.



3.7.1.2 Conversion des minutes et de l'heure en BCD

Comme les valeurs de l'heure sont en décimal, il faut les convertir en BCD pour pouvoir les envoyer. J'ai ainsi créé une fonction, à laquelle j'envoie la variable contenant les minutes et l'heure :

```

535 // Fonction de conversion Décimal -> BCD
536 int8_t DecimalToBCD (int valDecimal)
537 {
538     return (((valDecimal / 10) << 4) | (valDecimal%10));
539 }
```

Chaque décimal doit être traitée individuellement, par exemple si les minutes valent « 52 », cela équivaut en BCD : 5 -> 0101 2 -> 0010 soit 0x52

En divisant, tout d'abord, par '10' cela donne le premier chiffre, puis le modulo (%) par '10' donnera le deuxième chiffre. Dès que le premier chiffre est obtenu, le décalage à gauche le déplace de 4 bits vers l'avant. L'opération 'OU' permet d'obtenir le résultat.

1. $52 / 10 = 5.2$ => Comme nous travaillons avec des entiers, seul le '5' est gardé
2. Décalage de 4 bits sur la gauche, donc 0x05 devient 0x50
3. $52 \% 10 = 2$ => Le modulo permet d'obtenir le second chiffre
4. L'opération 'OU' donne le résultat => $0x50 | 0x02 = 0x52$

3.7.1.3 Extraction des bits

Comme les valeurs des minutes et des heures sont obtenues en BCD, après la conversion, il faut à présent extraire chaque bit pour pouvoir les envoyer les uns après les autres.

```

548 //Fonction d'extraction des bits
549 void ExtractBit (int Data, int numBit)
550 {
551     int MaskValBit = 0;
552
553     MaskValBit = 1 << numBit;
554     return (Data & MaskValBit) >> numBit;
555 }
```

En premier lieu, on envoie la data, donc la valeur en BCD, puis le numéro du bit.

Si l'on reprend la valeur précédente, soit « 0x52 », qui donne 0101 0010, et qu'on envoie numBit à '2', donc on extrait le troisième bit de poids faible.

1. $0000'0001 << 2 = 0000'0100$ => Décalage en fonction du Bit
2. $0101'0010 \& 0000'0100 = 0000'0000 = 0x00$ => Opération '&'
3. $0000'0000 >> 2 = 0$ => Décalage du résultat avec le bit

L'opération est donc répétée 7 fois pour les minutes et 6 fois pour les heures.

3.7.1.4 Vérification de la parité

Comme certaines données ont une parité (paire), j'ai décidé de créer une fonction qui permet de renvoyer, via une variable booléenne, un '1' si la parité est impaire pour qu'elle devienne paire, et un '0', si celle-ci est déjà paire.

```

532     //Fonction qui permet de vérifier si la donnée est paire ou impaire
533     bool getValParite(uint8_t Data, int NbBit)
534     {
535         bool bitParite = 0;
536         int i = 0;
537
538         for(i = 0 ; i < NbBit ; i++)
539         {
540             bitParite ^= (Data & 0x01);
541             Data >>= 1;
542         }
543         return bitParite;
544     }

```

On va donc envoyer à la fonction, les datas donc les données des minutes ou de l'heure en BCD, ainsi que le nombre de bit (7 bits pour les minutes et 6 bits pour les heures).

Ensuite, la boucle 'for' va traiter les bits un par un.

La seconde opération, qui est un 'OU EXCLUSIF' va comparer le « bitParite » avec la valeur du premier, qui par la suite sera décaler de '1' sur la droite.

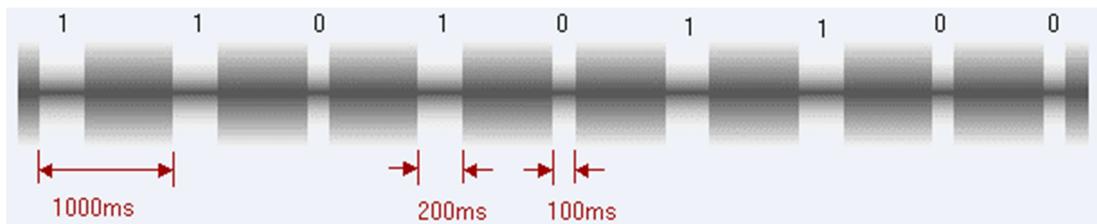
Exemple avec la valeur 0x52 : 0101 0010

Variable 'i'	Variable 'bitParite'	Data (valeur BCD)
0	0	0010'1001
1	1	0001'0100
2	1	0000'1010
3	1	0000'0101
4	0	0000'0010
5	0	0000'0001
6	1	0000'0000

On peut donc voir que la valeur BCD de 0x52 à une parité impaire (3 bits) et que la valeur renvoyée est un '1', ce qui permet d'obtenir une parité paire, comme souhaité.

3.7.2 Envoi des données

La durée d'une impulsion, détermine le niveau du bit reçu, à savoir qu'une impulsion de 100ms représente un bit à '0', et qu'une impulsion de 200ms représente un bit à '1'.



Pour ce faire, j'ai paramétré un Timer de 200ms, ainsi qu'OC, afin de gérer les durées des impulsions. De plus j'ai réalisé une fonction qui permet, selon l'état du bit, de définir la durée de l'impulsion.

(Voir point 3.6.2.1.4 Timer 3 (page 50) & point 3.6.2.1.6 OC4 (page 51))

```
525 // ***** FONCTION ENVOI DUREE BIT **** //
526 void sendBit (bool etatVal)
527 {
528     DRV_TMR2_CounterClear();
529     if(etatVal == 1)
530     {
531         //Valeur pour une impulsion de 200ms
532         PLIB_OC_Buffer16BitSet(OC_ID_4, 62500);
533     }
534     else
535     {
536         //Valeur pour une impulsion de 100ms
537         PLIB_OC_Buffer16BitSet(OC_ID_4, 31250);
538     }
539     DRV_OC1_Enable();
540 }
```

Tout d'abord, on met le compteur du Timer 3 à '0'. Puis on regarde si la valeur à envoyer vaut '0' ou '1' et on fonction de celle-ci, on attribue à l'OC4 la valeur pour obtenir la durée de l'impulsion voulue.

3.8 Travail restant à faire

- Usinage du boîtier (Boutons et micro USB)
- Dépanner la partie analogique (voir point 3.6.7.6)
- Implémenter la machine d'état, pour qu'elle soit appelée à la seconde
- Tester le code pour l'envoi des informations horaires

3.9 Conclusion

Ce module m'a permis de réaliser un projet complet, et ainsi d'être confrontée aux difficultés que cela peut représenter.

L'un des premiers problèmes que j'ai rencontrés est la partie Ethernet qui n'est pas fonctionnelle. En effet, j'ai passé beaucoup de temps à essayer de trouver d'où venait le dysfonctionnement, mais sans succès. En sachant que c'est une partie qui est reprise d'un projet fonctionnel (1630 Timbreuse RFID).

J'ai donc perdu un certain temps sur cette partie, et aurait préférer l'investir dans la partie analogique, afin de la rendre fonctionnelle.

Car en effet, lors du dimensionnement de celle-ci, je n'ai pas pensé à vérifier que l'amplificateur était capable de fonctionner, avec la fréquence que j'injecte dans le système, soit la porteuse à 77.5kHz. Il faut donc trouver un autre composant pour remplacer l'amplificateur U3, tout en portant une intention particulière au fait que celui-ci doit être capable de fonctionner à la fréquence voulue.

Le projet n'est donc pas totalement fonctionnel, étant donné que l'envoi des informations horaires n'a pas pu être testé. Mais malgré ça, la majeure partie du programme a été réalisée, et il ne reste que peu de code à implémenter, il reste essentiellement des tests à réaliser, dès que la partie analogique sera totalement opérationnelle.

Néanmoins, ce projet m'a aidé à être encore plus autonome et à m'améliorer dans divers domaines de l'électronique comme le développement, le design, la programmation, etc...

Lausanne, le 20 juin 2019

Signature :

3.10 Annexes

Annexe I : Cahier des charges

Annexe II : Schéma électrique

Annexe III : Journal de travail

Annexe IV : Planning

Annexe V : Liste de pièces

Annexe VI : Fichier des modifications

Annexe VII : Software (app.c ; app.h ; system_interrupt.c ;
Mc32DriverLcd.c ; bsp.h)

Annexe VIII : Normes euro-circuit

Annexe IX : Résumé du projet

Annexe X : Mode d'emploi

Annexe XI : Affiche