DOSSIER DE PROJET

Revue de projet n°2

Génération de tension de consigne pilotable

Electronique et communication	BONNET Nicolas
	LEROY Nicolas
Informatique et Réseaux	LANDO Gwendal

académie Aix-Marseille

Sommaire

Partie commune EC

Partie personnelle : LANDO Gwendal

Présentation du Projet

Il s'agit d'un projet développé en partenariat avec GEMALTO qui consiste à créer un équipement capable de générer des tensions de consigne destinées à régler la puissance de lasers.

GEMALTO est une société spécialisée dans la sécurité numérique. Elle propose des solutions pour aider ses clients à mettre en place des services sécurisés fondés sur les deux technologies de base que sont l'authentification et la protection.

Pour tester la fiabilité de leurs produits, les ingénieurs utilisent une technique appelée attaque par canal auxiliaire. De manière simplifiée, celle-ci consiste à observer de manière minutieuse le système afin de déterminer son fonctionnement et ainsi parvenir à dévoiler ce qu'il est censé garder secret (→ ex. : un code PIN pour une carte de téléphone portable). L'observation du système s'appuie naturellement sur l'analyse des signaux électriques mais aussi sur :

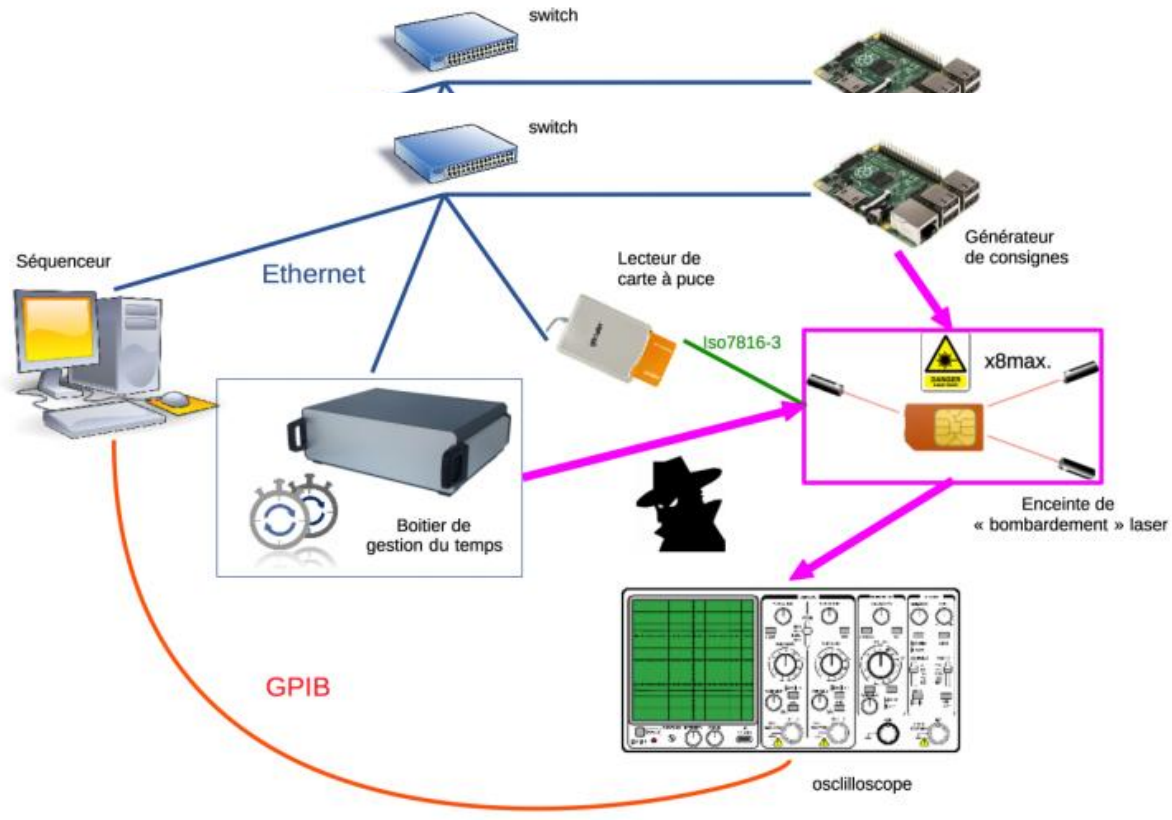
- Les variations de consommation électrique
- Les informations temporelles (ex. : durée/instants des pics de consommation, temps de réaction d'une sortie suite à un changement d'état d'une entrée…)
- Les émanations électromagnétiques voir acoustiques et/ou photoniques

L'attaque peut également se faire de manière plus invasive en perturbant le système. C'est dans ce dernier cas de figure que doit intervenir le générateur de tension de consignes dont la réalisation nous a été confiée par Pascal Moitrel, ingénieur sécurité. Le rôle du générateur de consignes est de piloter en tension la puissance de faisceaux Laser (8 au maximum) destinés à perturber le fonctionnement de la carte à puce par «bombardement» du circuit «nu» -- c'est-à-dire débarrassé de son boitier --- par des trains d'impulsions lumineuses de durée et d'intensité variables.

Situation à l'état initial

L'année dernière, les étudiants en charge du projet ont eu pour objectif de concevoir une carte de pilotage de lasers utilisable par GEMALTO pour effectuer des essais d'attaque et permettre de renforcer la sécurité des cartes à puce.
L'objectif final de ce projet était de piloter quatre sorties avec connectique SMA afin de remplacer les alimentations stabilisées utilisées jusqu'à présent par l'entreprise.

La Raspberry Pi comportait une carte d'extension accueillant quatre sorties analogiques contrôlant les lasers qui bombardent une carte à puce en vue de récupérer des informations de cette dernière. Les consignes des lasers étaient générées à partir d'une application Qt embarquée sur la Raspberry Pi qui pilotait le convertisseur numérique/analogique 8 voies (dont 4 voies étaient réellement utilisées) via le bus SPI.



Nos objectifs

Les objectifs du projet GEMALTO cette année :

- Equipement de petite taille et pilotable par Ethernet ;
- Cartes d'extension respectant le format HAT ;
- Extension capable de générer 8 tensions de consigne comprises entre 0V et 5V avec un convertisseur DAC de 12 bits ;
- 8 tensions de consigne pilotables délivrant chacune 100mA ;
- Commandes (resp. réponses) transmises à (resp. reçues de) l'équipement au format TLV ;
- Application écrite en C++ standard.

Le projet prend appui sur le projet réalisé l'an dernier afin de le poursuivre et répondre ainsi aux nouvelles exigences. Celles-ci sont :

- Pour la partie EC :

Proposer deux alternatives pour fournir les huit sorties de pilotage lasers dans un encombrement compatible avec le format HAT :

1. Association gigogne de 2 cartes identiques de 4 sorties chacune,
2. Association d'une carte ADC + amplification et d'une mezzanine disposant de 8 sorties **(une modification à fait que nous avons fait une carte de huit sorties sur cette même carte : voir page 21).**

- Pour la partie IR :

Reprendre complètement le codage du serveur TCP présent dans le générateur de consignes pour recevoir les commandes de pilotage depuis un client réseau (le Séquenceur).
Celui-ci devra :

1. Posséder une architecture orientée objet ;
2. Prendre en charge les commandes du cahier des charges ;
3. Permettre de piloter, via bus SPI, les huit sorties Laser quelle que soit la version de la carte d'extension utilisée (1x8 voies ou 2x4voies)

Changements à apporter

Certains changements sont à prévoir en raison des nouveaux objectifs fixés.

- Proposer deux systèmes différents pour l'ajout de quatre sorties supplémentaires sur la carte Gemalto 2017.
- Réduction d'une pastille sur la carte pour pouvoir faire passer une piste pour l'alimentation plus gros.
- Amélioration du plan de masse pour qu'il soit partout sur la carte afin de limiter l'effet condensateur.
- Changement du bloc d'alimentation par un bloc d'alimentation extérieur (de 7.5V) car les sorties demandent 100mA par sortie. Etant donné qu'il y a huit sorties il faut délivrer 800mA.
- Ajout d'un fusible dans la partie alimentation.
- Ajout de deux résistances de tirage pour l'EEPROM I2C.
- Relier les broches ID_SC et ID_SD à l'EEPROM.

Contraintes de réalisation & ressource

Contrainte de réalisation :

Contraintes de réalisation Contraintes financières (budget alloué) : Environ 200€.
Gemalto participe au financement des composants.

Contraintes de développement (matériel et/ou logiciel imposé / technologies utilisées) :
La spécification, conception et codage seront modélisés.

Contraintes qualité (conformité, délais, …) :
Maintenable, maniable (ergonomie).

Contraintes de fiabilité, sécurité :
Les accès logiciels seront sécurisés.

Ressources mises à disposition (logiciels / matériels / documents) :

Matériels :
Une carte RASPBERRY PI 2 B+ par étudiant.
Composants électroniques pour la réalisation.
Matériel de laboratoire (alimentation, oscilloscope, analyseur logique).

Logiciels :
Système d'exploitation Linux (Raspbian).
Logiciel de conception électronique : Kicad.
Logiciel de modélisation SysML/UML : MagicDraw v7.02.
SDK C/C++ standard POSIX (→ application serveur de génération de consignes).

Documents :
Les différentes documentations mises à disposition sur le site de la section de BTS SN.

Répartition des tâches par étudiant

Étudiant 1 **IR**	*Liste des tâches assurées par l'étudiant* • Concevoir/Coder/Tester librairies C++ de codage/décodage de trames TLV • Concevoir/Coder/Tester librairie pour s'interfacer par SPI avec la/les carte(s) HAT • Concevoir/Coder/Tester serveur TCP de génération de consigne en respectant une architecture orientée objet • Proposer une solution pour émuler le séquenceur : codage d'une application utilisation d'un logiciel existant, utilisation de commandes linux (ex. netcat)	**Installation** : Système Raspbian, SDK C++ **Mise en œuvre :** Langage C++ **Posix**, **Configuration** : **Réalisation** : Serveur TCP génération de consignes **Documentation** : Guide d'installation, manuel utilisateur, dossier de développement
Étudiant 2 **EC**	*Liste des tâches assurées par l'étudiant* • Analyser le schéma structurel produit par les étudiants de la promotion 2017, corriger les erreurs et l'adapter aux nouvelles consignes • Concevoir/Réaliser/Tester une carte d'extension RPi disposant de 4 sorties, conforme à la spécification HAT et pouvant être associée (montage gigogne) à une seconde carte identique pour pouvoir piloter au total jusqu'à 8 lasers. • Participer à la conception/au codage/au test d'une librairie de pilotage des cartes. • Concevoir/Coder/Tester une interface graphique permettant de vérifier à minima le bon fonctionnement des sorties analogiques	**Installation** : Librairies C/C++, BCM2835, framework Qt **Mise en œuvre :** BCM2835, C/C++ Qt **Configuration :** **Réalisation :** Circuit imprimé. Les documents de fabrication seront réalisés avec Kicad (logiciel utilisé par l'entreprise commanditaire). **Documentation :** Installation, prototypage/mise au point, documents de fabrication (pour sous-traitance industrielle des PCB) et programmes commentés
Étudiant 3 **EC**	*Liste des tâches assurées par l'étudiant* • Analyser le schéma structurel produit par les étudiants de la promotion 2017, corriger les erreurs et l'adapter aux nouvelles consignes. • Concevoir/Réaliser/Tester 2 cartes d'extension RPi, l'une conforme à la spécification HAT disposant du DAC *(unique)*, et l'autre montée sur la précédente *(association gigogne et/ou mezzanine)* disposant de la connectique permettant de pouvoir piloter jusqu'à 8 lasers. La répartition des composants *(autres que DAC et connecteurs SMA)* et la connectique de liaison des 2 cartes sont à l'initiative de l'étudiant. • Participer à la conception/au codage/au test d'une librairie de pilotage des cartes. • Concevoir/Coder/Tester une interface graphique permettant de vérifier à minima le bon fonctionnement des sorties analogiques	**Installation** : Librairies C/C++, BCM2835, framework Qt **Mise en œuvre :** BCM2835, C/C++ Qt **Configuration :** **Réalisation :** Circuit imprimé. Les documents de fabrication seront réalisés avec Kicad (logiciel utilisé par l'entreprise commanditaire). **Documentation :** Installation, prototypage/mise au point, documents de fabrication (pour sous-traitance industrielle des PCB) et programmes commentés
Tous les étudiants	✓ *Tâches à traiter par l'ensemble des étudiants de l'équipe projet pour le développement de la solution* • Définir **au plus tôt** les interfaces entre les tâches individuelles. Ex. : protocoles d'échange, valeurs délivrées (format, précision, fréquence,…) • Documentation globale. Ex. : documentation pour le déploiement de la solution (→ guide de mise en route rapide) ✓ *Domaines de physique à traiter par l'ensemble des étudiants de l'équipe projet :* • Ondes électromagnétiques ; dualité onde-corpuscule • Traitement des signaux analogiques • Numérisation	

Partie personnelle :
Nicolas Bonnet

GEMALTO

Présentation de ma partie personnelle

Ma partie du projet consiste à créer deux cartes d'extension Rpi, l'une conforme à la spécification HAT disposant du DAC et une seconde carte montée sur la précédente (association gigogne et/ou mezzanine) disposant de la connectique permettant de pouvoir piloter jusqu'à huit lasers. Une bonne partie de mon travail fût réalisé avec Nicolas Leroy car nous avions le même objectif à atteindre mais de façon différente, donc la partie étude de la carte de l'an dernier ainsi que l'apprentissage de Kicad et le choix des composants s'est fait à deux.

Le principe de fonctionnement est le même que pour la carte de l'année dernière, la différence majeure est le nombre de sorties.

A la demande de M.Moitrel, tous les schémas et routage sont faits avec le logiciel kicad.

Planification prévisionnelle

Le travail à effectuer se découpe de la manière suivante :

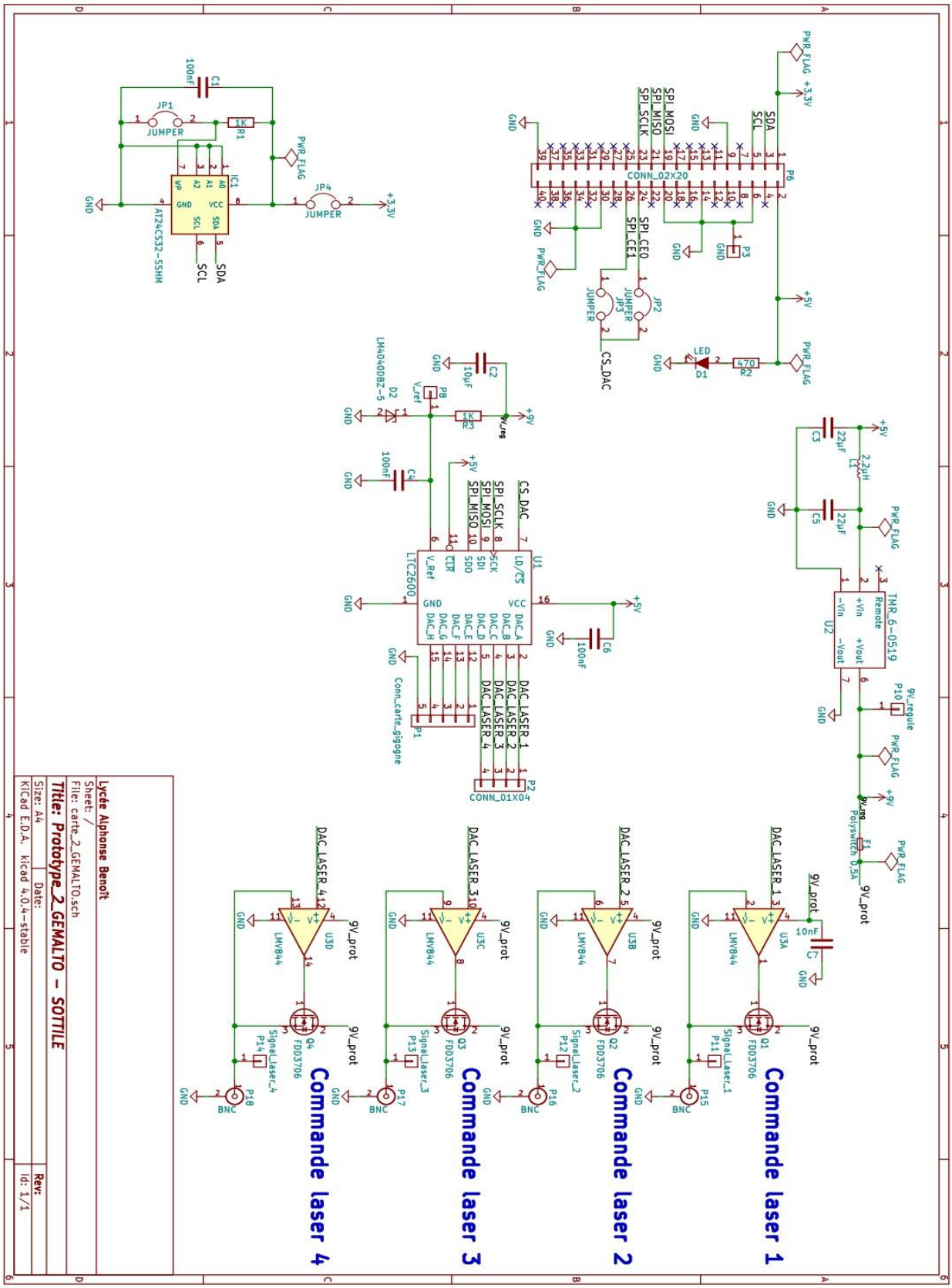
Etude de l'ancienne carte puis conception de la carte et test.
L'étude de l'ancienne carte s'est faite avec Nicolas Leroy car nous avons le même but à atteindre mais nous devions le faire de manière différente.

Diagramme de Gantt :

	Tâches BN	181 hr	Ven 12/01/1	Ven 11/05/1	
	Apprentissage logiciel Kicad	6 hr	Ven 12/01/18	Jeu 18/01/18	7
	Etude des composant à utiliser	28 hr	Jeu 18/01/18	Mer 31/01/18	22
	Etude de l'ancienne carte	40 hr	Mer 31/01/18	Ven 16/02/18	23
	Etude du AT	10 hr	Ven 16/02/18	Ven 23/02/18	24
	Réalisation du schéma avec kicad	25 hr	Mer 14/03/18	Jeu 22/03/18	25
	Réalisation de la carte	30 hr	Jeu 22/03/18	Jeu 05/04/18	26
	Participer à la conception/le codage/au test d'une librairie de pilotage des cartes	20 hr	Jeu 05/04/18	Ven 13/04/18	27
	Concevoir/coder/tester une interface graphique permettant de verifier à minima le bon fonctionnement des sorties analogique	22 hr	Mer 18/04/18	Ven 11/05/18	28

Cette planification fût réalisée au début du projet et a pour but de prévoir les tâches à venir et estimer leur durée pendant le projet.

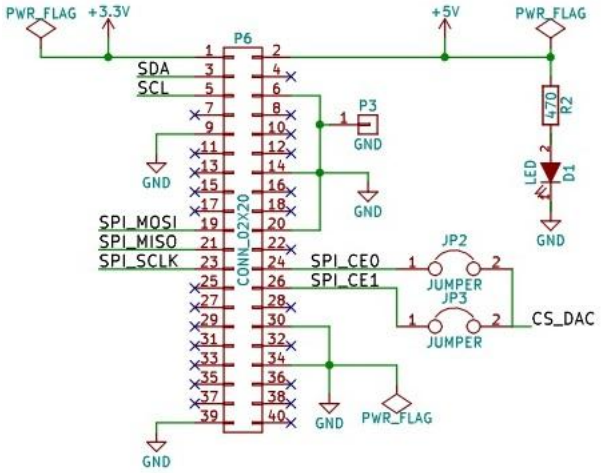
Etude de fonctionnement de la carte de 2017



Le parcours de la donnée dans la carte.

Le séquenceur envoi une commande à la Raspberry via un câble Ethernet afin de pouvoir piloter une des huit sorties de l'extension.

- La première étape est donc le GPIO qui est la liaison entre la carte Raspberry et l'extension.

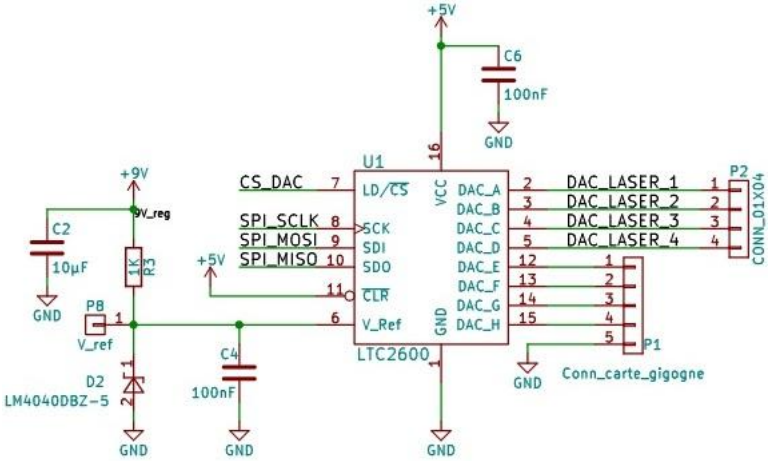


La donnée arrive donc par **les bus SPI sur les broches 19 21 23.**

Elle se dirige ensuite soit **sur SPI_CE0 ou SPI_CE1 (broche 24 et 26)**, les jumpers sont là pour **faire la sélection d'une des deux sorties**, une sortie correspond à une extension dans le cas ou plusieurs cartes sont l'une sur l'autre.

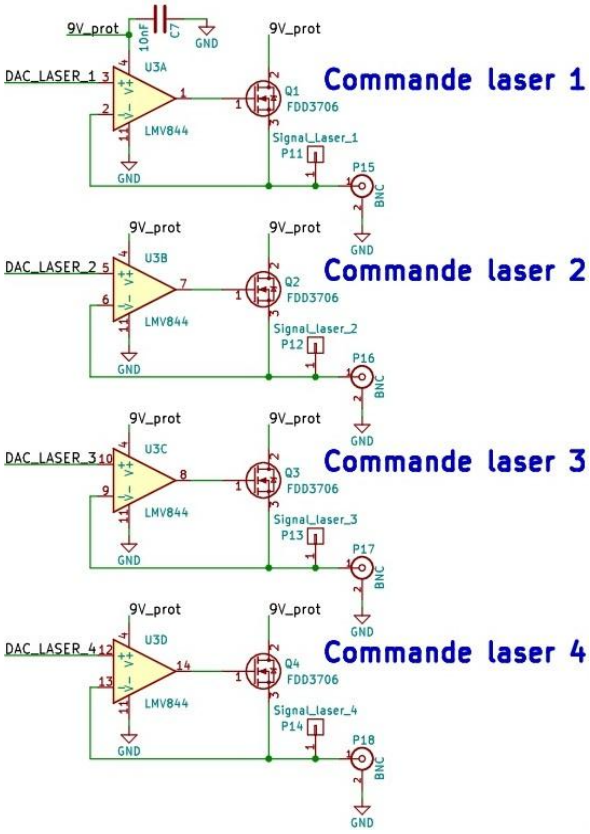
Les broches SDA et SCL permettent de relier l'EEPROM.

- Ensuite la donnée passe par le microcontrôleur.



La donnée rentre ensuite dans le microcontrôleur via l'entrée CS_DAC donc soit par SPI_CE0 ou SPI_CE1 du GPIO celons la carte sélectionnée.
Celons la valeur de la trame envoyée, la donnée passe par la sortie correspondant soit DAC_LASER_1/2/3/4.
(Voir l'analyse de la trame page 16).

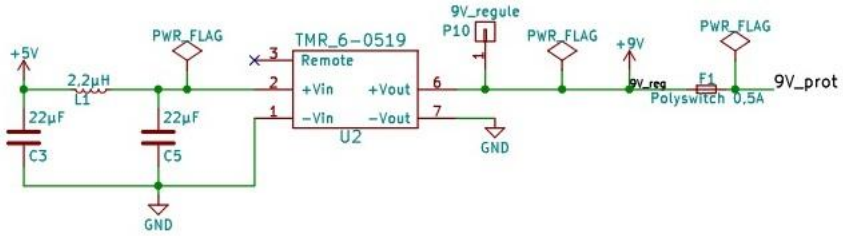
- Enfin la donnée passe dans l'ensemble Amplificateur/transistor qui lui correspond.



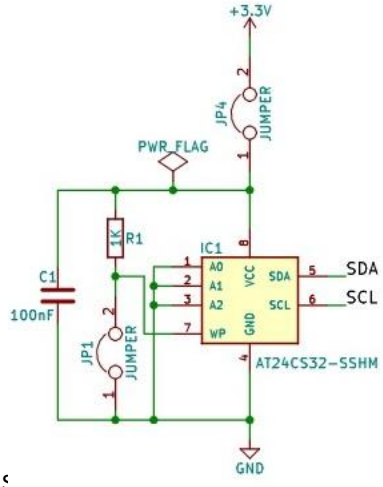
La donnée fini son trajet par l'ensemble amplificateur/transistor

L'analyse du fonctionnement se trouve dans la partie commune EC à la page 69

- Alimentation externe pour le pilotage laser



- La mémoire



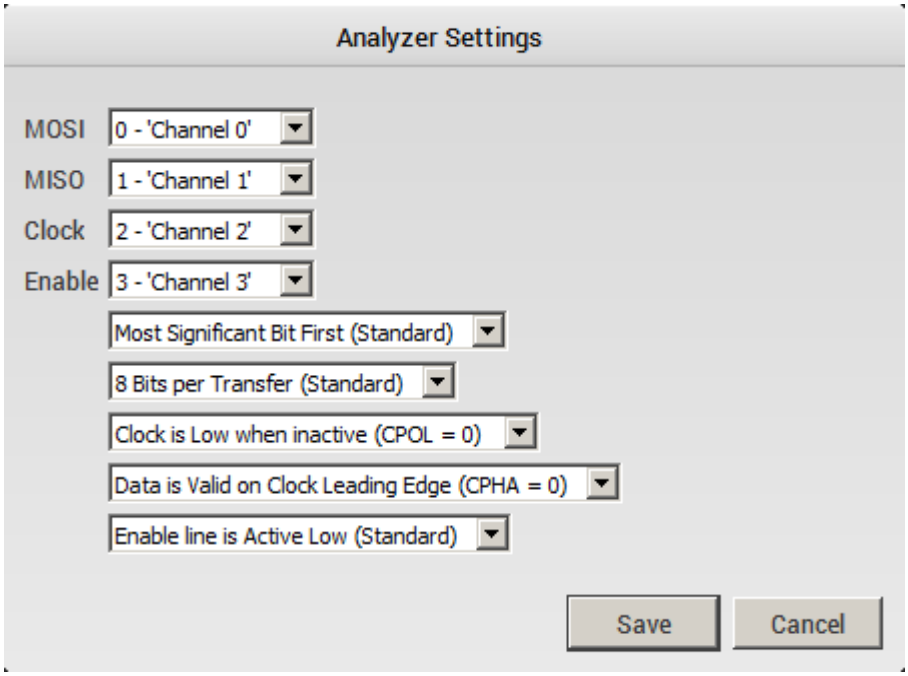
Nicolas

Analyse avec Salae

On a branché Saleae à la carte réalisée l'année dernière sur les ports SPI_MOSI (channel 0), SPI_MISO (channel 1), SPI_CLK (channel 2), SPI_CE (channel 3). La fréquence du signal de sortie provenant de l'analyseur logique doit être à 3.0625 MHz pour la Raspberry Pi2 et 6.250MHz pour la Raspberry Pi3 car Saleae ne parvient pas à relever les trames correctement au-delà de cette fréquence (ex : 7, 15MHz et 31MHz).

La sortie de la carte est reliée à l'analyseur logique.

On a configuré le bus SPI comme tel sur Saleae :



Nous procédons donc à l'analyse des trames.

Grâce au logiciel QT nous pouvons créer plusieurs signaux de sortie différent afin de créer plusieurs cas de figure (signal carré, triangle et en dents de scie).

Principe de fonctionnement :

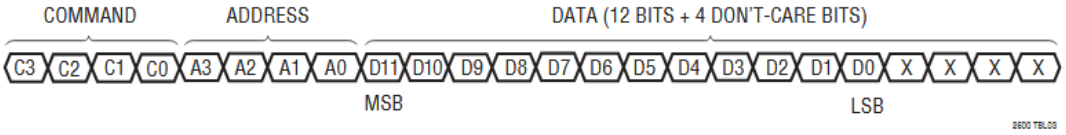
Table 1.

COMMAND*				
C3	C2	C1	C0	
0	0	0	0	Write to Input Register n
0	0	0	1	Update (Power Up) DAC Register n
0	0	1	0	Write to Input Register n, Update (Power Up) All n
0	0	1	1	Write to and Update (Power Up) n
0	1	0	0	Power Down n
1	1	1	1	No Operation

*Command and address codes not shown are reserved and should not be used.

ADDRESS (n)*				
A3	A2	A1	A0	
0	0	0	0	DAC A
0	0	0	1	DAC B
0	0	1	0	DAC C
0	0	1	1	DAC D
0	1	0	0	DAC E
0	1	0	1	DAC F
0	1	1	0	DAC G
0	1	1	1	DAC H
1	1	1	1	All DACs

INPUT WORD (LTC2620)



Les huit premiers bits sont les bits de commande et d'adresse répartis tous deux en quatre bits chacun.

Les quatre premiers correspondent donc aux bits de commande, pour notre cas la valeur de la commande sera 3 : Write to and Update (Power Up) n. La valeur des bits C0, C1, C2 et C3 donne un nombre compris entre zéro et quatre et aussi quinze en hexadécimal, chaque valeur correspond à une commande.

Ensuite, viennent les quatre bits d'adresse, comme on peut voir sur le tableau ci-dessus chaque DAC (correspondant aux sorties) à une adresse qui lui est propre allant de 0 à 8.

Les douze premiers bits de donnée correspondent à la valeur V_{min} ou V_{max}, les quatre derniers bits (Don't care bits) ne sont pas utilisés.

Exemple, si on a V_{min} à 4V et V_{max} à 5V.

4V correspond à la suite 0xCC 0xC0

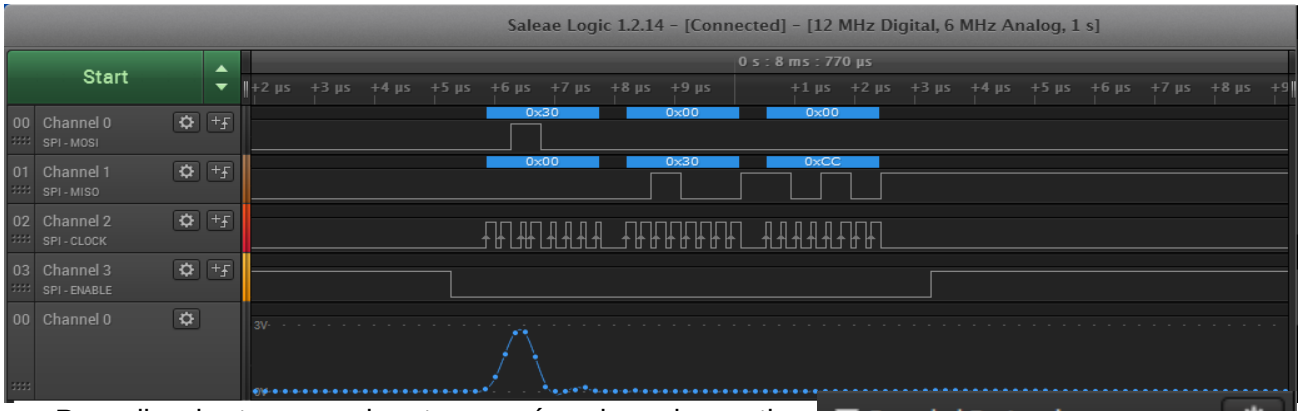
$(CCC)_{16} = (3276)_{10} \rightarrow 3276*(5/4096) = 4V$

5V correspond à la suite 0xFF 0xF0

$(FFF)_{16} = (4095)_{10} \rightarrow 4095*(5/4096) = 5V$.

Pour la suite de l'analyse nous avons pris plusieurs cas de figure.

- Signal carré – résolution 12 bits – DAC A – Vmin 0V, Vmax 4V – TLow 10 ms, THigh 15ms.



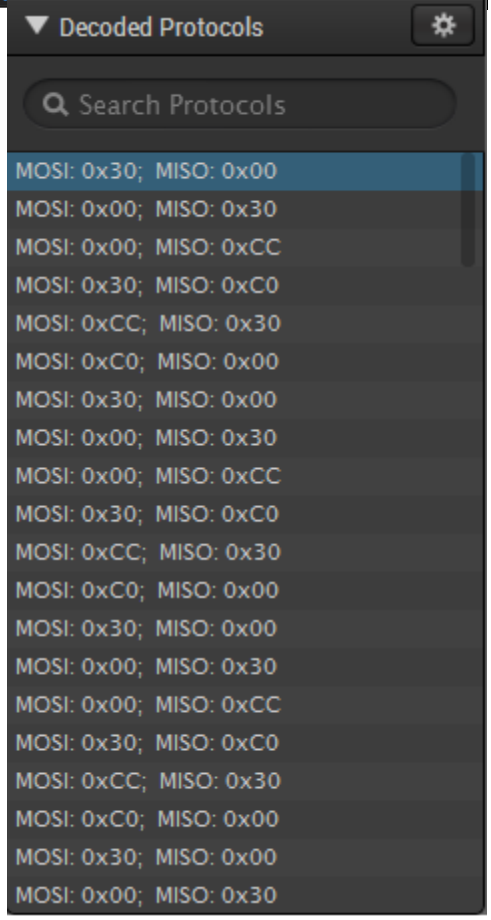
Pour lire la trame qui est envoyée, dans la partie « decoded protocole» **on s'intéresse à la partie MOSI**. On peut voir ici que les trois premiers octets en hexadécimale envoyé sont **0x30 0x00 0x00**.

Donc la commande est bien « Write to and Uptdate (Power Up) n » qui correspond au « 3 » et il est suivit de « 0 » qui équivaut au DAC A, ce qui donne « 0x30 ».

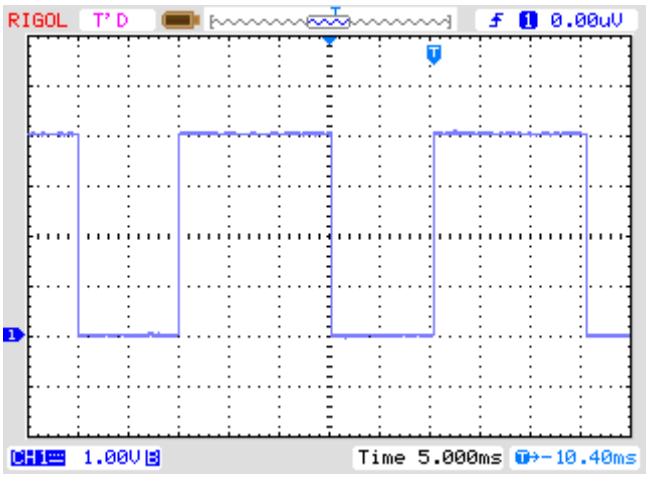
0x30 est suivit de 0x00 0x00 qui correspond à V_{min} étant donné que V_{min} est à 0V.

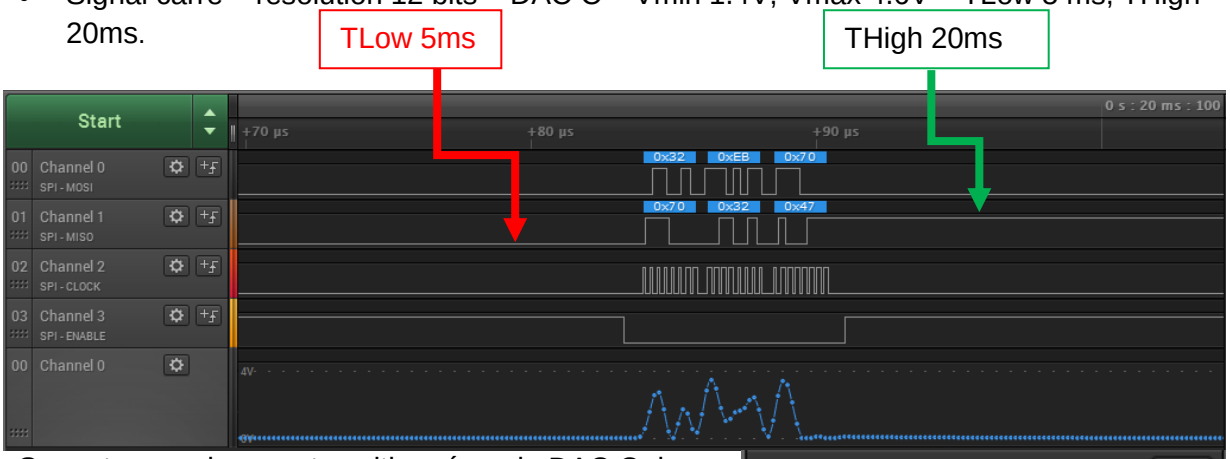
Ensuite, de nouveau il est affiché 0x30 correspondant au DAC A mais cette fois il est suivit de 0xCC 0xC0, on a vue au dessus que c'est égal à 4V.

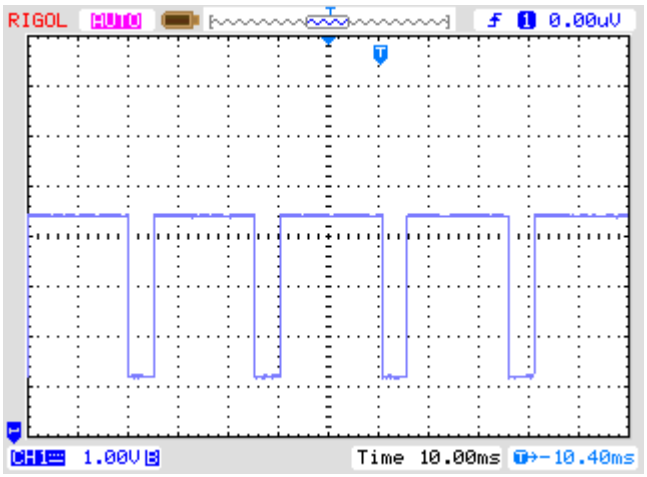
Etant donné que le signal envoyé est carré, la trame alterne tout le temps entre |0x30 et |0x30
|0x00 |0xCC
|0x00 |0xC0

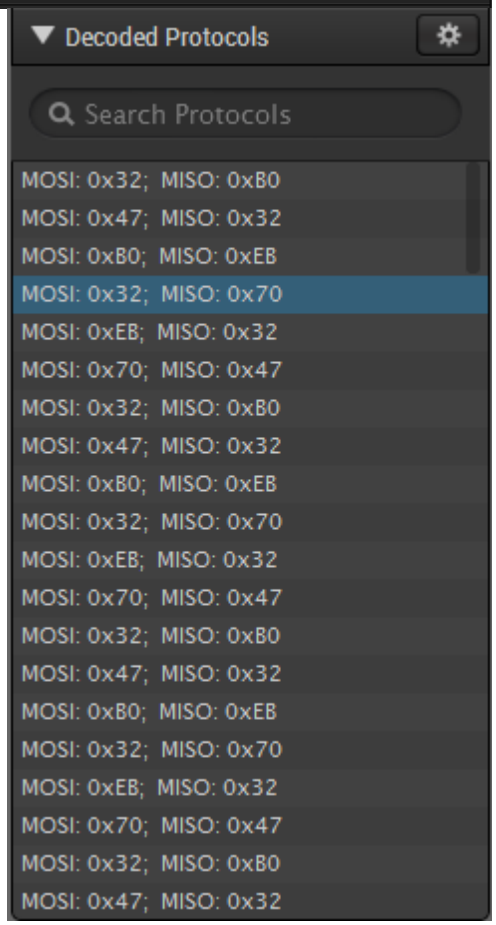


MOSI: 0x30; MISO: 0x00
MOSI: 0x00; MISO: 0x30
MOSI: 0x00; MISO: 0xCC
MOSI: 0x30; MISO: 0xC0
MOSI: 0xCC; MISO: 0x30
MOSI: 0xC0; MISO: 0x00
MOSI: 0x30; MISO: 0x00
MOSI: 0x00; MISO: 0x30
MOSI: 0x00; MISO: 0xCC
MOSI: 0x30; MISO: 0xC0
MOSI: 0xCC; MISO: 0x30
MOSI: 0xC0; MISO: 0x00
MOSI: 0x30; MISO: 0x00
MOSI: 0x00; MISO: 0x30
MOSI: 0x00; MISO: 0xCC
MOSI: 0x30; MISO: 0xC0
MOSI: 0xCC; MISO: 0x30
MOSI: 0xC0; MISO: 0x00
MOSI: 0x30; MISO: 0x00
MOSI: 0x00; MISO: 0x30









- Signal carré – résolution 12 bits – DAC C – Vmin 1.4V, Vmax 4.6V – TLow 5 ms, THigh 20ms.

Sur cet exemple on est positionné sur le DAC C donc le premier octet est « 0x32 »

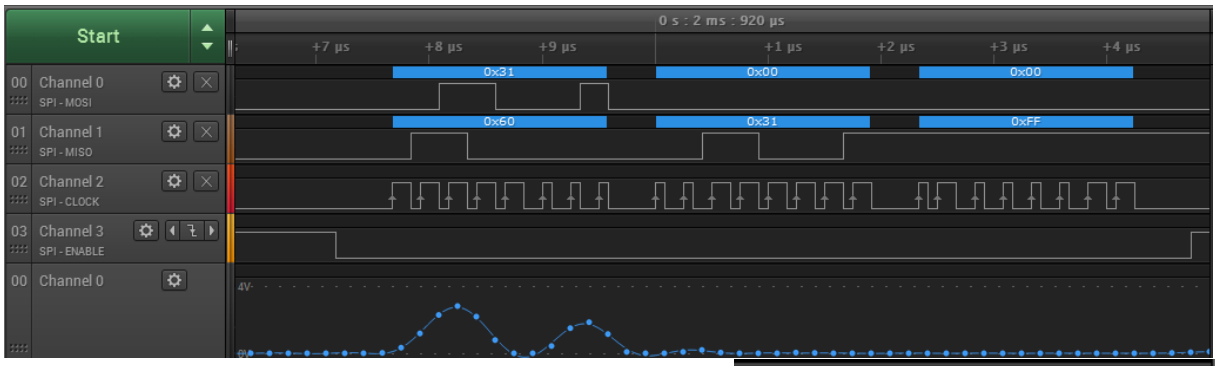
Vmin : $(47B)_{16} = (1147)_{10}$
1147*(5/4096) = 1.4V.

Vmax : $(EB7)_{16} = (3767)_{10}$
3767*(5/4096) = 4.6V.

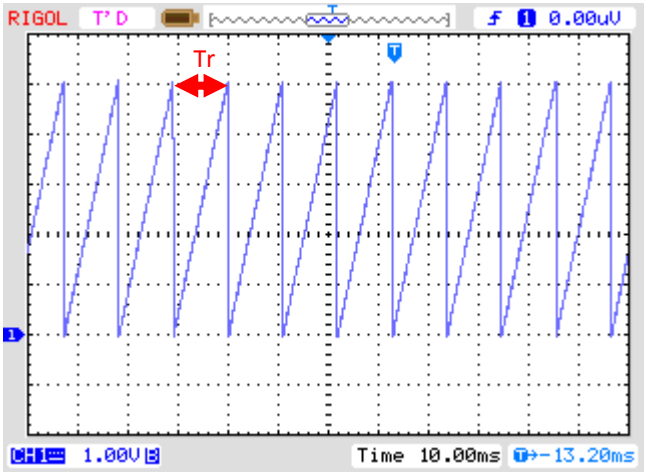
Tout est OK.

```
MOSI: 0x32;  MISO: 0xB0
MOSI: 0x47;  MISO: 0x32
MOSI: 0xB0;  MISO: 0xEB
MOSI: 0x32;  MISO: 0x70
MOSI: 0xEB;  MISO: 0x32
MOSI: 0x70;  MISO: 0x47
MOSI: 0x32;  MISO: 0xB0
MOSI: 0x47;  MISO: 0x32
MOSI: 0xB0;  MISO: 0xEB
MOSI: 0x32;  MISO: 0x70
MOSI: 0xEB;  MISO: 0x32
MOSI: 0x70;  MISO: 0x47
MOSI: 0x32;  MISO: 0xB0
MOSI: 0x47;  MISO: 0x32
MOSI: 0xB0;  MISO: 0xEB
MOSI: 0x32;  MISO: 0x70
MOSI: 0xEB;  MISO: 0x32
MOSI: 0x70;  MISO: 0x47
MOSI: 0x32;  MISO: 0xB0
MOSI: 0x47;  MISO: 0x32
```

- Signal en dent de scie – résolution 12 bits – DAC B – Vmin 0V, Vmax 5V – Tr(ms)= 10



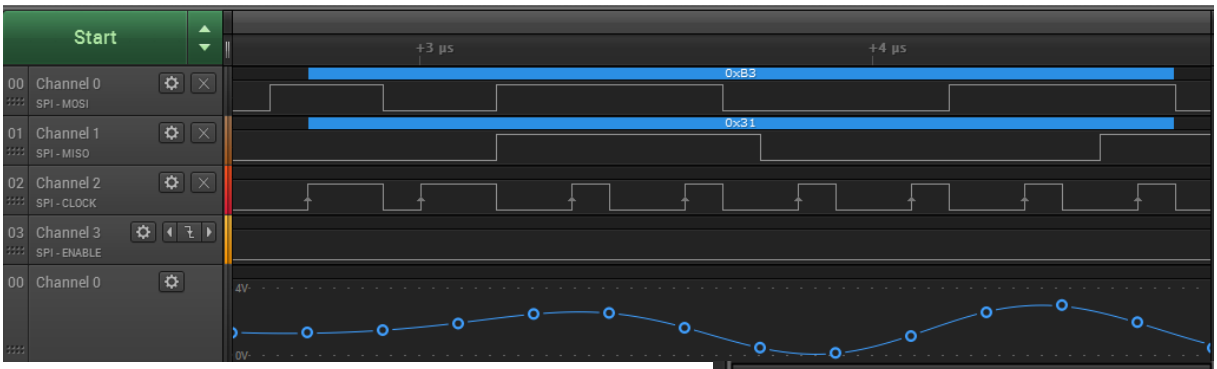
Les valeurs des octets de commande et d'adresse restent inchangés « 3 » pour la commande et « 1 » pour le DAC B



Le signal est en dent de scie, donc la valeur des 12 bits de data augmente petit à petit, dans ce cas la de 0 à 5V soit de |0x00 0x00| à |0XFF 0xF0|.

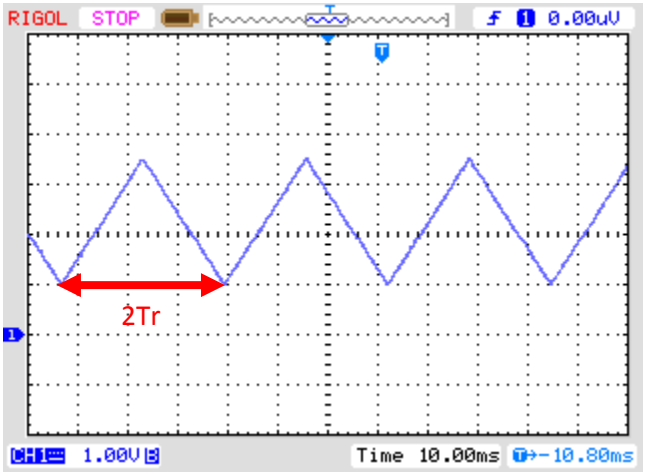
Tr = 10ms est le temps que le signal passe de 0V à 5V

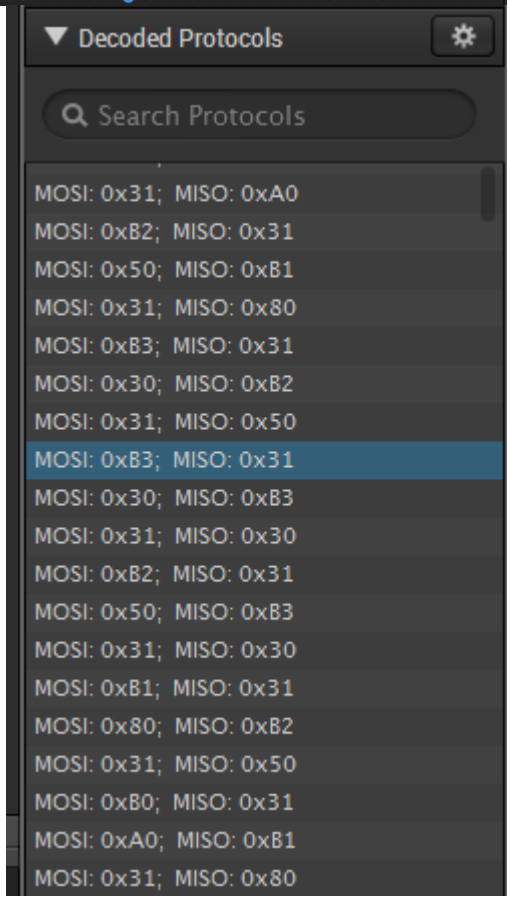
- Signal triangle – résolution 12 bits – DAC B – Vmin 1V, Vmax 3.5V – Tr(ms)= 15



Les valeurs des octets de commande et d'adresse restent inchangés « 3 » pour la commande et « 1 » pour le DAC B

Tr = 15ms est le temps que le signal met pour passer de 1V à 3.5V et pour passer de 3.5V à 1V. Donc il faut au signal 30ms pour partir de 1V, monter à 3.5V puis revenir à 1V soit 2Tr.



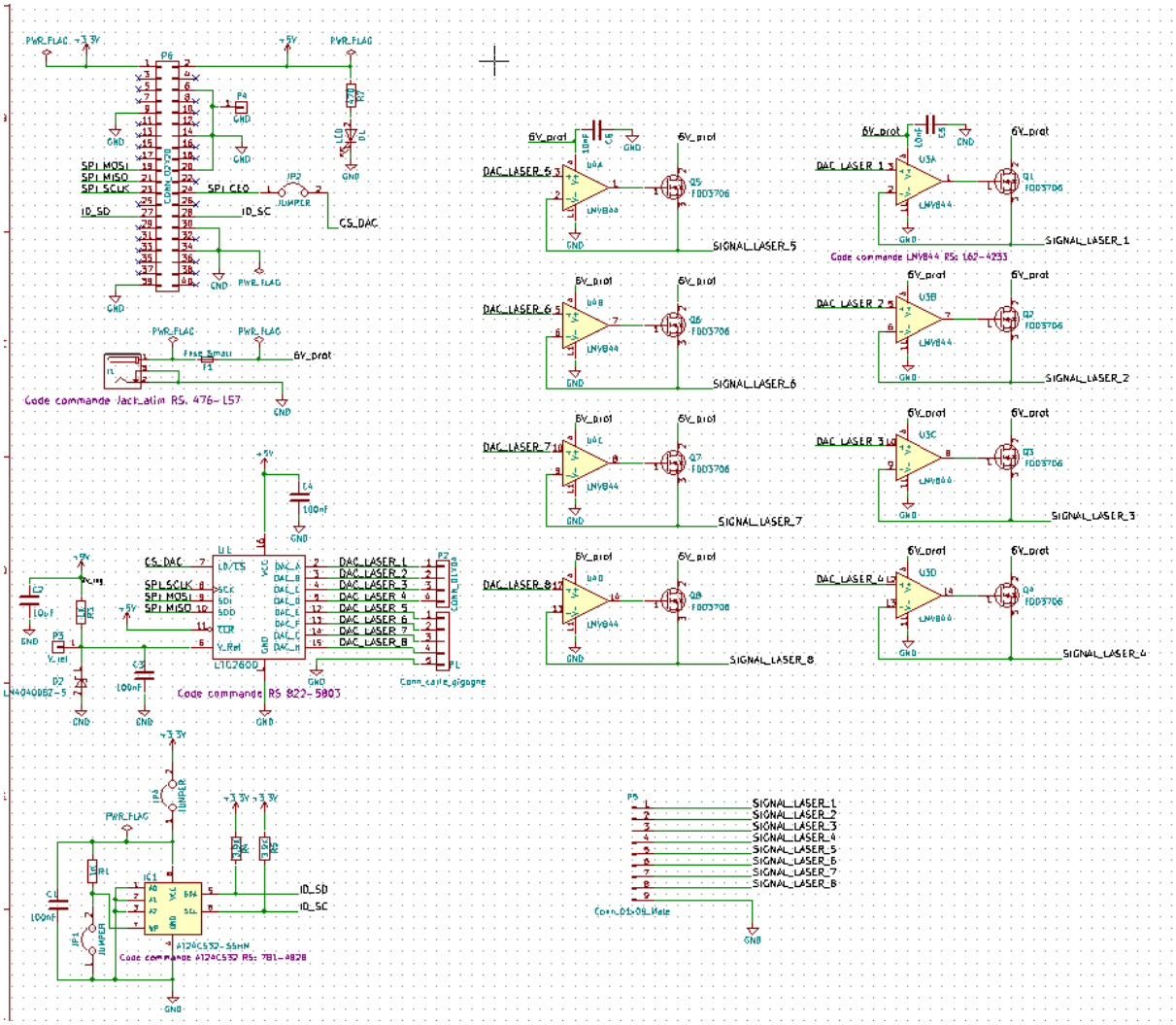


Pour décoder la trame il suffit de suivre le même procédé que le signal en dent scie, la différence avec le signal en dent de scie est qu'il ne repasse pas brutalement à 1V, il descend progressivement formant ainsi un signal triangulaire.

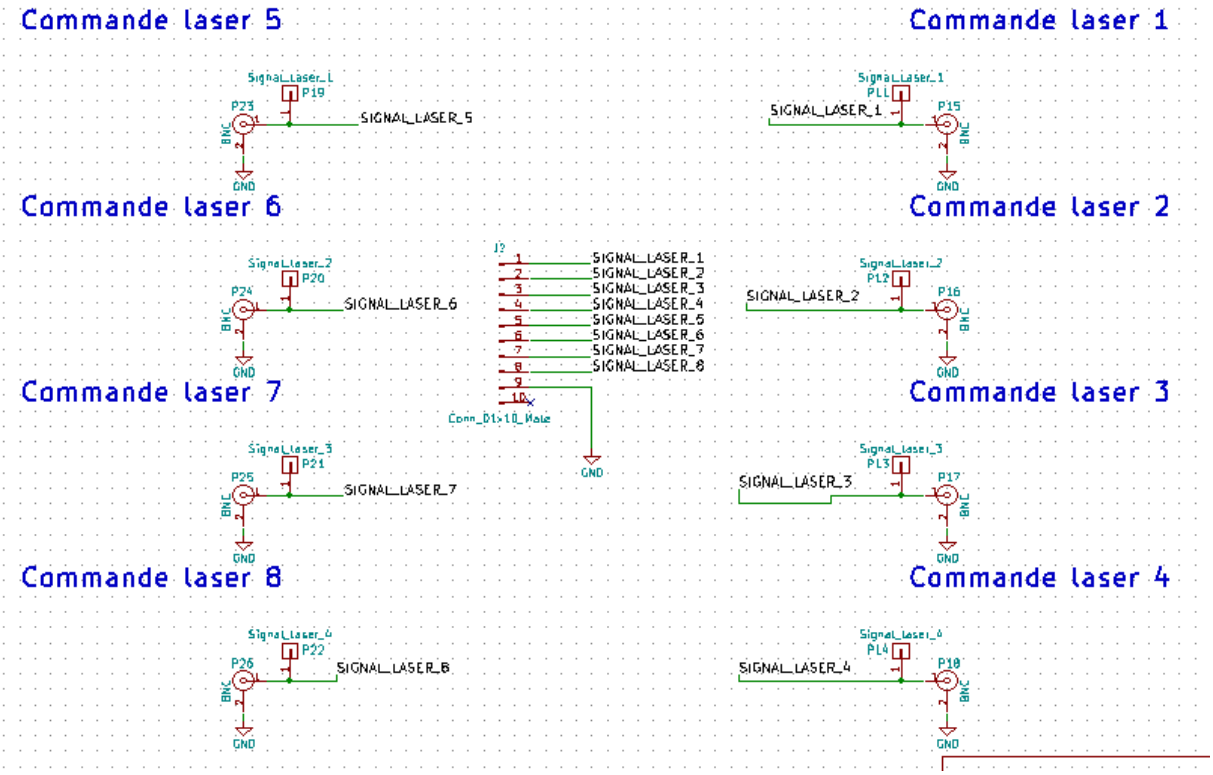
Schéma structurel

Après la prise en main de Kicad et l'étude de la carte de 2017, j'ai commencé mon projet et j'ai réalisé les deux schémas structurels de ma carte qui sont les suivants :

Schéma structurel de la carte disposant du DAC :



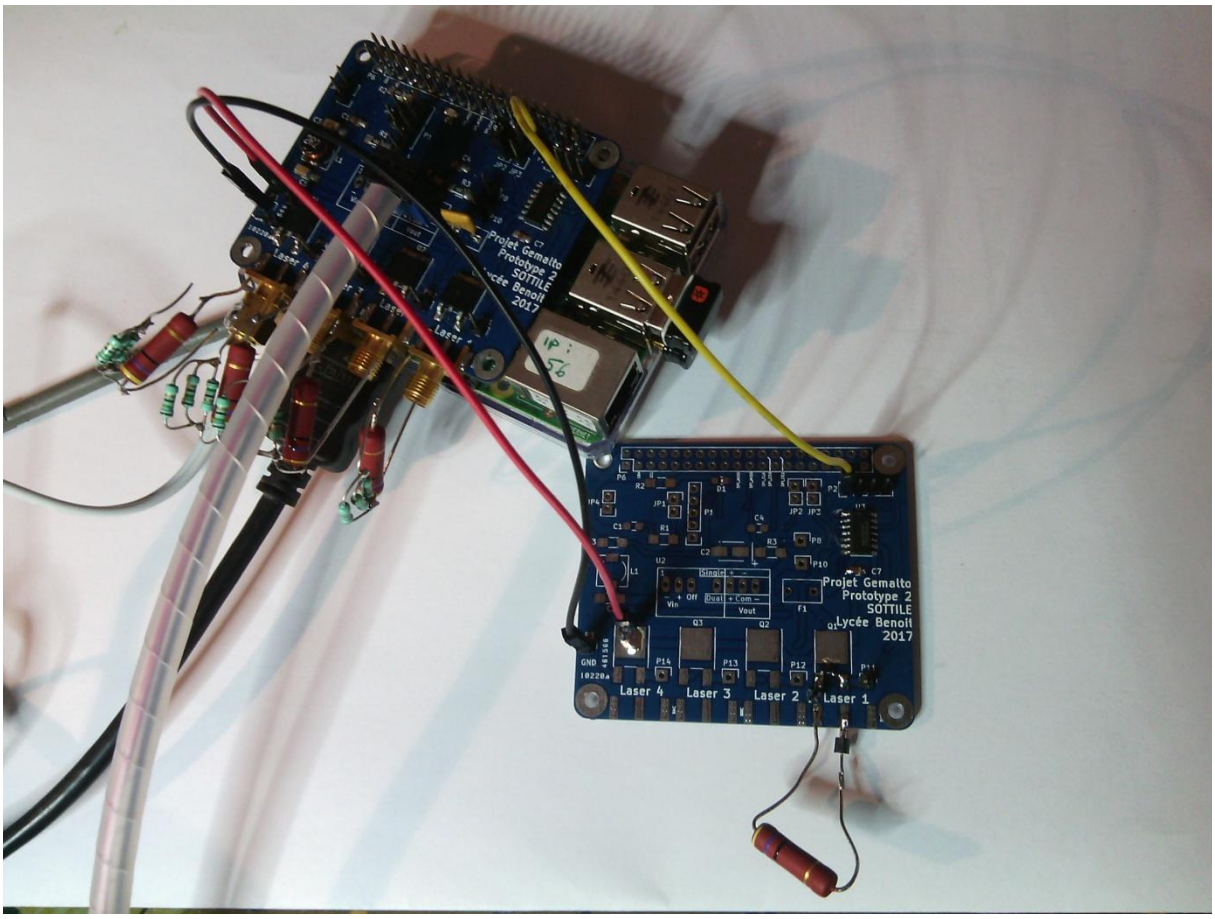
Je suis parti de la même base que le schéma structurel de l'année dernière en y apportant les modifications nécessaires pour piloter huit sorties et le relier à la seconde carte où se trouvent les sorties.



Le schéma structurelle de la seconde carte qui sera par-dessus la première et qui a pour but d'avoir les huit sorties SMA.

Changement apporté

A la demande de Monsieur Moitrel, nous avons pris un nouveau transistor moins surdimensionné, un test d'un nouveau transistor a été réalisé, le transistor MGSF2N02ELT1G. Le test à était réalisé en câblant au minimum (LMV844 + Transistor) une 2nde carte quatre lasers de l'année dernière. Le test s'est avéré bon.



Ce transistor a donc était choisi suite à ce test.

Caractéristique :

Type de canal : N

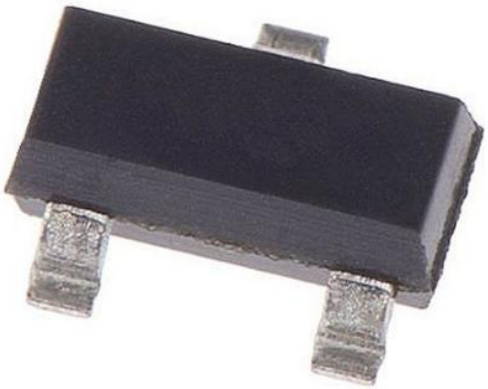
Courant maximum : 2.8A

Tension Drain : 20V

Résistance Drain 115mΩ

Type de montage : CMS

Dimension : 3.04 x 1.4 x 1.01mm



Ce transistor est **largement plus petit** que celui sélectionné auparavant et donc **apporte un gros changement pour ma partie du projet**. L'ancien transistor prenait beaucoup de place sur la carte et surdimensionné.

Après proposition de monsieur Moitrel et compte-tenu du gain de place, **la solution serait de ne réaliser qu'une seule carte et prendre exemple de la carte Redpitaya** disposant d'une partie arrondie et plus grande que le format HAT sur la longueur.

La carte Redpitaya

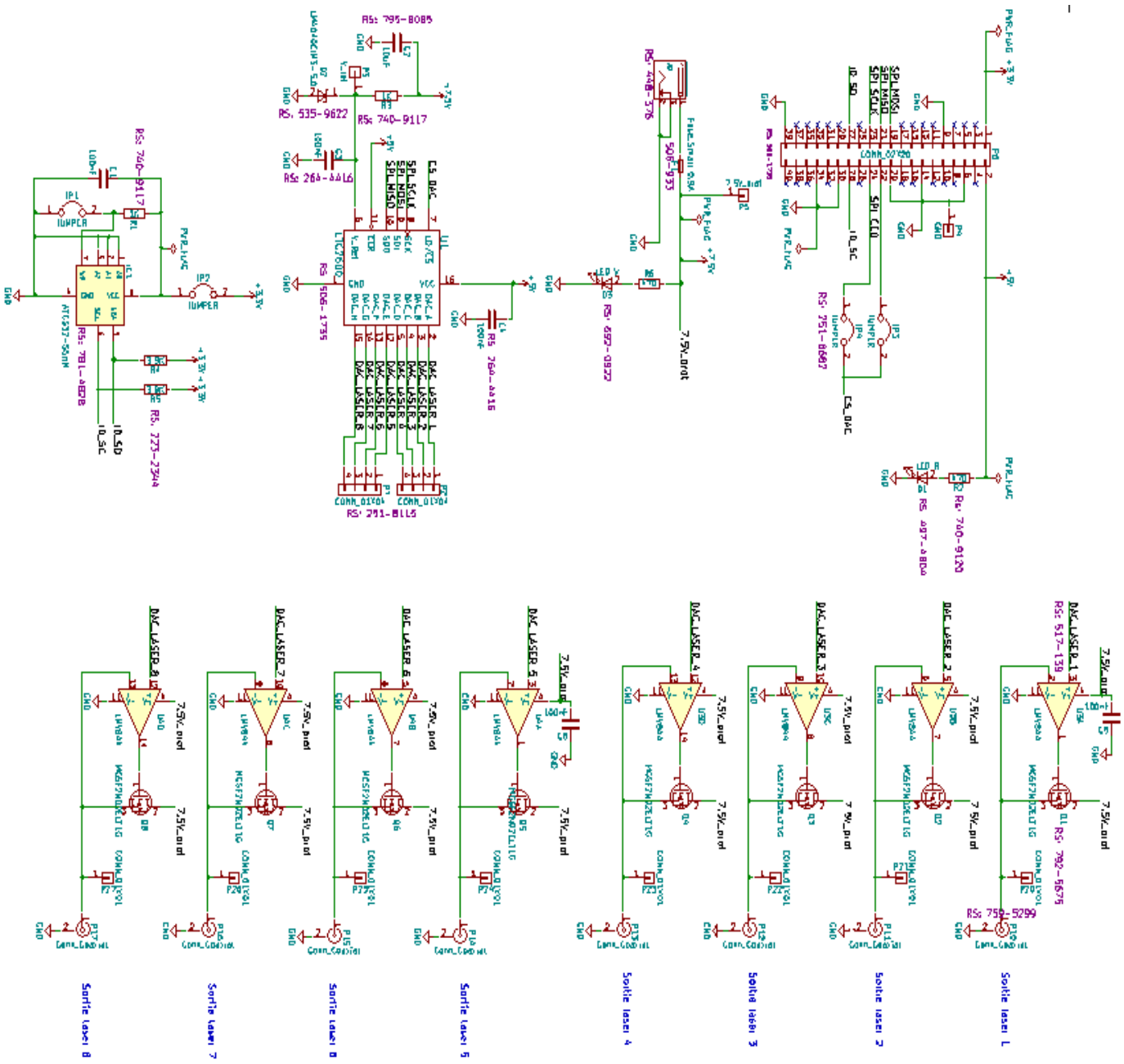
La RedPitaya est une petite carte rouge open source basée sur Linux pour le développement d'outils de mesure et de commande à l'aide d'applications trouvées sur **Bazaar** - le magasin de RedPitaya.

Dimensions : 107 x 21 x 60mm

Je vais donc m'inspirer de la forme de cette carte pour réaliser mon projet.

Nouveau schéma structurel

J'ai donc décidé de ne réaliser qu'une seule carte avec tous les composants à l'intérieur en m'inspirant de la forme de la carte RedPitaya.

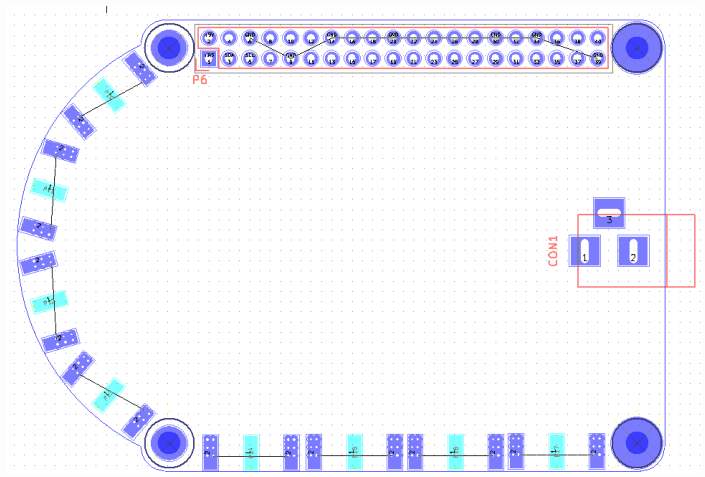


Routage de la carte

Après validation du schéma structurel, je dois procéder au routage.

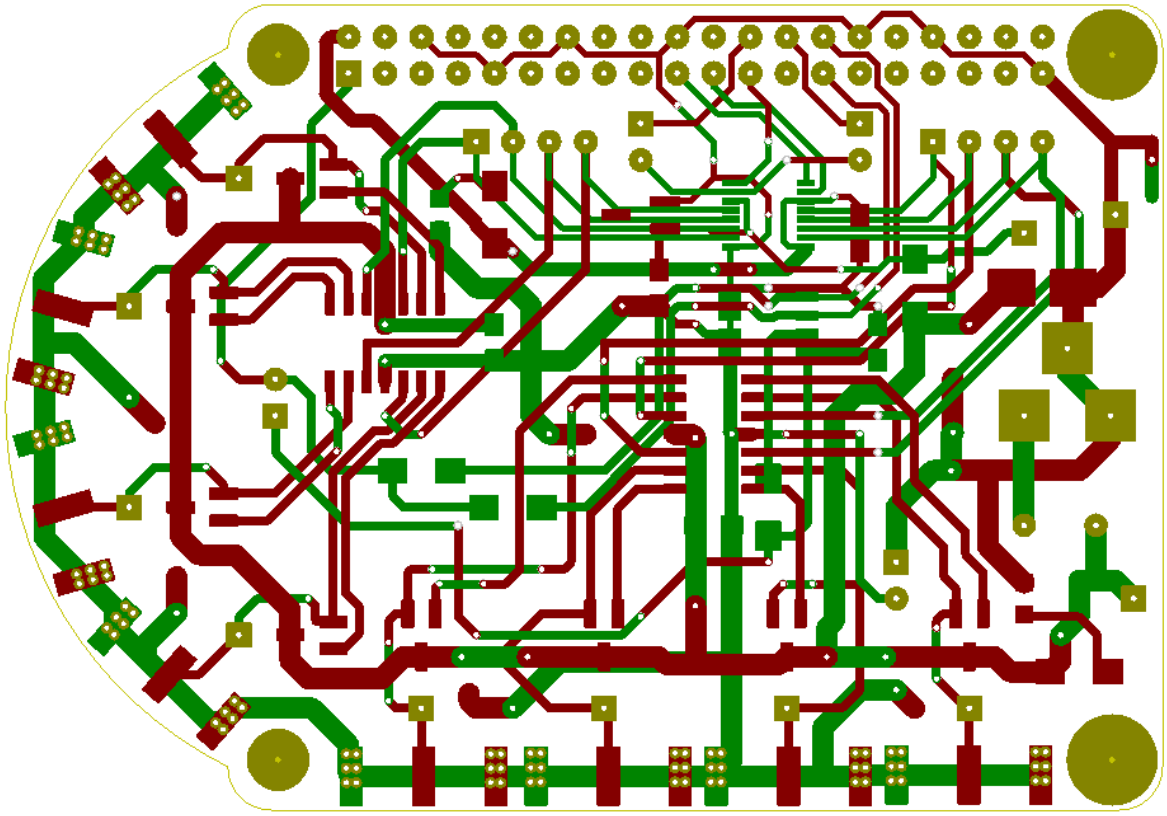
Pour cette carte huit voies j'ai dû partir d'un PCB contenant les contours de la carte, le GPIO, l'emplacement des connecteurs SMA, et l'alimentation externe pour router le reste de la carte.

Le PCB façon Red Pitaya utilisé :



Quatre connecteurs SMA sont sur la partie arrondie de la carte et quatre autres à l'opposé du GPIO par soucis de facilité. L'alimentation est donc logiquement sur le coté droit et sera donc juste au-dessus des ports USB de la carte Raspberry. Nous avons décidé de prendre un GPIO avec des broches plus longues, la carte sera donc un peu plus élevée et permettra de ne pas faire toucher l'alimentation et les ports USB.

Routage définitif :



Le routage m'a pris plusieurs semaines par rapport aux modifications qu'on m'a demandées, la carte devait être prête à envoyer au sous traitant sans réaliser de test.

Pour ma carte les problèmes qui ont été trouvés étaient d'abord en premier lieu ma disposition des composants qui était incorrecte, je les avais placés de manière que ça soit le plus pratique sans avoir respecté le schéma structurel qui a été fait ultérieurement.

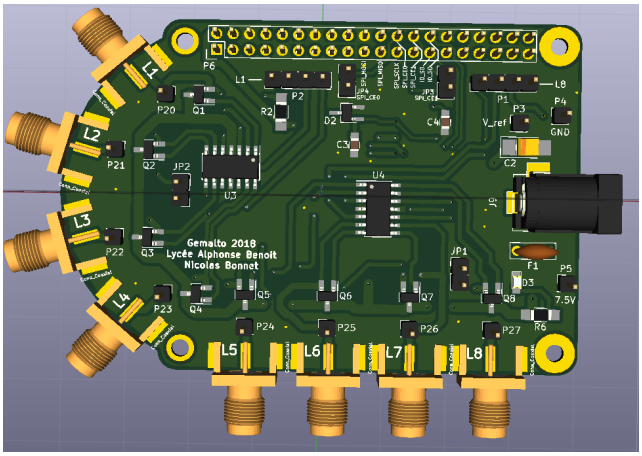
Après cette remarque j'ai utilisé mon schéma structurel afin de placer mes composants aux endroits correspondant car certains des composants devaient être placés côte à côte d'après les documentations du fabricant pour une utilisation correcte.

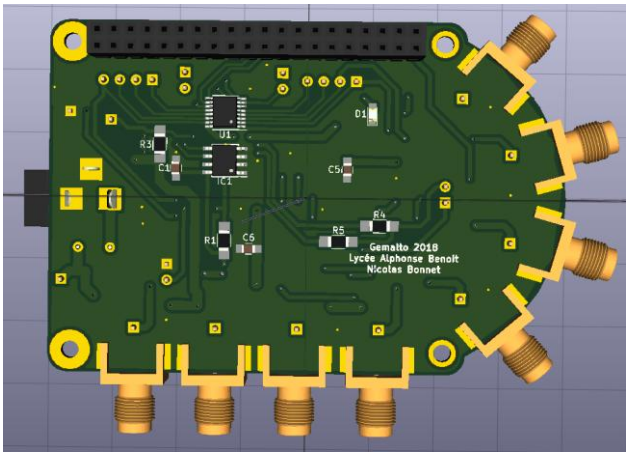
Après ces changements, j'ai dû refaire toutes mes pistes de routage de zéro; j'ai relié de manière à ce que ce soit le plus clair possible en respectant la largeur des pistes qui ont été données : GND → 1.000/1.500mm, 7.5V/5V/3.3V → 1.500mm et de 0.600/0.400mm pour les autres pistes.

J'ai rajouté des trous traversant afin d'avoir la masse partout sur ma carte et aussi des repères de texte pour les broches (ID_SD, ID_SC, MOSI, MISO, CLK, CE0 et CE1) pour une utilisation plus simple.

Visualisation 3D :

Grâce au mode 3D de Kicad on peut se faire une idée du résultat final, voir si tous les composants sont bien aux endroits voulus, voir si la taille des composants ne pose pas de problèmes particuliers qu'ont aurait pas forcément remarqué juste avec le routage. Le mode 3D permet aussi de voir si rien n'est à l'envers et que les références ne débordent pas ou bien qu'elles soient à l'envers.





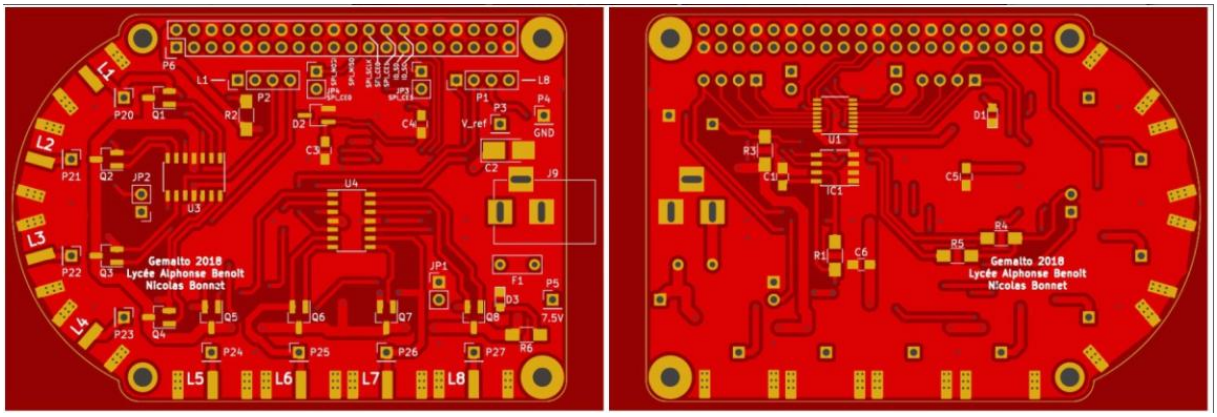
Après la vérification complète de la carte, il n'y a aucune erreur on peut créer un « tracer » qui permet de créer des fichiers gerber qui seront envoyer au sous traitant; puis on archive tous ces fichiers dans un dossier .zip.

Le site Seedstudio nous indique s'il y a des erreurs qu'on n'aurait pas remarquées avec le routage ou la visualisation 3D.

Fabrication de la carte

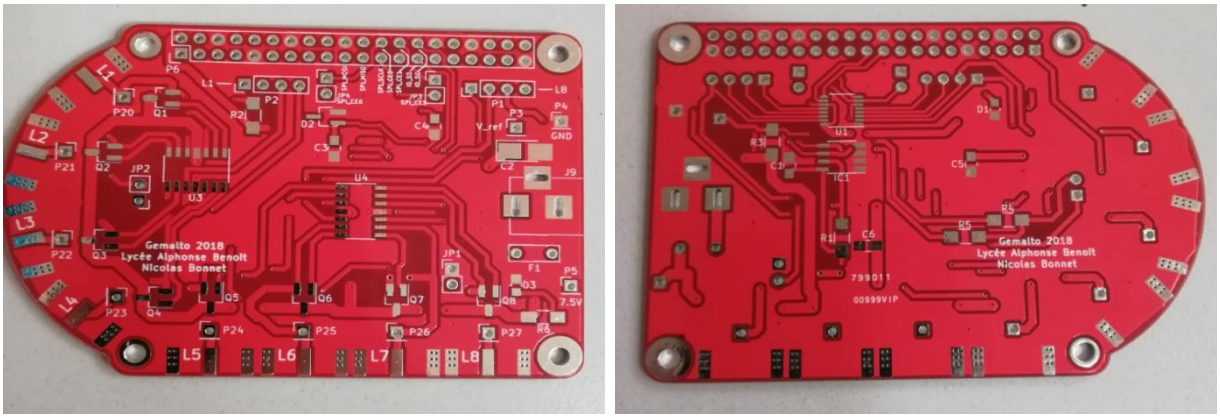
Après avoir étudié la carte de 2017, conçu la carte à huit sorties, il faut fabriquer la carte.

Cette carte nous l'a faisons fabriquer car nous n'avons pas le matériel nécessaire pour la fabrication donc nous transmettons nos fichiers aux fabricants afin qu'ils puissent réaliser la carte.



Voilà après la commande à quoi doit ressembler la carte.

Une fois fabriquée, la carte part de Shenzhen et met plusieurs jours pour arriver…

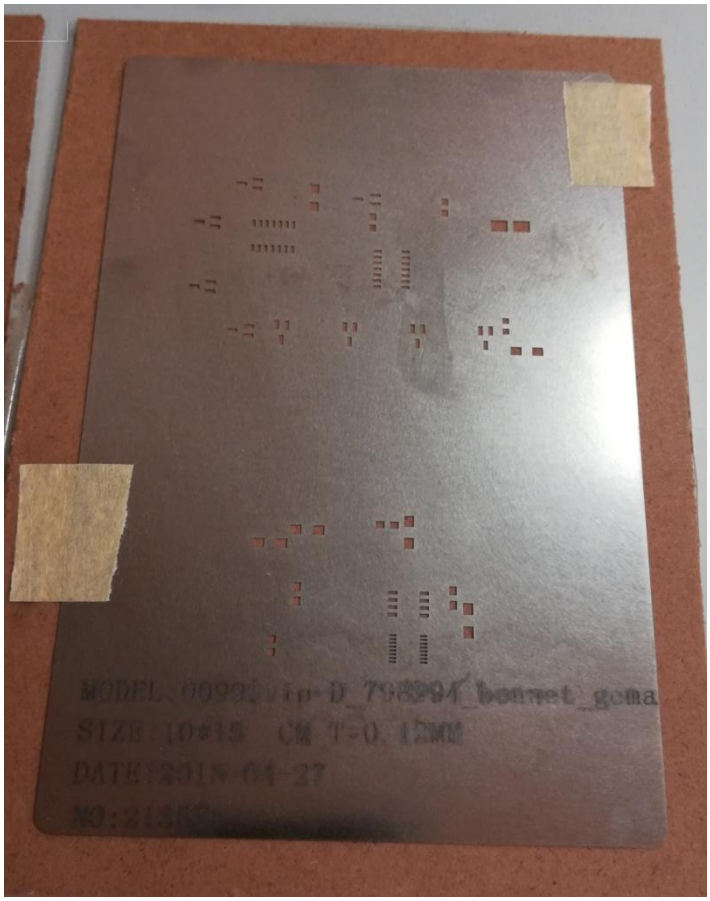


Pcb reçu

La prochaine étape consiste donc à souder les composants.

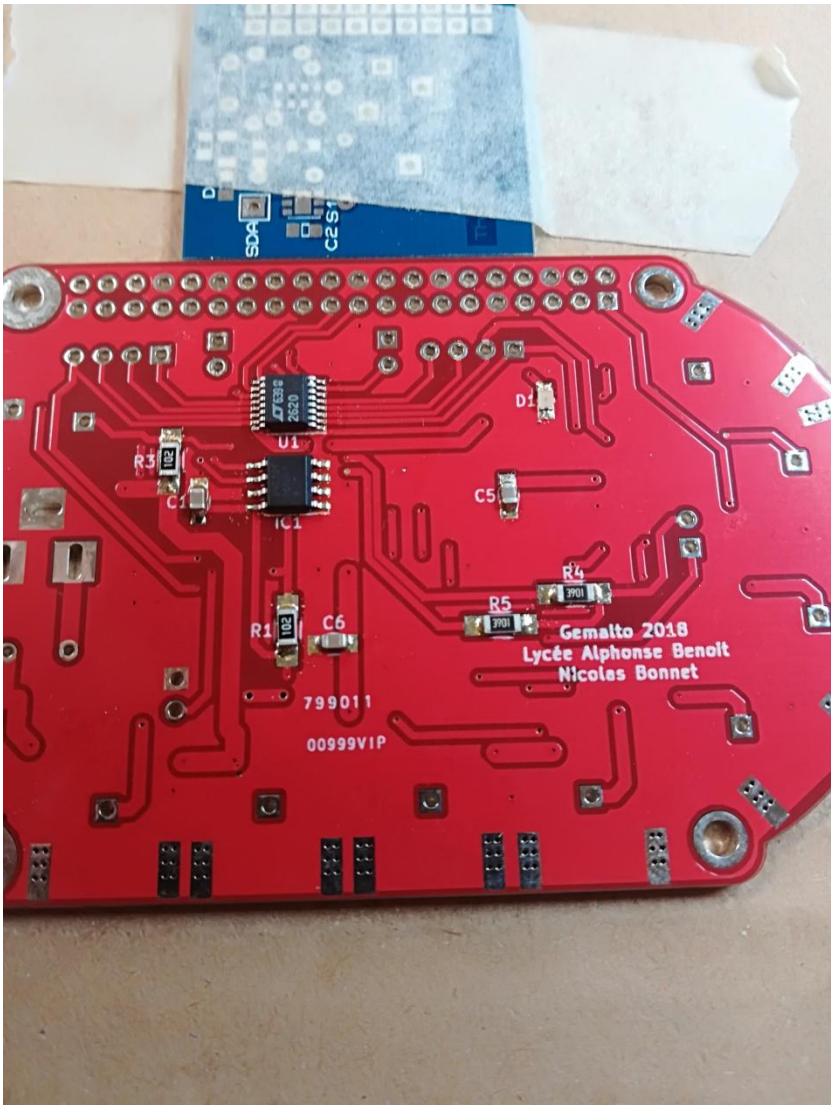
<u>Un pochoir à donc était commandé en plus de la carte.</u>

Le pochoir, qui est fabriqué exprès pour notre carte, permet de mettre de la pâte à braser uniquement sur les empreintes CMS. La pâte à brasée sera disposer que sur les empreintes CMS et donc donnera un rendu plus propre que de le faire à la pince.



On commence à souder les composants CMS sur une des deux faces.

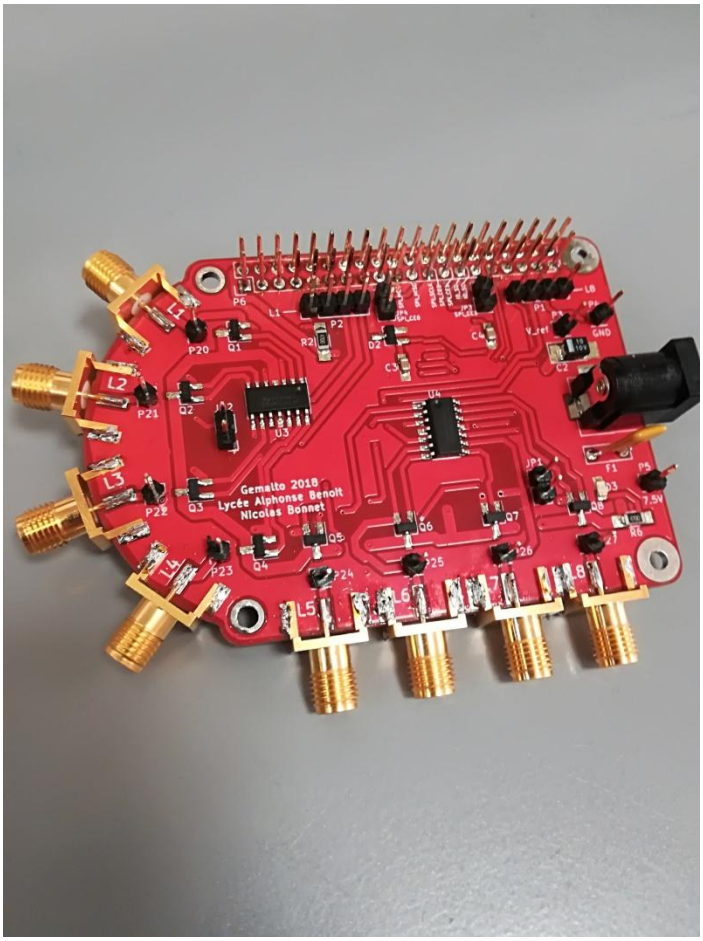
1. Aligner les trous du pochoir parfaitement avec les empreintes de la carte
2. Mettre une couche de pâte à braser sur le pochoir.
3. Retirer le pochoir.
4. Disposer les composants CMS sur la carte.
5. Placer la carte avec les composants CMS dans le four a fusion pour faire chauffer la pâte à braser.

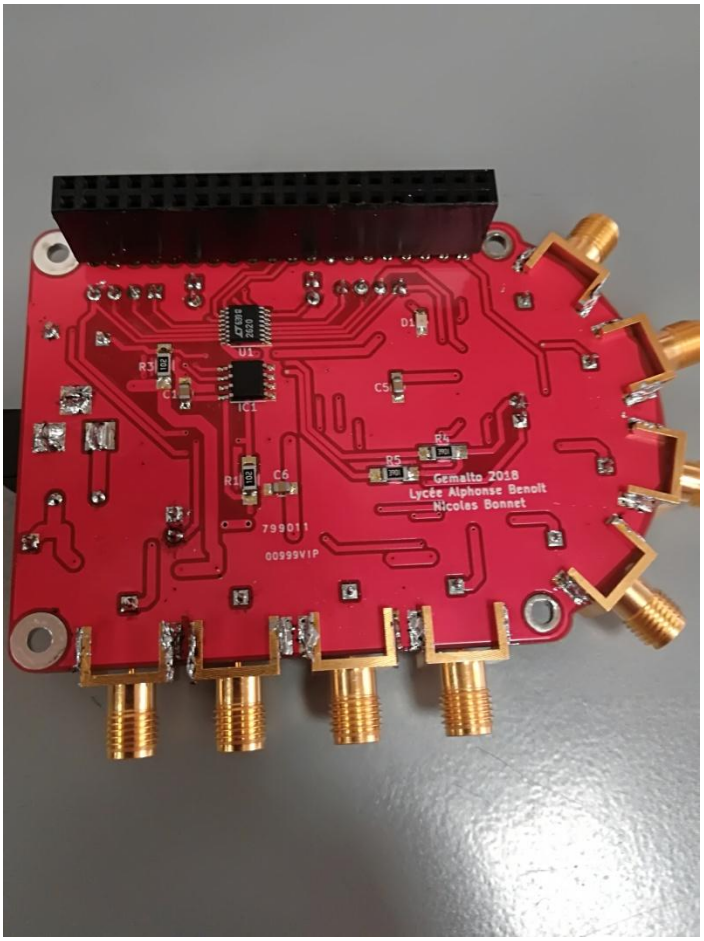


Il faut ensuite reproduire cette étape pour la seconde face.

Après avoir réalisé la soudure des composants CMS, il ne reste qu'à **souder les composants traversants.**

Résultat final

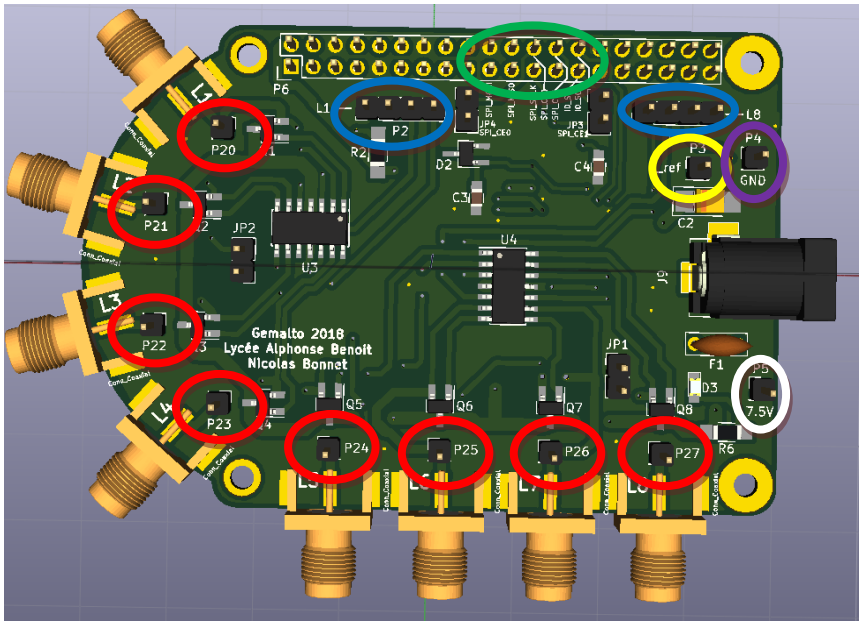




La carte huit sorties est donc prête à être testée.

Les points de test

La carte possède plusieurs points de test, pouvant ainsi en cas de défaillance repérer le problème.



Les points de test sont :

- de P1 à P2 : **Permet d'analyser la trame en sortie du microcontrôleur**
  Situé entre le microcontrôleur et l'ensemble amplificateur/transistor.
  L'ordre des points de test est de la gauche vers la droite. P2 a les sorties L1/L2/L3/L4 et P1 les sorties L5/L6/L7/L8.

- P3 : V_ref

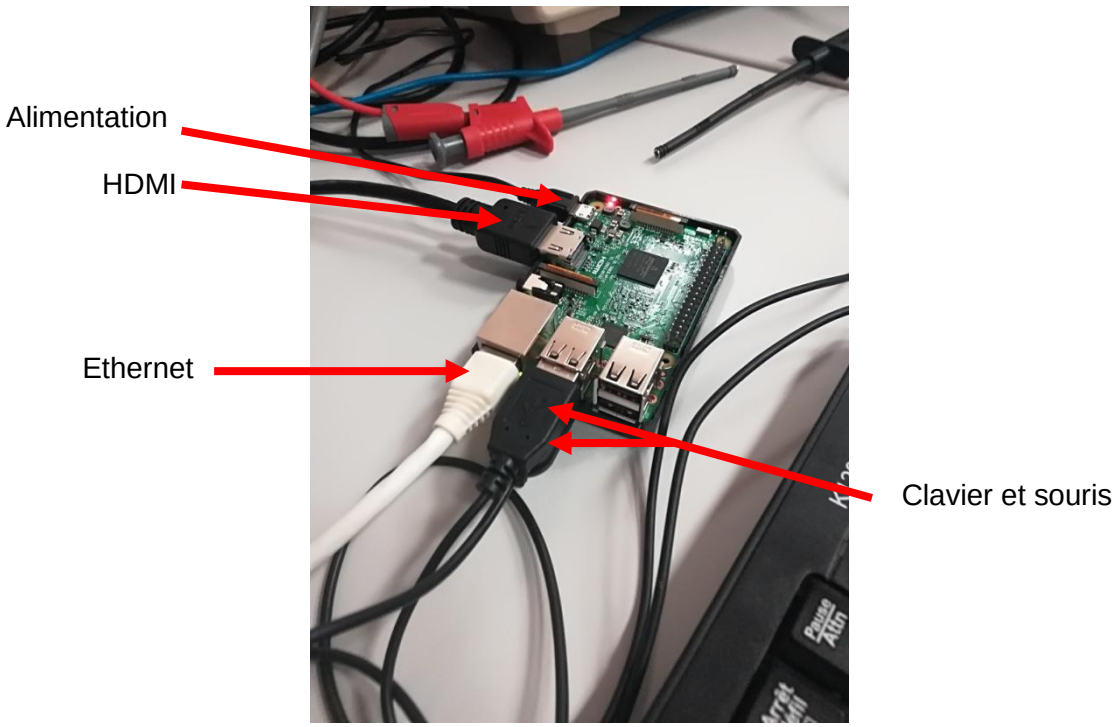
- P4 : GND

- P5 : 7.5V

- de P20 à P27 : Situé en sortie du transistor, **permet d'analyser la trame sur chaque sorties de la carte. P20 pour la sortie 1 jusqu'à P27 pour la sortie 8.**

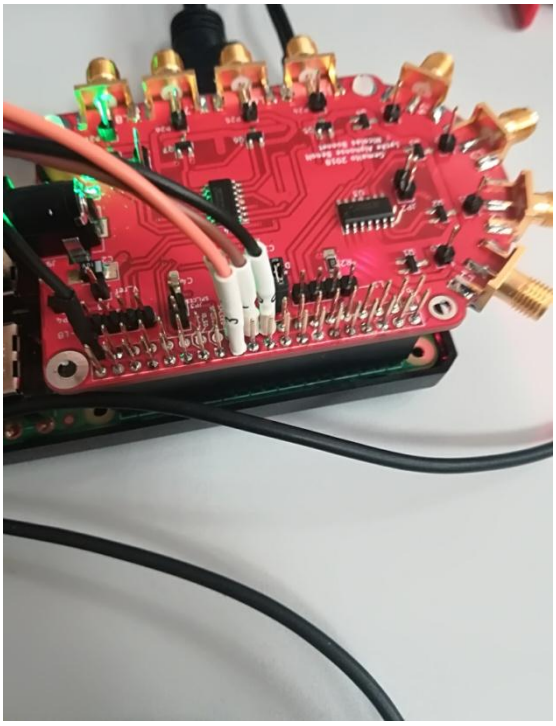
- Broches GPIO pour les bus SPI sont tous indiqués directement sur la carte.

Réalisation du test - recette



Brancher l'alimentation l'HDMI, un clavier et une souris.

Brancher la carte sur la Raspberry



Brancher Saleae (l'analyseur logique) sur les broches correspondantes du GPIO.

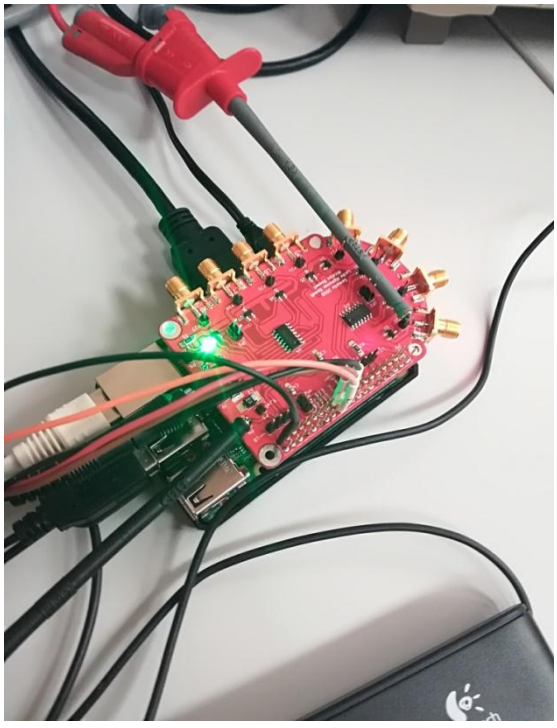
Les broches sont indiquées directement sur la carte.

SPI_MOSI (channel 0)

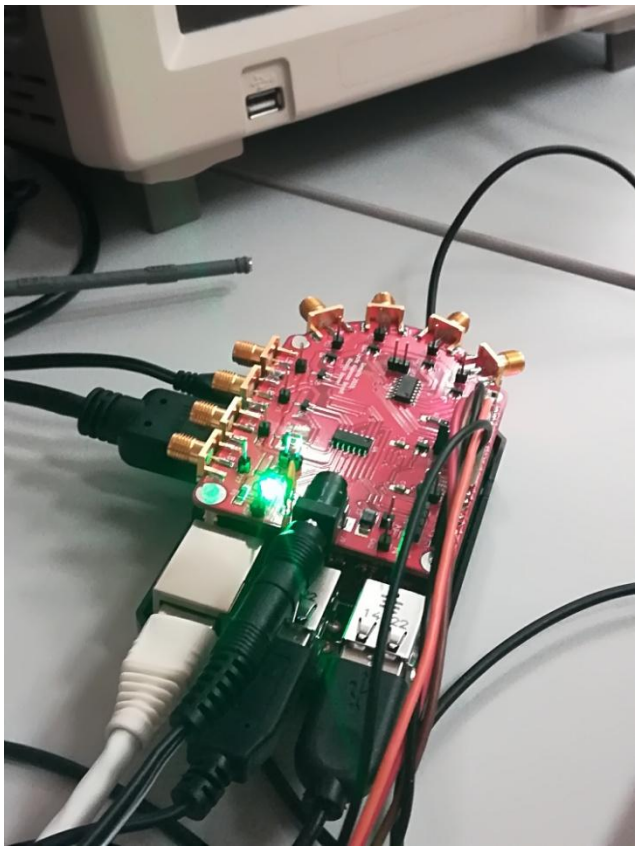
SPI_MISO (channel 1)

SPI_CLK (channel 2)

SPI_CE0 (channel 3)



Pour tester une sortie, brancher le noir sur le GND et le rouge sur le point de test souhaité.



Les tests peuvent être effectués.

Résultat des tests de la carte 2018

J'ai donc procédé au test de la carte de la même manière que l'étude de la carte de 2017

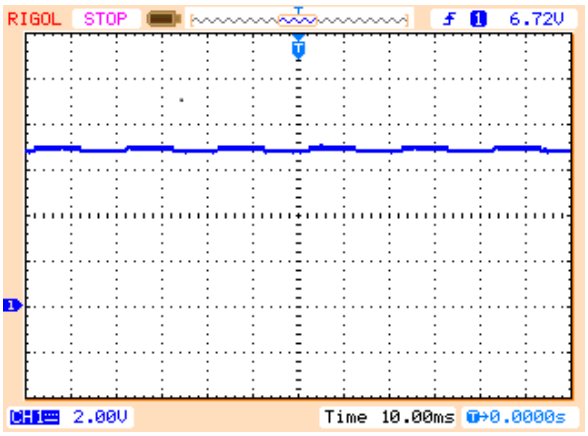
Nous avons eu un nouveau logiciel réalisé par Mr.Defrance permettant de faire ces tests.

Je peux piloter huit sorties, je donc réaliser les tests avec un oscilloscope et le logiciel Saleae.

Rapport des tests :

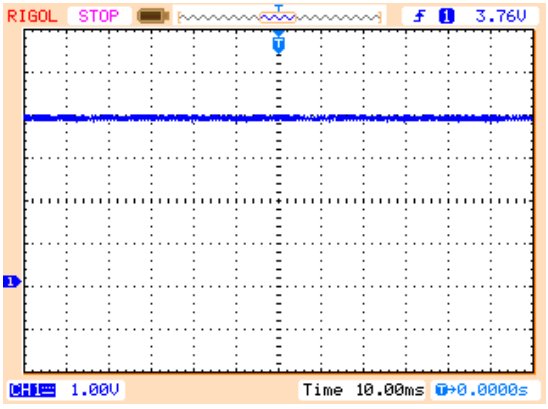
- **Rien d'anormale sur la sortie n°1.**
- Sur les sorties 2/3/6/7/8 il faut mettre **manuellement à zéro les sorties** (le problème vient certainement du logiciel). **Sinon rien d'anormale à l'oscilloscope ni sur Saleae.**
- **Les sorties 4/5 ne fonctionnent pas :**

Sortie n°4 : A l'oscilloscope on voit que la sortie est à 7.5V. Même si on ne commande pas la sortie n°4.
L'analyse de trame avec Saleae n'indique rien d'anormale
La raison de ce disfonctionnement est inconnue pour l'instant.



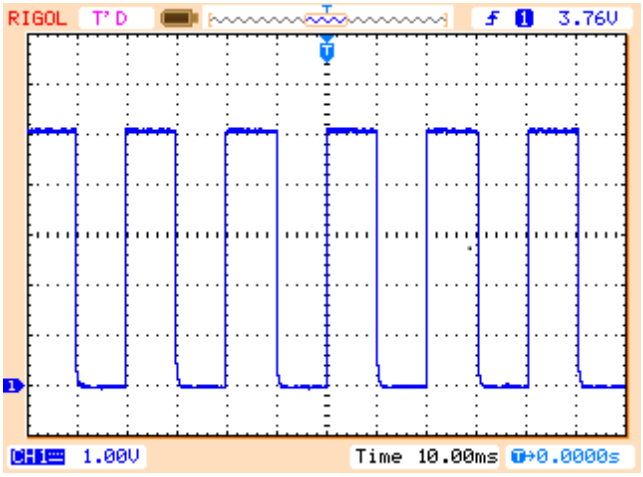
7.5V

Sortie n°5 : A l'oscilloscope la sortie n°5 est constamment à 4V. L'analyse de trame avec Saleae n'indique rien d'anormale
La raison de ce disfonctionnement est inconnue pour l'instant.



4V

Pour un fonctionnement normal le signal affiché à l'oscilloscope doit être l'image de la commande envoyé.

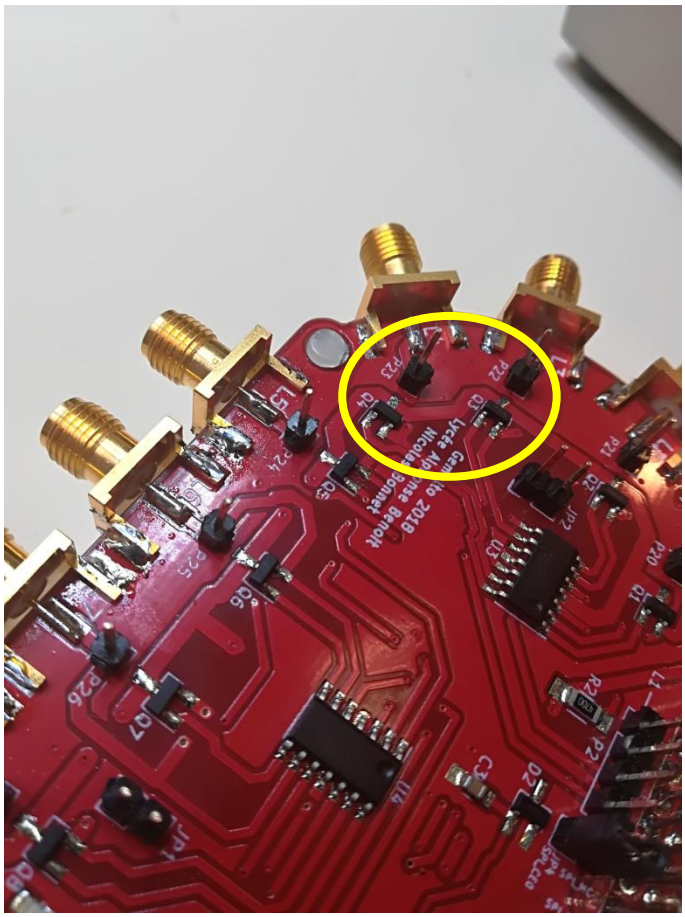


Le signal demandé est un signal carré, VL=0V, VH=5V, tL=10ms, tH=10ms.

Le signal à été envoyé au laser n°1, le signal à était visualisé avec un oscilloscope sur le point de test P20.

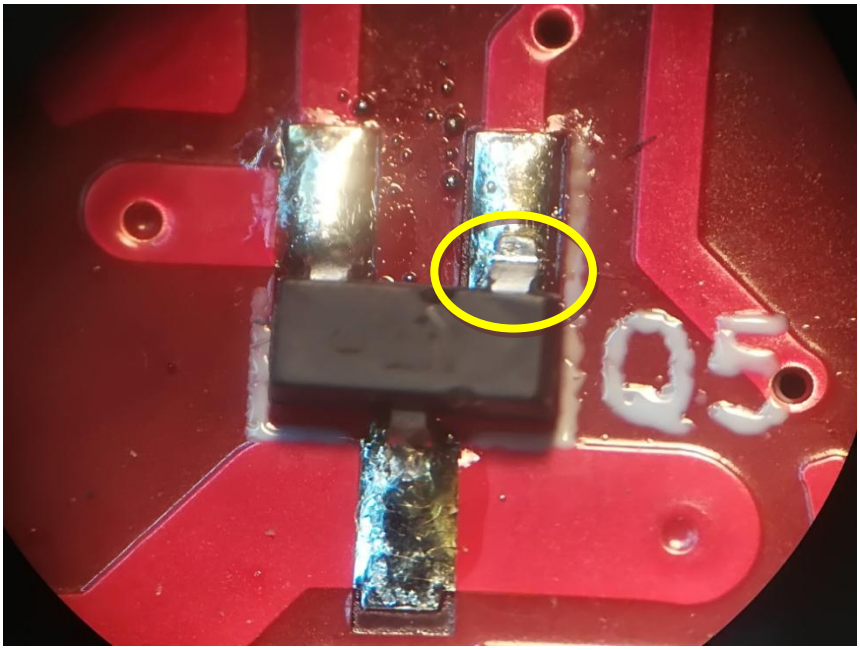
Après vérification on remarque qu'il n'y a pas de liaison entre l'amplificateur et le transistor.

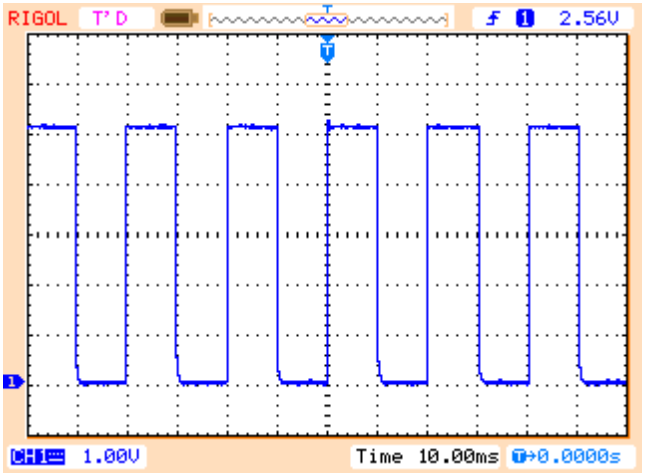
Le problème n'était pas visible à l'œil nu, nous avons donc essayé de voir si il y avait une liaison entre la l'amplificateur et le transistor pour les deux sorties avec un multimètre et nous nous sommes aperçu qu'ils n'étaient pas relié.



Il semble donc qu'il y ait un problème avec la soudure, mais à l'œil nu on ne voit pas du tout la différence avec les autres transistors.
Donc nous avons regardé à la loupe.

Nous avons repéré le problème. L'une des trois soudures à était mal faite, ce problèmes était sur le transistor et on peu voir sur la photo ci-dessous que l'une des broches est au dessus de l'éteint il suffit donc de chauffer l'étain et d'appuyer pour que ça prenne bien et que ça fonctionne.





A l'oscilloscope on remarque que dorénavant les signaux sur la sortie 4 et 5 sont bons.

Les lasers 4 et 5 ont un fonctionnement normal, donc tous les lasers sont opérationnels.

Modification

Mettre sur la face du dessus la led rouge.

Conclusion

Le projet s'est déroulé en quatre parties. Une partie étude de la carte de 2017, une partie conception (schéma structurel/routage) et fabrication puis test.

L'objectif premier de ce projet à été de réaliser deux cartes d'extension Rpi pouvant accueillir huit sorties pour l'entreprise GEMALTO en reprenant le projet réalisé par les étudiants de l'année dernière.
Cette extension était composée d'une carte avec le DAC et le reste des composants visant à traiter le signal, et une mezzanine avec les huit sorties.

Suite à des modifications, j'ai fini par ne réaliser qu'une seule carte avec tous les composants à l'intérieur.
Cette modification majeure du projet ne m'a pas vraiment gêné par rapport au fait qu'il suffisait simplement de rassembler mes deux schémas structurels en un seul et mon routage n'était pas encore réalisé donc je partais de zéro.

Le problème de la liaison entre la carte au format HAT et la mezzanine ne se posait plus.

Les problèmes suite au premier test sont réglés.

L'objectif de réaliser une carte avec les huit sorties opérationnelle est donc atteint.

Annexe

Listes des composants :

Composant Code RS	Valeur	Quantité	Prix
Jack alimentation RS: 448-376	7.5V	1	2.30€
Jumper RS: 251-8682		4	0.292€
CONN_02X20 RS: 681-6808 *GPIO*		1	4.94€
CONN_02X04 RS: 681-6808		1	0.175€
AT24CS32-SSHM RS: 781-4828 *Mémoire morte*		1	0.342€
LTC2600 RS: 506-1735 *Microcontroleur*		1	14.79€
MGSF2N02ELT 1G RS : 792-5675 *Transistor*	$2.8A_{max}$	8	0.27€
LM4040DBZ-5 RS: 535-9622 *Référence tension*	5V	1	0.792€
LMV844 RS: 517-139 *Ampli*		1	5.10€
Resistance RS: 740-9117	1K Ω	2	0.02€

Resistance RS: 740-9120	470Ω	2	0.02€
Resistance RS: 223-2344	3.9K Ω	2	0.008€
Condensateur RS: 264-4416	100nF	4	0.04€
Condensateur RS: 795-8085	10µF	1	0.0229€
Fusible RS: 506-933	0.9A	1	0.578€
LED rouge RS: 466-3920	1.9V	1	0.13€
LED verte RS: 692-0922	3.6V	1	0.353€
Jack SMA RS: 759-5299	50Ω	8	3.05€
Point de test RS: 863-2706		11	4.12€

4. Journal de bord.

Date	*Intervenant*	*Objet*
•Mercredi 10/01/2018	• BONNET • LEROY • LANDO	• Lecture du dossier • Prise en compte des objectifs
•Jeudi 11/01/2018	• BONNET • LEROY • LANDO	• Prise en compte des changements à réaliser • Installation du logiciel Kicad et récupération des docs nécessaires
•Vendredi 12/01/2018	• BONNET • LEROY • LANDO	• Elaboration du diagramme de Gantt • Apprentissage Kicad
• Mercredi 17/01/2018	• BONNET	• Etude des composants • Création d'un nouveau projet Kicad
•Jeudi 18/01/2018	• BONNET	• Conception du schéma de câblage sur kicad • Problème avec les librairies il manque des composants
•Vendredi 19/01/2018	• BONNET	• Fin du schéma sur kicad • Problème routage, pour lire la Netlist
• Mercredi 24/01/2018	• BONNET • LEROY	• Problème de librairie réglé • Etude des anciennes cartes • Teste de l'ancienne carte pour comprendre le fonctionnement
• Jeudi 25/01/2018	• BONNET • LEROY	• Utilisation du programme QT pour tester l'ancienne carte
•Vendredi 26/01/18	• BONNET • LEROY	• Elaboration des analyses de la carte avec Saleae et un oscilloscope, interprétation des résultats. • Problème de décalage des bits sur Saleae
•Mercredi 31/01/18	• BONNET • LEROY	• Problème de décalage des bits sur Saleae résolu →Data is valid on clock leading edge (CPHA = 0) active sur front montant • Tentative de correction de bug PNG sur le logiciel de test de la carte, sans conséquence pour l'avancer de l'analyse sur Saleae • Fin de l'analyse des trames sur Saleae et rédaction de la feuille explicative de fonctionnement
•Jeudi 01/02/18	• BONNET	• Mise en forme de la liste des composants • Présentation du fonctionnement de la carte (réception de la trame et interprétation) • Correction d'une erreur commit sur Kicad

•Vendredi 02/02/18	• BONNET	• Tuto routage • Routage
•Jeudi 08/02/18	• BONNET	• Avancement sur le routage • Transmission des schémas au professeur pour validé mes schémas pour finir le routage
•Vendredi 09/02/18	• BONNET	• Routage • Modification du schéma structurel
•Jeudi 15/02/18	• BONNET	• Prise en compte du mail envoyé par le professeur avec de nouvelles • Code commande RS mit à coté de chaque composant sur le schéma Kicad (pas fini) • Teste du bloque secteur alim 6V par encore validé car il y a un problème avec le logiciel qui ne garde pas en mémoire le signal généré, du coup on ne peut pas mesurer les courant sur plusieurs sorti
•Vendredi 16/02/18	• BONNET • LEROY • LANDO	• Elaboration du rapport pour la revue de projet n°1 • Recherche d'un système pour lier la carte à la mezzanine
•Mercredi 21/02/18	• BONNET • LEROY	• Installation du nouveau programme • Utilisation du programme • Routage
•Vendredi 23/02/18	• BONNET • LEROY	• Inventaire des composants
•Mercredi 14/03/18	• BONNET • LEROY	• Avancement du rapport de la revue de projet n°1 • Réalisation du diaporama
•Jeudi 15/03/18	• BONNET	• Fin de la rédaction du rapport de la revue de projet n°1 • Changement de plan, réalisation que d'une seule carte avec les huit sorties dessus. Reprise du schéma structurel. Un nouveau routage est nécessaire.
•Vendredi 16/03/18	• BONNET • LEROY	• Apprentissage de la présentation • Refaire les tests pour la démonstration.
•mercredi 21/03/18	• BONNET • LEROY	• Routage de la carte à huit sorties

•Jeudi 22/03/18	• BONNET • LEROY • LANDO	• Préparation de l'oral de la revue de projet • Préparation pour la partie démonstration e la revue de projet
•Vendredi 23/03/18	• BONNET • LEROY • LANDO	• Passage 1ère revue de projet
•Mercredi 28/03/18	• BONNET	• Modification à faire sur le schéma structurel • Correction des erreurs du rapport de la revue de projet n°1
•Jeudi 29/03/18	• BONNET	• Avancement sur le rapport de la revue de projet n°2
•Vendredi 30/03/18	• BONNET • LEROY	• Routage • Fin du routage (à faire vérifier)
•Mercredi 04/04/18	• BONNET • LEROY	• Modification routage • Fin du routage (à faire vérifier)
•Vendredi 06/04/18	• BONNET • LEROY	• Suite routage • Respecter les nouvelles consignes
•Mercredi 11/04/18	• BONNET • LEROY	• Finition apporté sur le routage • Suite rédaction de la seconde revue de projet
•Jeudi 12/04/18	• BONNET • LEROY	• Etude de la partie physique sur l'ensemble amplificateur – transistor • Rédaction de la partie physique à mettre en partie commune EC du rapport • Finition routage (routage achevé) • Création du fichier Gerber
•Vendredi 13/04/18	• BONNET	• Rédaction rapport de la revue de projet n°2
•Mercredi 18/04/18	• BONNET	• Rédaction rapport de la revue de projet n°2

•Mercredi 09/05/18	• BONNET	• Rédaction rapport de la revue de projet n°2 • Avancement sur le diaporama
•Vendredi 11/05/18	• BONNET • LEROY	• Rédaction rapport de la revue de projet n°2 • Révision de la partie physique du rapport de projet • Réception des cartes commandées
•Mercredi 16/05/18	• BONNET	• Rédaction rapport de revue de projet n°2
•Jeudi 17/05/18	• BONNET	• Rédaction rapport de revue de projet n°2 • Soudure CMS
•Vendredi 18/05/18	• BONNET • LEROY • LANDO	• Fin rédaction rapport de revue de projet n°2 • Fin soudure des composants traversant • Assemblage de toutes les parties de rapport de revue de projet n°2
•Mercredi 23/05/18	• BONNET	• Test de la carte 8 sorties, un problème à était trouver sur les sorties 4 et 5. • Diaporama • Rédaction de la partie test pour le rapport
•Jeudi 24/05/18	• BONNET • LEROY	• Mise en forme du rapport. • Fin diaporama • Problème trouver sur les sorties 4 et 5, les huit sorties sont maintenant opérationnelle

Partie Personnelle :
Leroy Nicolas

GEMALTO

Présentation de ma partie :

Notre projet est constitué d'une équipe de trois personnes composée de un IR et deux EC.

On m'a attribué la partie de concevoir deux cartes identiques qui permettent de piloter huit lasers en même temps ; on nous a confié la carte de l'année précédente afin de pouvoir apporter les nouvelles modifications données par le cahier des charges.

Ces cartes seront pilotées par une Raspberry Pi.

Une grande partie du temps j'ai effectué des tâches communes avec mon binôme sur ce projet.

Etapes préparatoires :

- Prise en compte du projet.
- Installation de KiCad.
- Récupération des documents de l'année précédente.

Modifications apportés :

- Ajout de quatre sorties supplémentaires.
- Suppression de l'ancienne partie d'alimentation sur le schéma.
- Ajout d'une nouvelle alimentation externe en 7,5V.

Modifications déportées en cours de projet après de nouvelles demandes de Mr Moitrel :

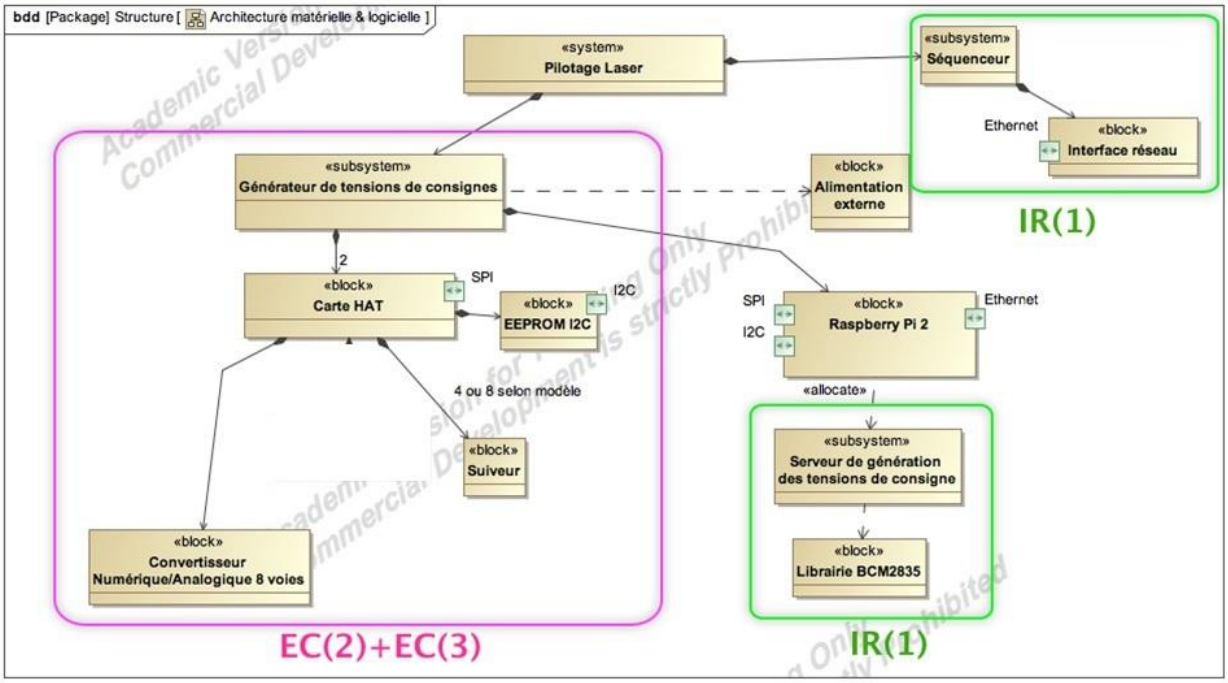
- Changement de transistors qui seront moins surdimensionné.
- Ne pas utiliser un système de nappe pour transmettre l'alimentation sur les deux cartes.

Planification :

Mon objectif est de concevoir deux cartes s'emboîtant l'une dans l'autre pouvant se connecter à une carte Raspberry et afin de pouvoir piloter huit lasers simultanément.

Pour cela j'ai réalisé un gantt « prévisionnel » afin de pouvoir gérer mon temps accordé aux différentes tâches à effectuer.

Tâches LN	191 hr	Jeu 11/01/18	Ven 11/05/18
Installation du logiciel Kicad et tuto du logiciel	6 hr	Ven 12/01/18	Jeu 18/01/18
Analyse du shéma structurel de 2017 , corriger les erreurs et l'adapter au nouvelles consignes.	60 hr	Jeu 18/01/18	Jeu 15/02/18
Etude du cas AT (mémoire morte)	8 hr	Jeu 11/01/18	Ven 12/01/18
Concevoir/Réaliser/Tester une cartes d'extension Rpi de 4 sorties (x2)	50 hr	Jeu 15/02/18	Mer 28/03/18
conception du codage/Test des pilotages des cartes.	40 hr	Mer 28/03/18	Ven 13/04/18
Concevoir/Coder/Tester une interface graphique pour vérifier le fonctionnements des sorties analogiques	25 hr	Ven 13/04/18	Ven 11/05/18



Logiciel

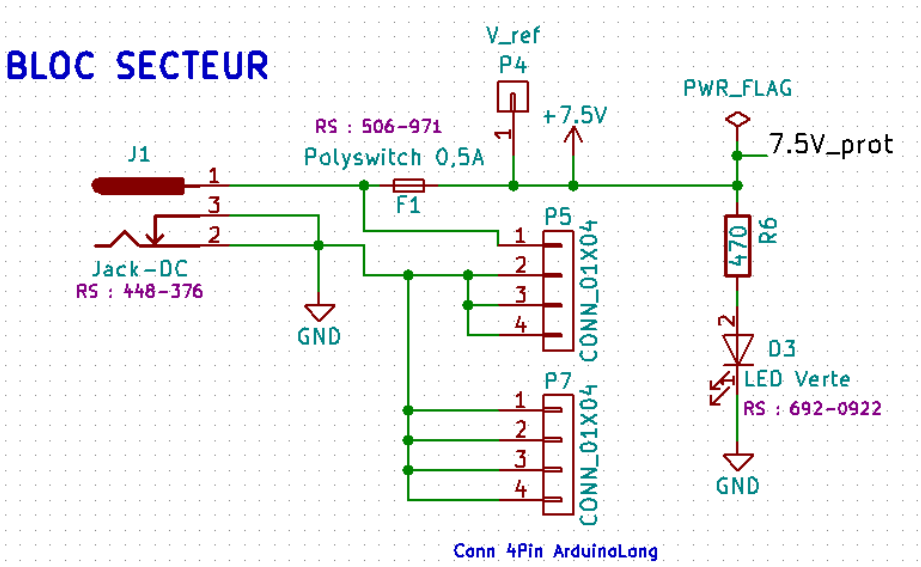
KiCad est un logiciel libre de conception pour l'électronique pour le dessin de schémas électroniques et la conception de circuits imprimés.

Il permet de réaliser toutes les étapes nécessaires à la conception d'un circuit imprimé:

Tout d'abord, nous avons repris le schéma de la carte précédente de 2017 afin d'y apporter les nouvelles modifications demandées pour la session 2018 du projet Gemalto qui est de changer la source d'alimentation pour des raisons de manque de puissance pour pouvoir alimenter toutes les sorties correctement , l'ajout de quatre sorties supplémentaires de deux manières différentes (deux cartes identiques ou une carte avec les huit sorties), utilisation de la mémoire morte.

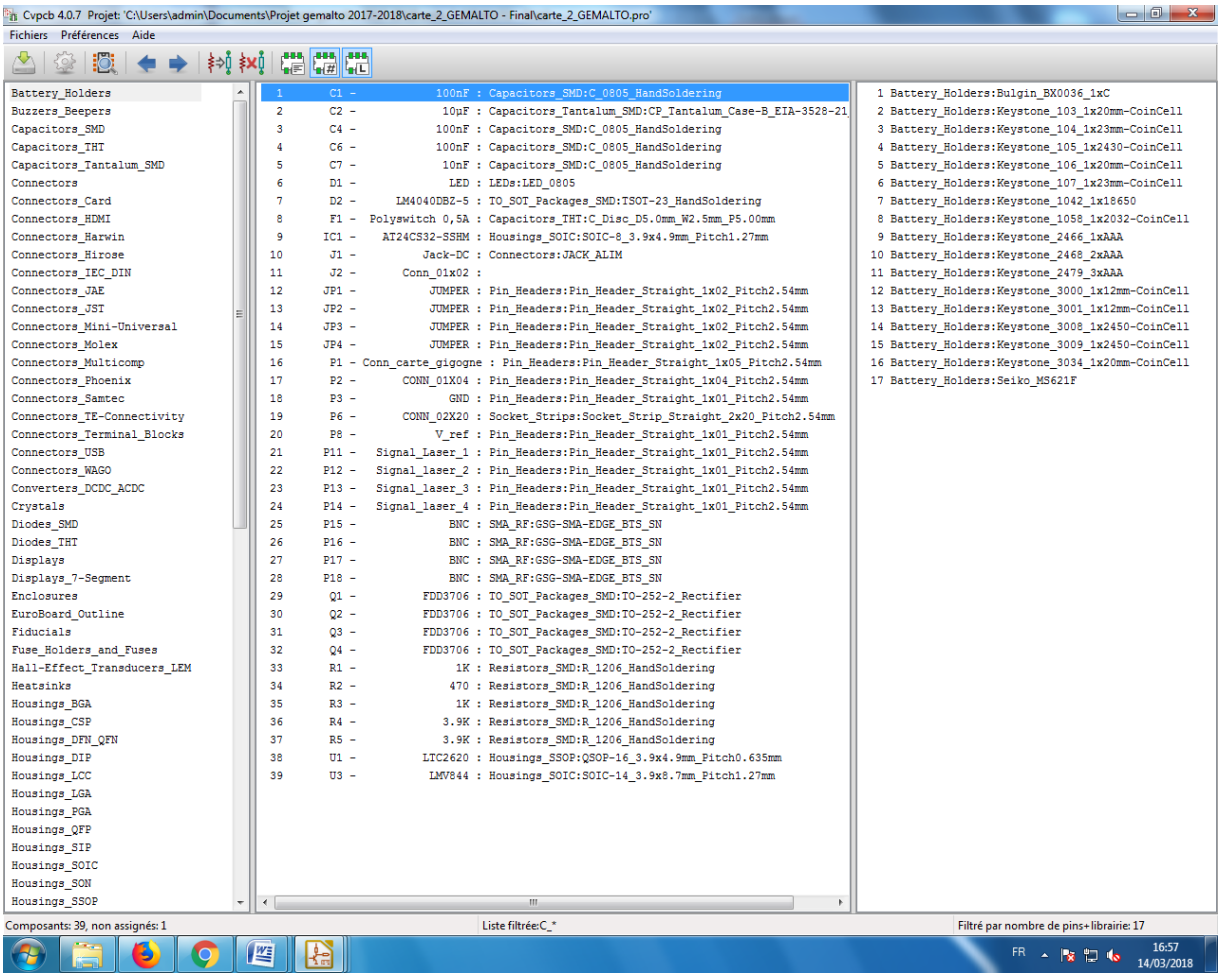
Nous avons connecté les broches 27 et 28 du GPIO (utilisation des broches ID_SD et ID_SC) aux bornes 5 et 6 du AT24CS32-SSHM qui correspondent à l'utilisation d'une mémoire morte avec l'I2C HAT.

Ajout de l'alimentation externe et suppression de l'ancien module.



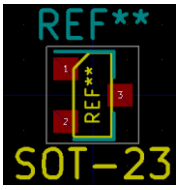
Nous avons repris les anciennes librairies afin de pouvoir obtenir les composants utilisés l'année dernière et ajouter de nouvelles pour les nouveaux composants.

Suite à ces modifications apportées nous devons ajouter les empreintes de ces nouveaux

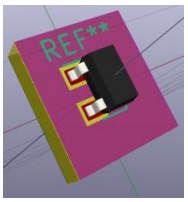


composants qui correspondent.

Une fois que tous les composants sont assignés a leur empreinte respective on peut visualiser



les composants en 2D et 3D pour composants correspondent bien à ceux voulus :

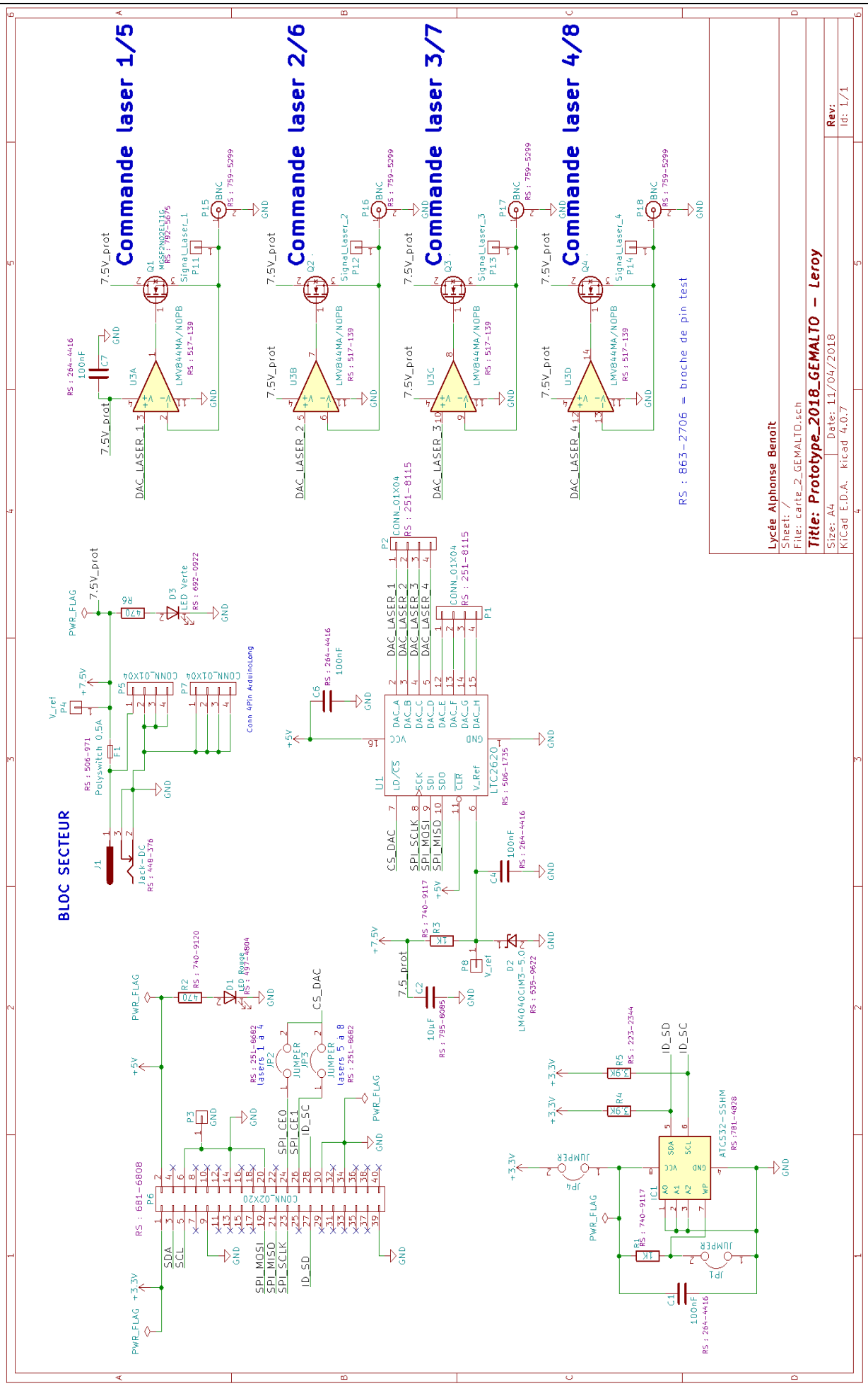


voir si les

Une fois terminé on génère une netlist afin de pouvoir placer nos composants voulus sur le schéma de routage ;

puis on peut lancer le pcb :

Schéma structurel de la carte 4 sorties laser (page suivante).


Commande laser 1/5
Commande laser 2/6
Commande laser 3/7
Commande laser 4/8
BLOC SECTEUR
RS : 863-2706 = broche de pin test
Lycée Alphonse Benoit
Sheet: /
File: carte_2_GEMALTO.sch
Title: Prototype_2018_GEMALTO - Leroy
Size: A4
Date: 11/04/2018
Rev:
KiCad E.D.A. kicad 4.0.7
Id: 1/1

Programme de pilotage de la carte Gemalto :

On utilise le nouveau programme créé par M.Defrance avec le terminal afin de pouvoir tirer le courant sur toutes les sorties en même temps car l'ancien programme utilisé ne permettait pas de faire fonctionner toutes les sorties en même temps.

On se dirige dans le dossier pour pouvoir l'utiliser :

« cd /home/pi/SendLtc2600 », puis ensuite rentrer

la commande : sudo ./dist/Debug/GNU-Linux/sendltc2600 –c 3 –n 0 -i 00 -d 0xFFF0

```
pi@raspberrypi:~/SendLtc2600 $ sudo ./dist/Debug/GNU-Linux/sendltc2600 -c 3 -n 3
 -d 0x0FFf0
cmd : WRITE_TO_AND_UPDATE_N
 ch : DAC D (pin 5)
val : 65520 (0xfff0) -> 5        V
```

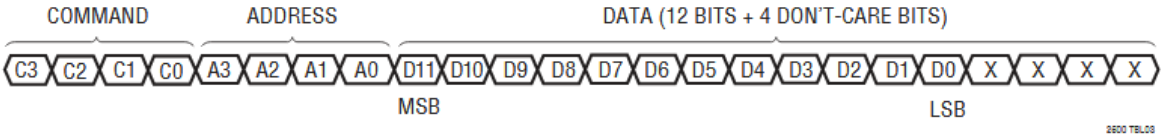
Le « c » représente la commande qui est le 3 dans notre cas (voir doc ci-dessous).

Le « n » représente l'adresse du DAC que l'on veut commander dans le cas présenté le « DAC D ».

Le « d » permet d'attribuer les données. (0xFFF0) = 5V.

Le « i » représente le cavalier qui permet de choisir quelle carte on veut piloter (00 ou 01).

INPUT WORD (LTC2620)



Calcul :

Allumé :

$0xFFF0 \rightarrow (4095)_{10} \times \frac{5}{4096} = 4.98V.$

A l'arrêt :

$0x0000 \rightarrow (0)_{10} \times \frac{5}{4096} = 0V.$

Documentation du 12 bits (LTC2620) :

Table 1.

COMMAND*				
C3	C2	C1	C0	
0	0	0	0	Write to Input Register n
0	0	0	1	Update (Power Up) DAC Register n
0	0	1	0	Write to Input Register n, Update (Power Up) All n
0	0	1	1	Write to and Update (Power Up) n
0	1	0	0	Power Down n
1	1	1	1	No Operation

*Command and address codes not shown are reserved and should not be used.

ADDRESS (n)*				
A3	A2	A1	A0	
0	0	0	0	DAC A
0	0	0	1	DAC B
0	0	1	0	DAC C
0	0	1	1	DAC D
0	1	0	0	DAC E
0	1	0	1	DAC F
0	1	1	0	DAC G
0	1	1	1	DAC H
1	1	1	1	All DACs

0 0 1 1 = il stock et il envoie.

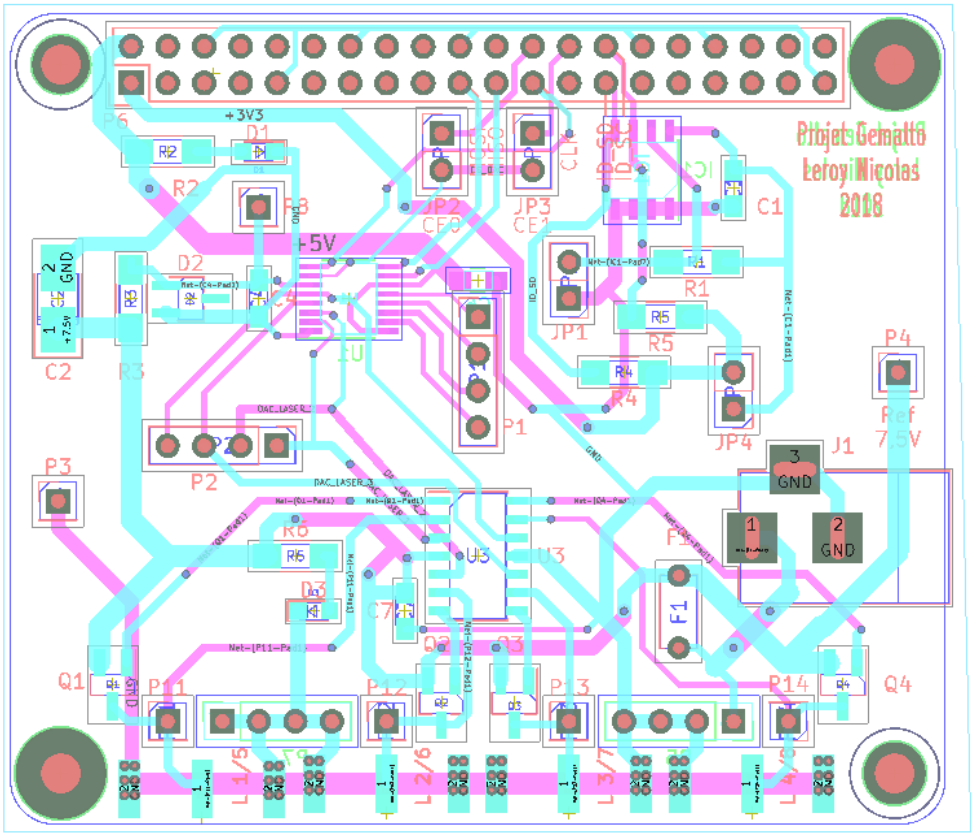
Pour cette 2ème revue, je suis reparti de mon schéma structurel car il n'était pas complet, j'ai ajouté certains composants comme 2 connecteurs de 01x04 afin de pouvoir emboiter mes cartes l'une sur l'autre mais ils serviront aussi à stabiliser mécaniquement la carte.

Nous avons aussi changé le symbole des transistors car il n'était pas le bon de base et rajouté les empreintes.

Ensuite nous avons ajoutés les finitions en ajoutant les codes commandes sur chaque composants de façon a simplifier la recherche des composants qui sont utilisé pour fabriquer cette carte et de pouvoir faire un bon de commande rapidement.

Suite a toutes c'est modifications apportées et la vérification on peut lancer le PCB et pouvoir commencer notre routage…

Routage de la carte Gemalto 2018



Pendant les semaines suivantes nous avons effectués plusieurs modifications pour concevoir ce routage afin qu'il soit opérationnel pour pouvoir l'envoyer au sous traitant.

Pour ma carte les problèmes qui ont été trouvés étaient d'abord en premier lieu ma disposition des composants qui étais incorrecte, je les avais placés de manière que ça soit le plus pratique sans avoir respecté le schéma structurel qui à était fait ultérieurement.

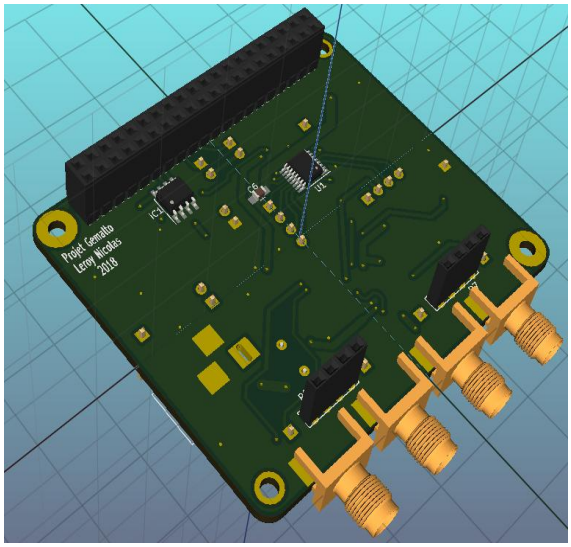
Après cette remarque j'ai utilisé mon schéma structurel afin de placer mes composants aux endroits correspondant car certains des composants devaient être placés à côté d'après les documentations du fabricant pour une utilisation correct comme par exemple mettre un condensateur de 100nF proche du LTC2620.

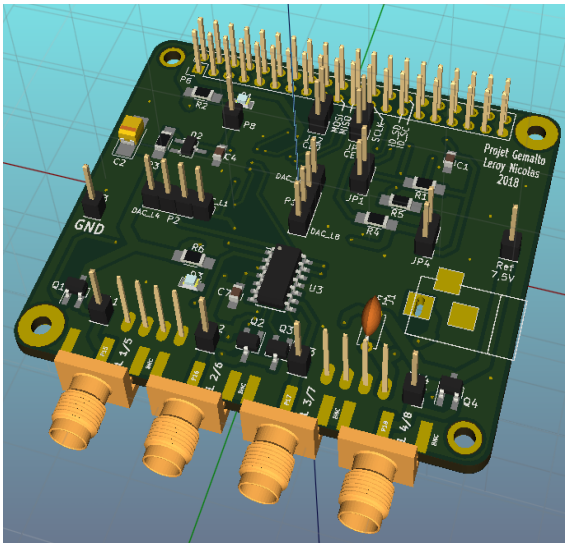
Suite à ce changement des composants j'ai dû refaire toutes mes pistes de routage de zéro ; j'ai relié de manière à ce que sa soit le plus claire possible en respectant la largeur des pistes qui a été donnée : GND → 1.000mm, 7.5V/5V/3.3V → 1.500mm et de 0.600/0.400 pour les autres pistes.

J'ai rajouté des trous traversant afin d'avoir la masse partout sur ma carte et aussi des repères de texte pour les broches (ID_SD, ID_SC, MOSI, MISO, CLK, CE0 et CE1) pour une utilisation plus simple.

Pour finir avec le mode 3D de Kicad on peut voir si tous les composants sont bien aux endroits voulus , que rien n' est à l'envers et que les références ne débordent pas ou bien qu'elles soient à l'envers…

Avec les empreintes assignées précédemment on peut visualiser notre carte avec tous ces composants.



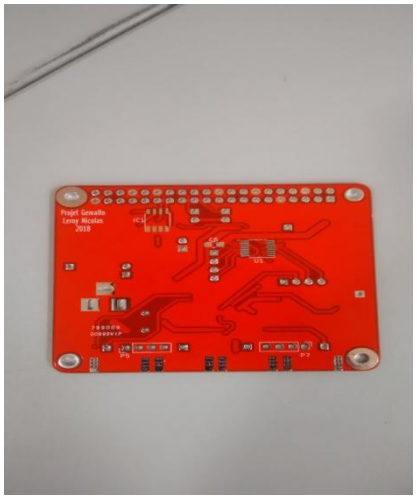


Après les vérifications complètes de la carte qu'il n'y ait aucune erreur on peut créer un « tracer » qui permet de créer des fichiers qui seront envoyés au sous traitant; puis on groupe tout ces fichiers dans un dossier .zip.

On va sur le lien Seed Studio afin de vérifier qu'avec nos fichiers gerber le routage n'ait pas d'erreurs et si le site accepte le fichier, on a un Gerber viewer pour voir que tout est correct et que cela nous convient; alors on peut sélectionner les critères dont on a besoin puis lancer la commande.

Fabrication de la carte :

Après avoir finalisé notre commande notre carte va être fabriquée industriellement et nous allons pouvoir commencer à monter notre carte.



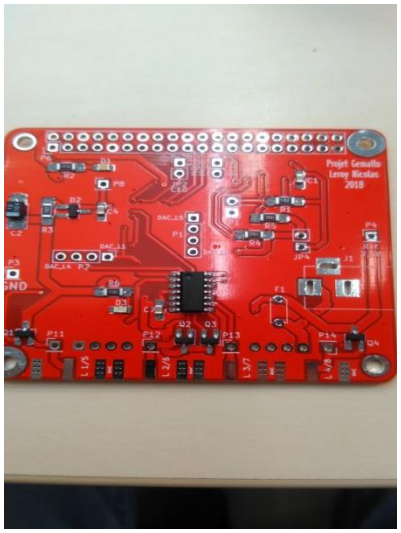
Pour commencer nous allons d'abord souder les composants en CMS a l'aide de pochoir que nous avons commandé en même temps que les PCB puis on ajoute de la pate à braser sur les empreintes CMS et de les souder a l'aide du four a refusion.

Aperçu du pochoir

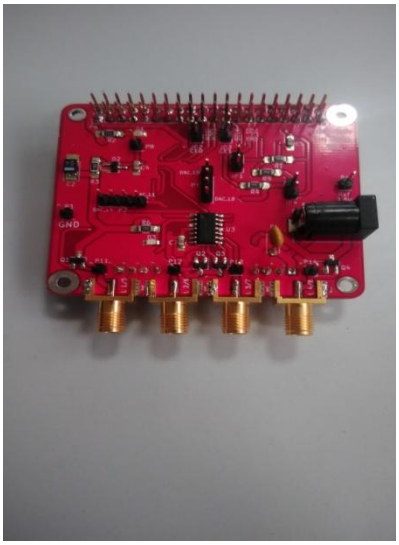
Four à refusion

Ensuite nous faisons de même pour le dessous de la carte.

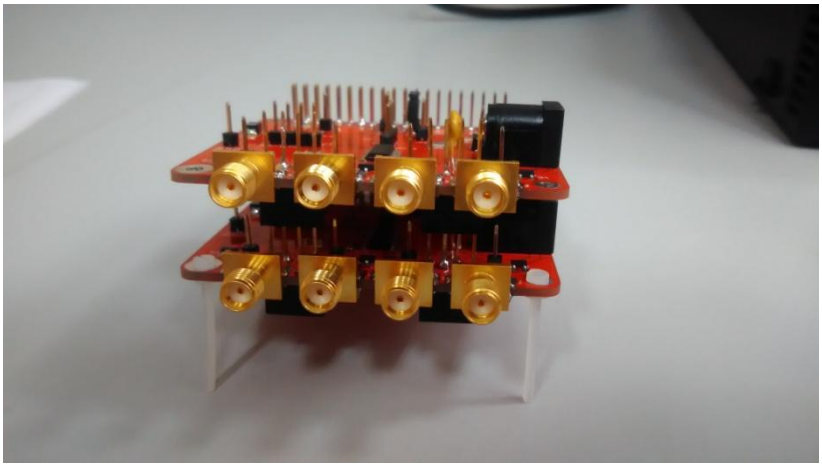
Pour finir nous soudons les composants traversants du plus courts aux plus grands pour ne pas qu'ils nous gênent pendant la manipulation, question de praticité.



Composant CMS



Avec les composants traversants

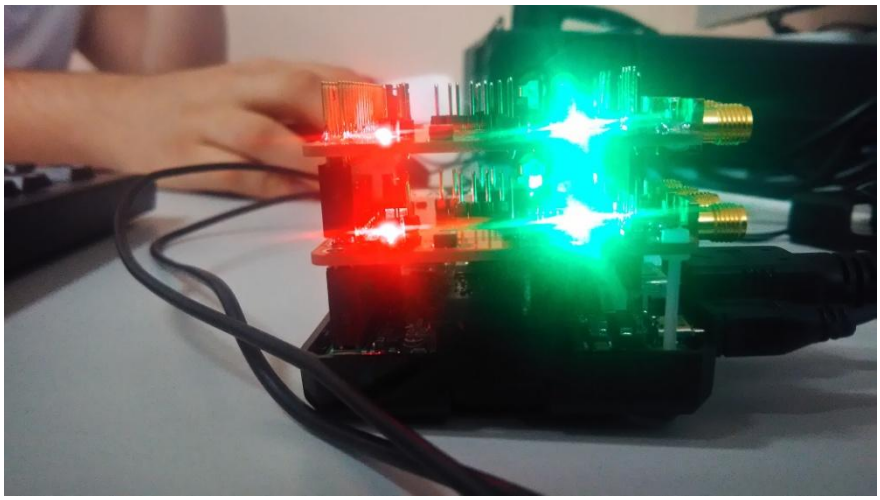


Rendu Final de la carte avec les 8 sorties

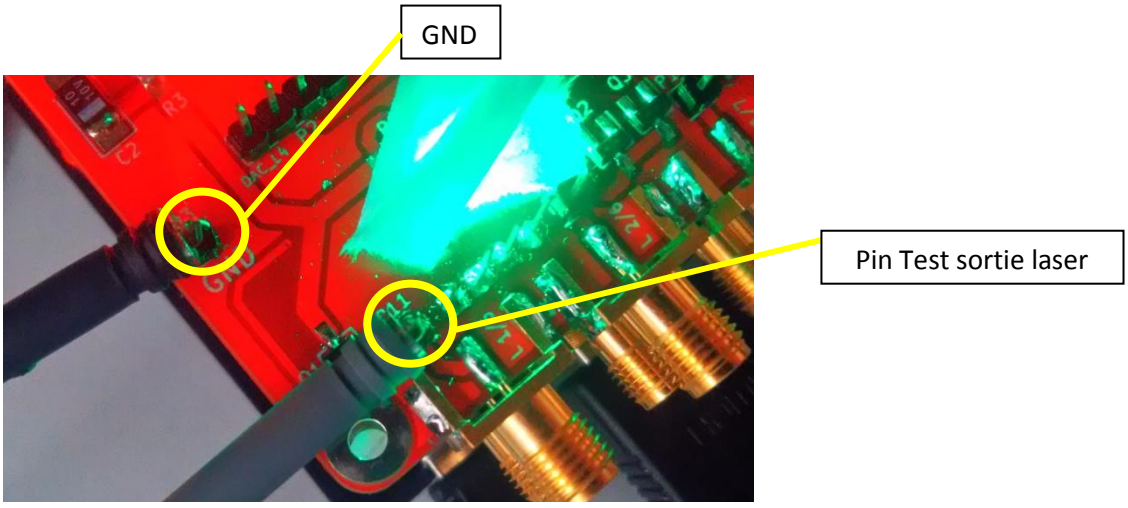
Après la fin de la fabrication de nôtre carte nous allons pouvoir faire nos tests pour voir si tout fonctionne correctement.

Fonctionnement de la carte Gemalto

On empile les 2 cartes l'une sur l'autre afin d'avoir nos 8 sorties fonctionnels puis les connectées à la carte raspberry.

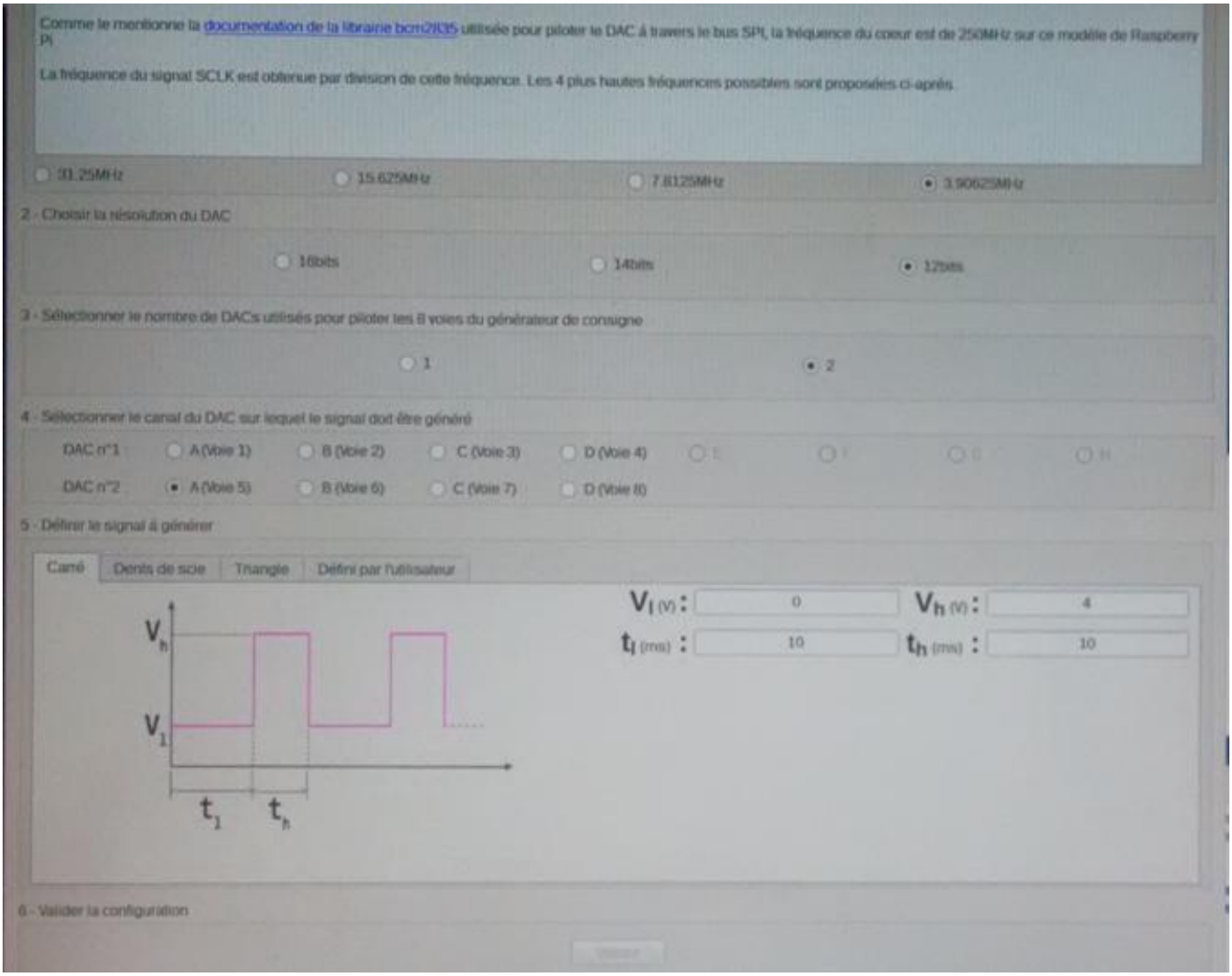


On effectue les tests de la carte en branchant l'oscilloscope, un sur le GND et l'autre sur un pin test de sortie du DAC que l'on veut mesurer. Une fois branché on utilise l'interface qui permettra de piloter les lasers.



Ensuite on ouvre la nouvelle l'interface créé par M.Defrance qui permet de piloter les 8 sorties; par exemple on choisit la sortie 5 de la carte.

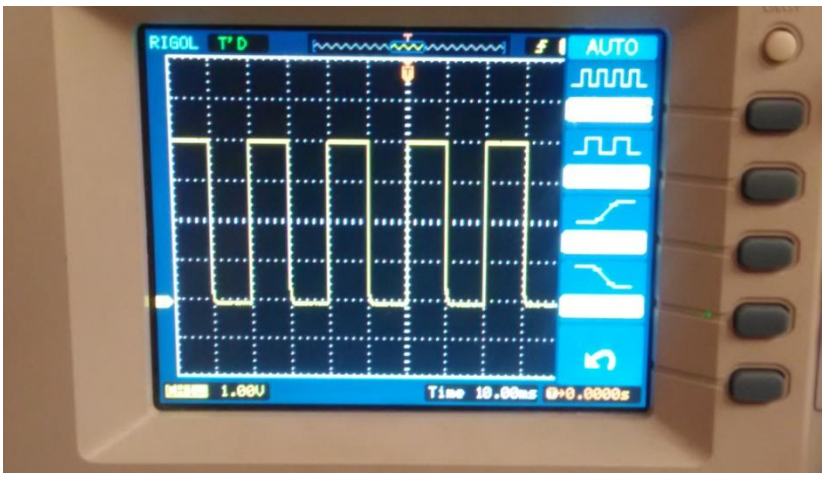
On sélectionne ensuite sur l'interface les coordonnées voulu, dans notre cas on prend la résolution 12bits, puis la solution 2 qui es celle avec les 2 cartes puis la voie que l'on veut tester (voie 5) et aussi de quel carte on choisit donc le DAC_2 (pour le cas de la voie 5 on met le cavalier sur CE1) et on met les données que l'on veut envoyer (signal, valeurs, temps),[voir interface ci-dessous].



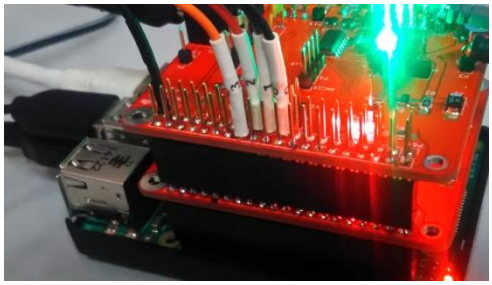
Voici la mesure obtenu avec les données que l'on a attribué, on obtient bien les 4V avec un temps de 10ms.

On effectue cela avec toutes les sorties afin de vérifié que tout fonctionnent correctement.

Le test est donc concluant.



Signal reçu



On peut aussi faire le test en numérique grâce a Saleae en utilisant le même principe.

En cas de défaillance de la carte

En cas d'un problème sur la carte nous pouvons faire des tests grâce au pin qui ont était rajouté au cas où cela arriverais comme le pin « ref 7.5V » pour voir si la tension est bien envoyé ou bien les pins des SMA « p11, p12, p13, p14 »

Conclusion :

La tâche qui nous à était donnée au début de ce projet Gemalto qui était de reprendre l'ancienne conception de la carte d'extension afin d'y ajouté 4 sorties supplémentaires.

Nous avons suivi les consignes du cahier des charges pour le bon déroulement des tâches à effectuer.

L'alimentation des lasers s'effectue désormais par un bloc secteur de 7.5V (RS : 448-376) pour que les 8 sorties obtiennes un courants identiques sur chaque sorties (100mA).

Les cartes ont était conçus dans les temps et sont opérationnels.

Annexe

Journal de bord		Leroy Nicolas
Date	***Intervenant***	***Objet***
• 10/01/2018 Mercredi	• Bonnet • Leroy •Lando	-Prise en compte du cahier des charges et des objectifs à réaliser Sur le projet GEMALTO. (2Heures)
• 11/01/2018 Jeudi	•Bonnet • Leroy	- installation de Kicad -Présentation des changements à apporter au schéma 2017. -Récupérations des documents nécessaire. (4Heures)
• 12/01/2018 Vendredi	• Lando •Bonnet •Leroy	-Réalisation du diagramme de Gantt. -Début du tuto sur kicad. (3Heures)
• 17/01/2018 Mercredi	• Leroy	-Analyse des composants. -compréhension des composants utilisés. -Changement apporté sur le schéma (test). (4heures)
• 18/01/2018 Jeudi	•Leroy	-Création d'une liste des composants utilisés sur Kicad. -Modification appliquée sur le schéma 2017. (2heures)
• 19/01/2018 Vendredi	• Leroy	ABS
• 24/01/2018 Mercredi	• Bonnet •Leroy	-Réglages des librairies et des empreintes. -Compréhension des changements qui ont étaient apportés sur la carte exigé par le client. - Test de la Carte. (4heures)
• 25/01/2018 Jeudi	•Bonnet •Leroy	-Utilisation du programme QT pour utiliser la carte. -Utilisation du mode Spi de la carte Gemalto 2017. (4heures)
		Leroy Nicolas
• 26/01/2018 Vendredi	• Bonnet •Leroy	-Relevé de la trame générée sur les sorties de l'oscilloscope. -Relevé des trames sur Saleae sur le mode spi (signal carré). -Compréhension des données reçues sur Saleae.

		(3heures)
• 31/01/2018 **Mercredi**	**•Bonnet** **•Leroy**	**-Problème de décalage sur Saleae résolu (SPI).** **-Relevé des trames sur Saleae sur le mode spi (signal dent de scie).** **- Relevé des trames sur Saleae sur le mode spi (signal triangle).** **-Tentative de régler les bugs des PNG du logiciel de test sur Qt.** **(4heures)**
• 01/02/2018 **Jeudi**	**•Leroy**	**- Régler un problème de librairie sur KiCad.** **- Fiche KiCad; réalisée.** **- Fiche des composants; réalisée.** **- Présentation de notre projet.** **-Assemblage du routage sur KiCad.** **(7heures)**
• 02/02/2018 **Vendredi**	**•Leroy**	**-Tuto sur le routage (KiCad).** **-Suite de routage.** **(3heures)**
•08/02/2018 **Jeudi**	**•Leroy**	**-Modification du schéma de câblage.** **-Suite de la carte du schéma routage de la carte 2017.** **(3heures)**
• 09/02/2018 **Vendredi**	**•Leroy**	**-Constatation d'un problème pour le routage du schéma de routage 2017.** **-Début du routage sur un nouveau schéma de routage (de 0).** **(3heures)**
• 15/02/2018 **Jeudi**	**•Leroy**	**-Prise en compte du mail et effectuer les modifications à apporter.** **-Test de l'alimentation extérieure (6V).** **(3heures)**
• 16/02/2018 **Vendredi**	**•Leroy** **•Lando** **•Bonnet**	**-Début du rapport de revue en équipe.** **-Rajout des codes commande RS sur le schéma KiCad.** **-Suite du routage.** **(3heures)**

• 21/02/2018 Mercredi	•Leroy •Bonnet	-Fin du routage. -Installation du nouveau programme. - Utilisation et compréhension du programme. (4heures)
• 23/02/2018 vendredi	•Leroy	-Compter combien de composants on a besoin pour nos cartes. -Inventaires des composants restants. -Début de la Revue de projet. (3heures)
• 14/03/2018 Mercredi	•Leroy •Bonnet	- Avancement sur la revue 1. -Création de la diapo. (4heures)
• 15/03/2018 Jeudi	•Leroy	-Reprise des schémas des nouvelles modifications données. -Fin de la revue 1 (partie personnelle). (3heures)
• 16/03/2018 Vendredi	•Leroy •Bonnet	-Présentation du dossier de la 1ére revue. -Refaire les essaies de test.
• 21/03/2018 Mercredi	•Leroy •Bonnet	-Apprentissage routage KiCad.
• 22/03/2018 Jeudi	•Leroy •Bonnet •Lando	-Préparation de l'oral de la revue. - Refaire les essaies de test.
• 23/03/2018 Vendredi	•Leroy •Bonnet •Lando	Passage 1ère revue
• 28/03/2018 Mercredi	•Leroy	-Petite modification sur le schéma structurel. -Correction des fautes a rectifié sur la revue 1.
• 29/03/2018 Jeudi	•Leroy	-Avancement sur le rapport 2èmerevue.
• 30/03/2018 Vendredi	•Leroy •Bonnet	-Commencement du routage. -Fin du routage (à faire vérifier).

•04/04/2018 Mercredi	•Leroy •Bonnet	-Modification sur le routage de 0. -Fin du routage (à faire vérifier).
•05/04/2018 Jeudi	•Leroy •Bonnet	-Modification sur le routage de 0. -Fin du routage (à faire vérifier).
•06/04/2018 Vendredi	•Leroy •Bonnet	-Suite routage. -Respectée les nouvelles consignes.
•11/04/2018 Mercredi	•Leroy •Bonnet	-Finition apporté au routage. -Suite sur la 2ème revue.
•12/04/2018 Jeudi	•Leroy •Bonnet	-Commencement sur la partie physique. -Routage Complet. -Création en fichier Gerber.
•13/04/2018 Vendredi	•Leroy	- Suite sur la 2ème revue.
•18/04/2018 Mercredi	•Leroy	-Suite sur la 2ème revue.
•09/05/2018 Mercredi	•Leroy	-Suite sur la 2ème revue. -Avancement sur le diaporama.
•11/05/2018 Vendredi	•Leroy	-Suite sur la 2ème revue.
•16/05/2018 Mercredi	•Leroy	-Montage de la carte Gemalto
•18/05/2018 Vendredi	•Leroy	-fin du montage de la carte Gemalto
•23/05/2018 Mercredi	•Leroy •Bonnet	-Test de la carte Gemalto 2018.

Partie commune :

Bonnet Nicolas

Leroy Nicolas

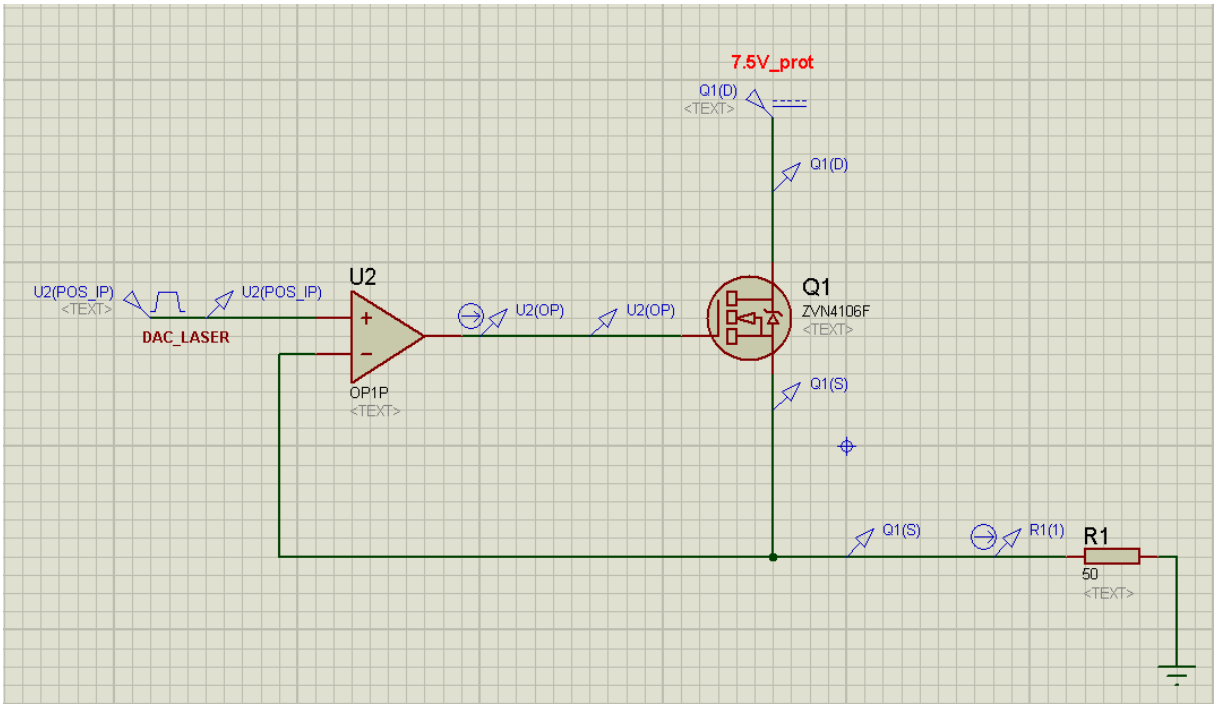
GEMALTO

Etude de la structure amplificateur/transistor

Nous avons réalisé une étude de la structure amplificateur/transistor présent dans notre schéma structurel.

Cette étude a été réalisée avec le logiciel Proteus afin de pouvoir disposer de la fonction Analogue Analysis permettant de retranscrire les variations de courant et de tension qui nous intéresse.

Le schéma structurel ci-dessous permet la simulation de la réception d'une commande sur une des sorties. Cette structure associant l'amplificateur et le transistor est un suiveur de tension.



La commande DAC_LASER entre par l'entrée + de l'amplificateur alimenté en 7.5V, ici la commande est un signal continu de 5V. Cette commande est ensuite envoyée au transistor et va jusqu'à la résistance R1 qui représente la sortie SMA. **D'après la documentation technique l'impédance du jack SMA est de 50Ω donc R1 = 50Ω.**

Tout d'abord, nous avons étudié le comportement de cet ensemble lorsqu'un signal continu est envoyé.

Simulation d'un signal continu de 5V résistance avec une R1 = 50Ω.

Commande 5V en continue

5V en sortie du transistor

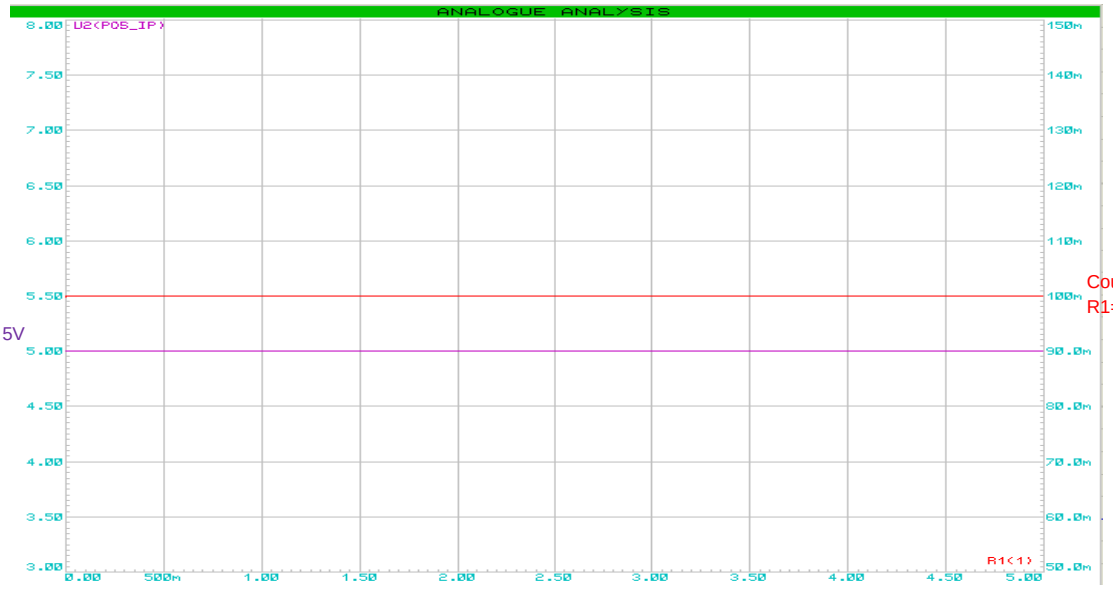
En sortie du transistor on retrouve les 5V continue de la commande.
U2(POS_IP) = 5V→ commande en entrée de l'amplificateur.
Q1(S) = 5V→ sortie du transistor.

On ne remarque pas de différence de tension entre la commande et la sortie du transistor.

Mesure du courant en sortie du transistor pour un signal continue de 5V :

$I_{max} = \frac{5V}{50\Omega}$ = 100mA.

100mA est la valeur maximale du courant en sortie attendu d'après le cahier des charges.



Commande 5V en continue

Courant au niveau de R1=100mA

Simulation d'un signal continu de 5V avec une résistance R1 = 25Ω

En sortie du transistor on retrouve les 5V continu de la commande comme précédemment.

Commande 5V en

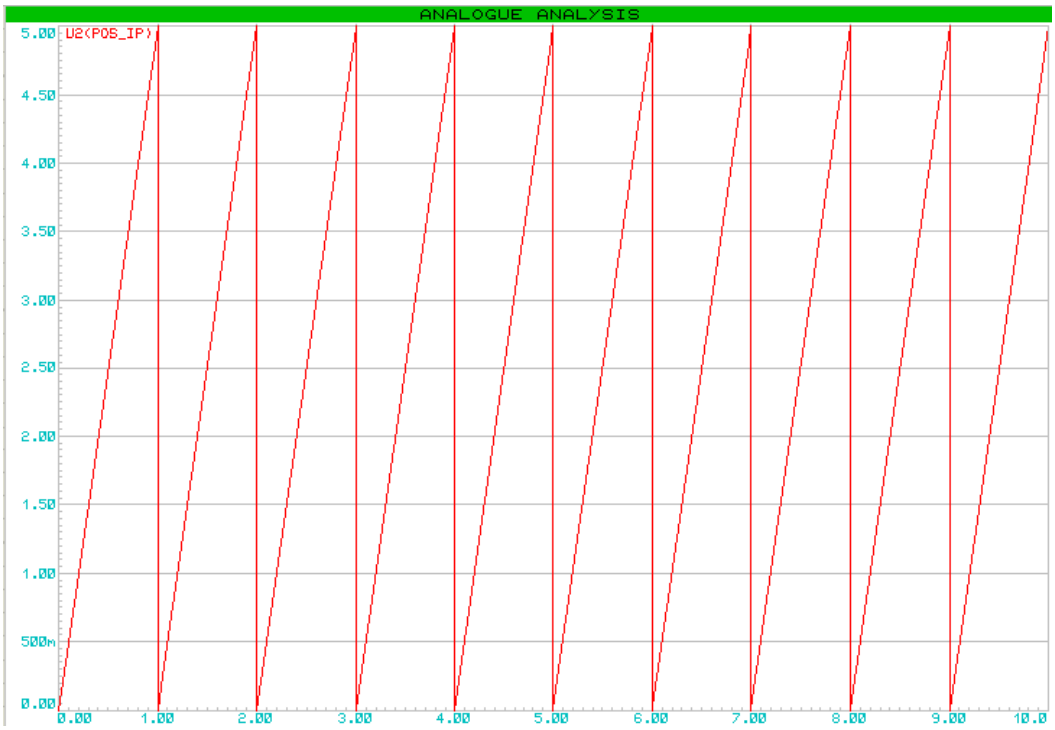
Courant au niveau de R1=200mA

Le courant I_{R1} est de 200mA continu.

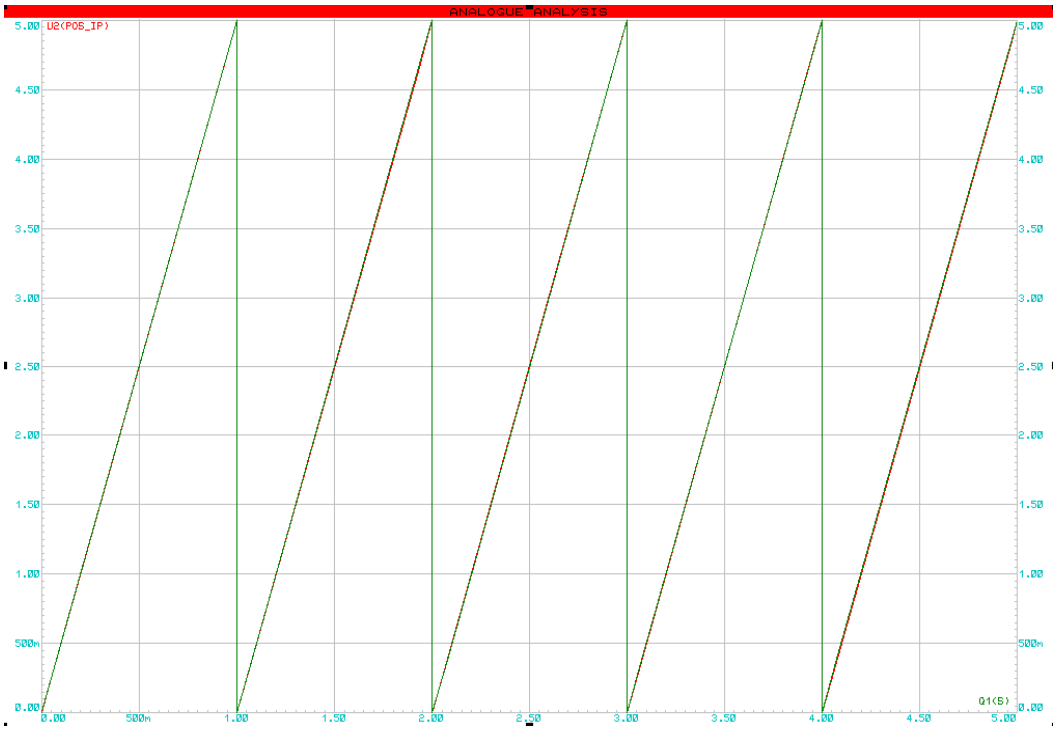
On a **diminué de moitié la résistance R1 ce qui a doublé la valeur du courant** en sortie du transistor.

On observe donc que l'amplificateur est linéaire. Plus la résistance est faible, plus le courant est fort.

Simulation d'un signal en dent de scie variant de 0V à 5V est envoyé à l'amplificateur:



Signal émit par le DAC.



Signal en sortie de l'ampli et du transistor

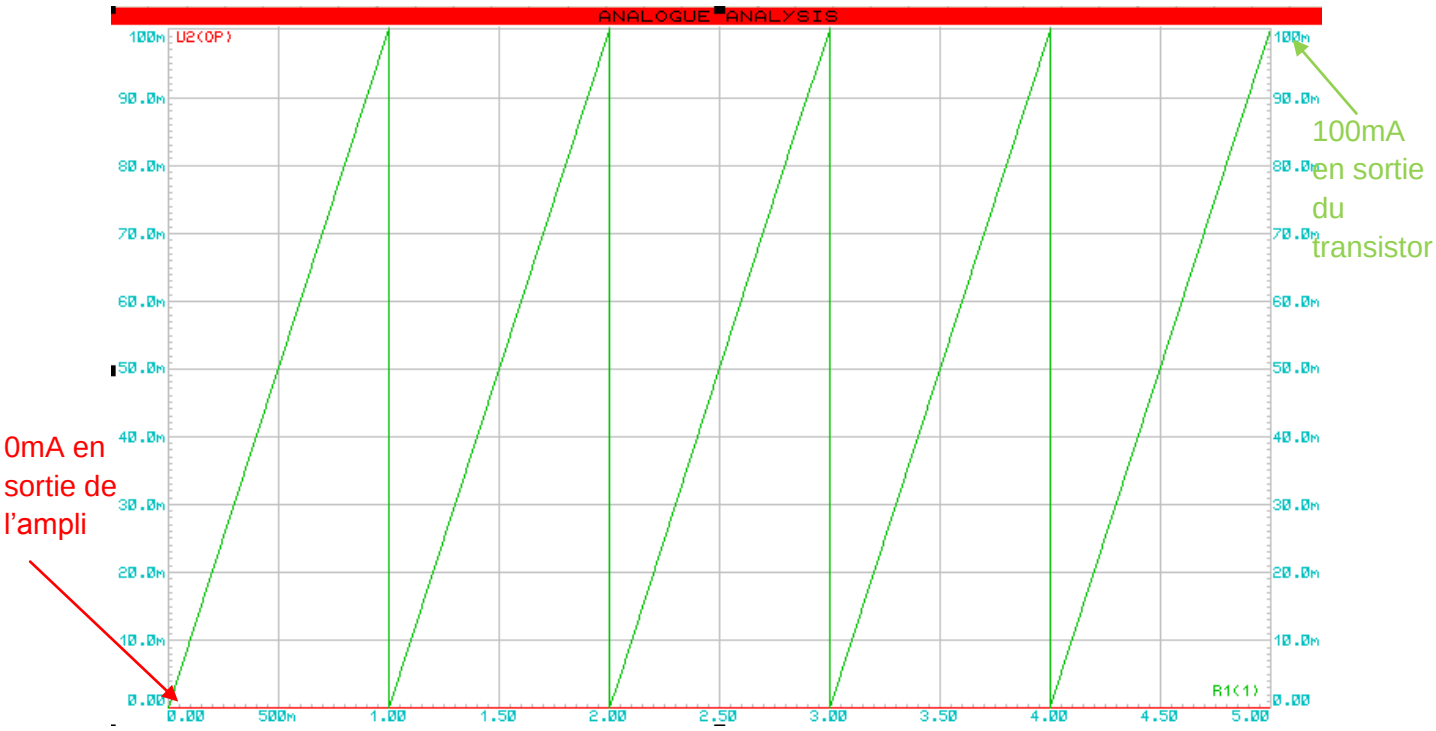
Les tensions en sortie de l'amplificateur et en sortie du transistor ne changent pas.

La différence se fera pour le courant.

Les deux signaux ci-dessous représentent **le courant qui passe avant et après le transistor**, on peut voir qu'**entre l'amplificateur et le transistor le courant est de quasiment 0mA.**

Juste après le transistor le courant est en dent de scie et varie entre 0 et 100mA.

On sait que lorsqu'on met le maximum (5 volts) en signal d'entrée de l'ampli on doit avoir 100mA au niveau de la sortie du SMA.



On envoi un signal en dent de scie variant entre 0 et 5V.

La tension passe par l'amplificateur et le transistor sans modification apparente.

Le courant est quasi inexistant en sortie de l'amplificateur mais en sortie du transistor le courant créer un signal en dent de scie variant entre 0 et 100mA.

Le courant en sortie du transistor est image de la commande demandé et permet de piloter le laser.

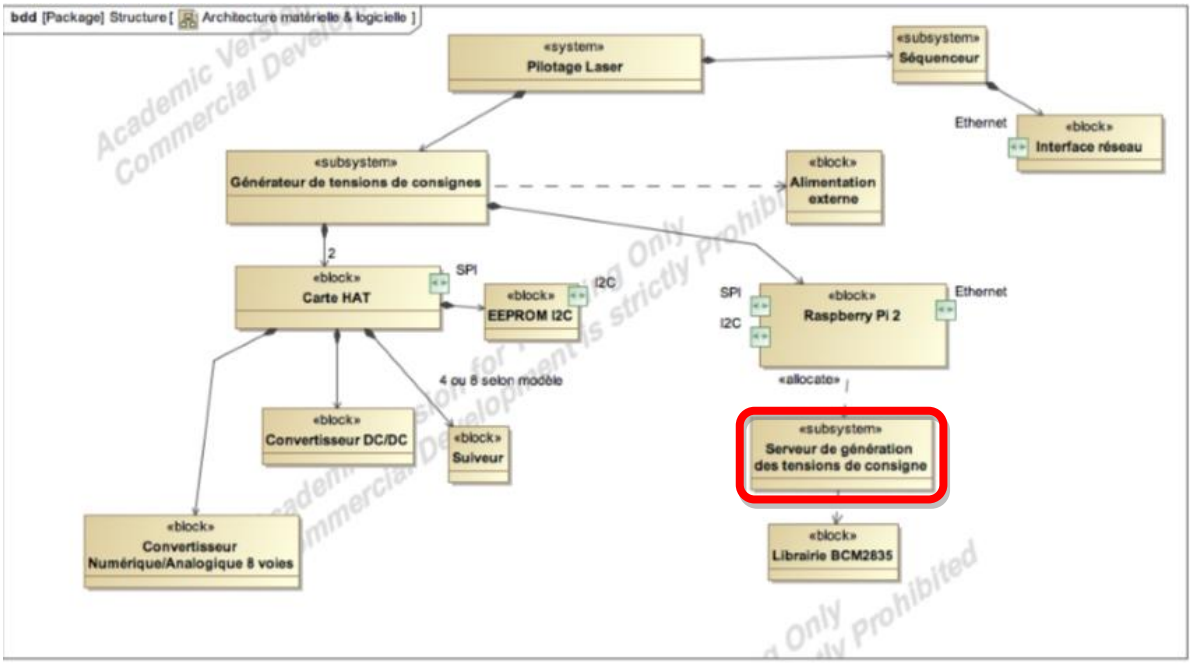
L'intérêt de mettre cette structure entre le DAC et les sorties SMA est de ne pas dénaturé la consigne à l'entrée de l'amplificateur, le DAC, de lui-même ne peut fournir les 800mA attendu si les huit sorties sont à 5V. C'est la structure qui permet de fournir ces 100mA par sorties à partir du bloc secteur externe tout en respectant les consignes.

Partie personnelle :
Lando Gwendal

GEMALTO

Présentation de ma partie :

Ma partie apparait en rouge dans le diagramme de blocs ci-dessous :



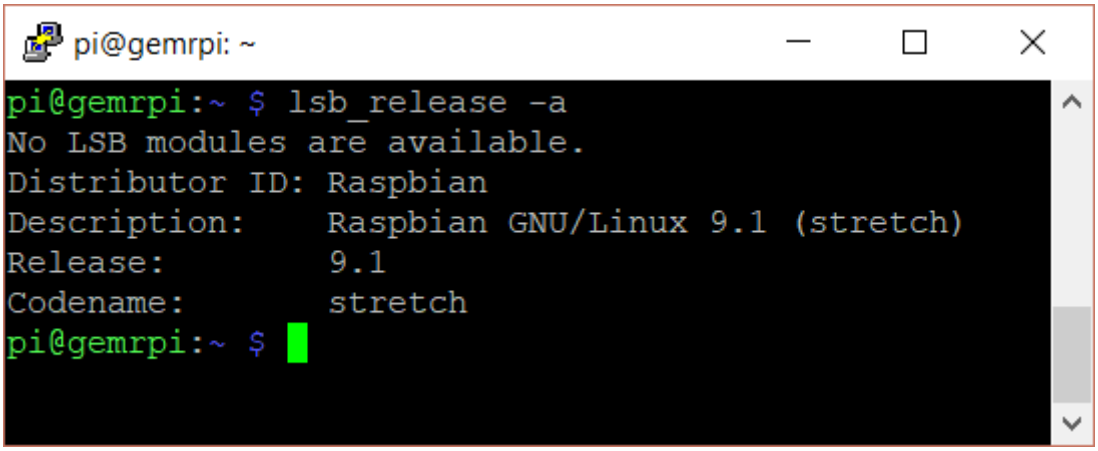
Comme évoqué dans les objectifs de la partie commune du dossier, ma tâche consiste donc à :

- reprendre le codage (en C++ standard) d'un serveur TCP destiné à recevoir puis interpréter des commandes issues du séquenceur dont le rôle est de superviser à distance les consignes de pilotage des lasers
- implémenter les différentes commandes du cahier des charges à savoir :
  - CmdVersionSoft : permet au séquenceur de récupérer la version logicielle du générateur de consignes
  - CmdVersionHard : récupère la version matérielle du générateur de consignes
  - CmdSetPowerLevel : fixe la puissance d'un ou de plusieurs lasers (voire tous)
  - CmdGetPowerLevel : récupère la puissance courante d'un ou de plusieurs Lasers (voire tous)
  - CmdPowerDown : coupe un ou des lasers (voire tous)
  - CmdClose : ferme la communication avec le générateur de consignes
- Prise en charge transparente des deux configurations matérielles possibles (1x8 voies ou 2x4voies).

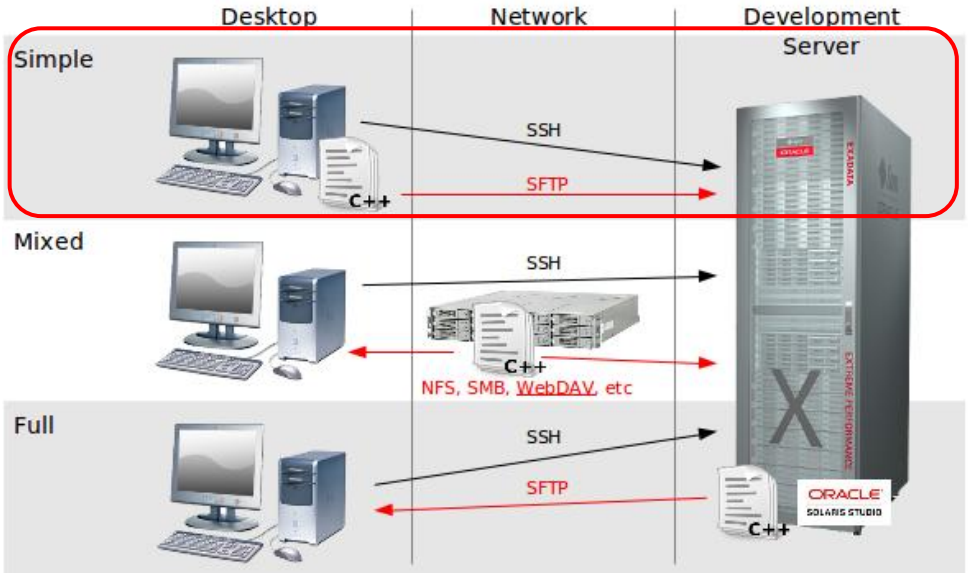
Le code C++ devra se répartir sur différentes classes pour respecter les principes de la programmation orientée objet.

Environnement logiciel utilisé

- **Windows 10** pour la machine de développement personnelle / **Windows 7** pour la machine de développement du lycée
- **OS Raspbian v9.1** (stretch) pour la Raspberry Pi

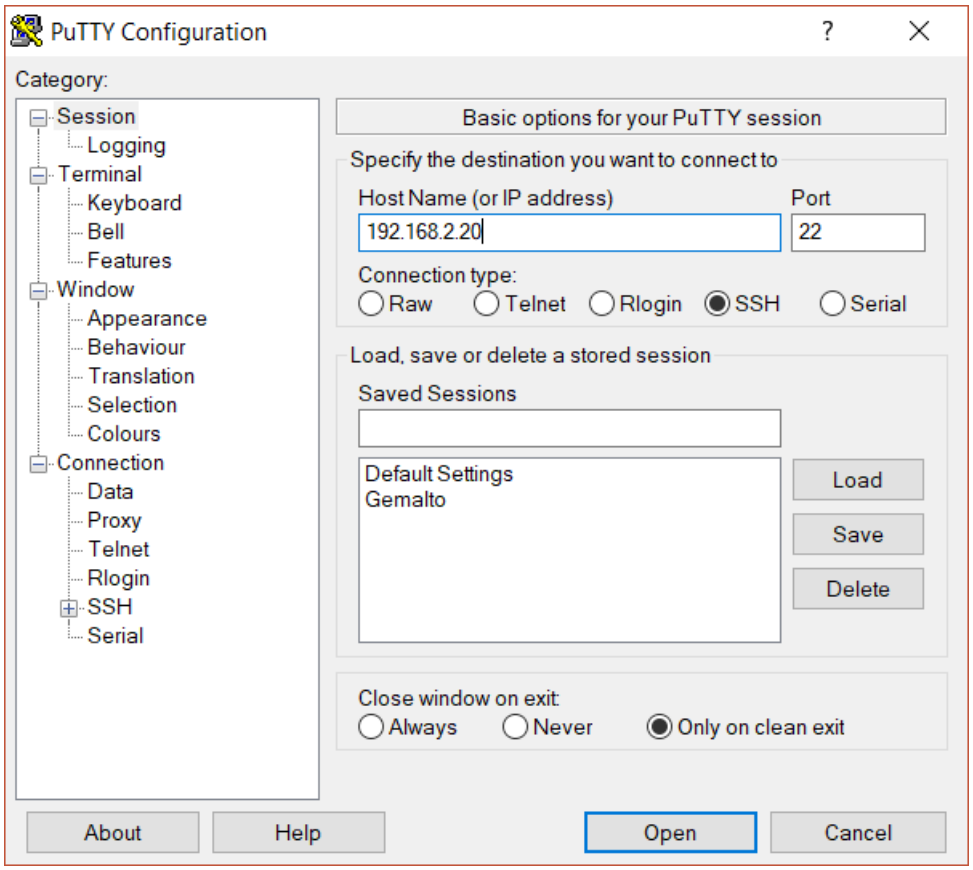


- **Netbeans 8.2** : Environnement de développement Intégré C++ d'Oracle (passé sous licence Apache récemment) qui autorise le développement à distance selon différents modes

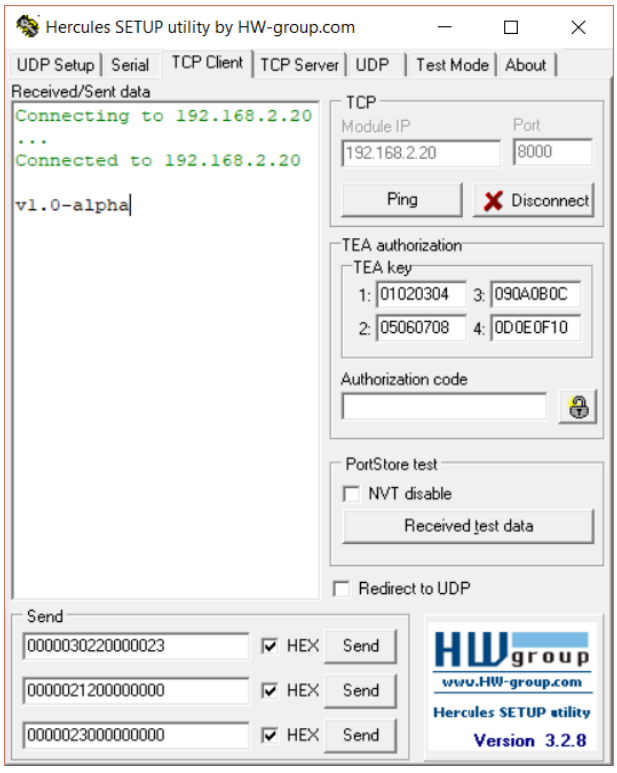


Dans le cadre du projet, nous utilisons le mode « Simple » :
Les fichiers source résident sur la machine de développement (mon PC). Lorsqu'on demande de construire l'exécutable, les fichiers sources sont tranmis sur la Raspberry Pi (« Development server » sur le schéma ci-dessus) via le protocole SFTP (protocole de transfert de fichiers de SSH).

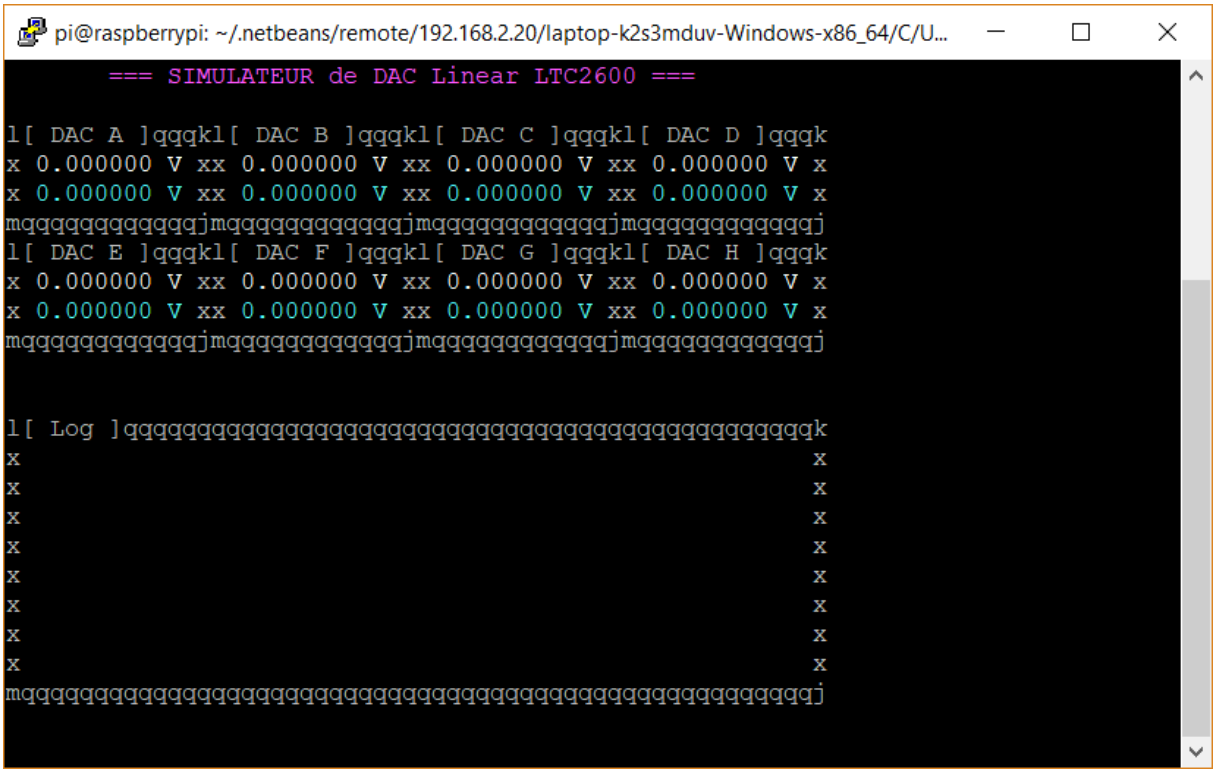
- **PuTTY** : client SSH pour prendre le contrôle à distance de la Raspberry Pi utilisée en mode Headless (pas de clavier, pas d'écran)



- **Hercules** : utilitaire réseau de « HW Group »pouvant jouer le rôle de client ou server TCP



- **Simulateur de DAC** : simulateur codé l'an dernier pour permettre de valider le fonctionnement de l'application en l'absence de la carte d'extension réelle



Planning

Ceci est le planning prévu à la base avant de recevoir de nombreux conseils et révisions de priorités de tâches :

Tâches LG	192 h	Ven 12/01/18	Jeu 17/05/18	7
Installation des logiciels nécéssaires	2 h	Ven 12/01/18	Mer 17/01/18	7
Revoirle l'ancien code	50 h	Mer 17/01/18	Jeu 08/02/18	9
Création du serveur	50 h	Jeu 08/02/18	Mer 21/03/18	10
Mise en TLV des comunication	40 h	Mer 21/03/18	Ven 06/04/18	11
Implémentation des commandes	50 h	Ven 06/04/18	Jeu 17/05/18	12

Celui-ci en n'est un plus proche de la réalité :

Tâches LG	122h	Ven 12/01/18	Jeu 17/05/18	7
Installation des logiciels nécéssaires	2 h	Ven 12/01/18	Mer 17/01/18	7
Revoirle l'ancien code	8 h	Mer 17/01/18	Jeu 08/02/18	9
Création et mise en place des tag	36 h	Jeu 08/02/18	Mer 21/03/18	10
Etude des nouveux codes fournit	8 h	Mer 21/03/18	Ven 06/04/18	11
Implemantation pour reconnaitre les différentes configuration	68 h	Ven 06/04/18	Jeu 17/05/18	12

Plan d'itération

Itérations	Tâches
1	Programmation des fonctions manquantes
2	Assurer la comptabilité du serveur sur les configurations matérielles possible
3	Assurer le bon fonctionnement du serveur sur le hardware

Itération n°1

Test du simulateur et des fonctions du serveur déjà implémentés.

Etudes du code pour créer les fonctions manquantes :

Le client envoie une trame :

00000212 |	00000001 |	03
Tag indiquant à quel fonction elle s'adresse	Le Lenght qui indique que le Value contient un octet	Le Value contenant les données à transmettre au serveur ici ''3'' en héxadécimal

Le serveur reçoit la trame, identifie le Tag, active la fonction relier et si besoin, renvoie une réponse, par exemple, imaginons qu'elle active la fonction qui renvoie le chiffre ''500''
Le serveur envoie la trame suivante :

00000212 |	00000002 |	01F4
Le Tag indiquant Qu'il s'agit de la même Fonction	Le Length indiquant que le Value à une taille de 2 octets	Le Value contenant ''500'' en héxadécimal

Ceux-ci fut suivit des tests de chaque fonctions via le simulateur et le logiciel ''Hercules'' comme client.

Itération n°2

Intégration d'arguments au lancement du simulateur pour lui indiquer à quelle configuration matérielle (1x8 voies ou 2x4 voies) il doit se conformer.

Intégration dans les trames de valeurs indiquant sur quelle configuration matérielle on s'adresse, telle que DAC_B ou DAC_A qui indique sur quelle DAC écrire et CE0 ou CE1 qui indique à quelle configuration on s'adresse.

Itération n°3

Implémentation du code sur les deux configurations matérielles réelles.

Vérification du fonctionnement du serveur via l'utilisation de l'interface homme-machine fournit.

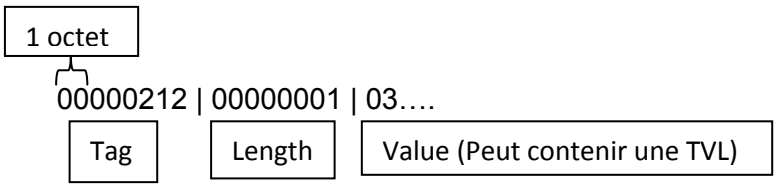
Les activités mises en œuvre :

Dans un premier temps, je me suis concentré sur le codage de l'interface en me basant sur des codes C++ réalisés durant les cours sur les serveurs TCP/IP, mais ceci fut mis de côté suite à une discussion avec mon professeur référent qui m'a plutôt conseillé de me concentrer sur le serveur du coup j'ai passé en revue le code effectué l'année dernière et compris comment rajouter des fonctions dans le code et ainsi j'ai pu créer le tags des fonctions manquantes.

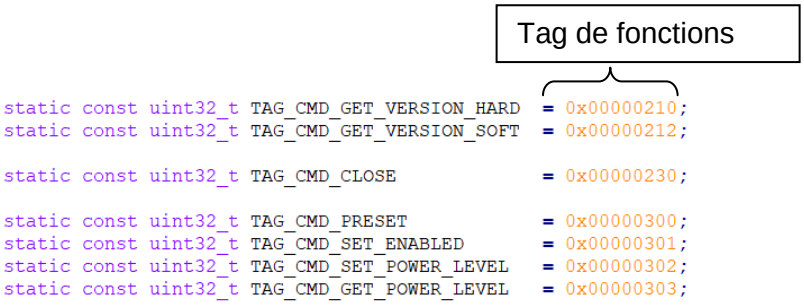
Celle-ci s'intègre dans le format d'envoie de commande demandée qui est le TLV (Type-Length-Value) le tag correspond au Type, le Value correspond à la réponse attendue par la demande, par exemple afficher ''v1.1'' quand on lui demande la version du matériel et/ou les informations pour paramétrer une voies telle que la numéro de voie, la valeur de la tension et etc. Le Length quantifie la taille en octet du Value.

Suite à des modifications acceptées par l'entreprise, le Value peut lui aussi contenir une TLV pour implémenter des arguments aux seins des réponses.

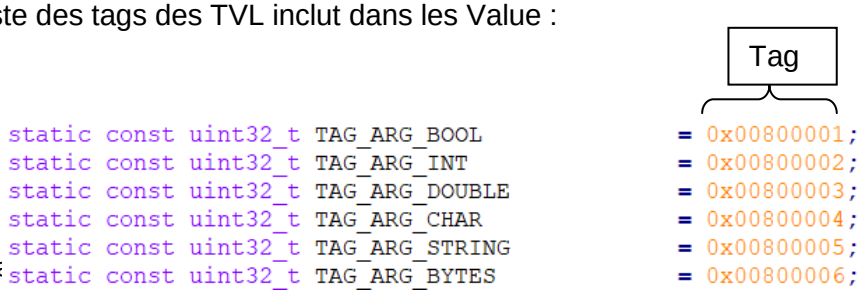
Exemple d'une trame envoyé sous base hexadécimal :



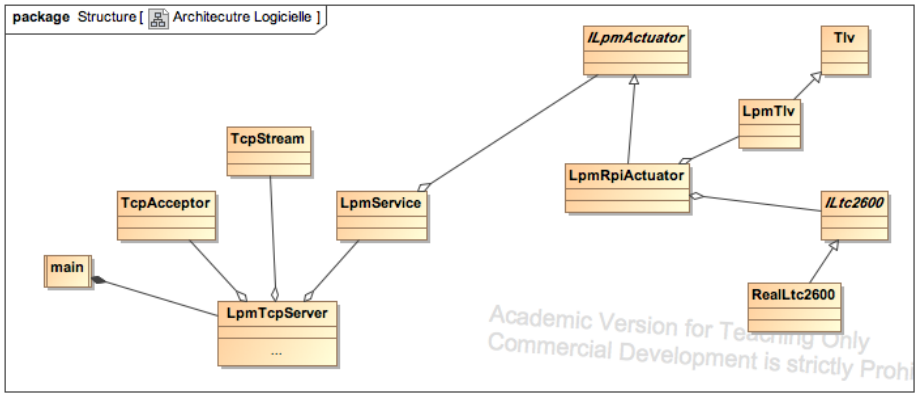
Liste des tags des fonctions :



Liste des tags des TVL inclut dans les Value :



De plus, l'organisation du diagramme de classe fut modifiée par le prof référent comme ceux-ci :



Le serveur devra être allumé au démarrage de la raspberry, celle-ci examinera les arguments utilisés lors de son lancement, pour savoir quelle configuration machine est utilisé – soit 1x8 voies ou 2x4voies - et se mettra en attente d'une connexion de la part d'un client, dans le cas du projet, une interface homme-machine Qt fut fournis pour le teste du bon fonctionnement du serveur et sera normalement opérationnel pour divers démonstration, puis se mettra en attente d'un tag correct envoyé par le client.

Journal de bord :

Vendredi 19 Jan	Installation de l'environnement de développement (Putty, netbeans…)
Mercredi 24 et Jeudi 25 Jan	Compréhension des codes fournit avec quelques explications de la part du prof référent
Vendredi 26 à 9 Février	Codage de l'interface
Mercredi 14 au 23 Fév	Codage du serveur avec mise en place des tags des fonctions manquantes pour qu'au minimum elles répondent
Mer 14 Mars	Les tags répondent.
Jeu 15 au 22 Mars	Etude des nouveaux codes fournit par Gemalto
Ven 23 Mars au 20 Avril	Implémentation pour reconnaitre les différentes configurations