

Sablier Electronique



Table des matières :

1	But du projet.....	Erreur ! Signet non défini.
2	Pré étude	1
2.1	Schéma bloc.....	1
2.2	Matrice à leds	2
2.2.1	Schéma	2
2.2.2	PBC Matrice	3
2.3	PIC32MX795F512L.....	4
2.4	L3GD20H – Gyroscope – (Exclu du projet)	5
2.4.1	Communication.....	5
2.4.2	SPI - L3GD20H.....	6
2.4.3	Principe de fonctionnement d'un gyroscope.....	6
2.4.4	Registres du L3GD20H.....	7
2.5	IAM-20680 – Centrale inertielle. (choix final, 6 axes détection)	8
2.5.1	Communication.....	8
2.5.2	Schéma block	9
2.5.3	Interface SPI – IAM 20680.....	10
2.5.4	Registres du IAM-20680.....	11
2.5.5	Fonctionnement d'un accéléromètre.....	12
2.6	Buzzer	13
2.7	RN4020 - Module Bluetooth	14
2.7.1	Interface recommandée.....	14
2.7.2	Schéma RN4020	14
2.7.3	Schéma block	15
2.7.4	Consommation du module Bluetooth.....	15
2.8	Alimentation step-down.....	16
2.8.1	Estimation de la consommation maximale	16
2.8.2	Choix de l'alimentation step down	17
2.8.3	Calcul de la résistance R12	19
2.9	Schéma de commande de leds.....	20
2.9.1	Multiplexeur - Fonctionnement.....	20
2.9.2	Mosfets principe - commande	21
2.9.2.1	Calculs leds.....	22
3	Conception.....	22
3.1	Module Bluetooth	22
3.2	Alimentation à découpage	23
3.3	Découplage	25
3.3.1	Microcontrôleur	25
3.4	Positionnement	28
3.5	Montage + Modifications	29
3.6	Boîtier 3D.....	30
3.6.1	Bot	30
3.6.2	Top.....	31
3.6.3	Assemblage	32
4	Software.....	33
4.1	Configuration Harmony	33
4.1.1	Timers.....	33
4.1.2	Pinout Harmony	36
4.2	Algorithmes de particules.....	37

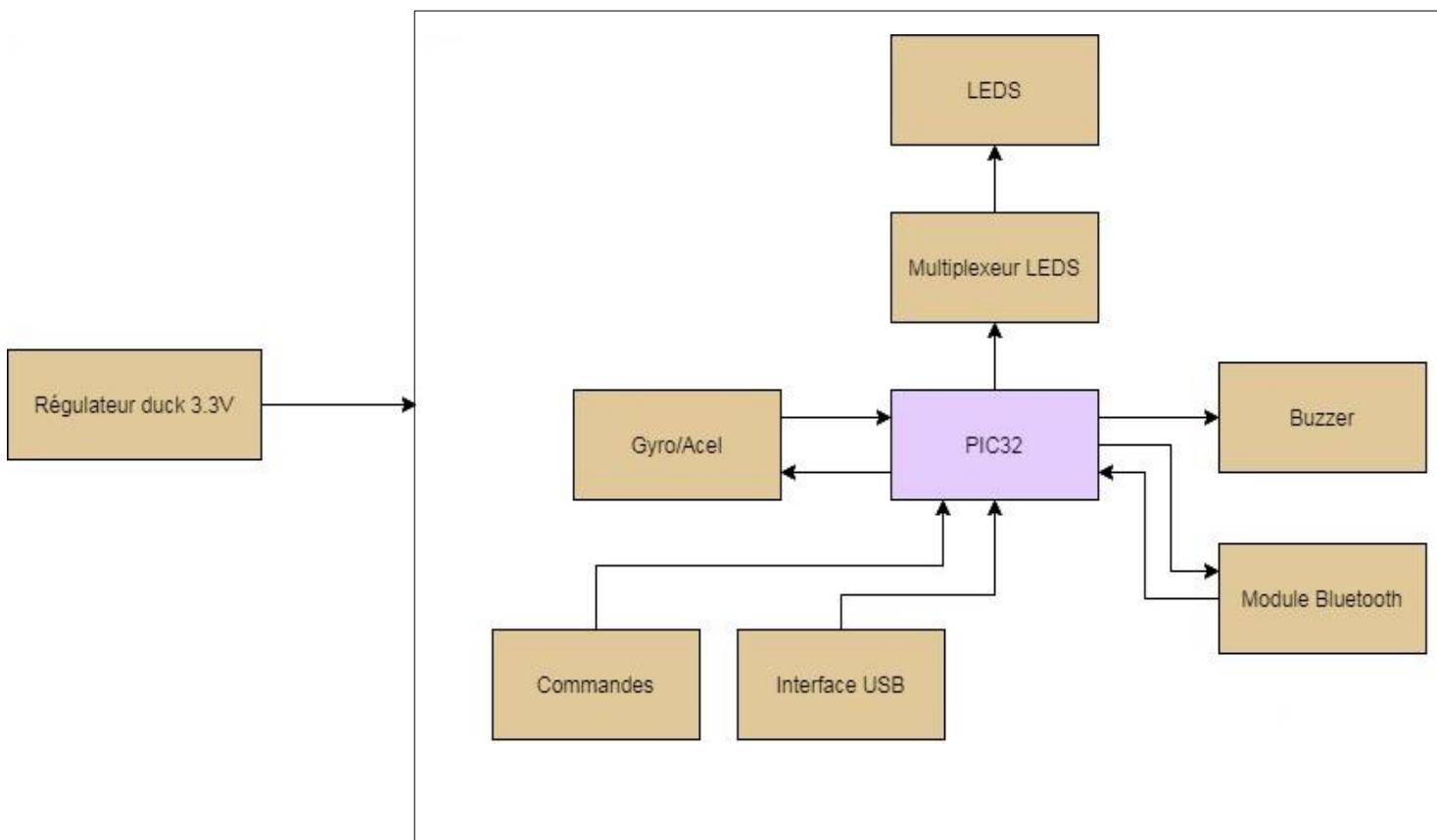
ETML	Sablier Electronique	
4.2.1	Effacement de la matrice	37
4.2.2	DropParticule, transférer une particule d'une matrice à une autre	37
4.2.3	SetXY, fonction pour imposer un état sur un case de l'affichage matriciel	38
4.2.4	UpdateParticules, fonction pour mettre à jours les particules dans l'affichage	38
4.2.5	fillParticules	39
4.3	App USB	39
5	Résultats + mesures	40
5.1	Mesure de l'alimentation	40
5.1.1	Schéma de mesure	40
5.1.2	Résultat	40
5.2	Mesure de la communication SPI	42
5.2.1	Schéma de mesure	42
5.2.2	Mesure	42
5.3	Mesure de la commande leds – demux	43
5.3.1	Schéma de mesure	43
5.3.2	Mesure	43
5.4	Mesures BT	44
5.4.1	Uart to USB converter	44
5.4.2	Configuration Putty	45
5.4.3	Test de configuration	45
5.5	Fonctionnement du sablier	48
5.5.1	Transfert des particules d'un affichage à l'autre	48
6	Bilan	50
6.1	Coûts et liste des pièces	50
6.2	Liste de matériel	50
6.3	Améliorations	50
6.3.1	Software	50
6.3.2	Hardware	51
6.4	Conclusion	52

1 INTRODUCTION

Nous allons voir dans ce projet un sablier électronique. Ce projet est destiné à contrôler le temps lors des épreuves pour les élèves de l'ETML-ES. Nous allons voir un sablier électronique qui réagit selon le sens actuel selon 3 axes. Le Sablier pourra émettre un son à la fin de temps pour indiquer que le délai est achevé.

2 PRÉ ÉTUDE

2.1 SCHÉMA BLOC

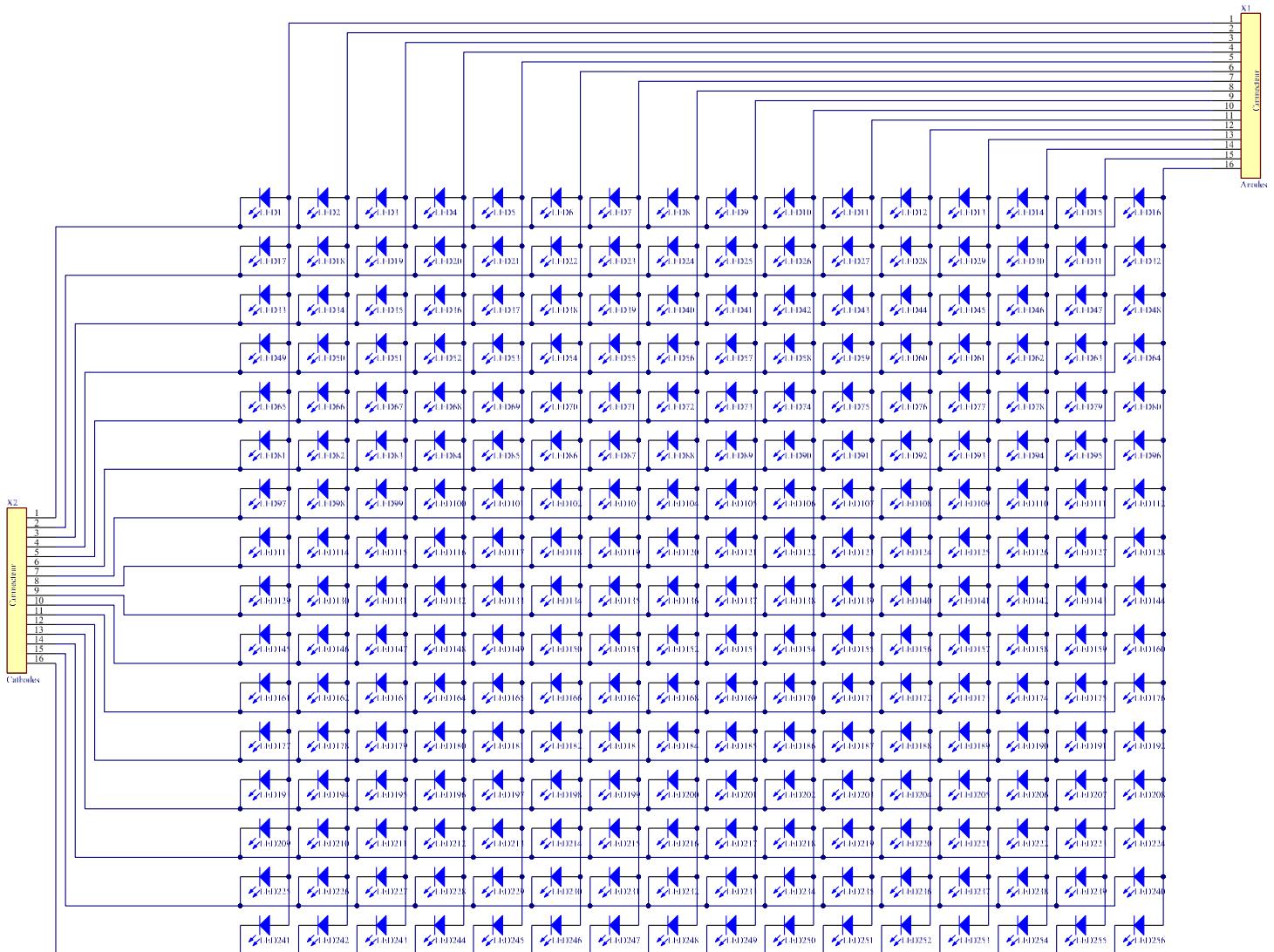


Nous pouvons voir tous les blocs dans ce schéma pour le bon fonctionnement du projet. Nous avons un PIC32 comme microcontrôleur. Nous avons un Gyroscope et/ou Accélémètre pour déterminer le sens de rotation du sablier. Nous avons une partie de commande/multiplexeurs pour les leds. Le buzzer sera commandé avec un signal logique "1 ou 0". Nous allons utiliser l'interface USB du PIC32. Le Module Bluetooth sera utilisé pour envoyer les données sur le temps sur une possible application Android. Les commandes serviront à sélectionner le temps désiré.

2.2 MATRICE À LEDS

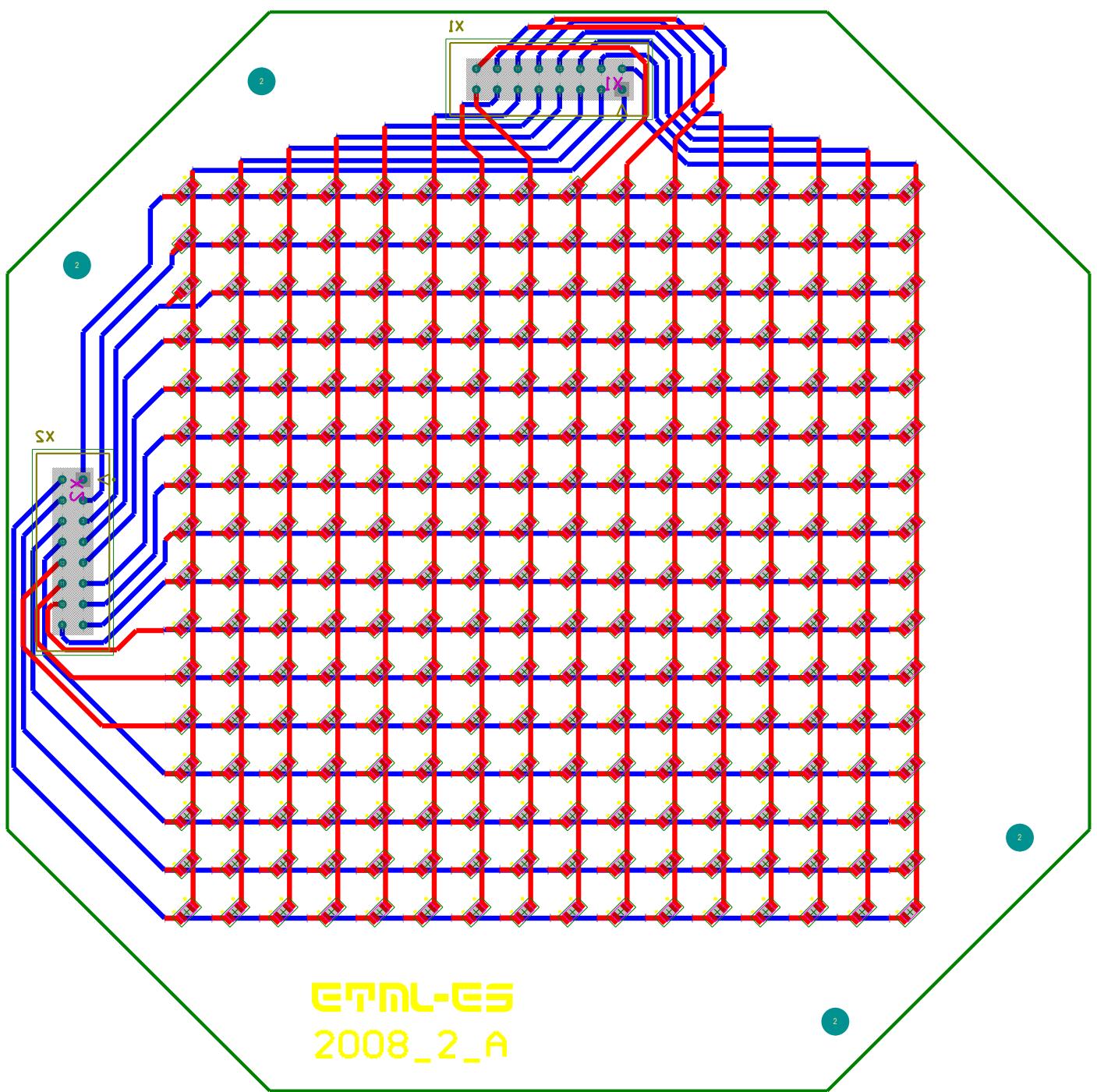
Nous n'avons pas trouvé une matrice à leds qui répond à nos besoins. Pour le correct déroulement du projet, nous avons décidé de créer notre propre matrice à leds.

2.2.1 Schéma



Nous avons le schéma traditionnel pour une matrice à leds. Il y a dans ce schéma, 256 leds. La commande de leds sera implémentée dans la carte principale.

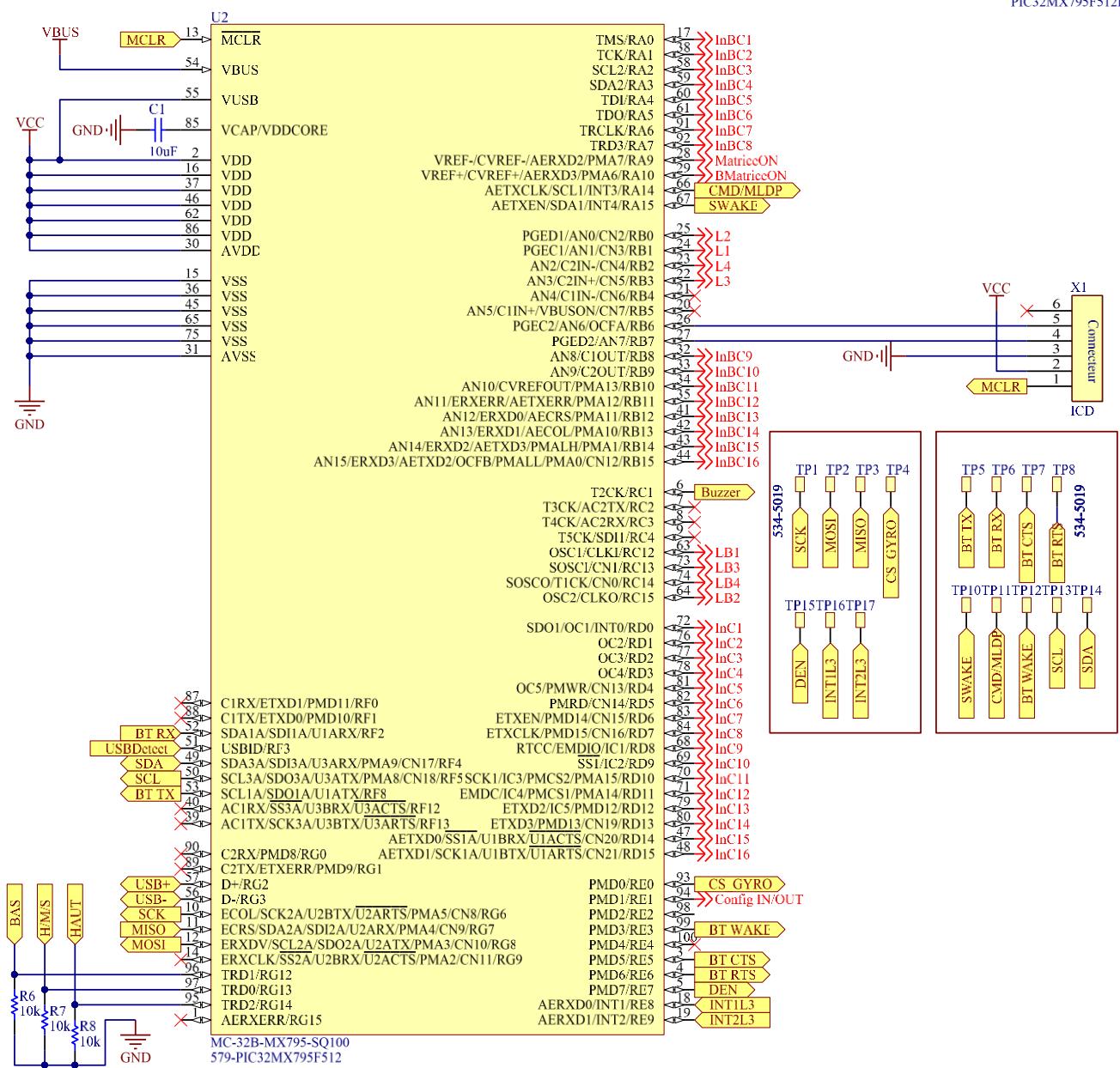
2.2.2 PBC Matrice



Notre PCB a une taille de 132mm*132mm. Il n'y a pas une longueur supérieure à 132mm dans ce PCB.
Nous avons choisi un diamètre de 132mm parce que les deux PCB mis à côté ne font pas plus de 300mm.

2.3 PIC32MX795F512L

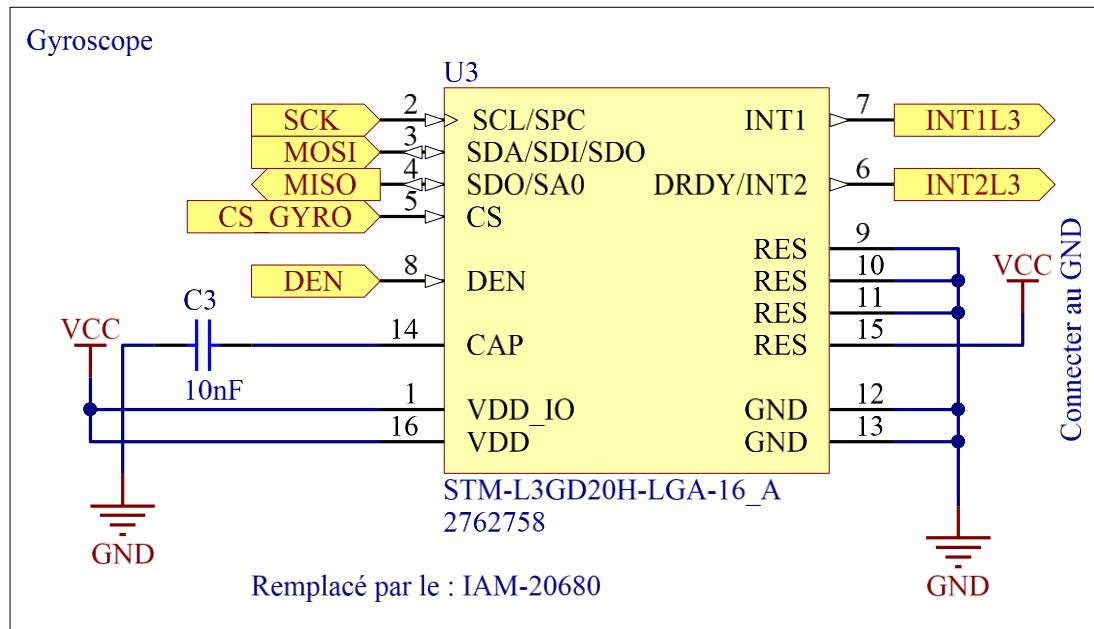
PIC32MX795F512L



J'ai choisi le PIC32MX795F512L. J'ai choisi ce pic parce que ce dernier est le même que celui utilisé dans le kit pic32 de l'ETML. Ce pic est un pic TQFP100, ces nombreuses entrées et sorties me seront utiles pour commander facilement les leds.

J'ai déjà implémenté un algorithme USB sur le kit pic32 de l'etml. Je pourrais normalement reprendre cet algorithme sans problèmes.

2.4 L3GD20H – GYROSCOPE – (EXCLU DU PROJET)



Le L3GD20H est un gyroscope proposé par ST. Ce gyroscope permet de mesurer le moment d'inertie sur notre circuit. Ceci me permettrait de connaître la rotation de mon produit.

2.4.1 Communication

Notre gyroscope nous permet d'utiliser, comme moyens de communication, le SPI et le I²C.

Pin name	Pin description
CS	SPI enable I ² C/SPI mode selection (1: I ² C mode; 0: SPI enabled)
SCL/SPC	I ² C Serial Clock (SCL) SPI Serial Port Clock (SPC)
SDA/SDI/SDO	I ² C Serial Data (SDA) SPI Serial Data Input (SDI) 3-wire Interface Serial Data Output (SDO)
SDO/SA0	SPI Serial Data Output (SDO) I ² C less significant bit of the device address

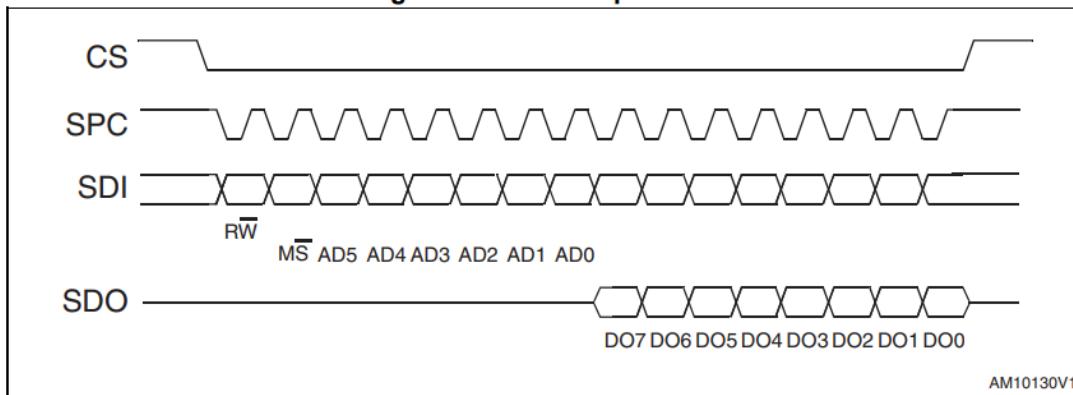
Nous pouvons voir que la pin CS de notre gyroscope contrôle le mode de communication de ce dernier. Les interfaces de communication possible partageant plusieurs points. Le clock, et le port des données. Dans le cas du SPI, nous avons une pin de donnée en plus (SDO). SA0 sert à changer un bit d'adresse dans le cas du I²C.

2.4.2 SPI - L3GD20H

Le SPI de notre gyroscope est toujours en slave. Nous pouvons accéder, au travers du SPI, aux valeurs des registres.

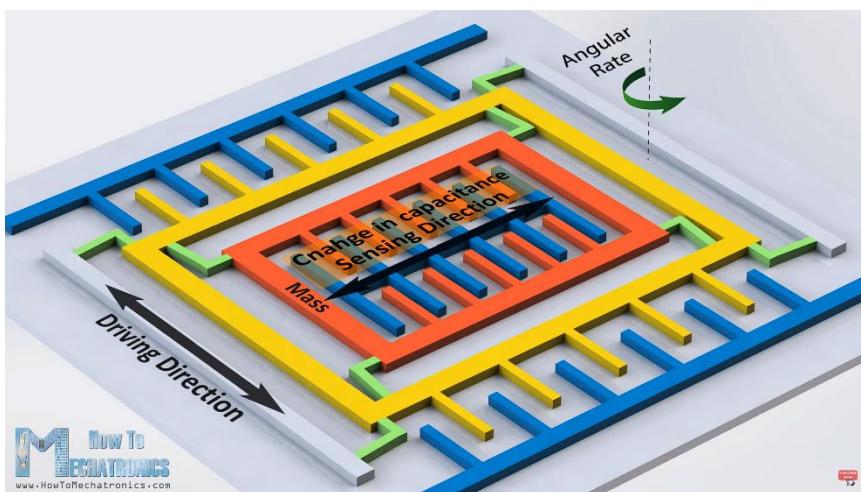
SPI read

Figure 18. SPI read protocol



Pour la lecture, le premier bit envoyé au slave doit être 1. Le dixième bit déterminera si l'on continuera à lire l'adresse suivante. Les 6 prochains bits détermineront l'adresse à lire. Par la suite, notre gyroscope répondra avec le registre choisi et les suivants dans le cas d'une lecture multiple.

2.4.3 Principe de fonctionnement d'un gyroscope



Pour le fonctionnement d'un gyroscope, nous avons une oscillation constante d'un circuit interne. Sous l'effet gyroscopique, lorsque nous appliquons une rotation angulaire, notre circuit vibrant veut continuer à vibrer dans la même position qu'auparavant.



Ce principe est le même que celui utilisé par les insectes pour rester stable lorsqu'ils volent. Lorsque ces derniers sont déstabilisés en plein vol, l'effet gyroscopique de leurs 4 ailes (lorsqu'une paire est en haut, l'autre paire est en bas) leur permet d'exercer une force dans le sens contraire du déplacement. Les gyroscopes MEMS utilisent le même principe.

2.4.4 Registres du L3GD20H

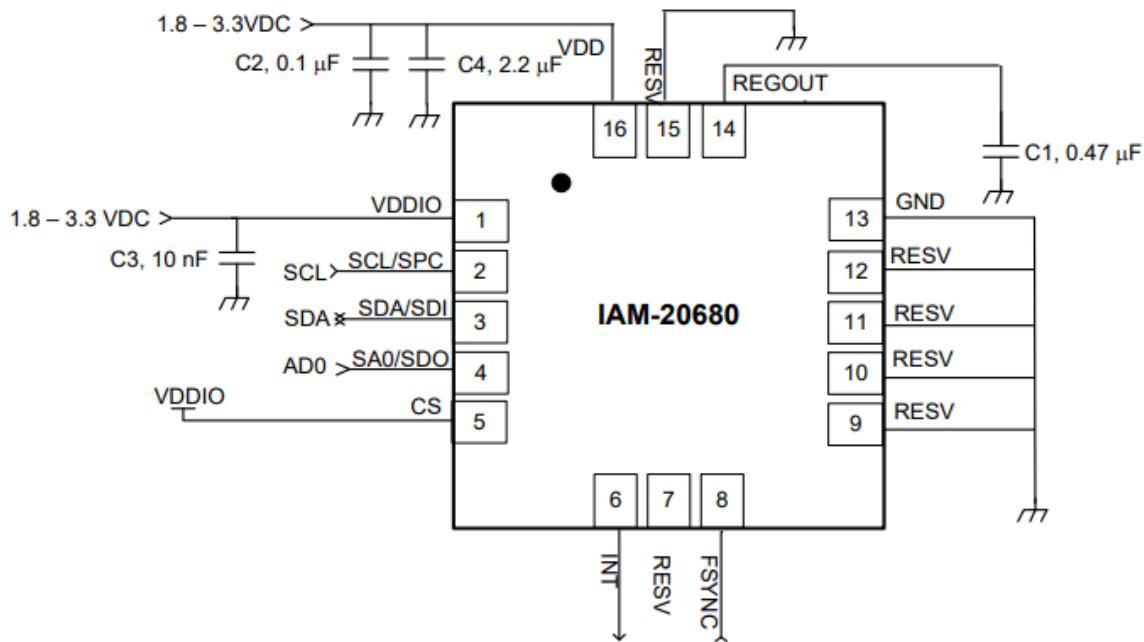
Table 17. Register address map

Name	Type	Register address		Default
		Hex	Binary	
Reserved	-	00-0E	-	-
WHO_AM_I	r	0F	000 1111	11010111
Reserved	-	10-1F	-	-
CTRL1	rw	20	010 0000	00000111
CTRL2	rw	21	010 0001	00000000
CTRL3	rw	22	010 0010	00000000
CTRL4	rw	23	010 0011	00000000
CTRL5	rw	24	010 0100	00000000
REFERENCE	rw	25	010 0101	00000000
OUT_TEMP	r	26	010 0110	Output
STATUS	r	27	010 0111	Output
OUT_X_L	r	28	010 1000	Output
OUT_X_H	r	29	010 1001	Output
OUT_Y_L	r	2A	010 1010	Output
OUT_Y_H	r	2B	010 1011	Output
OUT_Z_L	r	2C	010 1100	Output
OUT_Z_H	r	2D	010 1101	Output
FIFO_CTRL	rw	2E	010 1110	00000000
FIFO_SRC	r	2F	010 1111	Output
IG_CFG	rw	30	011 0000	00000000
IG_SRC	r	31	011 0001	Output
IG_THS_XH	rw	32	011 0010	00000000
IG_THS_XL	rw	33	011 0011	00000000
IG_THS_YH	rw	34	011 0100	00000000
IG_THS_YL	rw	35	011 0101	00000000
IG_THS_ZH	rw	36	011 0110	00000000
IG_THS_ZL	rw	37	011 0111	00000000
IG_DURATION	rw	38	011 1000	00000000
LOW_ODR	rw	39	011 1001	00000000

Nous pouvons voir que nous avons uniquement 57 registres sur notre gyroscope. Les registres qui nous importent le plus sont les registres 40 au registre 46.

2.5 IAM-20680 – CENTRALE INERTIELLE. (CHOIX FINAL, 6 AXES DÉTECTION)

Le IAM-20680 est-ce que l'ont appelle, une centrale inertielle. Ce dernier nous permet de détecter la rotation angulaire sur 3 axes, comme notre L3GD20H, mais en plus, nous pouvons détecter l'accélération sur 3 axes.



J'ai choisi ce composant après avoir commandé le PCB, en effet, j'ai commis une erreur dans l'estimation sur la difficulté d'utiliser uniquement un gyroscope pour ce projet. Ce composant a un pinout plus au moins compatible avec celui choisi dans un premier lieu. Dans le premier schéma, nous avions la pin 15 au VCC. Dans ce cas, ils nous indiquent que cette dernière doit être mise au GND. En réalité, les pins RESV, sont des pins NC. Ces pins ne sont pas utilisés, mais il n'est pas correct de les laisser flottantes. Il ne devrait pas avoir un problème quelconque à laisser la pin 15 au VCC, mais nous allons tout de même effectuer une modification afin de limiter les risques.

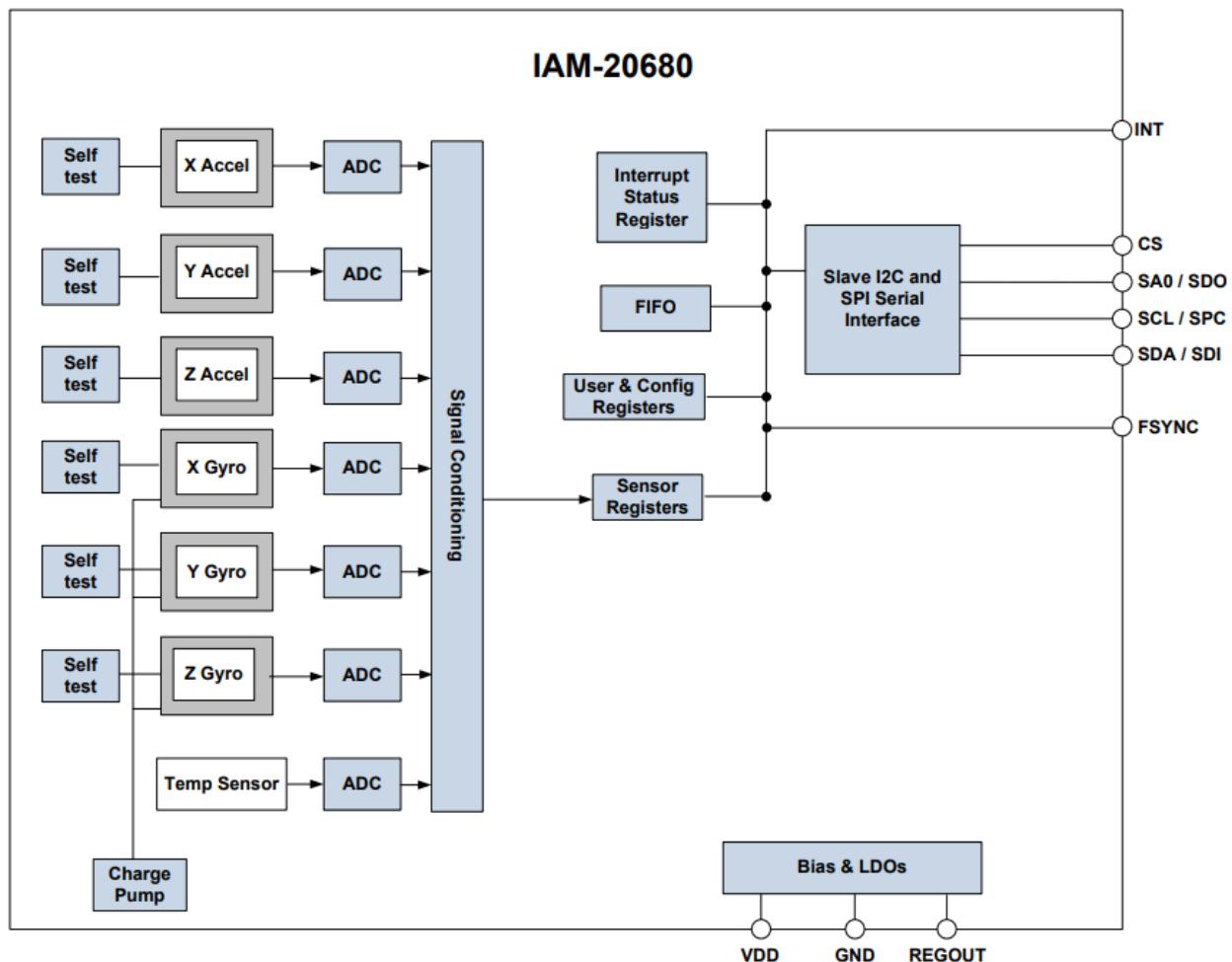
2.5.1 Communication

Ce composant utilise le même principe de communication que le L3GD20H à quelques exceptions près. Dans ce cas, nous allons utiliser le SPI afin d'avoir la plus grande bande passante.

2	SCL/SPC	I ² C serial clock (SCL); SPI serial clock (SPC).
3	SDA/SDI	I ² C serial data (SDA); SPI serial data input (SDI).
4	SA0/SDO	I ² C slave address LSB (SA0); SPI serial data output (SDO).
5	CS	Chip select (0 = SPI mode; 1 = I ² C mode).

Pour ceci, il suffit de mettre notre cs à 0.

2.5.2 Schéma blocs



Le principe de ce composant est qu'il a 6 capteurs intégrés qui lui renvoient une certaine tension selon la mesure qu'ils effectuent. Par la suite, il y a une convention analogique digitale. Les données sont traitées et envoyées aux registres correspondants.

2.5.3 Interface SPI – IAM 20680

Dans ce cas, la communication est plus simple qu'avec le L3GD20H. Les données envoyées par et au slave sont en MSB first. Pour indiquer au composant que l'opération est une lecture, nous devons envoyer 0 comme premier bit, suivi de l'adresse à lire. Pour configurer les registres, nous devons donc envoyer 1 comme premier bit.

SPI Address format

MSB							LSB
R/W	A6	A5	A4	A3	A2	A1	A0

SPI Data format

MSB							LSB
D7	D6	D5	D4	D3	D2	D1	D0

Envoyer plusieurs adresses sur une seule communication résultera sur une réponse multiple (toutes les adresses demandées). La communication commence avec la mise à 0 de la pin CS et se termine avec la mise à 1 de cette dernière.

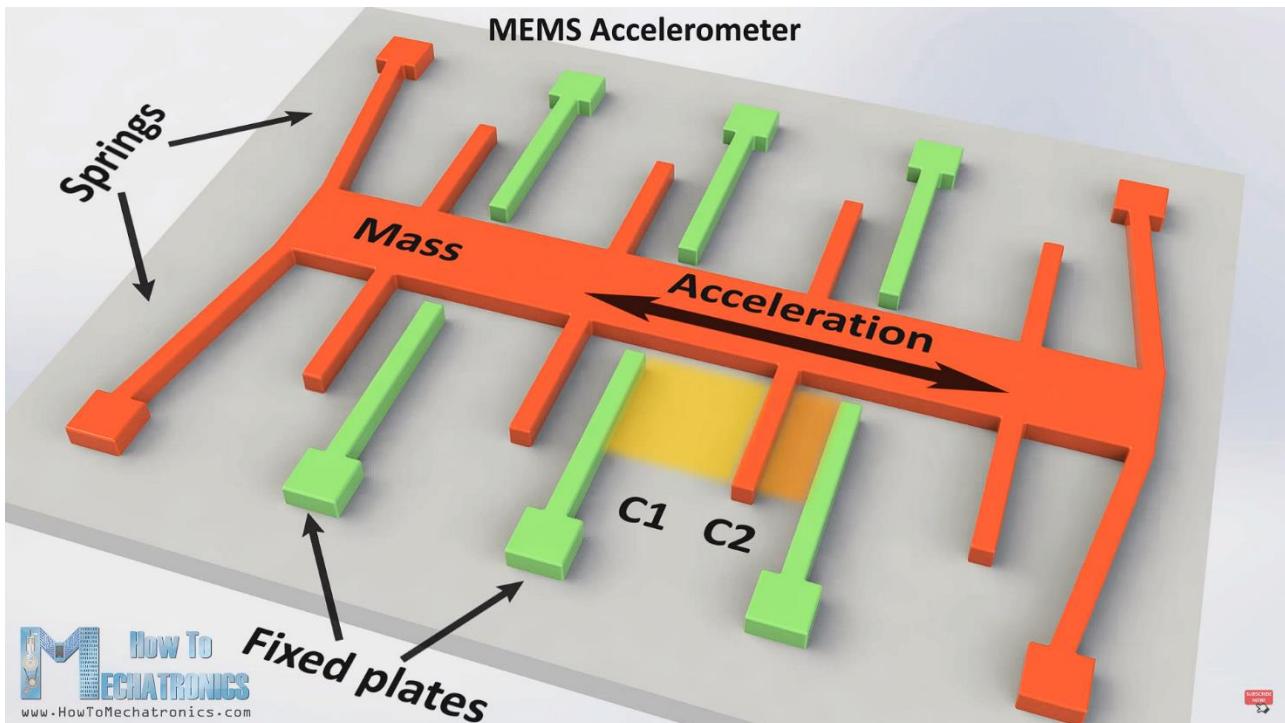
2.5.4 Registres du IAM-20680

Addr (Hex)	Addr (Dec.)	Register Name	Serial I/F	Accessible (writable) in Sleep Mode	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
00	00	SELF_TEST_X_GYRO	R/W	N				XG_ST_DATA[7:0]				
01	01	SELF_TEST_Y_GYRO	R/W	N				YG_ST_DATA[7:0]				
02	02	SELF_TEST_Z_GYRO	R/W	N				ZG_ST_DATA[7:0]				
0D	13	SELF_TEST_X_ACCEL	R/W	N				XA_ST_DATA[7:0]				
0E	14	SELF_TEST_Y_ACCEL	R/W	N				YA_ST_DATA[7:0]				
0F	15	SELF_TEST_Z_ACCEL	R/W	N				ZA_ST_DATA[7:0]				
13	19	XG_OFFSET_USRH	R/W	N				X_OFFSET_USR [15:8]				
14	20	XG_OFFSET_USRL	R/W	N				X_OFFSET_USR [7:0]				
15	21	YG_OFFSET_USRH	R/W	N				Y_OFFSET_USR [15:8]				
16	22	YG_OFFSET_USRL	R/W	N				Y_OFFSET_USR [7:0]				
17	23	ZG_OFFSET_USRH	R/W	N				Z_OFFSET_USR [15:8]				
18	24	ZG_OFFSET_USRL	R/W	N				Z_OFFSET_USR [7:0]				
19	25	SMPLRIT_DIV	R/W	N				SMPLRIT_DIV[7:0]				
1A	26	CONFIG	R/W	N	-	FIFO_MODE		EXT_SYNC_SET[2:0]			DLPF_CFG[2:0]	
1B	27	GYRO_CONFIG	R/W	N	XG_ST	YG_ST	ZG_ST	FS_SEL[1:0]	-		FCHOICE_B[1:0]	
1C	28	ACCEL_CONFIG	R/W	N	XA_ST	YA_ST	ZA_ST	ACCEL_FS_SEL[1:0]			-	
1D	29	ACCEL_CONFIG 2	R/W	N	-			DEC2_CFG	ACCEL_FCHOICE_B		A_DLPF_CFG	
1E	30	LP_MODE_CFG	R/W	N	GYRO_CYCLE			G_AVGCFG[2:0]			-	
1F	31	ACCEL_WOM_THR	R/W	N					WOM_THR[7:0]			
23	35	FIFO_EN	R/W	N	TEMP_FIFO_EN	XG_FIFO_EN	YG_FIFO_EN	ZG_FIFO_EN	ACCEL_FIFO_EN	-	-	-
36	54	FSYNC_INT	R/C	N	FSYNC_INT	-	-	-	-	-	-	-
37	55	INT_PIN_CFG	R/W	Y	INT_LEVEL	INT_OPEN	LATCH_INT_EN	INT_RD_CLEAR	FSYNC_INT_LEVEL	FSYNC_INT_MODE_EN	-	-
38	56	INT_ENABLE	R/W	Y			WOM_INT_EN[7:5]		IFO_OFLOW_EN	GDRIVE_INT_EN	-	DATA_RDY_IN
3A	58	INT_STATUS	R/C	N			WOM_INT[7:5]		IFO_OFLOW_INT	GDRIVE_INT	-	DATA_RDY_IN
3B	59	ACCEL_XOUT_H	R	N					ACCEL_XOUT_H[15:8]			
3C	60	ACCEL_XOUT_L	R	N					ACCEL_XOUT_L[7:0]			
3D	61	ACCEL_YOUT_H	R	N					ACCEL_YOUT_H[15:8]			
3E	62	ACCEL_YOUT_L	R	N					ACCEL_YOUT_L[7:0]			
3F	63	ACCEL_ZOUT_H	R	N					ACCEL_ZOUT_H[15:8]			
40	64	ACCEL_ZOUT_L	R	N					ACCEL_ZOUT_L[7:0]			
41	65	TEMP_OUT_H	R	N					TEMP_OUT[15:8]			
42	66	TEMP_OUT_L	R	N					TEMP_OUT[7:0]			
43	67	GYRO_XOUT_H	R	N					GYRO_XOUT[15:8]			
44	68	GYRO_XOUT_L	R	N					GYRO_XOUT[7:0]			
45	69	GYRO_YOUT_H	R	N					GYRO_YOUT[15:8]			
46	70	GYRO_YOUT_L	R	N					GYRO_YOUT[7:0]			
47	71	GYRO_ZOUT_H	R	N					GYRO_ZOUT[15:8]			
48	72	GYRO_ZOUT_L	R	N					GYRO_ZOUT[7:0]			
68	104	SIGNAL_PATH_RESET	R/W	N	-	-	-	-	-	-	ACCEL_RST	TEMP_RST
69	105	ACCEL_INTEL_CTRL	R/W	N	ACCEL_INTEL_MODE							
6A	106	USER_CTRL	R/W	N	-	FIFO_EN	-	I2C_IF_DIS	-	FIFO_RST	-	SIG_COND_RST
6B	107	PWR_MGMT_1	R/W	Y	DEVICE_RESET	SLEEP	ACCEL_CYCLE	GYRO_STANDBY	TEMP_DIS			CLKSEL[2:0]

Addr (Hex)	Addr (Dec.)	Register Name	Serial I/F	Accessible (writable) in Sleep Mode	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
6C	108	PWR_MGMT_2	R/W	Y	FIFO_LP_EN	-	STBY_XA	STBY_YA	STBY_ZA	STBY_XG	STBY_YG	STBY_ZG
72	114	FIFO_COUNTH	R	N		-						FIFO_COUNT[12:8]
73	115	FIFO_COUNTL	R	N								FIFO_COUNT[7:0]
74	116	FIFO_R_W	R/W	N								FIFO_DATA[7:0]
75	117	WHO_AM_I	R	N								WHOAMI[7:0]
77	119	XA_OFFSET_H	R/W	N								XA_OFFSET[14:0]
78	120	XA_OFFSET_L	R/W	N								XA_OFFSET[6:0]
7A	122	YA_OFFSET_H	R/W	N								YA_OFFSET[14:0]
7B	123	YA_OFFSET_L	R/W	N								YA_OFFSET[6:0]
7D	125	ZA_OFFSET_H	R/W	N								ZA_OFFSET[14:0]
7E	126	ZA_OFFSET_L	R/W	N								ZA_OFFSET[6:0]

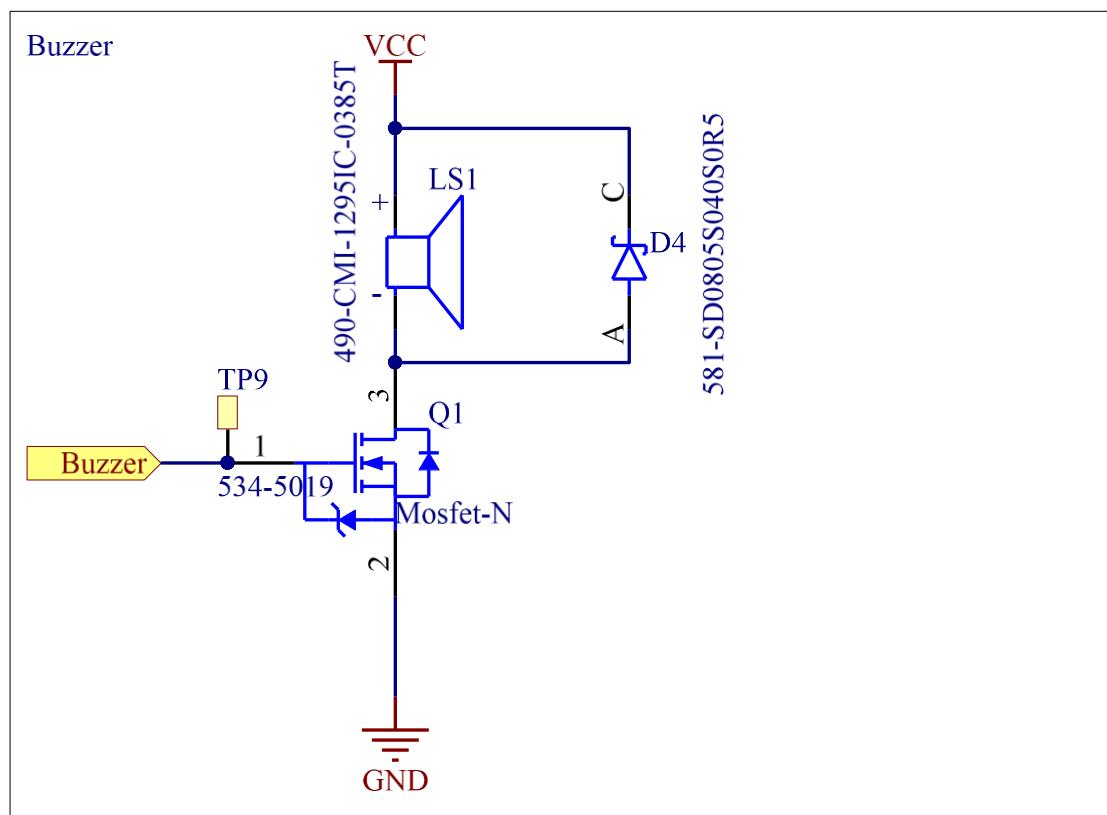
Nous pouvons voir que nous avons deux fois plus de registres qu'auparavant, d'où la perte du bit MS. Avec 7 bits d'adresse, nous pouvons accéder jusqu'à 128 adresses. Dans ce cas, nous avons 126. Les registres que l'on va utiliser sont les registres 59 au registre 64.

2.5.5 Fonctionnement d'un accéléromètre.



Le principe de fonctionnement d'un accéléromètre est plus simple que celui d'un gyroscope. En effet, nous avons sur 3 axes, une masse avec des filaments. Lorsqu'il y a une force appliquée sur la masse en question, les filaments de cette dernière se rapprochent ou s'éloignent des plaques fixes. Ceci a comme résultat, une modification de la capacité mesurée aux bornes des plaques fixes. Dans ce cas nous allons apercevoir l'accélération constante de la terre. Ceci rend notre projet beaucoup plus simple dans un premier temps. En effet, nous pourrons savoir la position de notre appareil sans avoir à le calibrer auparavant.

2.6 BUZZER



Pour le buzzer, j'ai trouvé un buzzer avec un prix très réduit et qui intègre directement un driver.

MODEL: CMI-1295IC-0385T | **DESCRIPTION:** MAGNETIC BUZZER INDICATOR

FEATURES

- 3 Vdc rated
- 85 dB
- through hole
- magnetic
- internally driven
- narrow frequency range
- used in medical & security applications



Nous pouvons voir que dans les spécificités, ils indiquent qu'il est contrôlé par un circuit interne.

SPECIFICATIONS

parameter	conditions/description	min	typ	max	units
rated voltage			3		Vdc
operating voltage		2		5	Vdc
current consumption	at rated voltage			30	mA
rated frequency	at rated voltage	2,300	2,400	2,500	Hz

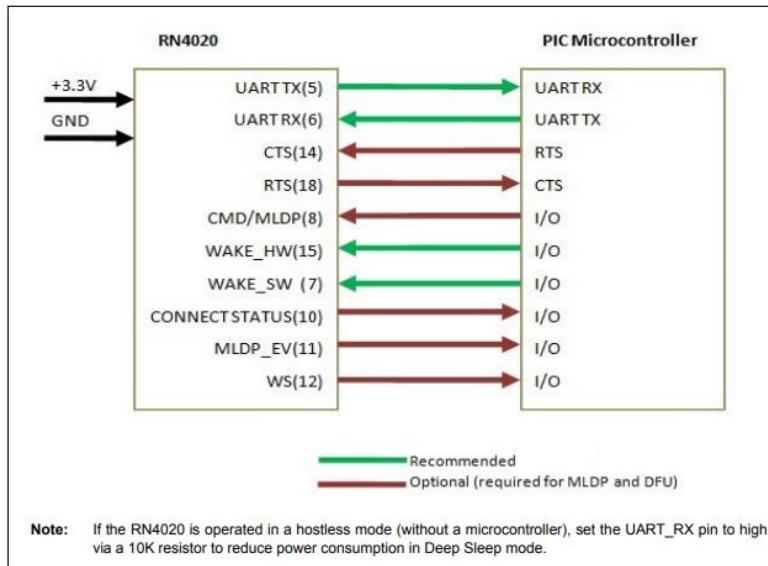
La plage de tension est entre 5V et 2V. Nous ne dépassons donc pas la tension maximale du buzzer.

La fréquence de 2400Hz me paraît correcte pour avertir les élèves sur la fin de l'examen.

Ne sachant pas comment est le circuit interne du buzzer, j'ai placé une diode de protection afin de protéger le mosfet. Ce dernier devra être mis à 0V pour l'activer. Il faut donc initialiser la pin qui contrôle le mosfet à 1 afin d'éviter un bruit à l'allumage. La consommation maximale est de 30mA.

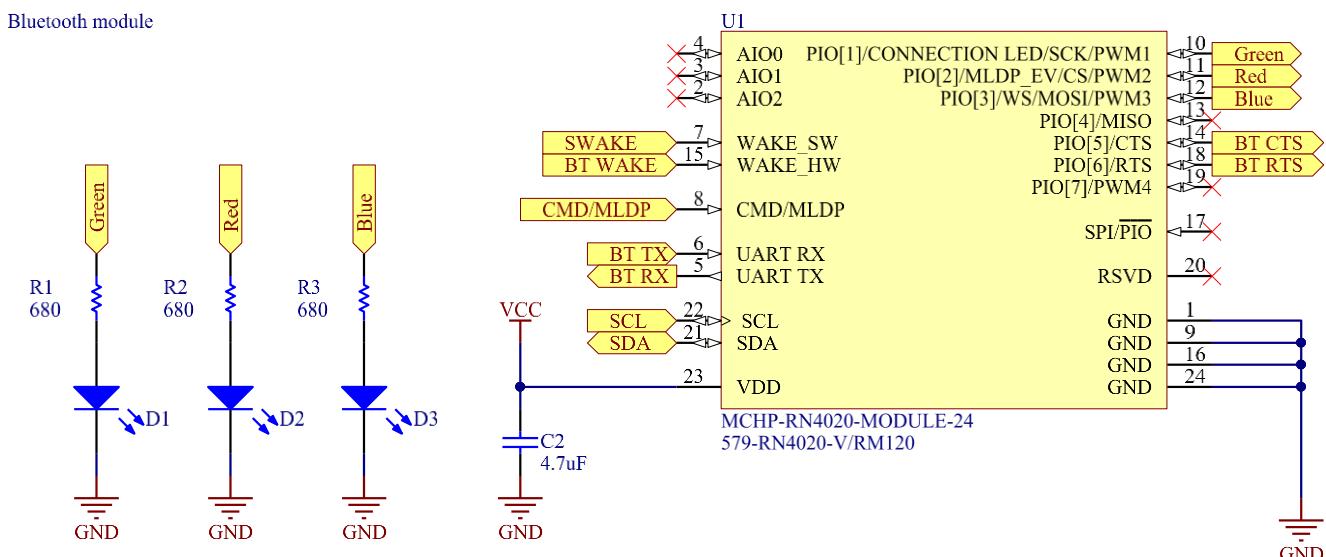
2.7 RN4020 - MODULE BLUETOOTH

2.7.1 Interface recommandée



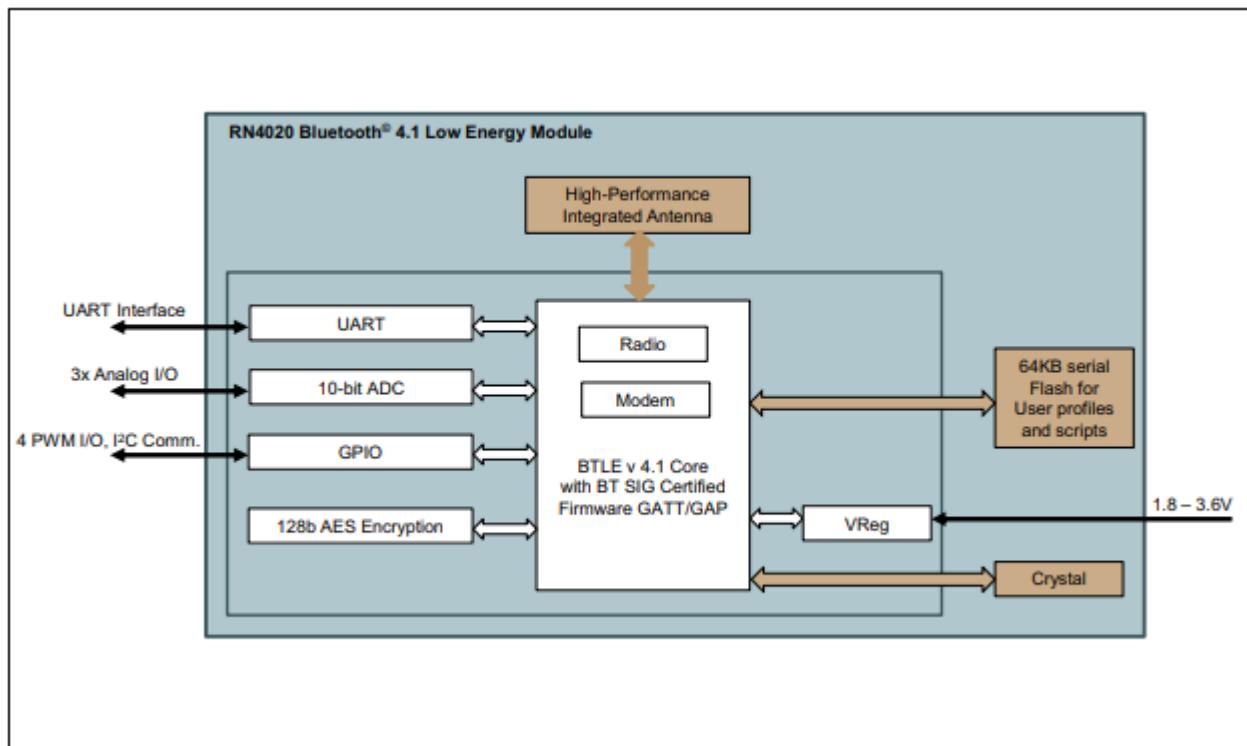
Nous pouvons voir que le fabricant propose une interface uniquement avec le TX,RX et les pins de contrôle WAKE_HW, WAKE_SW.

2.7.2 Schéma RN4020



Dans notre cas, nous avons choisi d'utiliser CTS et RTS pour un contrôle du flux sur l'UART. Nous avons choisi de connecter l'interface I2C pour prévenir des problèmes. Il y a des leds de contrôle qui ont été implémentées. Elles nous permettront de savoir l'état de notre module.

2.7.3 Schéma blocs



Notre module Bluetooth RN4020 contient plusieurs modules de communication. Nous pouvons utiliser le module en lui indiquant les ordres grâce à l'UART. Il est possible d'utiliser l'I2C également. Nous avons 3 ADC sur le RN4020 et des sorties PWM utilisables. Il y a une mémoire de 64KB.

2.7.4 Consommation du module Bluetooth

TABLE 2-4: CURRENT CONSUMPTION

Mode	Typical Current at 3V
Dormant	<900 nA
Deep Sleep	<5.0 µA
Idle	<1.5 mA
TX/RX active	16 mA

TABLE 2-5: CURRENT CONSUMPTION VS RF TX POWER

RN4020, VDD = 3.3V, 25°C	
TX Power (dBm)	Id (mA)
-19.1	14.0
-15.1	14.4
-10.9	15.0
-6.9	15.9
-2.5	17.6
1.6	20.7
5.8	26.9
7.5	33.6

Nous pouvons voir, selon la table du datasheet 2-5, que la consommation maximale de notre module est de 33.6mA.

La consommation habituelle est néanmoins de 16mA.

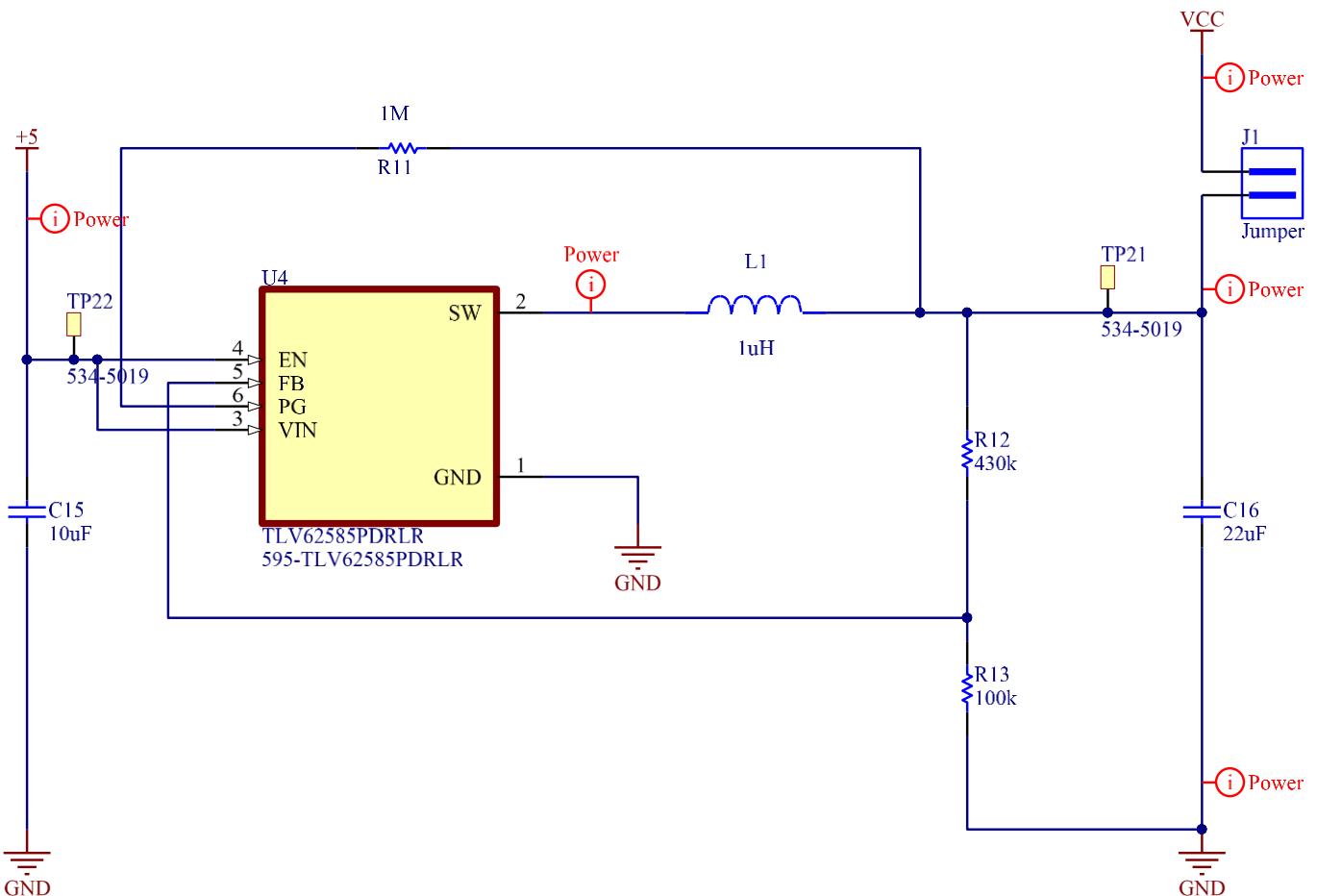
2.8 ALIMENTATION STEP-DOWN

2.8.1 Estimation de la consommation maximale

DC CHARACTERISTICS			Standard Operating Conditions: 2.3V to 3.6V (unless otherwise stated)			
Param. No.	Typical ⁽³⁾	Max.	Units	Conditions		
Operating Current (IDD)^(1,2,4) for PIC32MX575/675/695/775/795 Family Devices						
DC20	6	9	mA	Code executing from Flash	-40°C, +25°C, +85°C	—
DC20b	7	10			+105°C	
DC20a	4	—		Code executing from SRAM	—	4 MHz
DC21	37	40	mA	Code executing from Flash	—	—
DC21a	25	—		Code executing from SRAM	—	
DC22	64	70	mA	Code executing from Flash	—	—
DC22a	61	—		Code executing from SRAM	—	
DC23	85	98	mA	Code executing from Flash	-40°C, +25°C, +85°C	—
DC23b	90	120			+105°C	
DC23a	85	—		Code executing from SRAM	—	80 MHz
DC25a	125	150	μA	—	+25°C	3.3V
					LPRC (31 kHz)	

Le microcontrôleur a une consommation maximale de 120mA à une fréquence de 80MHz. Nous savons que notre microcontrôleur commande uniquement des entrées en haute impédance. Pour prendre une marge, nous allons compter 200mA. Nous avons une consommation de 2.25mA par led. En sachant que nous pourrons avoir uniquement 16 leds allumées en même temps. Nous avons un courant de 36mA pour les 16 leds. Pour le module RN4020, nous pouvons prendre le courant maximal de 33.6mA. Le module Bluetooth consomme à peine 3mA. Le multiplexeur est un circuit analogique, de se fait, tant que les sorties ne consomment pas du courant, la consommation de ce dernier est minime. Le buzzer a une consommation maximale de 30mA. Nous avons donc une consommation maximale d'environ 300mA. Pour le dimensionnement de l'alimentation, nous allons prendre 1A.

2.8.2 Choix de l'alimentation step down



Cette alimentation Step-Down a été choisie grâce au site WEBENCH. Dans ce site, nous pouvons choisir les paramètres d'entrée pour notre alimentation à découpage et nos paramètres de sortie.

Input

Supply type is DC AC

Vin Min *	4.5	V	Vin Max *	5.5	V
(-50 - 500)			(-50 - 500)		

Advanced

Output

Vout * 3.3 V

Iout Max * 1 A

Isolated Output

Advanced

Nous devons mettre comme paramètres d'entrée, la tension d'entrée minimale et maximale. Il nous permet aussi de choisir entre une tension DC et AC.

En sortie, nous devons lui indiquer la tension voulue et le courant maximal en sortie.

Nous pouvons aussi lui indiquer

Design Consideration

I want my design to be Balanced Low Cost High Efficiency Small Footprint

Design Parameters

I agree that the use of TI's WEBENCH tools is subject to the [Webench Notice](#), [TI's Site Terms and Conditions of Use](#), and [TI's Privacy Policy](#).

VIEW DESIGNS

l'oscillation maximale que l'on veut avoir en sortie.

TLV62585PDRL

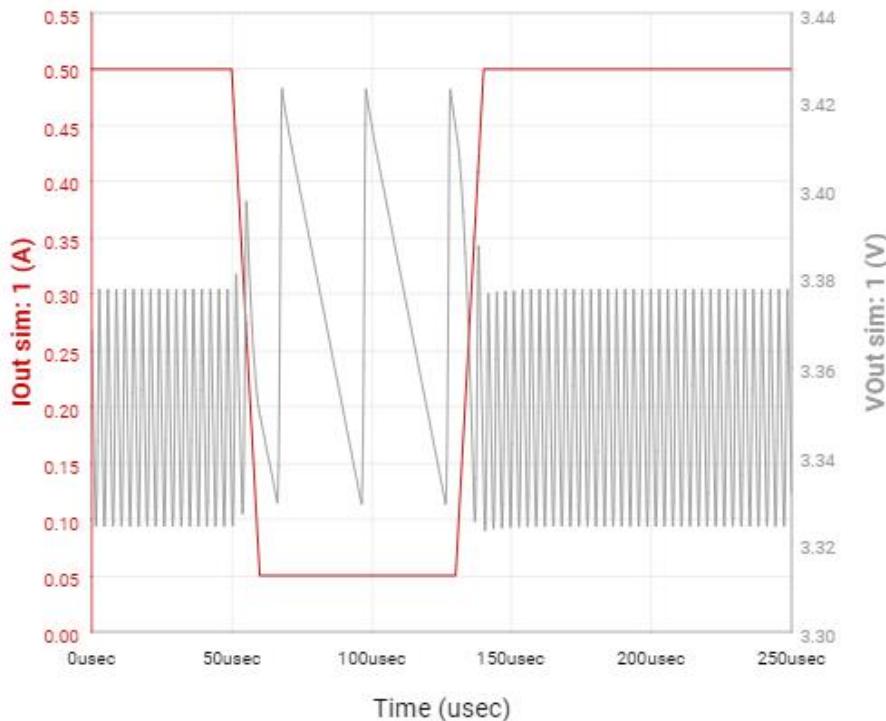
2.5V-5.5V input, 3-A high efficiency step-down converter in 2x2 QFN or SOT563 package

Efficiency: 95.2% BOM Cost: \$2.87 Footprint: 113 mm² BOM Count: 7 Topology: Buck

Frequency: 1.53 MHz IC Cost: \$0.16 | 1ku

[CUSTOMIZE](#) [SIMULATE](#) [EXPORT](#)

Personnellement, j'ai choisi ce parce que la taille du footprint (tous les composants sur la carte) pour cette alimentation est acceptable (pas trop petit ni trop grand). Dans notre cas, nous avons choisi une tension d'entrée à 5V pour pouvoir l'alimenter en USB.



Avec une tension de sortie à 3.3V, nous pouvons voir qu'il y a un pic de tension lorsque nous diminuons la charge. Ceci est normal puisque nous avons un step down. Dans une alimentation step down, elle aura plus de peine à abaisser la tension lorsque la consommation est minime. L'oscillation dans ce cas, est d'environ 0.1V maximum. Ceci

correspond à une oscillation de 3%. C'est une oscillation qui est considérée acceptable.

2.8.3 Calcul de la résistance R12

La résistance de 453kOhms proposée par Webench pour atteindre une tension cible de 3.3V n'est pas une résistance dans la série E24. J'ai choisi de changer de résistance pour qu'elle soit dans la série E24.

Typical Application

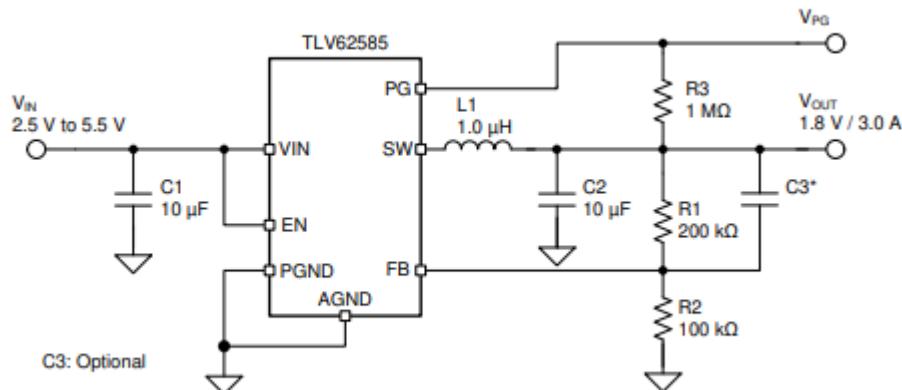


Figure 4. 1.8-V Output Voltage Application

8.2.2.2 Setting The Output Voltage

The output voltage is set by an external resistor divider according to [Equation 2](#):

$$V_{OUT} = V_{FB} \times \left(1 + \frac{R1}{R2}\right) = 0.6V \times \left(1 + \frac{R1}{R2}\right) \quad (2)$$

R2 must not be higher than 100 kΩ to achieve high efficiency at light load while providing acceptable noise sensitivity.

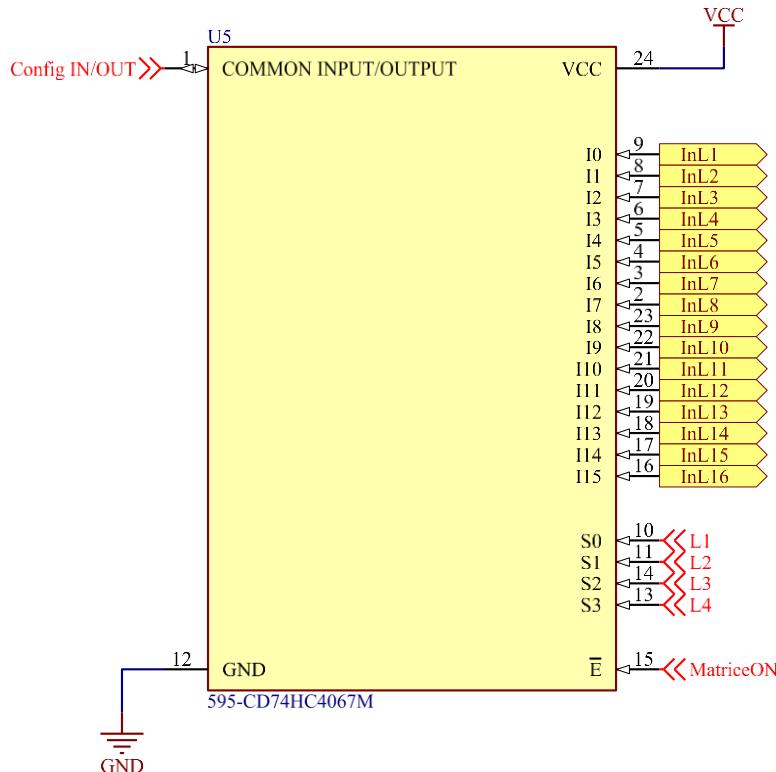
Dans le datasheet, ils nous indiquent que la résistance, dans notre cas R13, ne doit pas être plus grande que 100kohms pour avoir une bonne efficience.

Nous avons vu que la tension de sortie pouvait dépasser 3.3V dans cette configuration lorsque le courant de sortie est faible. La résistance utilisée dans la simulation étant 453kOhms, nous allons choisir 430kOhms pour réduire la tension de sortie.

Nous aurions donc : $V_{out} = 0.6V * \left(1 + \frac{430k}{100k}\right) = 3.18V$. En sachant que dans le cas précédent. Nous avions une tension maximale de 3.42V, nous pouvons espérer à un pic jusqu'à 3.3V dans ce cas. Il faudra reconfirmer ceci grâce à une mesure finale.

2.9 SCHÉMA DE COMMANDE DE LEDS

2.9.1 Multiplexeur - Fonctionnement



Le 74HC4067 est un multiplexeur/démultiplexeur analogue.

TRUTH TABLE

S0	S1	S2	S3	E	SELECTED CHANNEL
X	X	X	X	1	None
0	0	0	0	0	0
1	0	0	0	0	1
0	1	0	0	0	2
1	1	0	0	0	3
0	0	1	0	0	4
1	0	1	0	0	5
0	1	1	0	0	6
1	1	1	0	0	7
0	0	0	1	0	8
1	0	0	1	0	9
0	1	0	1	0	10
1	1	0	1	0	11
0	0	1	1	0	12
1	0	1	1	0	13
0	1	1	1	0	14
1	1	1	1	0	15

Nous pouvons voir, selon la table de vérité, comment sélectionner les entrées/sorties de notre multiplexeur.

Les pins S0 à S3 nous permettent de choisir la pin que l'on veut utiliser sur notre multiplexeur.

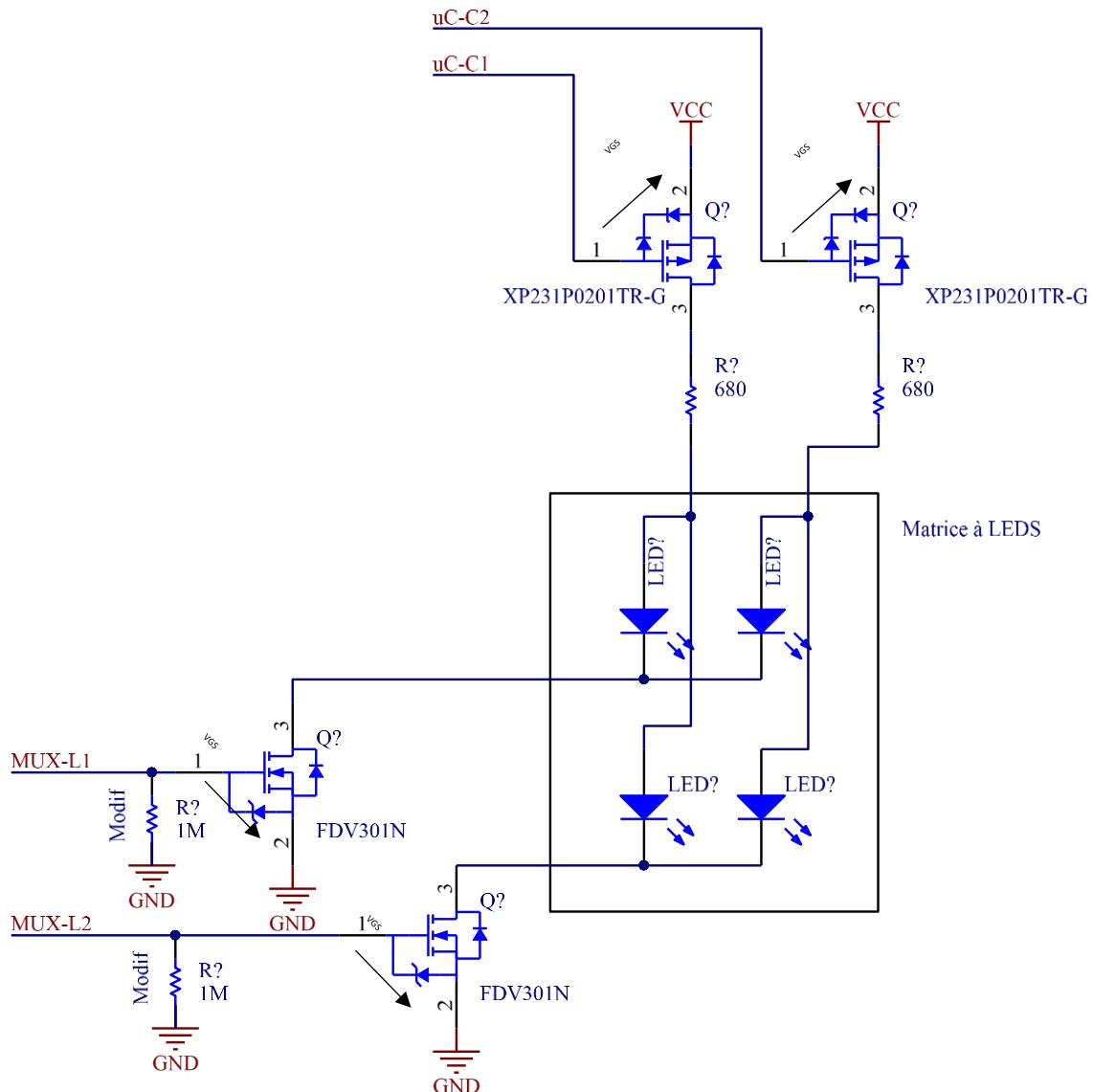
Le niveau logique présent sur la pin choisie sera déterminé par la pin COMMON INPUT/OUTPUT (pin 1). En effet, les sorties ou entrées de ce multiplexeur ont toutes en commun la pin 1. Cette dernière déterminera l'état de la pin choisie.

Lorsqu'une pin n'est pas choisie, l'état de cette dernière est « high impédance »

- HC Types
 - 2V to 6V Operation
 - High Noise Immunity: $N_{IL} = 30\%$, $N_{IH} = 30\%$ of V_{CC} at $V_{CC} = 5V$
- HCT Types
 - 4.5V to 5.5V Operation
 - Direct LSTTL Input Logic Compatibility, $V_{IL} = 0.8V$ (Max), $V_{IH} = 2V$ (Min)
 - CMOS Input Compatibility, $I_I \leq 1\mu A$ at V_{OL}, V_{OH}

Lorsque nous choisissons le type à utiliser, il faut choisir uniquement les types HC et non HCT. En effet, les tensions d'alimentation maximales supportées sont différentes.

2.9.2 Mosfets principe - commande



Pour contrôler nos matrices à leds, nous avons un mosfet N par ligne et un mosfet P par colonne. Dans ce schéma de principe, nous pouvons retrouver la résistance de 680 ohms pour limiter le courant de nos leds. Les résistances de 1M ohms ont été rajoutées par après. En effet, je ne les ai pas incluses dans le schéma principal. Ceci créait des problèmes lorsque nous voulions éteindre les leds.

Nous avons des mosfets N pour les lignes et des mosfets P pour les colonnes. Pour le choix de ces derniers, il faut choisir des mosfet avec un VGS inférieur à 3V. Dans notre cas, le mosfet P a un vgs de -1.25V max et le mosfet N, un vgs de 1.06V max. Nous pouvons voir, dans le cas du mosfet P, qu'il n'y a pas de résistance pour limiter le courant de charge de la grille. Il faut vérifier que le courant de charge du mosfet P n'est pas excessif pour éviter de l'endommager.

Gate-Source Leakage Current	I_{GSS}	$V_{GS} = \pm 8V, V_{DS} = 0V$	-	-	± 10	μA
-----------------------------	-----------	--------------------------------	---	---	----------	---------

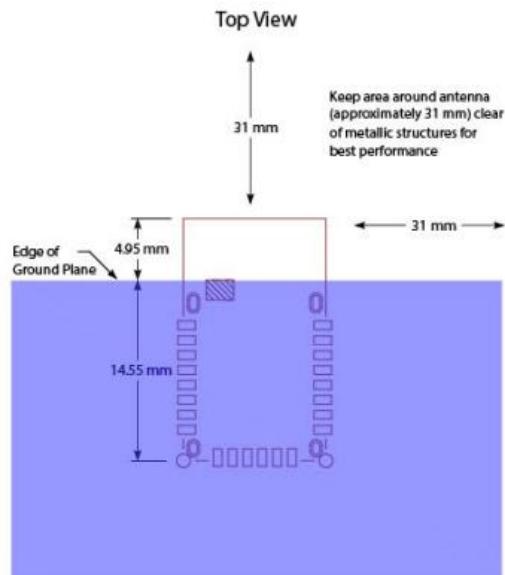
Nous pouvons voir que le courant de leakage est de 10uA lorsque l'on mesure le vgs maximal de 8V. Le courant sera plus petit avec un vgs d'environ 3V. Il faut savoir que la fréquence de switch ne sera pas très élevée. Une fréquence de 1kHz sera plus que suffisante.

2.9.2.1 Calculs leds

Calculs des leds : $R_{leds} = ((3.3V - 1.75V) / 2mA) - 10\text{ohms(mosfets)} = 765 \text{ ohms.} \Rightarrow 680(\text{E12})$
 Vrai courant = $(3.3V - 1.75V) / (680 + 10) = 2.25 \text{ mA.}$

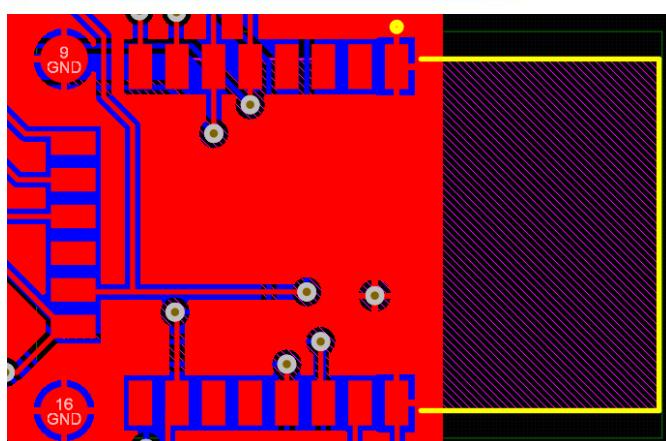
3 CONCEPTION

3.1 MODULE BLUETOOTH



Nous pouvons voir qu'il est indiqué, dans le datasheet du RN4020, qu'il ne faut pas avoir du cuivre sous l'antenne du RN4020.

Il faut avoir une zone dégagée devant de l'antenne et sur les cotées. Ceci est compréhensible puisque le cuivre peut agir comme cage de Faraday.



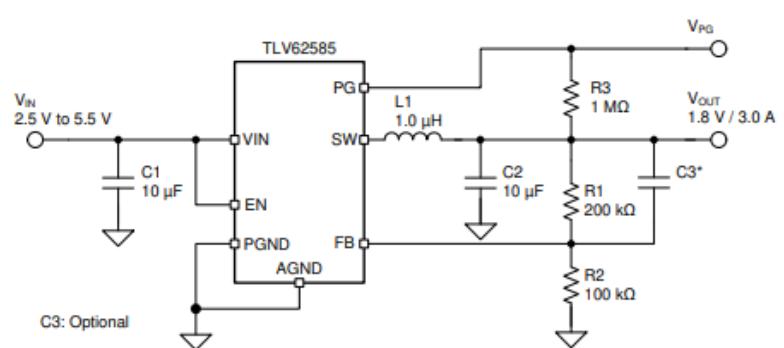
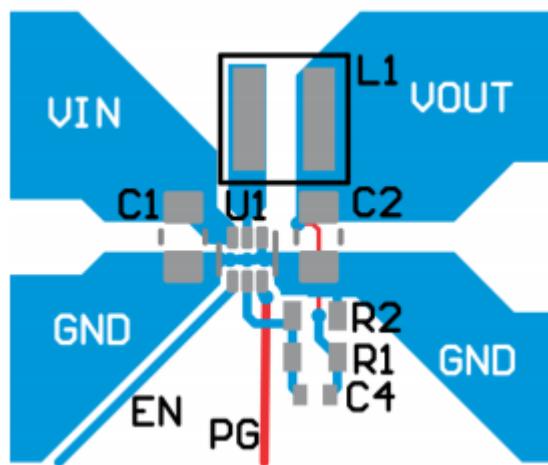
Dans la conception du PCB, nous avons dégagé toute la zone autour de l'antenne Bluetooth pour garantir que les signaux de l'antenne puissent avoir une bonne portée.

(Le boîtier de l'appareil est en plastique).

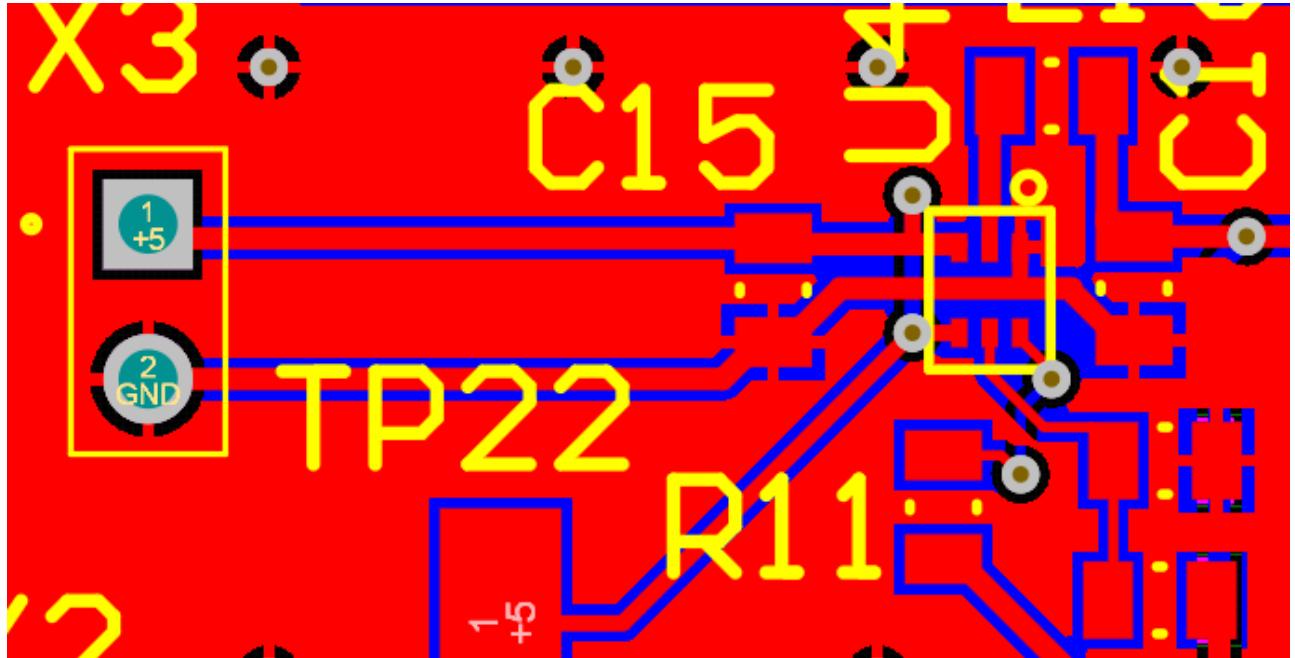


Dans un exemple, nous pouvons voir qu'il est possible d'avoir du cuivre au bottom assez proche de l'antenne. Mais je ne l'ai pas trouvé nécessaire.

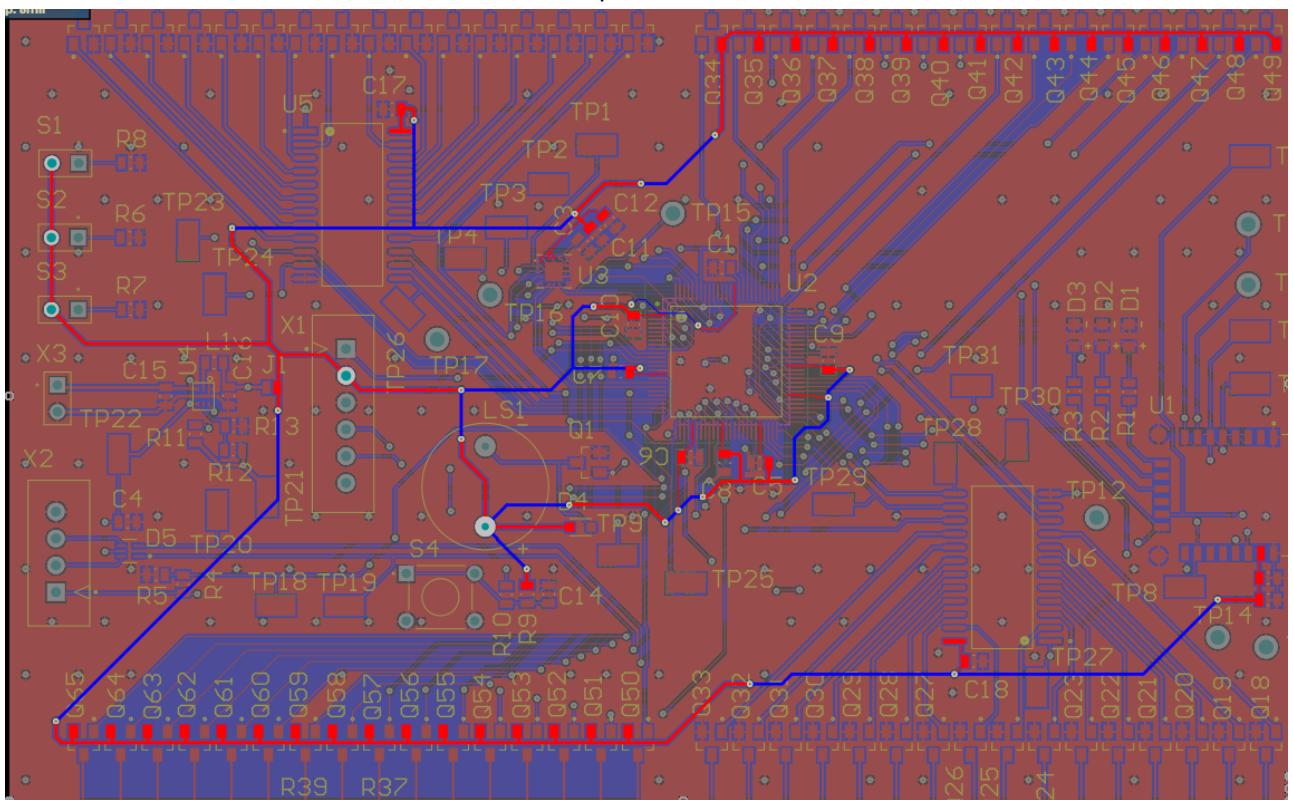
3.2 ALIMENTATION À DÉCOUPAGE



Nous pouvons voir que dans le datasheet, il nous est proposé un dessin pour le PCB. Nous allons router le PCB de la même manière afin d'éviter le plus possible que l'alimentation à découpage parasite notre carte.



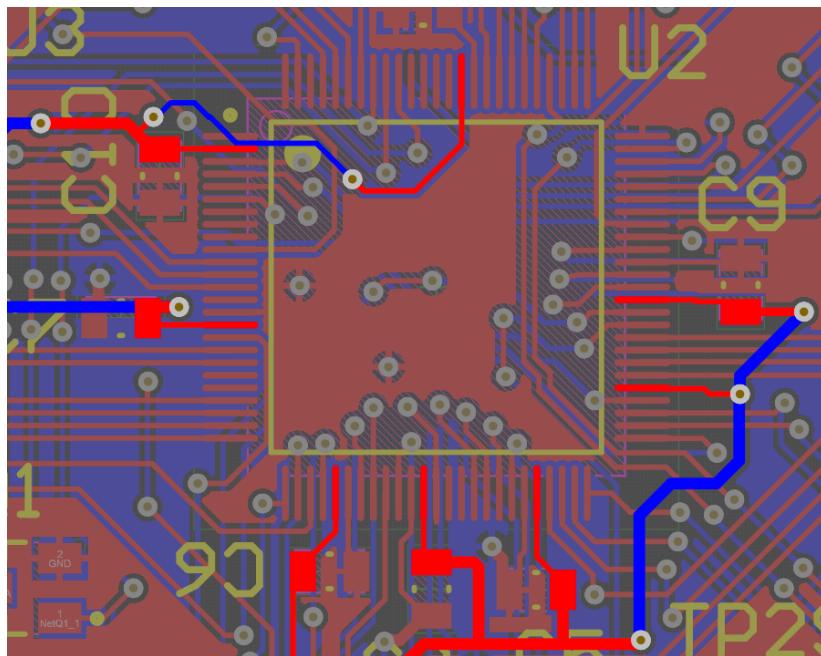
Nous pouvons voir que le dessin de la carte n'est pas exactement le même que proposé dans le datasheet, mais nous faisons passer les pistes au même endroit et plaçons les composants comme indiqué dans le datasheet.



Une autre précaution à prendre lors du routage de l'alimentation, c'est d'éviter les boucles. Notre alimentation 3.3V part depuis la droite et s'étend jusqu'à la partie gauche du circuit. La forme que l'alimentation est routée est comparable à l'expansion des racines d'un arbre. Il est simple de comprendre comment router une alimentation en suivant cette règle.

3.3 DÉCOUPLAGE

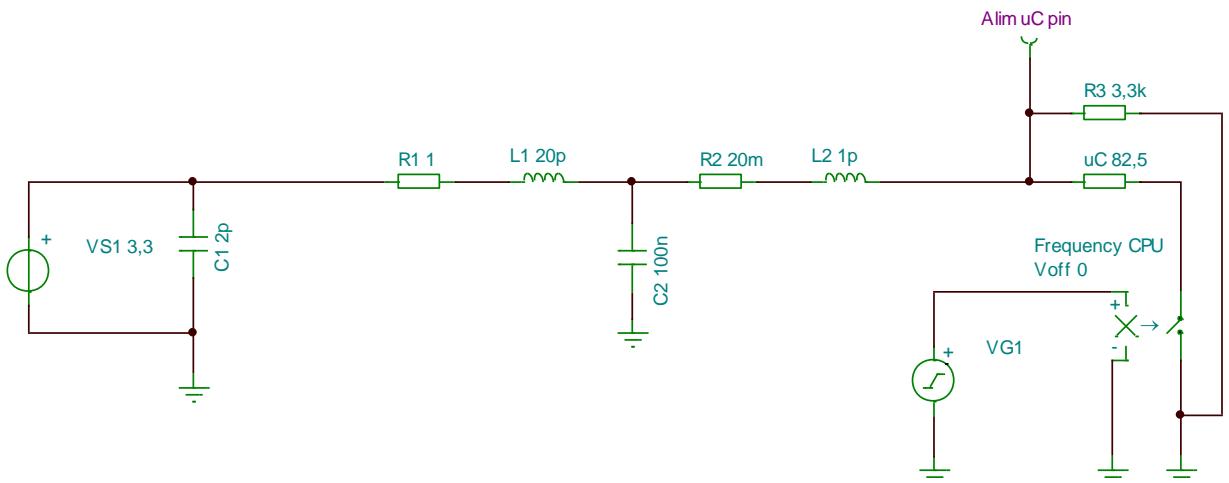
3.3.1 Microcontrôleur



Il faut faire attention au découplage du microcontrôleur. Dans notre cas, l'alimentation de ce dernier passe toujours par les condensateurs de découplage. Il est vrai, qu'il serait plus simple de router l'alimentation sous le microcontrôleur, mais ceci n'est pas correct.

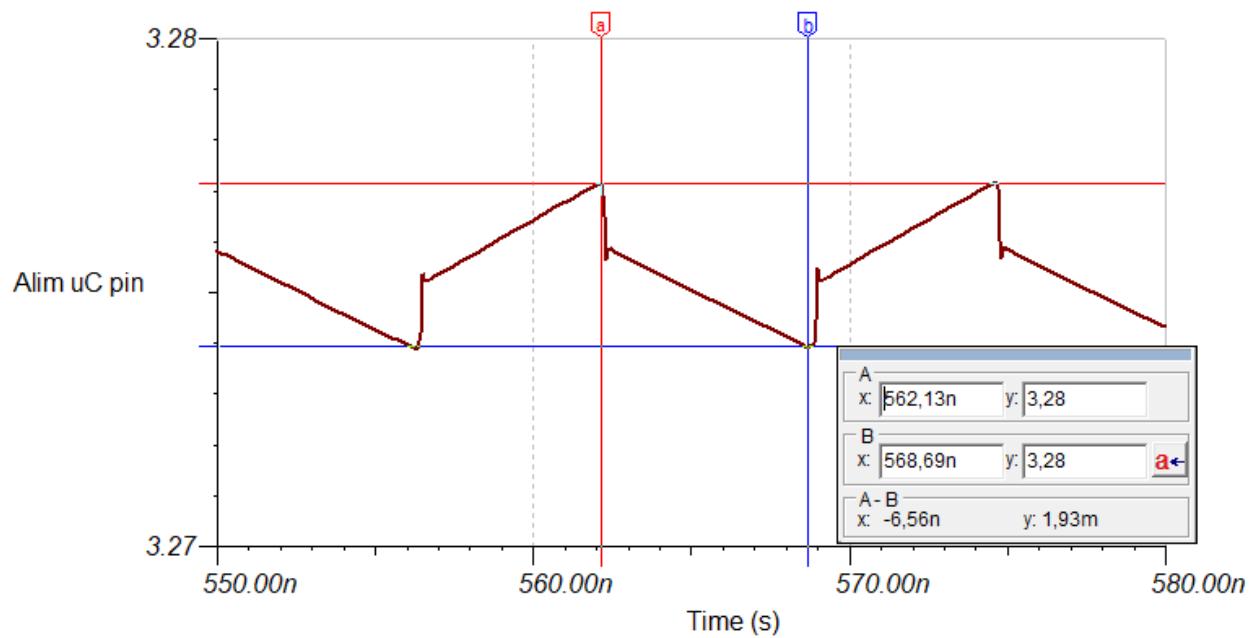
Il faut également faire attention à avoir une distance courte entre le condensateur de découplage et la pin d'alimentation du microcontrôleur.

Il faut savoir que nous pouvons représenter nos pistes comme des inductances.

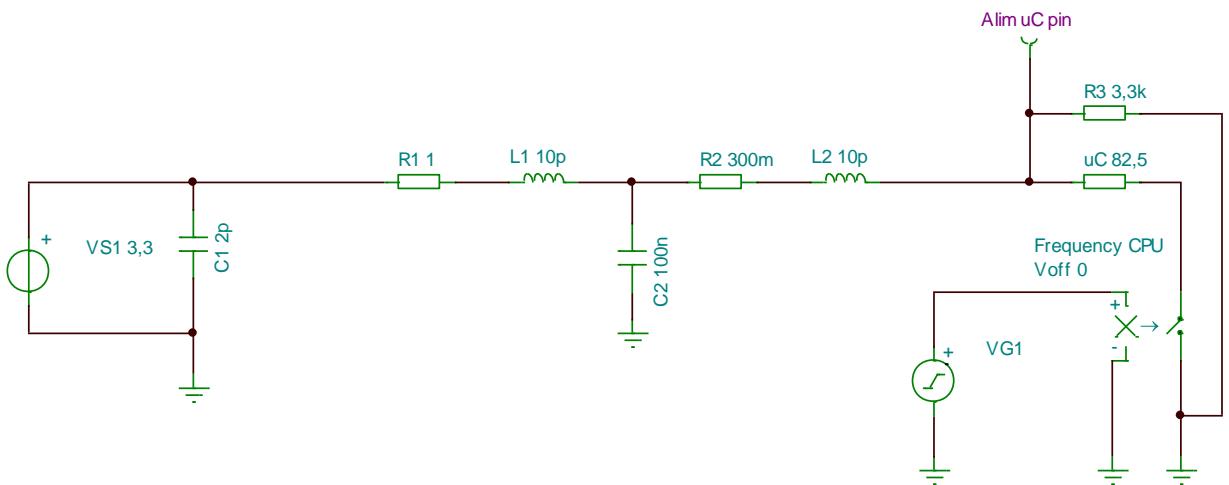


Dans un premier cas, nous allons modéliser un découplage où l'on fait très attention à la distance des pistes après le condensateur de découplage.

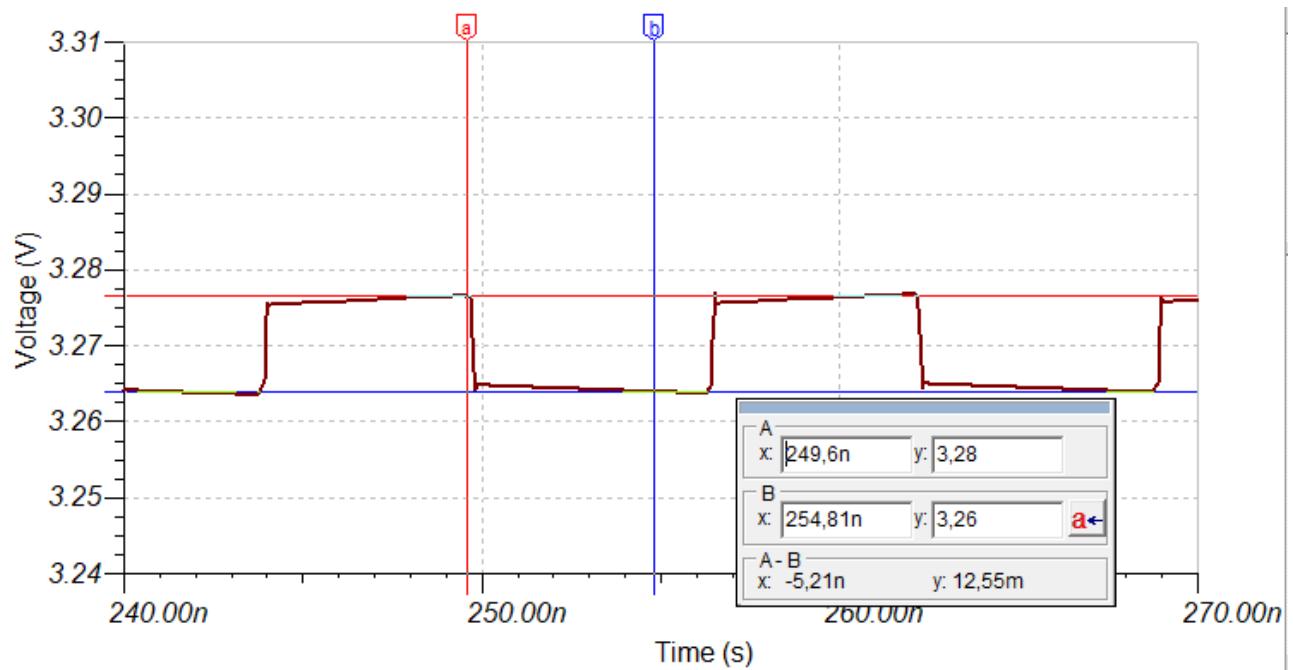
Sablier électronique



Nous pouvons voir que l'oscillation sur la pin d'alimentation du microcontrôleur est d'environ 2mV dans ce cas.



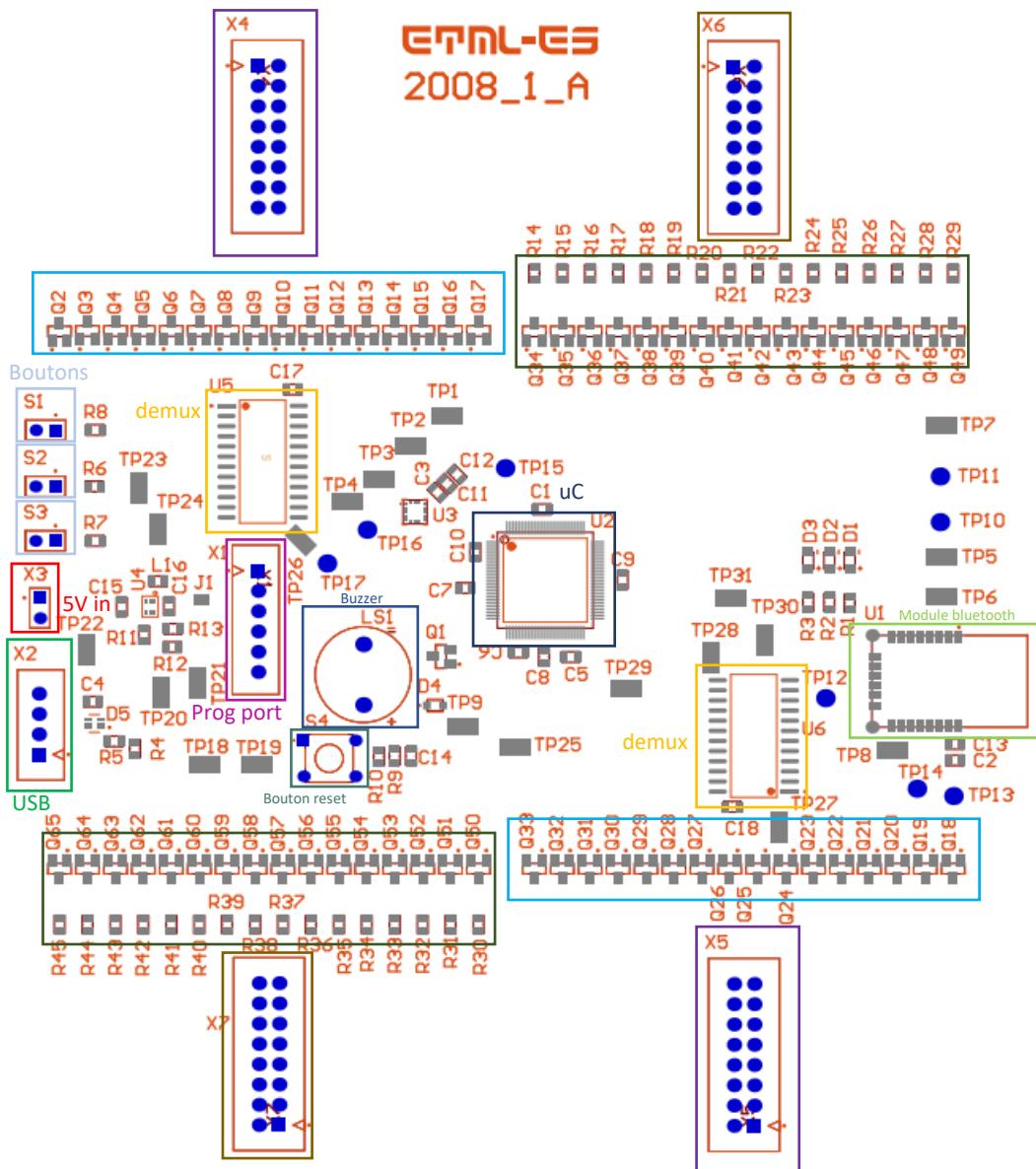
Dans le cas où l'on distante le condensateur de découplage de la pin d'alimentation du microcontrôleur, nous allons voir l'effet que ceci a sur l'alimentation du microcontrôleur. Afin de simuler un éloignement du condensateur de découplage, j'ai augmenté l'inductance parasite, ainsi que la résistance de la piste.



Nous pouvons voir que nous avons 10mV de plus d'oscillation sur l'alimentation du microcontrôleur. Ceci, uniquement dans le cas où l'ont éloigne le découplage. Les inductances utilisées dans cette simulation ne sont pas tirées de l'inductance présente sur mon circuit. Cette simulation a pour but de montrer l'effet d'un mauvais découplage.

Il faut savoir, également, que l'alimentation de mon circuit est assurée par une alimentation à découpage. Cette alimentation à découpage à environ 200mV d'oscillation. Il est donc impératif d'avoir un bon découplage.

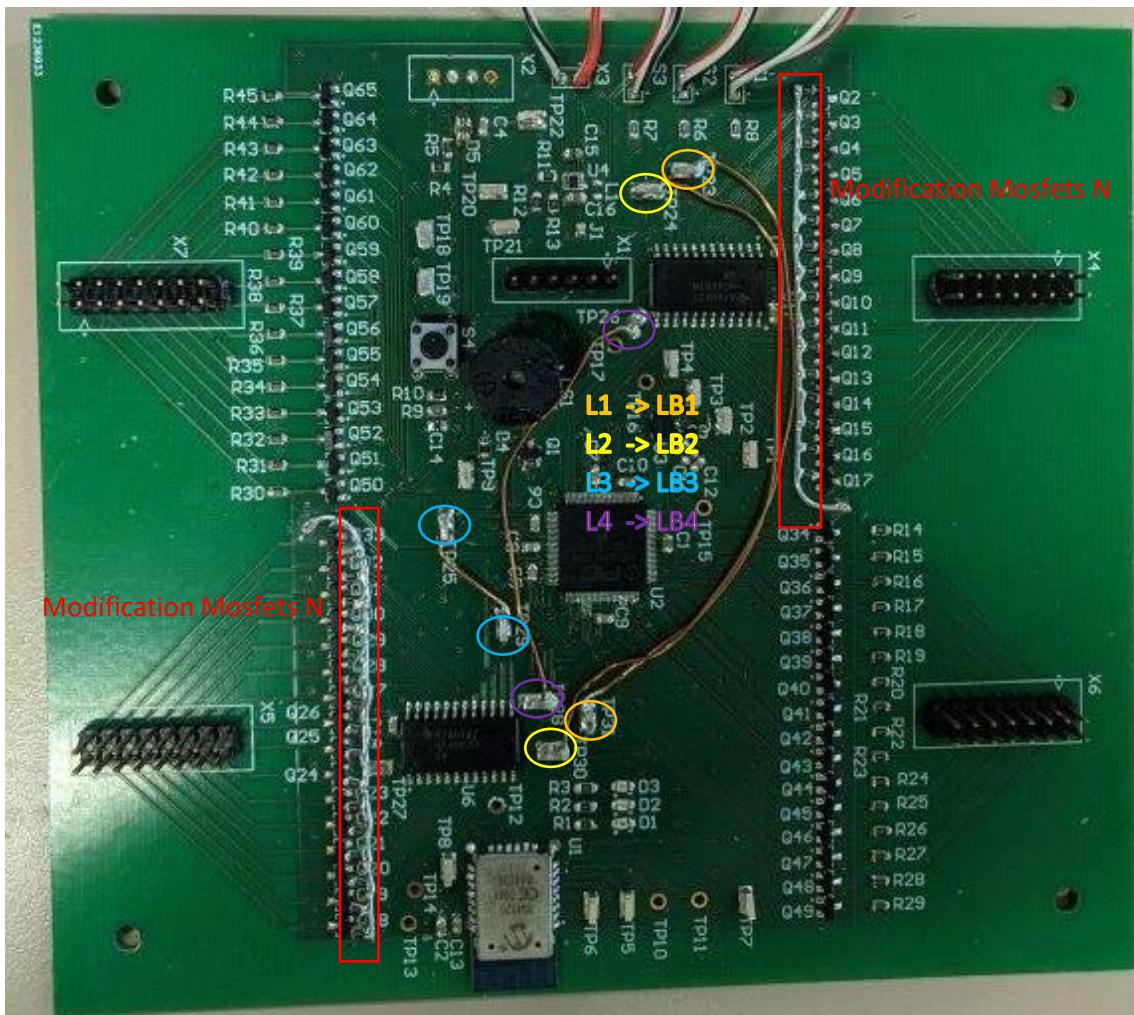
3.4 POSITIONNEMENT



Pour le placement des composants, nous avons placé le microcontrôleur au centre du circuit. Nous avons placé les autres composants en pensant à la meilleure manière pour la connectique.

Les entrées du circuit sont placées à gauche de ce dernier. Les connecteurs pour la matrice à leds sont placés en haut et en bas du circuit.

3.5 MONTAGE + MODIFICATIONS



La fonctionnalité de base de cette carte est fonctionnelle. Nous pouvons communiquer avec la centrale inertuelle en SPI et configurer le temps que l'ont veut sur le sablier. Après avoir monté la carte, j'ai dû effectuer une série de modifications pour pouvoir atteindre l'un des objectifs.

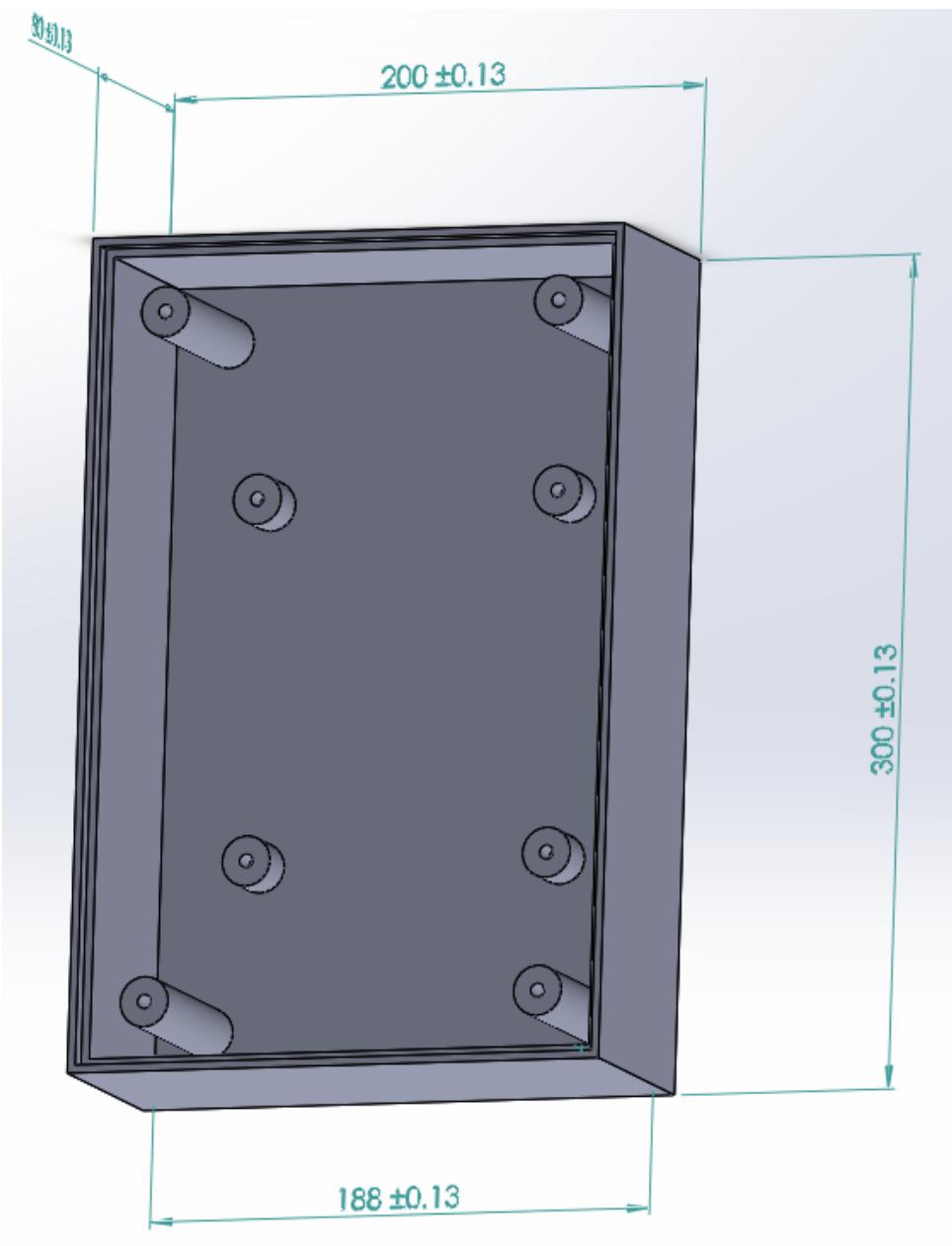
Nous avons placé une résistance de 1Mohms sur chaque grille des mosfets N. L'autre extrémité de la résistance, nous l'avons placée à la masse. Ceci est dû au fait que notre multiplexeur, lorsque le canal n'est pas défini, est en haute impédance.

Nous pouvons commander les multiplexeurs en même temps, mais la modification n'est pas due uniquement à ce fait. En effet, les sorties LB3 et LB4 ne fonctionnaient pas sur mon circuit. J'ai recherché de nombreuses fois, mais je n'ai pas trouvé un problème apparent. Afin de ne pas perdre énormément de temps sur ce fait, j'ai mis toutes les pins LB en haute impédance, et j'ai ponté ces derniers avec les pins L.

J'ai testé l'interface USB, mais l'USB n'est pas reconnu sur ma carte. J'ai testé le même programme USB sur le kitpic32 et il fonctionnait. Par manque de temps, j'ai dû malheureusement abandonner l'interface USB.

3.6 Boîtier 3D

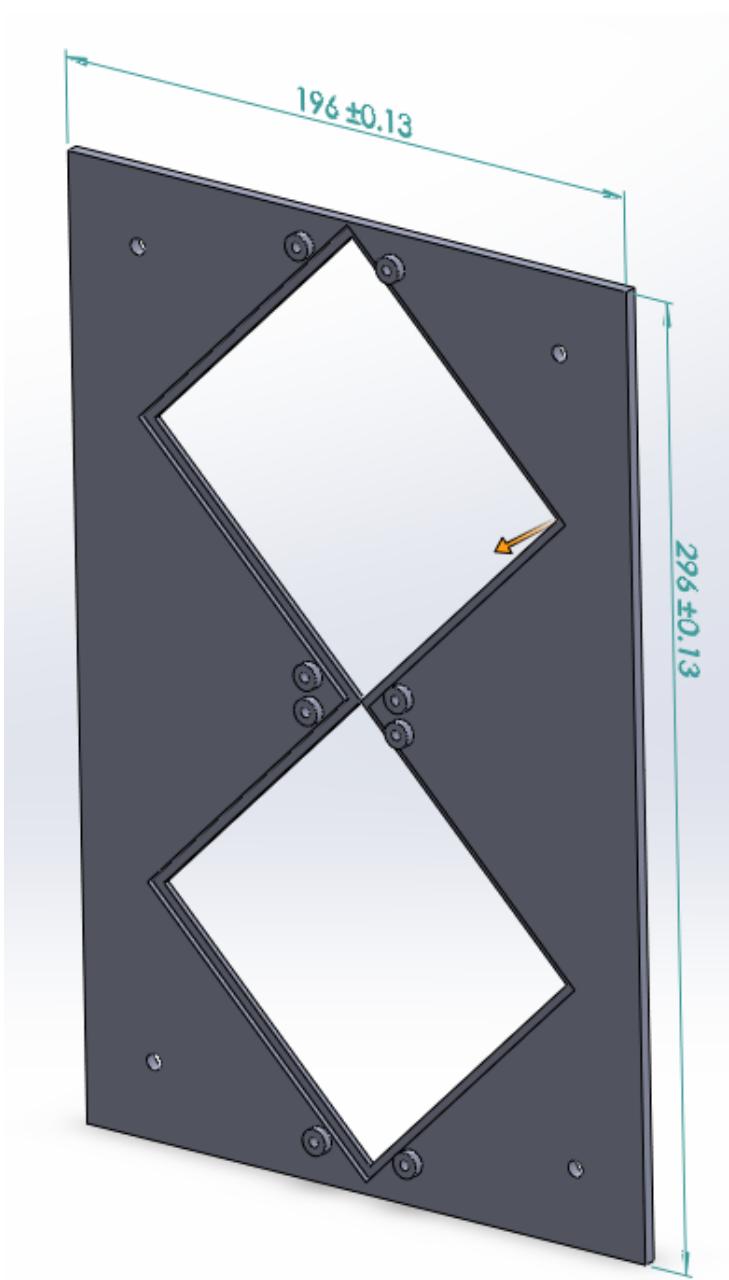
3.6.1 Bot



J'ai dessiné un boîtier 3D qui aura nos 3 PCB à l'intérieur. Je n'ai pas dessiné les sorties USB et autres, puisque je voulais les faire personnellement. La taille du boîtier est de 300 par 200 par 80, comme indiqué dans le cahier de charges.

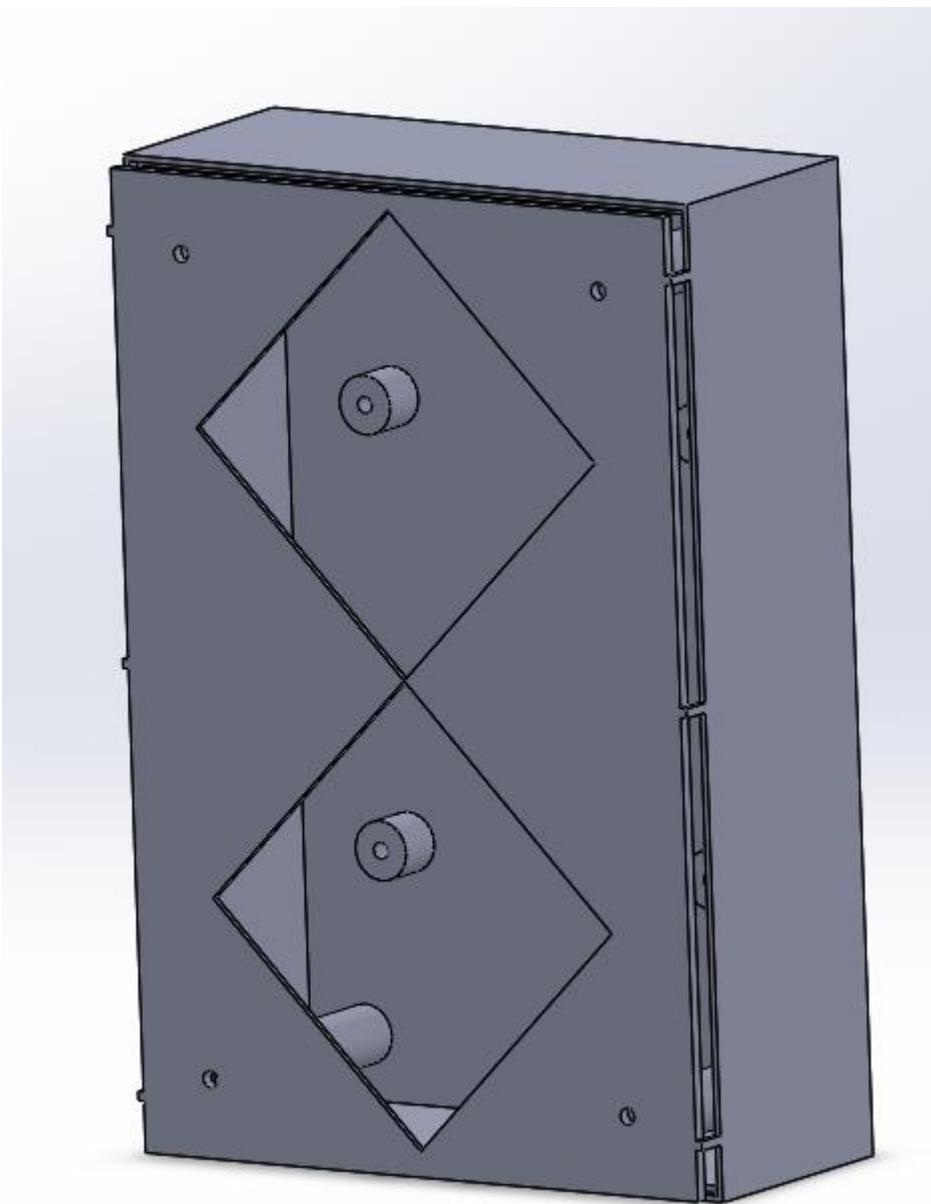
Malheureusement, l'imprimante 3D est tombée en panne avant que je puisse imprimer le boîtier. Je n'ai donc pas des photos du boîtier terminé. Il sera normalement présent pour la présentation.

3.6.2 Top



Le couvercle du boîtier a une épaisseur de 3mm. Sur ce couvercle, nous viserons les deux matrices à leds.
J'ai prévu des rainures afin de placer un plexiglas.

3.6.3 Assemblage



Pour l'impression 3D, j'ai réalisé un assemblage que j'ai fusionné en une seule pièce. Nous pourrons ainsi, imprimer le tout en une seule fois.

4 SOFTWARE

4.1 CONFIGURATION HARMONY

4.1.1 Timers

Le calcul d'un timer se fait grâce à l'équation : $\frac{Time(us)*tFreq}{Prescaler} - 1 = Valrecharge$

Sur Excel, j'ai réalisé un document qui me permet de calculer directement la valeur des timers en suivant cette équation

TIMER 0				
Délai souhaité(us)	3000	Fréquence timer(MHz)	80	
Résolution du timer	16 bits			
Valeur du prescaler	Fréquence utilisable [Hz]	Période d'incrémentation [s]	Délai souhaité [us]	Valeur de recharge
1	80.0E+6	12.5E-9	3000	239'999.00
2	40.0E+6	25.0E-9	3000	119'999.00
4	20.0E+6	50.0E-9	3000	59'999.00
8	10.0E+6	100.0E-9	3000	29'999.00
16	5.0E+6	200.0E-9	3000	14'999.00
32	2.5E+6	400.0E-9	3000	7'499.00
64	1.3E+6	800.0E-9	3000	3'749.00
128	625.0E+3	1.6E-6	3000	1'874.00
256	312.5E+3	3.2E-6	3000	936.50

Si nous prenons l'exemple ci-dessous, nous voulons que l'interruption du timer soit appelée tous les 3ms.

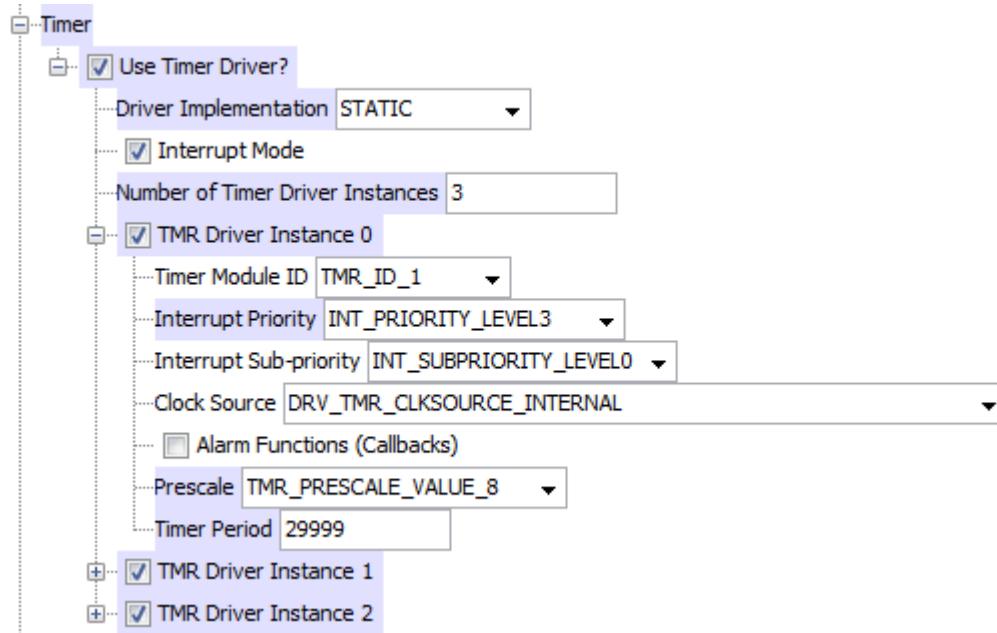
$$\frac{Time(us)*Ftimer}{Prescaler} - 1 = \frac{3000*80}{64} - 1 = 3749$$

Avec un prescaler de 64, nous devons charger 3749 sur un timer 16bits pour avoir un temps de 3ms.

ETML

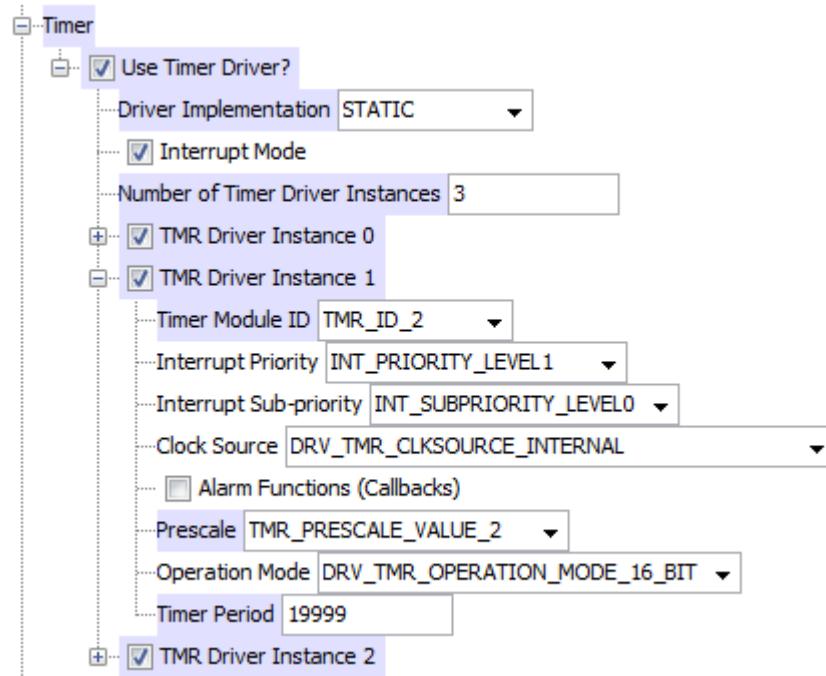
Sablier électronique

//Timer 1



Pour le timer 1, nous voulions avoir un temps de 3ms. Soit : $\frac{Time(\mu s)*Ftimer}{Prescaler} - 1 = \frac{3000*80}{8} - 1 = 29999$
Dans ce cas, j'ai choisi un prescaler de 8. La priorité d'interruption est 3.

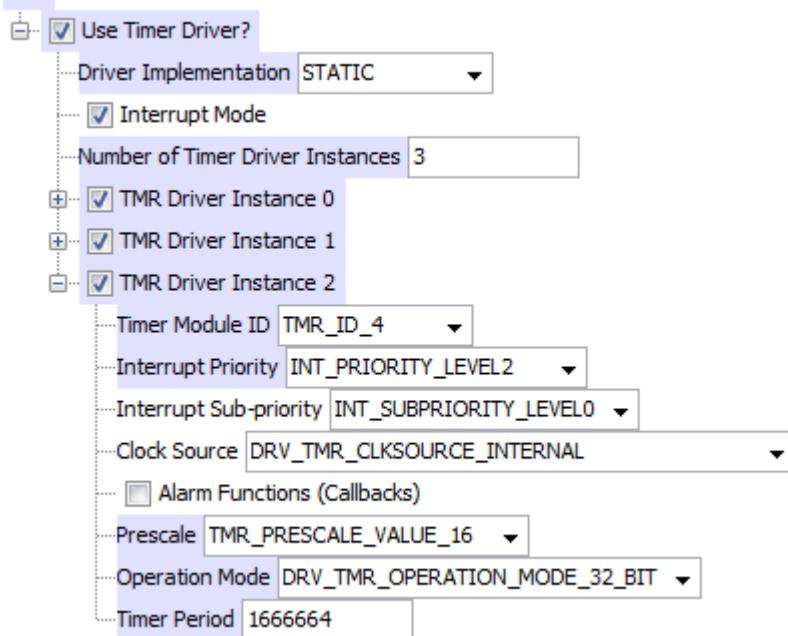
//Timer 2



Pour le timer 2, nous voulions une fréquence de 2kHz. Cette interruption nous sert à afficher ligne par ligne les valeurs de notre tableau sur les matrices à leds. En sachant que nous avons une matrice de 16 lignes, nous avons donc 125Hz d'affichage.

$$\frac{Time(\mu s)*Ftimer}{Prescaler} - 1 = \frac{500*80}{2} - 1 = 19999$$

//Timer 4 et 5 – DropParticule time.

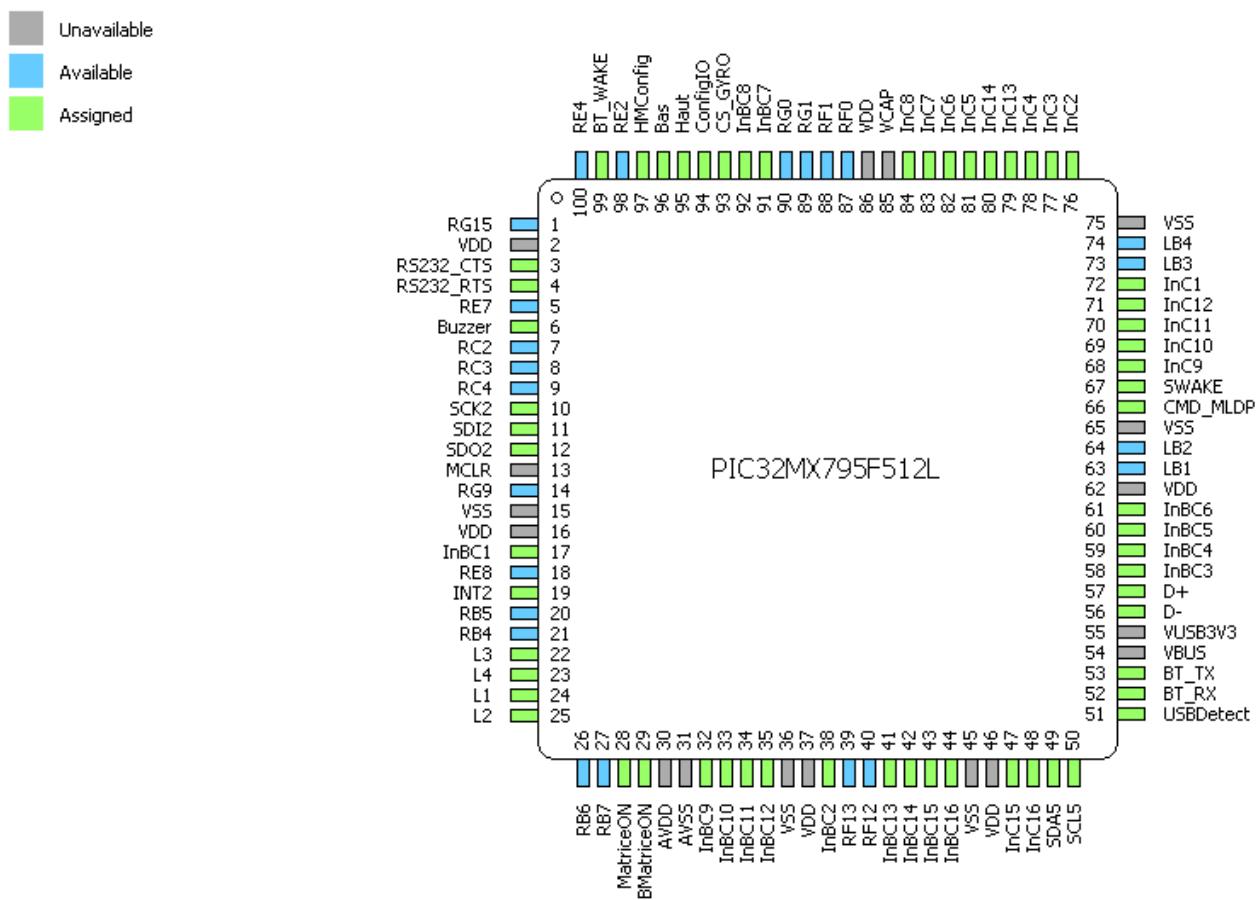


Ce timer nous sert à avoir une référence de temps pour la fonction DropParticule. Ce timer est appelé 3 fois par secondes. La vitesse d'écoulement de particule sera divisée par un nombre donné dans l'application APP_Task. Si la variable vaut 2, l'écoulement sera de 3 particules toutes les deux secondes.

$$\frac{Time(\mu s) * Ftimer}{Prescaler} - 1 = \frac{333333 * 80}{16} - 1 = 1666664$$

Il faut savoir que nous avons utilisé dans ce cas, un timer 32bits. Soit, le timer 4 et 5 lié.

4.1.2 Pinout Harmony



Presque toutes les pins sur notre microcontrôleur sont utilisées. Nous pouvons voir des pins comme tous les LB1 à LB4 qui sont en bleu. Ces pins sont disponibles. Les pins LB1 à LB4 ont été mis dans cet état puisque nous ne voulions pas utiliser ces dernières.

4.2 ALGORITHMES DE PARTICULES

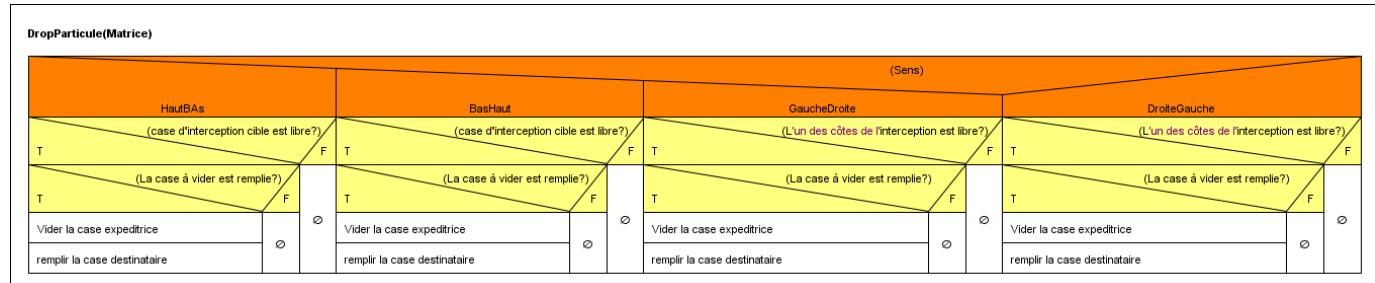
4.2.1 Effacement de la matrice

```
clearMatrice(matrice)

for i ← 0 to 15
    MatriceaxeY[i] := 0xFFFF //eteindre les leds
```

Cette fonction nous permet d'éteindre toute la matrice à leds en même temps. Dans ce projet, l'affichage de la matrice à leds est géré par un tableau de 16 lignes de valeurs à 16bits. Puisque nous commandons des mosfet P, nous devons mettre un 1 en sortie si l'on veut couper le passage du courant. Afin d'éteindre toute la matrice à leds, nous devons procéder au balayage de tout le tableau grâce à une boucle for pour placer FFFF (que des 1 sur tous les bits du tableau).

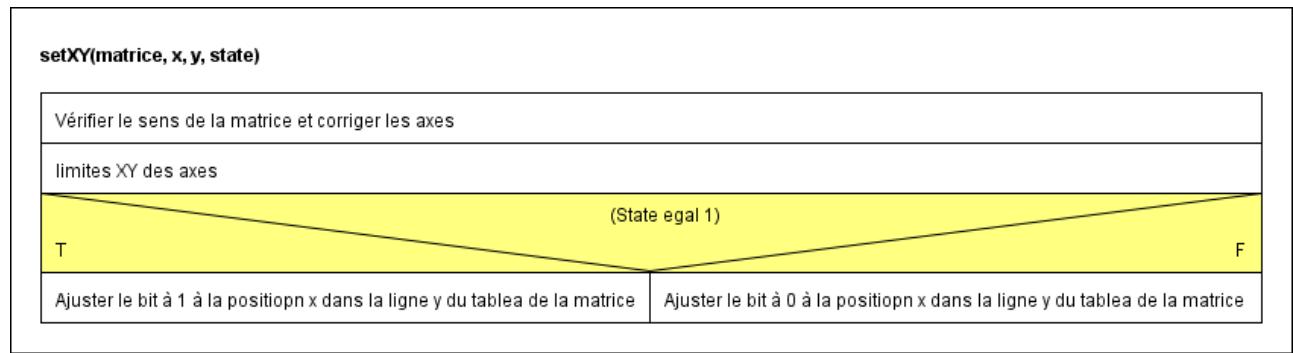
4.2.2 DropParticule, transférer une particule d'une matrice à une autre



Dans ce spectrogramme, nous pouvons voir le principe du transfert d'une particule d'une matrice à une autre. Lorsque la fonction est appelée, puisque les coordonnées de l'affichage variant selon l'orientation, il faut contrôler l'orientation actuelle de l'affichage. En effet, les coordonnées de l'interception entre les deux affichages varient selon l'orientation de ce dernier. Prenons l'exemple de l'orientation HautBas, dans ce cas, l'interception de l'affichage top se trouve aux coordonnées XY(15,15) et pour l'affichage bot XY(0,0). Lorsque l'orientation change à DroiteGauche, l'interception de l'affichage top se trouve désormais aux coordonnées XY(15,0) et de l'affichage bot XY(0,15).

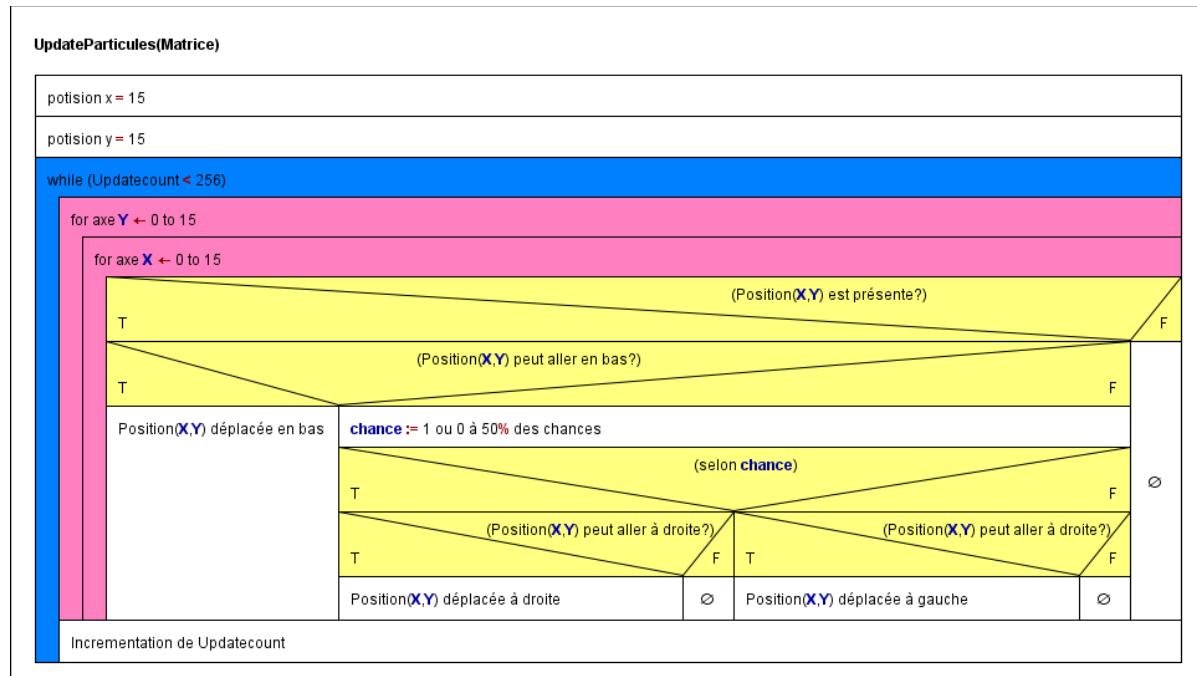
Lorsque nous sommes dans le cas de HautBas(même chose pour les autres cas), nous vérifions si la case cible est libre (la case où la particule va venir se loger). Nous vérifions par après, si oui, si la case à vider contient une particule. Dans le cas où la case contient une particule, nous transférons la particule de la case expéditrice à la case destinataire.

4.2.3 SetXY, fonction pour imposer un état sur un case de l'affichage matriciel



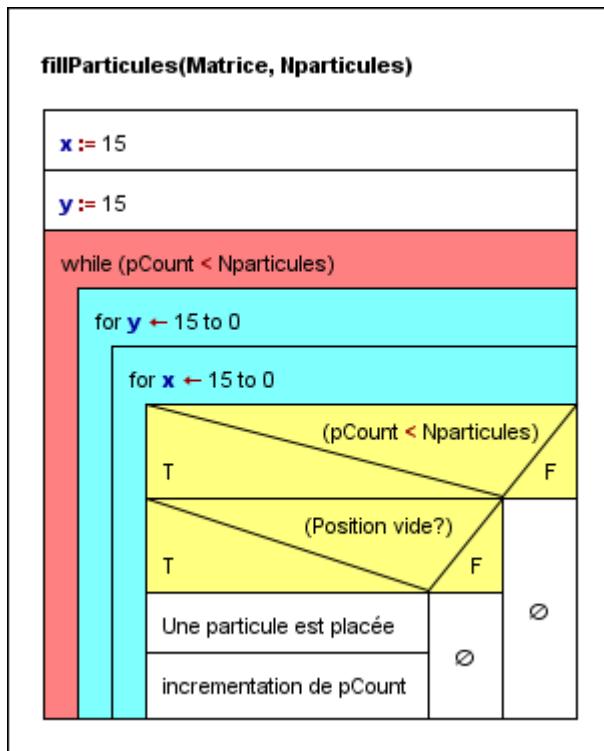
Cette fonction est utilisée pour imposer un état sur l'une de led de la matrice choisie. Il y a une vérification du sens de la matrice et après ceci, nous corrigons les axes de la matrice pour garder toujours la position XY(0,0) en haut à droite de la matrice. Il y a une vérification des limites des axes qui est faite après ceci afin de garantir que les axes soient dans les limites de l'affichage. Si l'état choisi est égal à 1, nous appliquons cet état sur le bon bit 2^x de la ligne y(sur le tableau). S'il est égal à 0, nous appliquons l'état 0.

4.2.4 UpdateParticules, fonction pour mettre à jours les particules dans l'affichage



Nous fixons le départ du balayage sur la position XY(15,15) qui correspond à la position se trouvant la plus basse sur l'affichage (selon l'orientation). Après, nous entrons dans une boucle jusqu'à 255 depuis 0. Nous initialisons deux boucles for pour pouvoir couvrir tout le tableau de 16x16. Le balayage commence sur la première ligne (y) et finit sur le dernier (y). Lorsque nous sommes dans la boucle for, nous allons vérifier à chaque fois s'il y a une particule présente sur la position XY actuelle. S'il y a une particule, nous vérifions si elle peut être déplacée en bas. Dans le cas où elle peut être déplacée en bas, nous la déplaçons vers le bas. Si elle ne peut pas être déplacée vers le bas, nous allons établir une valeur entre 0 et 1 à 50 % des chances. Selon le résultat de cette valeur, nous allons vérifier si la particule peut aller à droite (1) ou à gauche (0). Nous allons déplacer la particule dans ce sens si elle peut être déplacée dans le sens choisi. Sinon, nous n'allons pas vérifier l'autre sens et nous allons passer à la position suivante.

4.2.5 fillParticules



Cette fonction permet de remplir notre matrice avec un nombre des particules demandée sur les paramètres. Nous allions rentrer dans une boucle. Cette boucle prendra fin lorsque le nombre des particules demandées sera atteint. Nous retrouvons deux boucles `for` dans cette fonction pour faire le balayage de toute la matrice à led, dans l'axe x et l'axe y. Pour chaque position, nous allons vérifier si le nombre des particules a été atteint afin de ne plus mettre aucune particule. Dans le cas où le nombre des particules n'a pas été atteint, nous allons vérifier si la case actuelle est vide, nous allons placer une particule et incrémenter le compteur des particules.

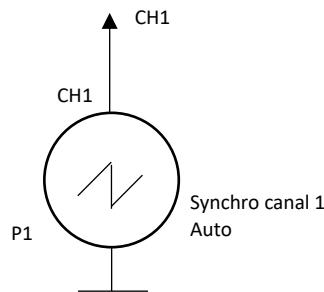
4.3 APP USB

J'ai essayé d'implémenter l'interface USB. Malheureusement, je me suis rendu compte qu'il y avait un problème hardware sur l'interface USB. L'application USB que j'ai testée sur le kitpic32 fonctionnait, mais pas sur ma carte. Il faut savoir que le uC du kitpic32 est le même que celui de mon projet. Je n'ai donc malheureusement pas pu implémenter l'interface USB dans ce projet.

5 RÉSULTATS + MESURES

5.1 MESURE DE L'ALIMENTATION

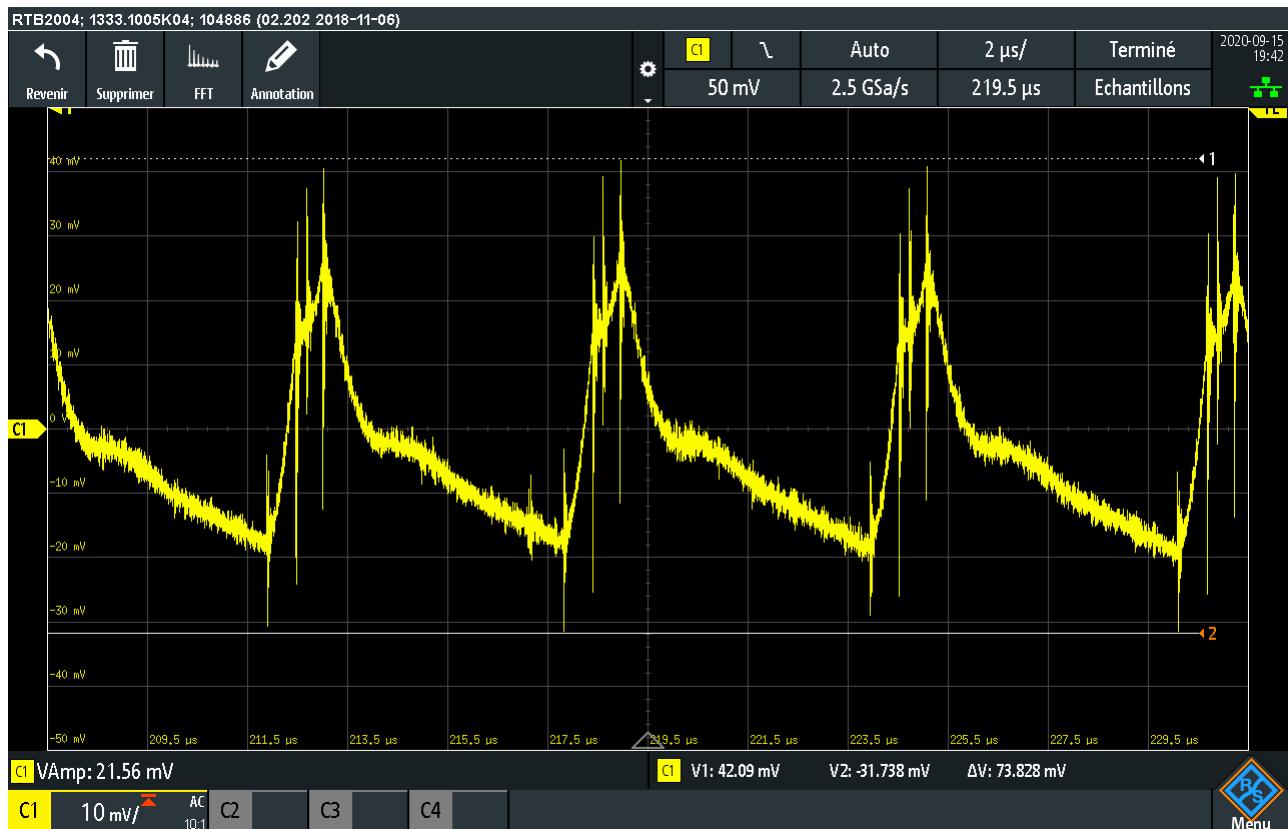
5.1.1 Schéma de mesure



5.1.2 Résultat



Nous pouvons observer dans cette image une alimentation plutôt stable de 3.21V.



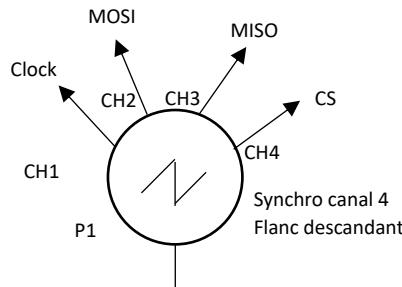
Lorsque nous mesurons l'oscillation sur l'alimentation (en charge), nous avons un delta de 74mV. Si nous calculons le % d'oscillation par rapport à la tension d'alimentation, nous avons :

$$(0.074/3.21)*100 = 2.3\%.$$

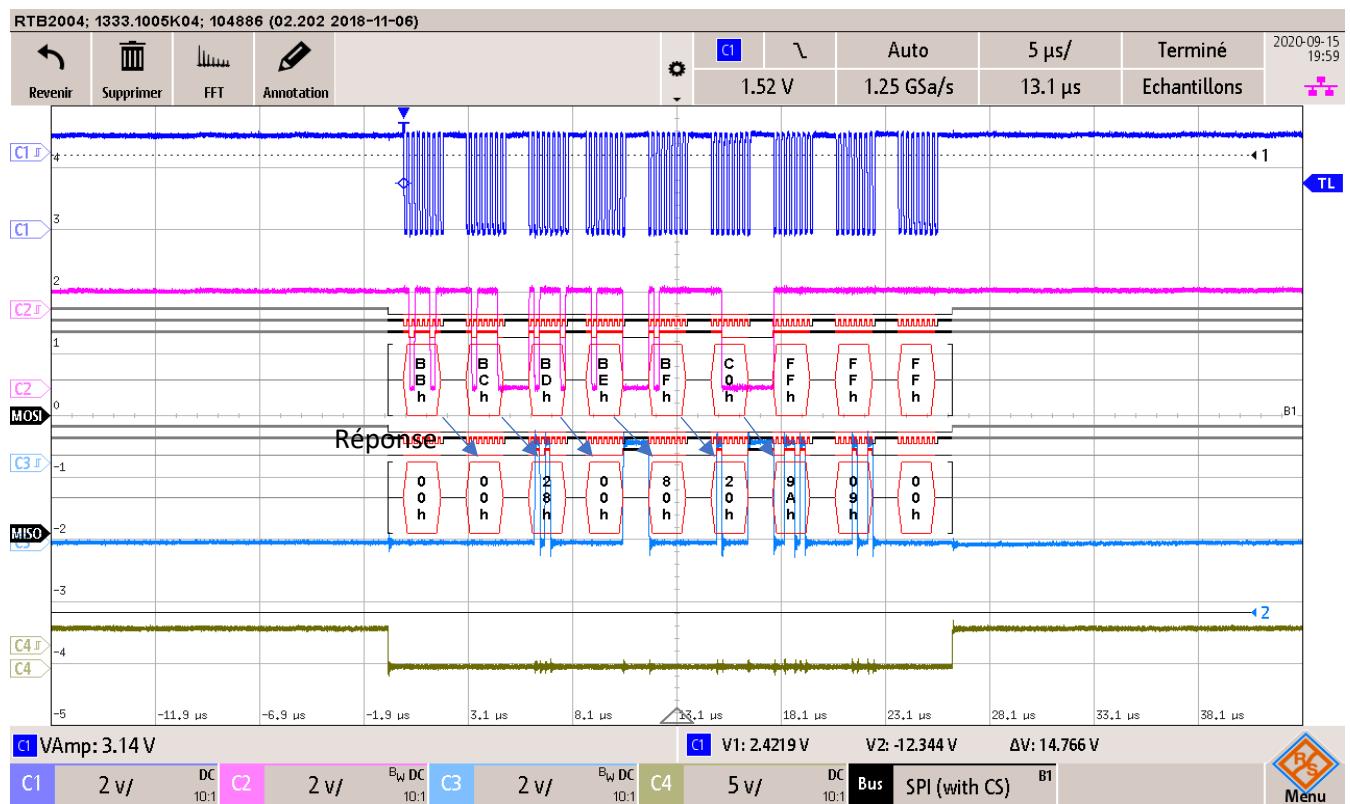
Nous avons une oscillation sur l'alimentation de 2.3%. Ceci est une oscillation normale pour une alimentation à découpage.

5.2 MESURE DE LA COMMUNICATION SPI

5.2.1 Schéma de mesure



5.2.2 Mesure

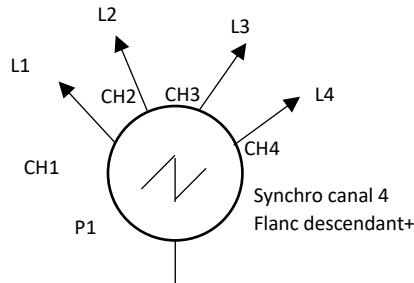


Le canal 2 de notre oscilloscopogramme contient les registres (+1 au MSB) que l'on veut obtenir. Le premier registre demandé, si l'on pense aux registres indiqués au point 2.5.4, est le registre 59(3B en hex). Si l'on ajoute la valeur 128 décimale (1 au bit MSB d'un octet), nous obtenons BB en hexadécimal.

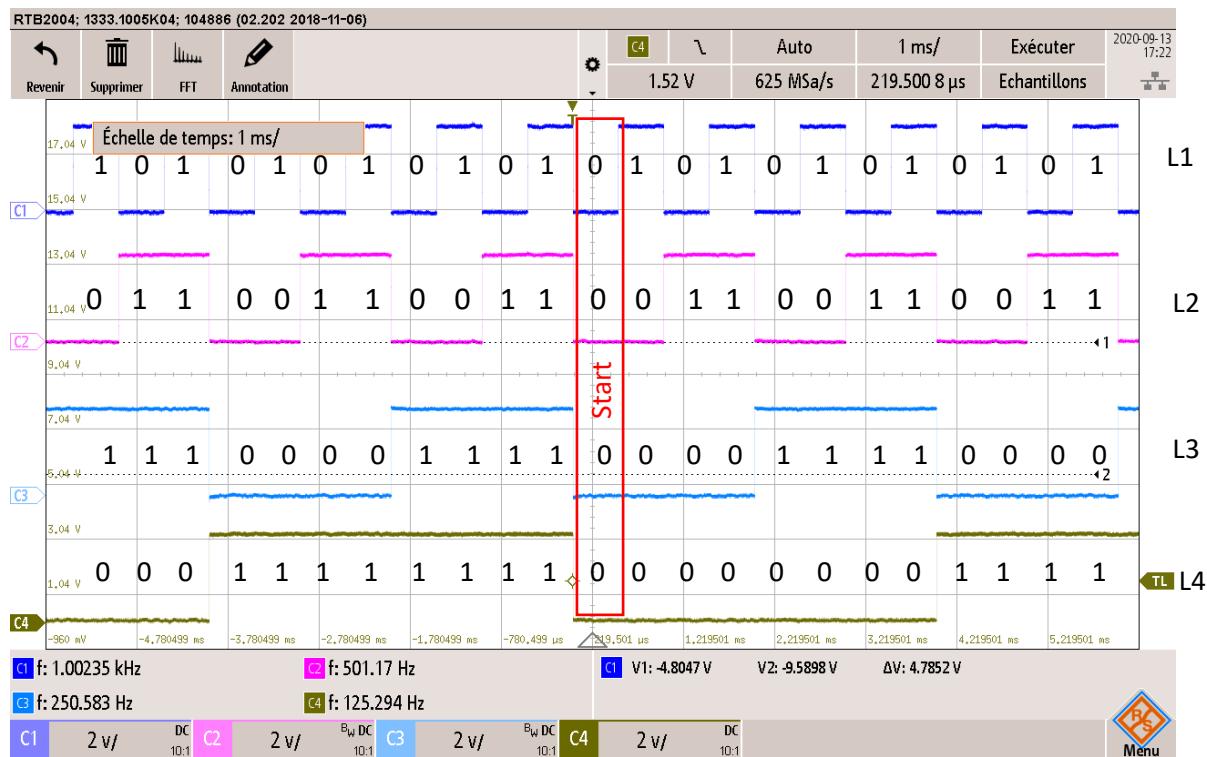
La communication de la centrale inertuelle est en Half-Duplex, nous avons donc la réponse de ce dernier sur les 8 coups de clock qui suivent l'envoi du registre. Nous pouvons voir la réponse numéro 5 et la réponse numéro 6. Ces réponses indiquant l'axe Z. Dans cet axe, nous pouvons voir la valeur 209A ce qui correspond à une valeur de 8361. La sensibilité de notre accéléromètre est fixée sur +4g. Nous avons environ 1g sur l'axe Z.

5.3 MESURE DE LA COMMANDE LEDS – DEMUX

5.3.1 Schéma de mesure



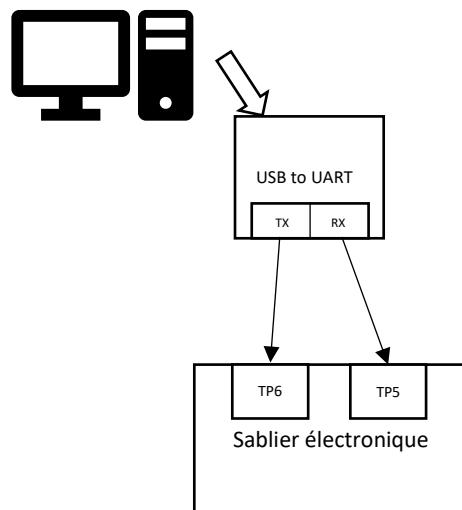
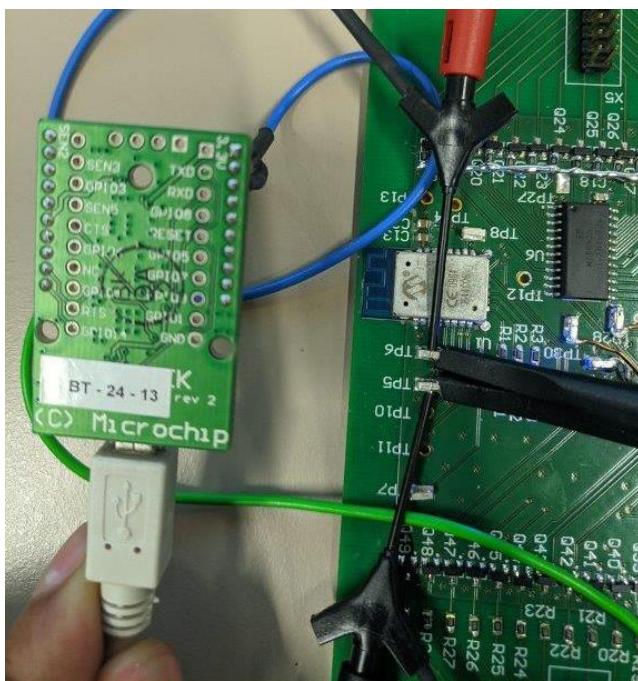
5.3.2 Mesure



Nous pouvons voir qu'il y a une séquence de comptage entre L1 et L4. L1 étant le bit LSB et L4 le bit MSB. Cette incrémentation est gérée de manière software afin d'allumer les bons bits sur les colonnes. Lorsque le compteur vaut 0(ligne 0 en y), nous allons mettre sur les colonnes, les bits pour la ligne 0.

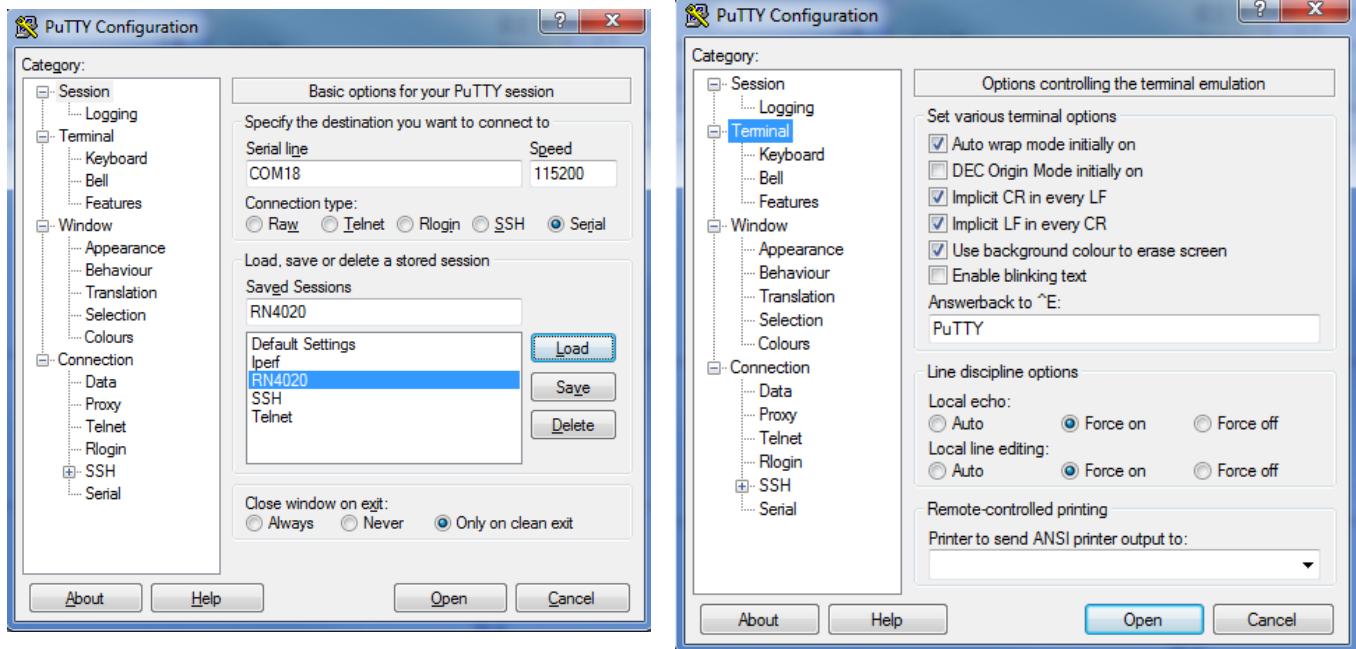
5.4 MESURES BT

5.4.1 Uart to USB converter



Puisque je n'avais pas le temps de réparer l'interface USB, j'ai voulu essayer de mettre en place une interface Bluetooth simple. Pour faire ceci, j'ai utilisé un convertisseur USB à UART afin de configurer facilement mon module Bluetooth.

5.4.2 Configuration Putty



Sur putty, nous devons choisir le port COM qui a été attribué au convertisseur USB à UART. Le débit de communication de base avec le module USB est de 115200 bauds.

5.4.3 Test de configuration

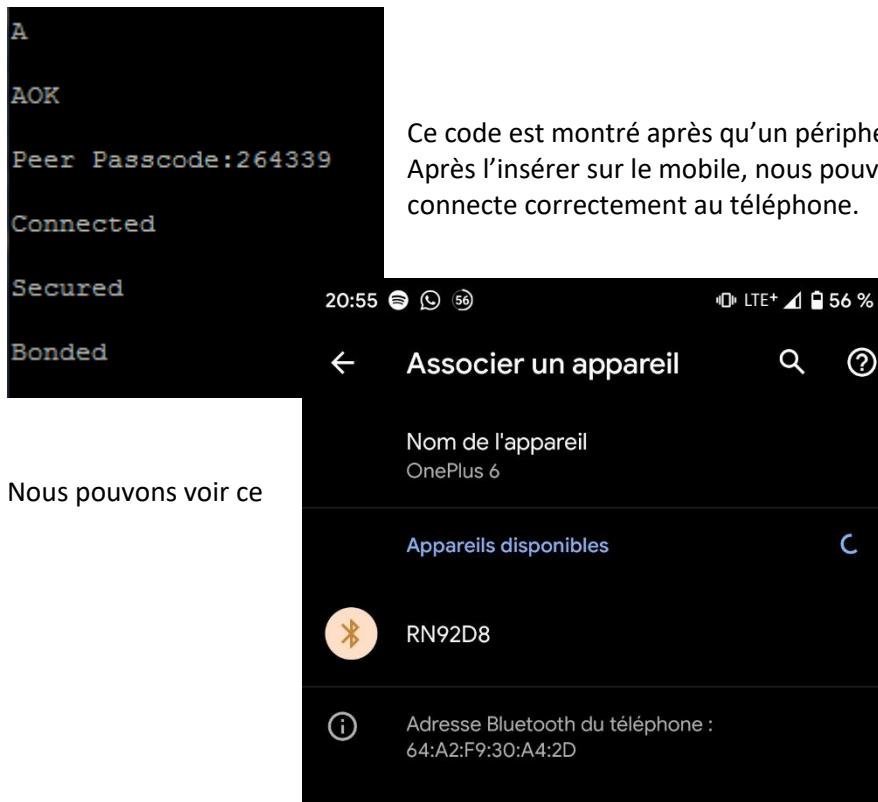
```
+          // echo on
SF,2      // perform complete factory reset
SS,C0000000 // enable support of the Device Information and Battery services
SR,00000000 // set the RN4020 module as a peripheral
R,1      // reboot to apply settings
```

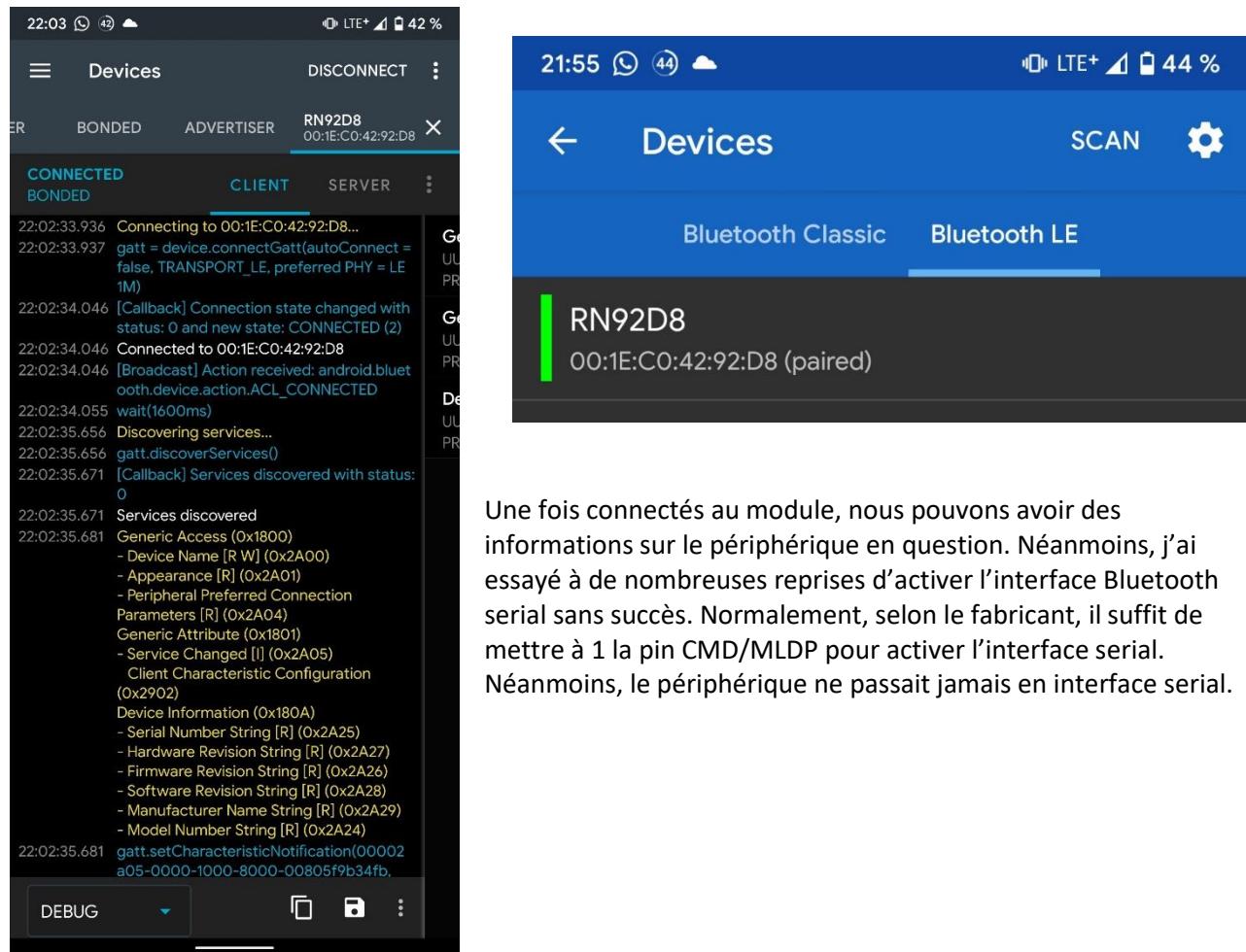
Nous allons utiliser l'exemple donné par le fabricant microchip sur le site :
<https://microchipdeveloper.com/ble:rn4020-app-example-smartphone-basic-demo>

```
SF,2  
SF,2  
AOK  
  
SS,C0000000  
SS,C0000000  
AOK  
  
SR,00000000  
SR,00000000  
AOK  
  
R,1  
R,1  
Reboot  
  
CMD
```

Nous pouvons communiquer facilement avec le module Bluetooth

Par la suite, nous devons envoyer la commande A afin de le rendre visible à l'appariement.

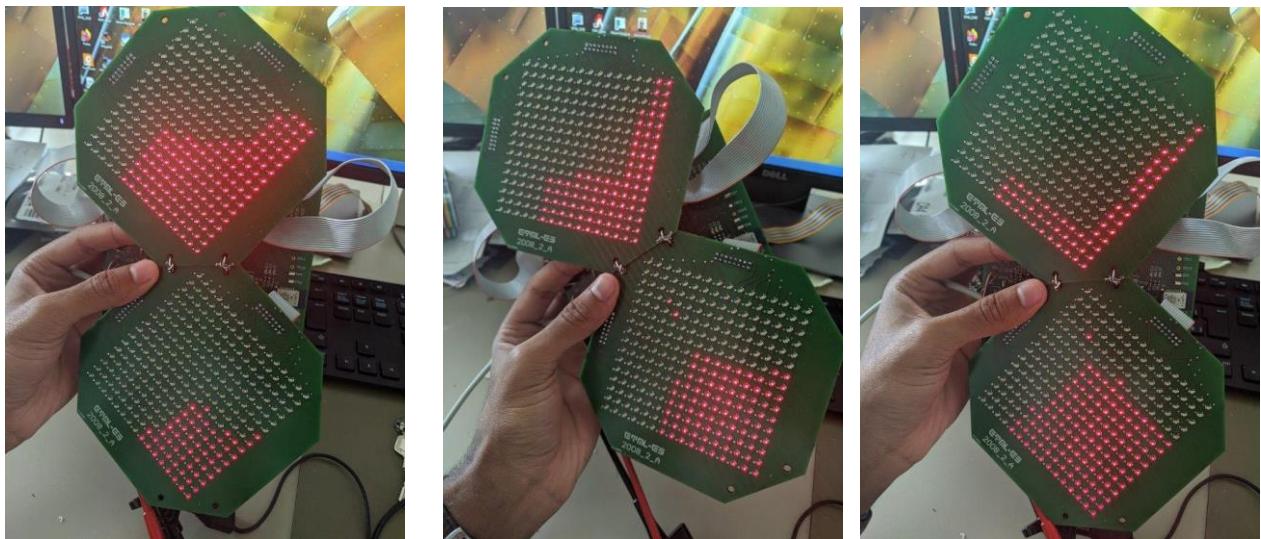




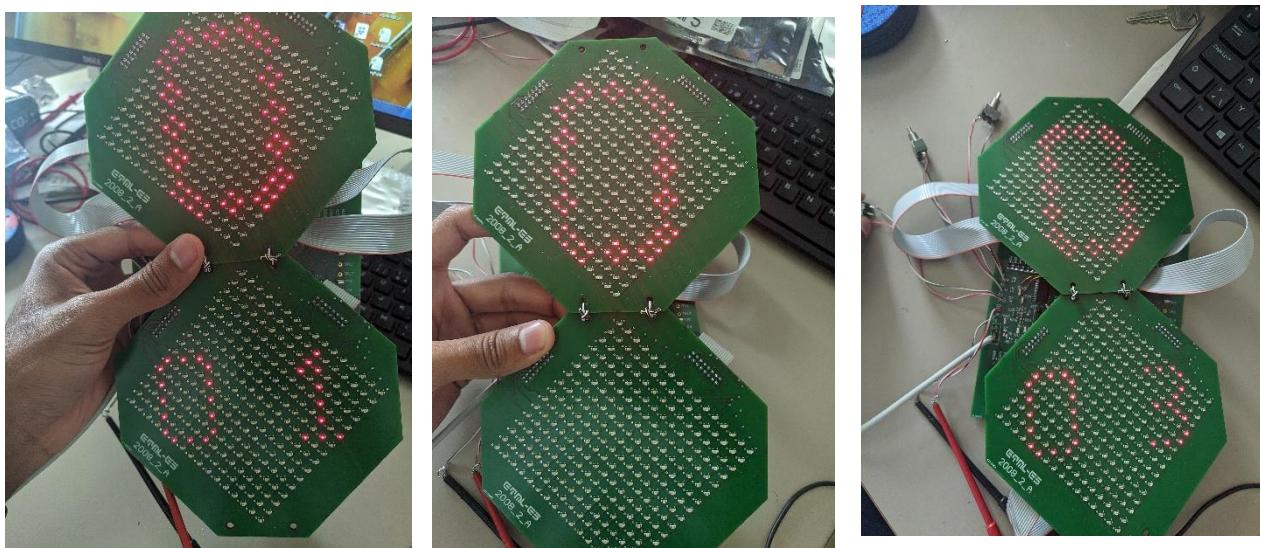
Une fois connectés au module, nous pouvons avoir des informations sur le périphérique en question. Néanmoins, j'ai essayé à de nombreuses reprises d'activer l'interface Bluetooth serial sans succès. Normalement, selon le fabricant, il suffit de mettre à 1 la pin CMD/MLDP pour activer l'interface serial. Néanmoins, le périphérique ne passait jamais en interface serial.

5.5 FONCTIONNEMENT DU SABLIER

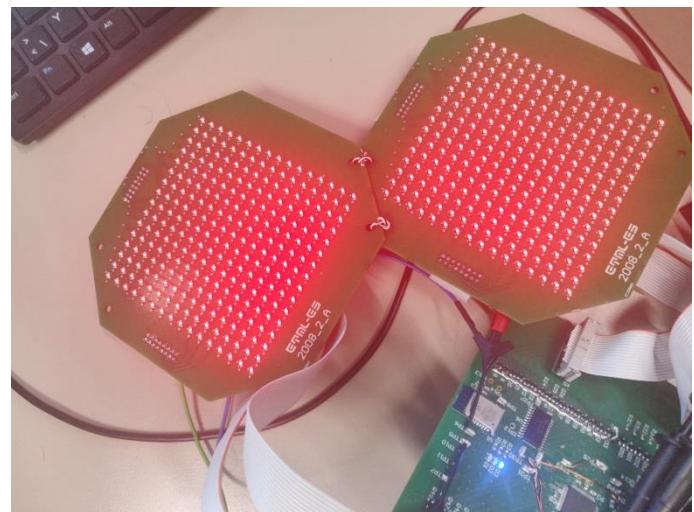
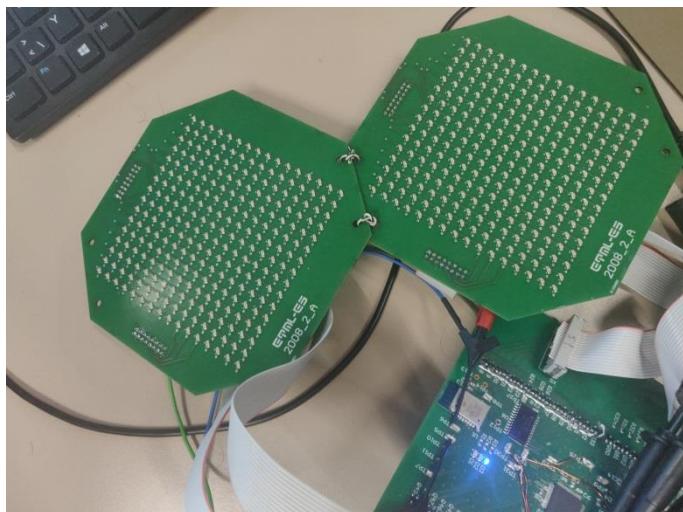
5.5.1 Transfert des particules d'un affichage à l'autre



Lorsque nous alimentons l'affichage, fixe 180 particules sur la matrice haute. Il y a un écoulement des particules de 3 particules par secondes, au départ, ce qui nous donne un temps d'une minute.



Lorsque nous appuyons sur le bouton HMconfig(S3), nous entrons dans la configuration des heures et des minutes. En appuyant sur les boutons Haut et Bas (S1 et S2), nous pouvons incrémenter et décrémenter le paramètre choisi actuellement. Lors du premier appui sur le bouton S3, nous configurons les minutes. Lorsque nous appuyons une seconde fois, nous configurons les heures. Le troisième appui nous fait rentrer dans le mode sablier.



Lorsque l'écoulement finit, les matrices à leds commencent à clignoter avec une fréquence de 0.5 Hz. Le buzzer commence à sonner en même temps. Cet état d'alarme dure 15 secondes. Une fois l'alarme finie, nous retournons dans l'état de configuration des minutes. Le temps fixé au préalable reste affiché dans la matrice et nous pouvons relancer le temps en appuyant deux fois sur S3.

6 BILAN

6.1 COÛTS ET LISTE DES PIÈCES

Annexe.

Nous avons un coût total de 108 franc pour toutes les pièces de ce projet.

6.2 LISTE DE MATÉRIEL

P1 - Oscilloscope - RTB2004 2.5GSa/s - ES.SLO2.05.01.10

2 Cartes matrice à leds 16x16 - 2008_2_A

1 Carte commande matrice - 2008_1_A

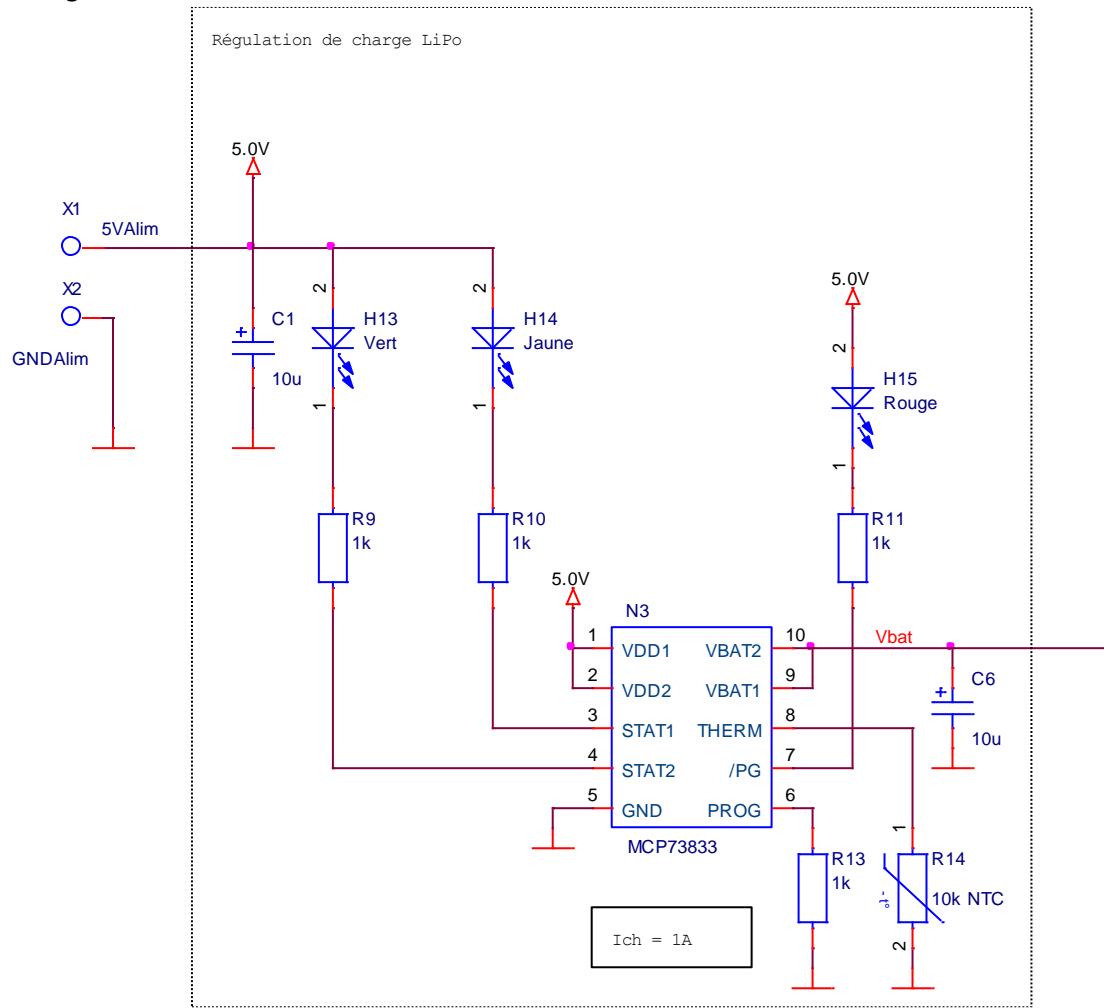
6.3 AMÉLIORATIONS

6.3.1 Software

Il serait possible de créer une fonction qui contrôlerait le sens indiqué pendant le fonctionnement, en effet, nous pourrions avoir une meilleure interaction des particules si l'on échange le sens perçu selon un pourcentage du sens actuel. Par exemple, si la valeur de l'axe Y est actuellement de 10000 et celui de l'axe X de 6000, en sachant que nous avons 1g à une valeur d'environ 16000, indiquer que notre sablier est en position haut pendant 10/16 du temps. Ainsi, les particules devraient respecter plus le sens actuel de l'affichage. Nous pouvons, actuellement, faire une interaction sur les 4 sens de fonctionnement de notre sablier sans devoir reset les particules présentes.

6.3.2 Hardware

Chargeur de batterie



Au départ du projet, j'avais prévu d'ajouter un chargeur d'accu dans le circuit. En effet, il serait préférable que ce sablier tourne sur accu. Le produit final serait beaucoup plus réussi que maintenant.

Il faudrait modifier le schéma afin d'implémenter les modifications hardware que j'ai réalisées. Enlever les commandes pour LB1 à LB4 et mettre ensemble les commandes du multiplexeur.

Il faudrait modifier le schéma et le PCB pour ajouter les résistances de Pull-Down que j'ai ajouté dans les modifications.

6.4 CONCLUSION

Dans ce projet, j'ai pu m'aventurer dans la conception d'un sablier électronique. L'algorithme de ce circuit a été réfléchi par blocs. En effet, il est plus simple de réaliser des blocs qui réalisent des actions spécifiques. 75% du cahier des charges est atteint, en effet, je n'ai pas réussi à implémenter l'interface USB. Le reste des fonctionnalités demandées dans le cahier des charges sont implémentées. Dans ce projet, le fait de devoir concevoir le boîtier 3D et la matrice à leds m'a pris beaucoup de temps. Le montage de la matrice à leds a été réalisé avec une placeuse manuelle (axe Y verrouillable). Le projet reste quand même utilisable sans l'interface USB.

Lausanne le 15 Septembre 2020
Carlos Antonio Reyes Peña