Rapport de laboratoire

Ecole supérieure
Électronique

Projet de diplôme ES
Salle R110

2003 – Capteur de mouvements basé sur CCD

Technicien :

Erwann Chancerel

Responsable :

M. Serge Castoldi

Experts :

M. Philippe Déglon
M. Flavien Baumgartner

Mandataire :

M. Philippe Déglon (PHYD Electronics)

Dates :

10 août 2020 – 14 septembre 2020

Table des matières :

1 Introduction

Dans le cadre de la formation de Technicien en Génie Electrique à l'ETML-ES, les étudiants sont amenés à travailler sur un projet à plein temps sur une durée de 5 semaines. La réalisation de ce projet vise à déterminer l'aptitude de l'étudiant à être autonome sur un projet. Ce travail a également pour but de valider les acquis de l'étudiant sur sa capacité à réaliser un circuit imprimé, programmer un microcontrôleur et procéder à une mise en service.

Le projet qui m'a été assigné a le numéro 2003 dans la base de données de l'ETML-ES et est sous la dénomination Capteur de mouvements basé sur CCD (CapteurMvtCcd).

Pourquoi ? Suite à mon projet de semestre visant à réaliser une commande de moteur de précision pour une application horlogère, il fallait trouver un moyen de déterminer si la rotation du moteur est linéaire. C'est la raison pour laquelle M. Philippe Déglon de PHYD Electronics a mandaté l'ETML-ES à la suite du projet de semestre pour réaliser ce système.

Le principe proposé pour le système est le suivant : utiliser un capteur CCD (Charge-coupled Device) linéaire afin de détecter les transitions ombre-lumière sur un disque à fente couplé mécaniquement sur l'axe du moteur dont on souhaite connaitre la position/vitesse.

PHYD Electronics fournit le système d'éclairage Infra-rouges qui permet d'éclairer les fentes de l'obturateur couplé au moteur et générer ainsi les transitions ombre-lumière sur le capteur CCD. Ce système est relié au circuit développé via un bus de communication sérielle (afin de pouvoir ajuster différents paramètres au besoin).

Le travail réalisé est défendu dans le cadre d'une présentation orale qui se déroule entre le 5 et le 7 octobre 2020.

Cahier des charges

Voir le cahier des charges détaillé en annexe " 2003_CdC_CapteurMvtCcd_Phyd.pdf"

Ressources utilisées

Software

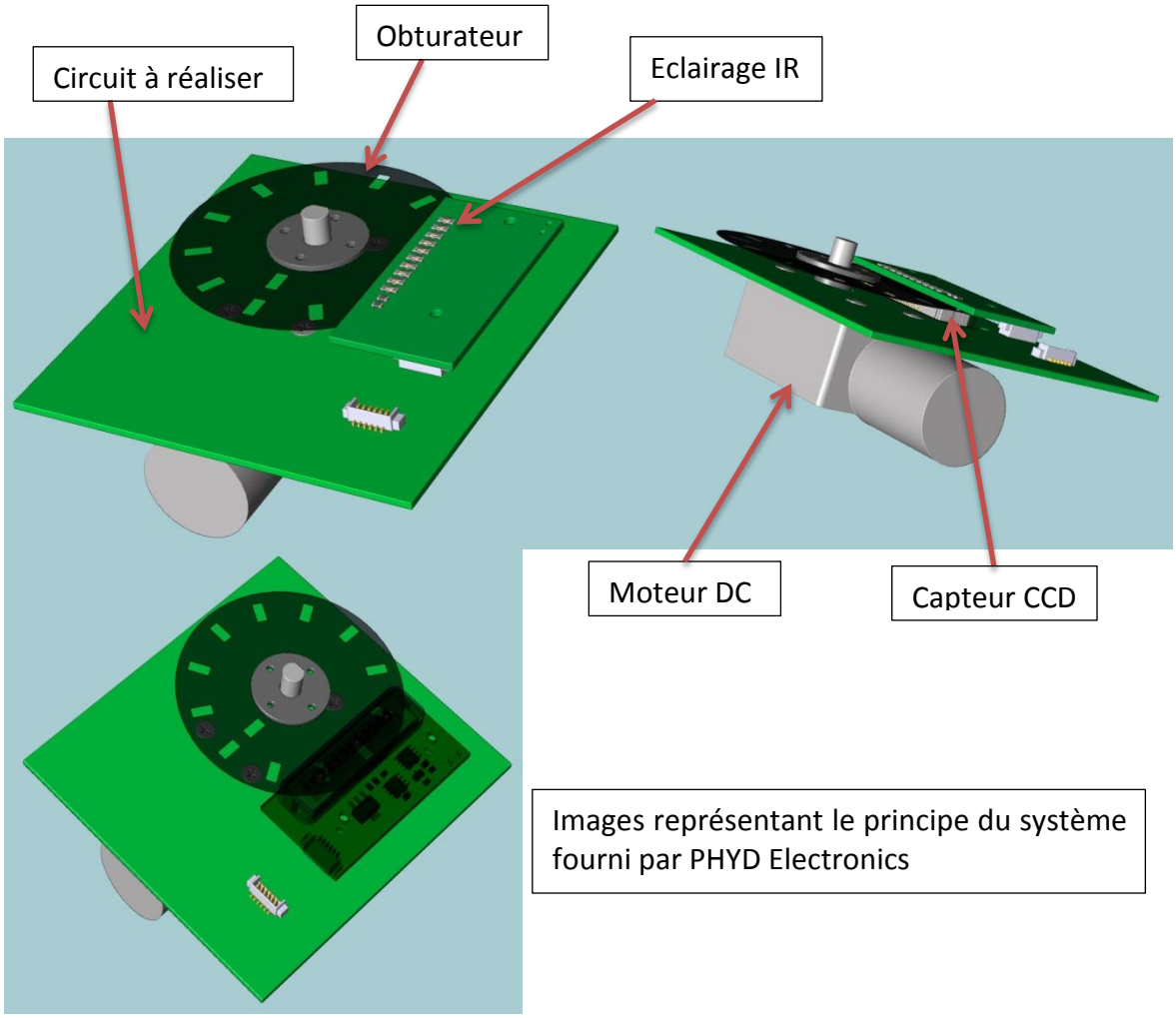
- Altium Designer 19
- MPLAB X IDE v5.15 avec le framework Harmony v2.06
- PuTTY

Hardware

- Oscilloscope Rhode & Schwartz RTB2004 4x25GSa/s
- Alimentation de laboratoire Gw INSTEK GPS-3303

2 Pré-étude du projet

Représentation du système



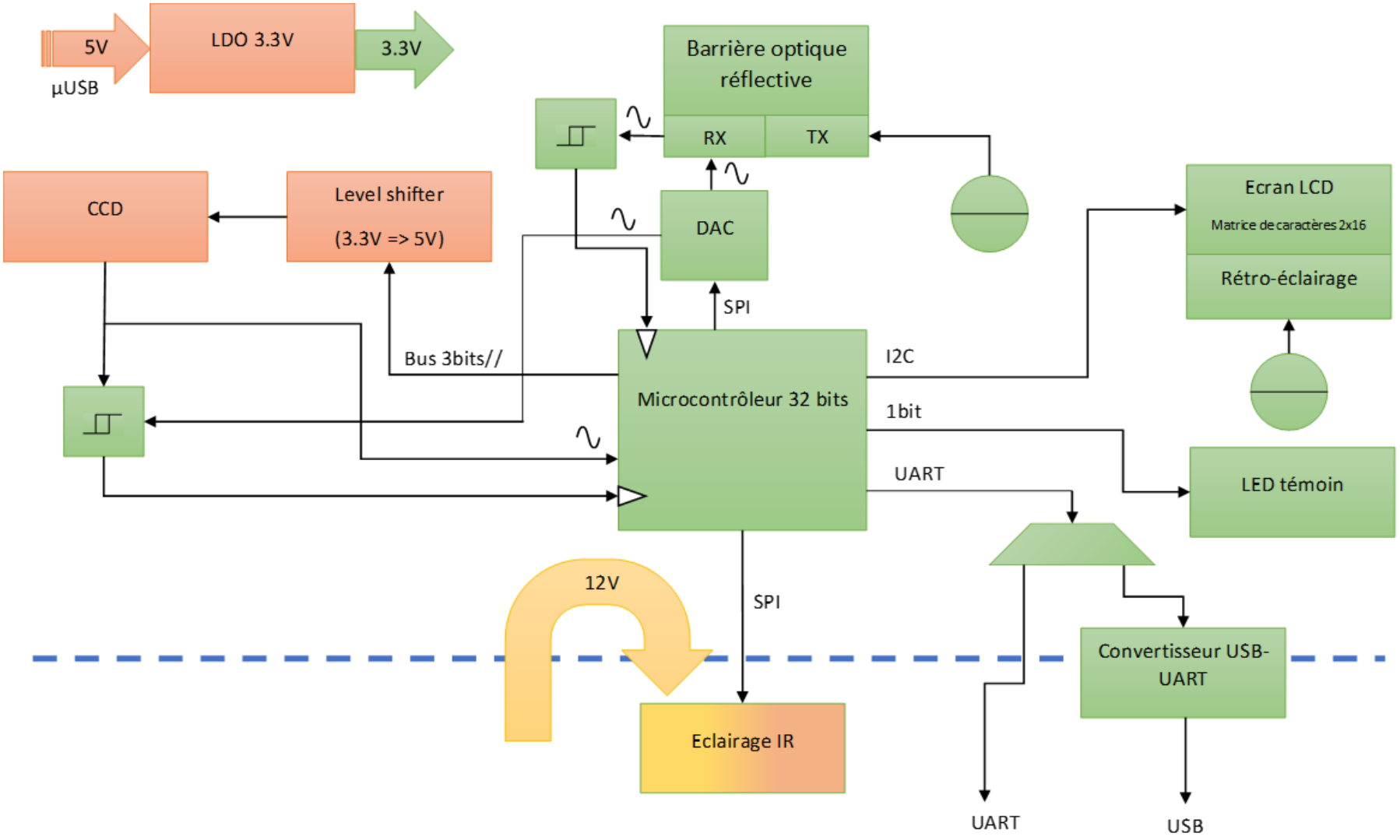
Images représentant le principe du système fourni par PHYD Electronics

Le principe est le suivant : l'axe du moteur, couplé mécaniquement à l'obturateur, tourne. L'obturateur, consiste en un disque à fentes découpés de manière précise. Le module d'éclairage Infra-Rouges éclaire le capteur CCD sur toute sa longueur. L'obturateur a pour rôle de cacher dynamiquement cet éclairage afin de pouvoir faire des détections de transitions ombre-lumière. En outre, son objectif est de créer des flancs sur l'amplitude des signaux mesurés par chaque pixel du CCD.

Au niveau software, nous avons donc la possibilité de mesurer l'évolution d'une fente dans son passage au-dessus du capteur CCD, ainsi que de mesurer la distance entre deux fentes. Connaissant, la taille des pixels, nous pouvons déterminer la position angulaire de l'obturateur et par conséquent de l'axe du moteur. Connaissant la position, nous pouvons déterminer la vitesse angulaire en dérivant cette valeur par rapport au temps.

Ainsi, en récoltant les données acquises sur une révolution du moteur, nous pouvons déterminer s'il y a des variations de vitesse.

Schéma bloc du circuit



Choix des composants principaux

Capteur CCD

Demandé dans le cahier des charges, le **TCD1304** est un capteur d'image linéaire de 3648 pixels. Interfacé et cadencé par le µC, il permet d'obtenir les informations de transitions ombre-lumière de l'obturateur.

Microcontrôleur

Après échange avec M. Déglon, il recommande un microcontrôleur avec un CPU pouvant tourner à au moins 80MHz pour avoir une vitesse d'acquisition et de traitements des données suffisamment rapide.

Suite à cela, le µC **PIC32MX370F512H** a été choisi car il peut tourner à une fréquence de maximum 120MHz. Bien qu'il n'existe pas de version 32 ou 48 pattes (64 ou 100 pattes uniquement), la priorité a été mise sur la puissance de traitement et de calcul plutôt que sur la surface de l'empreinte du composant.

Interface USB-UART

Demandé par M. Déglon de PHYD Electronics, le **CP2102N** est utilisé pour avoir une interface USB série. Ceci afin de pouvoir faire communiquer le système avec un PC sur un port série avec un câble USB.

Régulateur 3.3V

Une tension de 3.3V étant nécessaire afin d'alimenter la partie logique du circuit. Il est nécessaire d'intégrer un régulateur entre le 3.3V et le 5V de l'USB. La puissance en jeu étant relativement faible, il est possible d'utiliser un régulateur linéaire puisque les pertes seraient minimes.

Le **MCP1700T** est un LDO de Microchip en boitier SOT89 qui permet d'obtenir du 3.3V en sortie et de débiter un courant de max 250mA sans transistor Ballast.

Ce composant a été utilisé dans le projet n°1921 PilotagePrecisionMoteurLin. Sa fiabilité a donc été testée et validé.

Voir la section "Estimation de la consommation du courant" plus loin dans ce document pour plus de détails.

Ecran LCD

Afin d'afficher les différentes valeurs mesurées, l'ajout d'un écran LCD a été demandé.

Le **NHD-C0216CiZ-FSW-FBW-3V3** a été choisi en raison de sa faible empreinte et de son interface I2C (faible empreinte de layout => meilleure niveau d'intégration). Il s'agit d'un écran LCD à matrice de caractère 2x16 avec rétro-éclairage intégré.

Convertisseur numérique-analogique externe

Afin de pouvoir ajuster les seuils de basculement des trigger de schmitt pour le capteur CCD et la barrière réflective, un DAC, interfacé par le µC est implémenté sur le circuit.

Le **LTC2624**, la version 12bits du LTC2604 qui est elle-même déjà utilisée sur des projets existants de l'ETML-ES (dont le kit PIC32) a été choisie. Il est interfacé via un bus SPI.

Estimation de la consommation du courant

Sur l'alimentation 3.3V

Main components	Max current [mA]
Microcontrôleur	84
Convertisseur USB-UART	15
Ecran LCD	0.5
DAC externe	2
Total	**101.5**

En arrondissant avec une marge de sécurité suffisante et en tenant compte des autres composants actifs négligés. Un courant de **120mA** peut être considéré.

Sur l'alimentation 5V

Main components	Max current [mA]
Capteur CCD	15[1]
Rétro-éclairage LCD	18
Led capteur réflectif	18
Total	**51**

En additionnant le courant directement soutiré du 5V et celui prit sur le 3.3V, un courant d'environ **171mA** peut être considéré sur le 5V de l'USB.

Sur l'alimentation 12V

La seule partie alimentée en 12V est le circuit d'éclairage IR PHYD, qui ne devrait pas dépasser 20mA.

1 Calculé à partir d'une puissance dissipée max de 75mW et d'une alimentation de maximum 5.25V

Estimation des coûts

Coût du PCB

La dimension estimée du PCB est de 100x100mm. Ceci afin qu'il puisse entrer dans le rail d'un boitier pour circuit formats Europe.

En entrant cette information dans le calculateur de prix Eurocircuit, nous obtenons le prix suivant :

Price summary

Service	PCB proto	
Delivery term	3 working days	
Estimated shipment date	27-08-2020	
Quantity	1 PCBs	
Board surface / Order surface	1.00 dm² / 1.00 dm²	
Prices	Net	Gross*
Single PCB	€ 48.93	€ 59.20
Total boards	€ 48.93	€ 59.20
Express transport	€ 0.00	€ 0.00
Total	€ 48.93	€ 59.20

* The gross prices include 21.0% VAT.

Nous avons donc un prix total du PCB de 59.20 Euros (63,67CHF avec le cours actuel).

Coût du montage

Most expensive components	Price [CHF]
PCB	63.67
µC	5.45
DAC	13.53
Capteur CCD	22.67
Ecran LCD	10.92
Total	**116.24**

En tenons compte des composants principaux les plus chers, nous obtenons un coût total du circuit de 116.24CHF. Arrondissons à **150CHF** pour absorber le coût des composants qui ne sont pas pris en compte, des charges et des consommables.

Risques et faisabilité

Ce projet reste du domaine de la recherche, il sert principalement à effectuer des tests et des essais. Des réserves sont émises quant aux futurs résultats obtenus. Notamment concernant le problème de parallaxe[2].

2 Parallaxe : changement d'incidence d'observation (Déplacement de la position apparente dû au changement de position de l'observateur ou de l'objet mesuré).

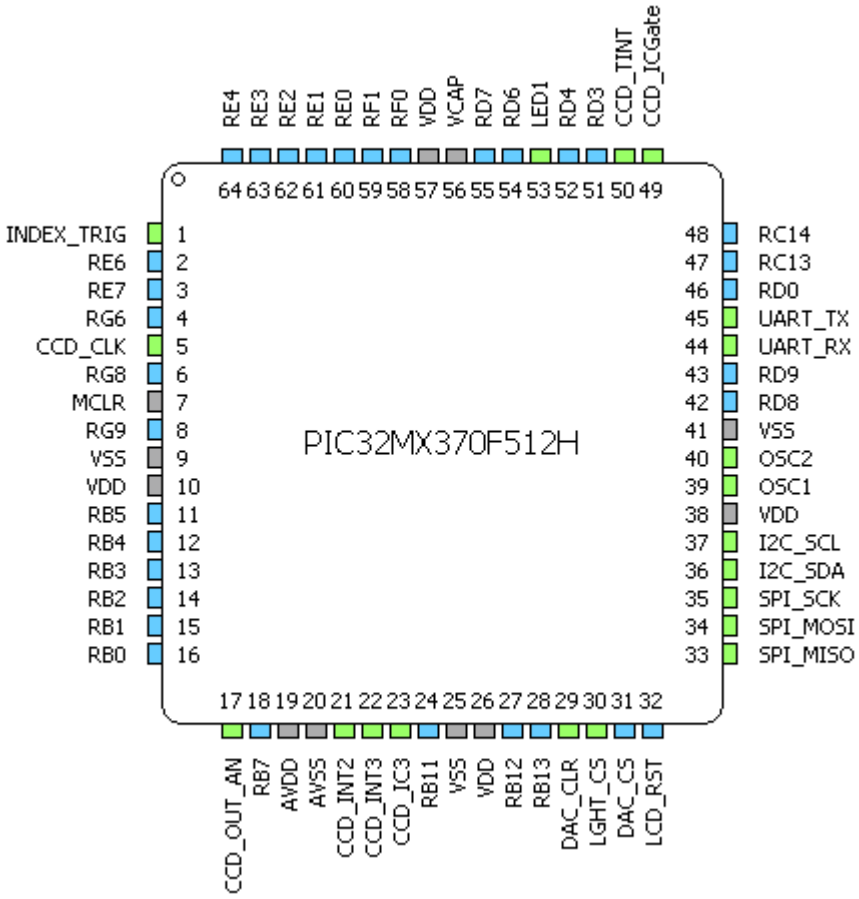
3 Design

Microcontrôleur

Pinout

Comme mentionné dans le chapitre pré-étude, le µC choisi est le PIC32MX370F512H.

En listant les interactions et besoins du circuit avec le µC, le diagramme du pinout peut être établi :

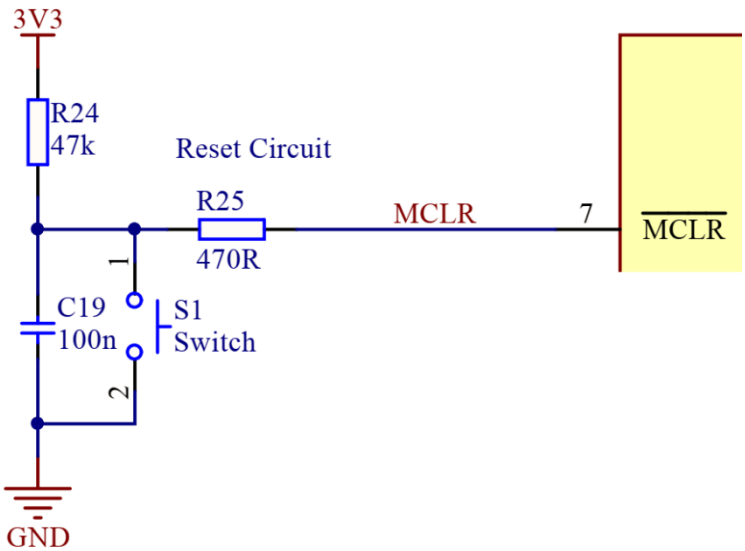


Circuit de reset

Le circuit de reset a été repris des anciens projets de l'ES, dont celui utilisé dans le projet n°1921 de la base de données de l'ETML-ES.

Il consiste en un basique circuit RC permettant de maintenir la patte reset du µC pendant la durée recommandée par le fabricant.

Un bouton poussoir a été ajouté afin de pouvoir aider pendant la phase de code/debug.



Quartz

Après des essais avec le configurateur de clock du framework Harmony et discussion avec M. Castoldi, il a été déduit qu'un quartz 10MHz était nécessaire afin d'exploiter la fréquence maximale du µC (en multipliant avec la PLL interne).

Le quartz ABLS-10.000MHZ-B2-T de chez Abracon a été choisi pour remplir cette tâche. La donnée du fabricant nous recommande des capacités de charge de 18pF.

Or le datasheet du µC nous recommande de revoir ces valeurs de la manière suivante :

EXAMPLE 2-1: CRYSTAL LOAD CAPACITOR CALCULATION

Crystal manufacturer recommended: $C1 = C2 = 15\ pF$ 18pF

Therefore:

$C_{LOAD} = \{([C_{IN} + C1] * [C_{OUT} + C2]) / [C_{IN} + C1 + C2 + C_{OUT}]\}$
$+$ *estimated oscillator PCB stray capacitance*

$= \{([5 + 15][5 + 15]) / [5 + 15 + 15 + 5]\} + 2.5\ pF$

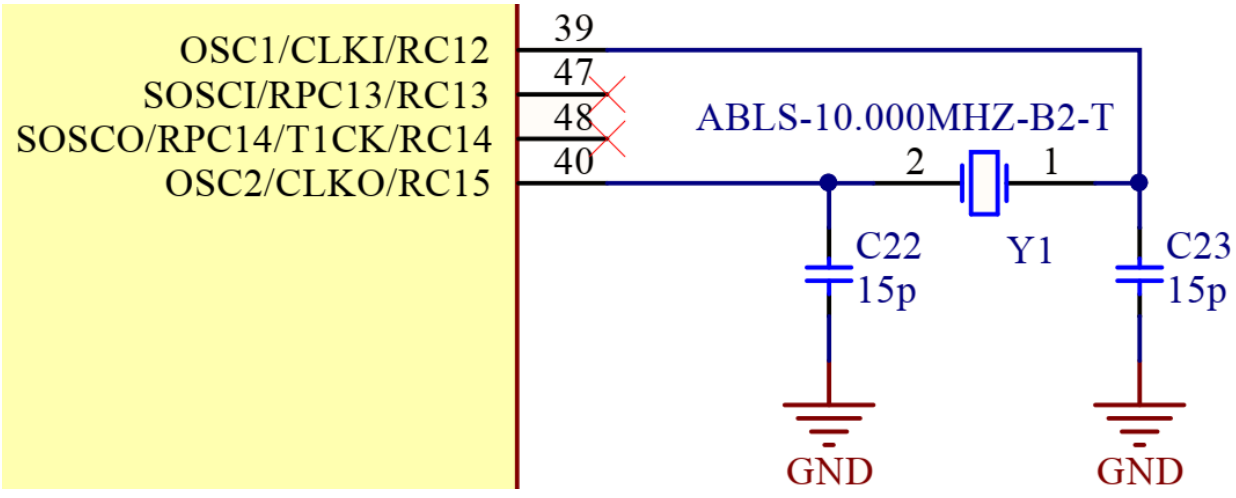
$= \{([20][20]) / [40]\} + 2.5$

$= 10 + 2.5 = 12.5\ pF$ 14pF => **15pF**

Rounded to the nearest standard value or 13 pF in this example for Primary Oscillator crystals "C1" and "C2".

En outre, reprendre la valeur conseillée (18pF) et la repasser dans la formule indiquée. En obtenant des capas de 14pF, j'ai arrondi à 15pF pour être dans la série E6.

Microchip, le fabricant du µC, nous recommande de baisser cette valeur pour tenir compte des capacités d'entrée et de sortie (CIN = 5pF et COUT = 2.5pF) des pattes du quartz dans la charge capacitive totale du quartz.

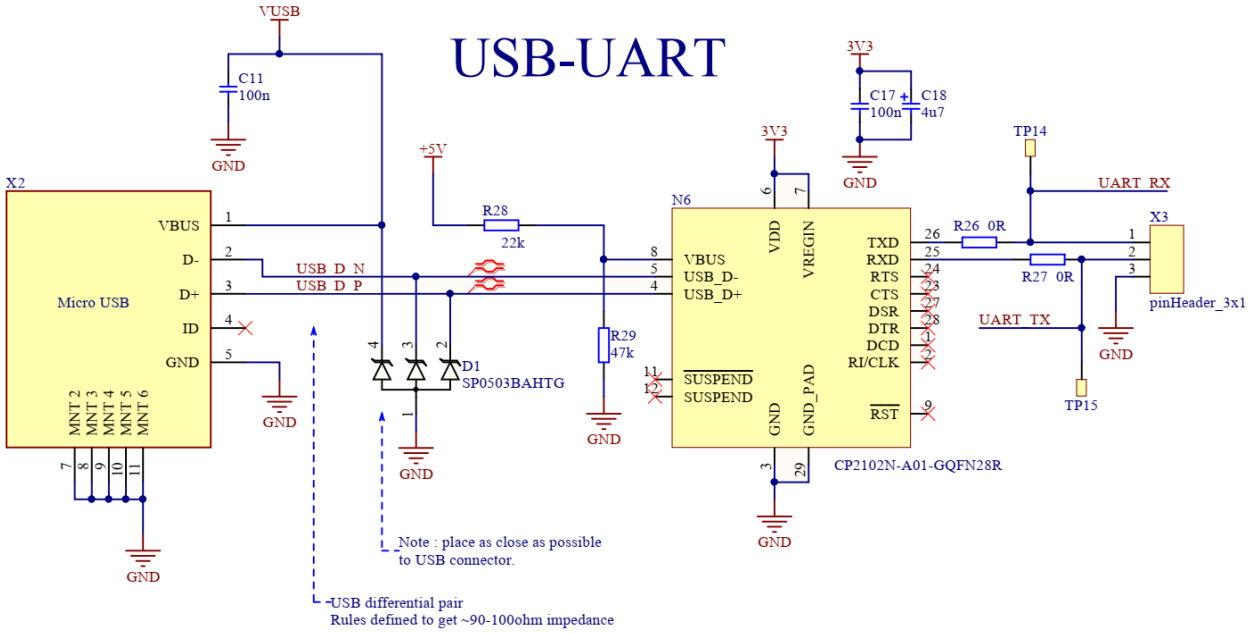


Interface USB

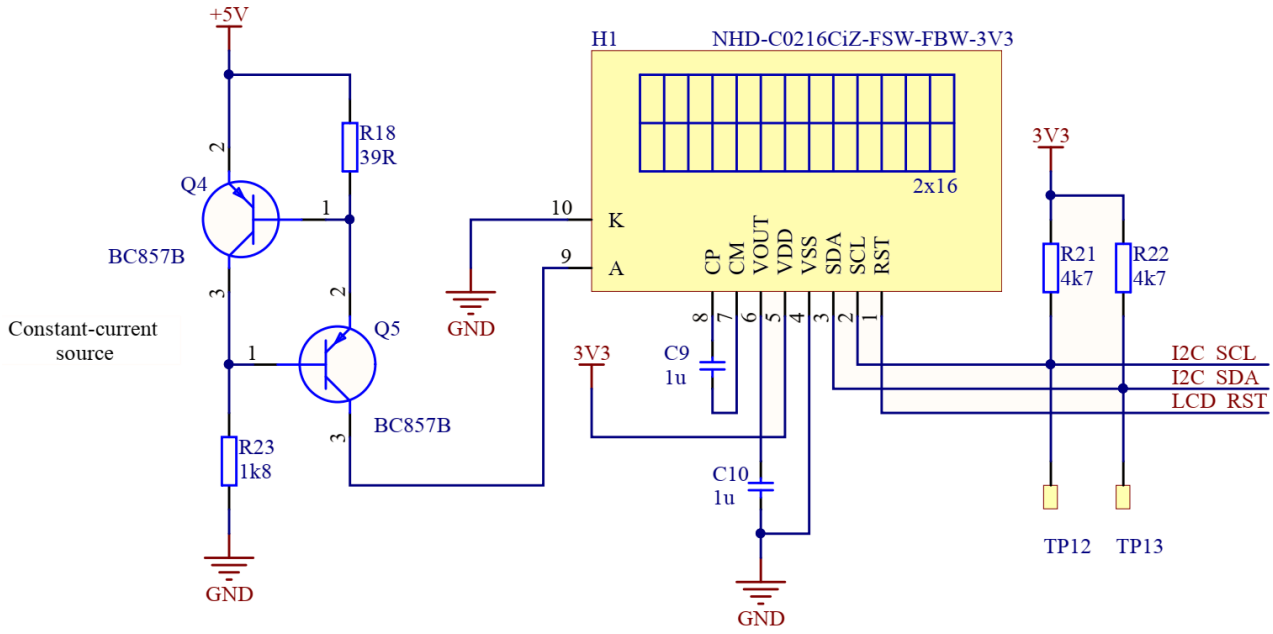
Schéma repris du projet n°1921. Adapté pour mettre un connecteur µUSB à la place du connecteur type B femelle.

Permet d'avoir une interface série virtuelle vue depuis un PC.

Les résistances 0Ω R26 et R27 ainsi que le connecteur X3 ont été ajoutés afin de pouvoir dévier par la suite le bus UART vers l'extérieur. Notamment afin de communiquer avec le circuit de commande du projet n°1921.



Ecran LCD



Le schéma de l'écran a été repris des recommandations du fabricant. Les résistances Pull-Ups du bus I2C ont été choisies les plus faibles possible afin de pouvoir communiquer à vitesse élevée.

Le rétro-éclairage de l'écran fonctionne avec un courant type de 20mA et une tension directe de 3.12V. Une source ce courant a été implémenté afin de piloter la led de backlight.

La jonction émetteur-base de Q4 sert de référence de tension pour la source de courant, environ 0.7V. Avec un courant cible de 20mA, la résistance R18 est calculée comme suit :

$$R18 = \frac{U_{REF}}{I_{CIBLE}} = \frac{0.7}{0.02} = 35\Omega => \mathbf{39\Omega}$$

En arrondissant la valeur vers le haut pour être dans la série de résistance E12, nous sacrifions simplement légèrement l'intensité lumineuse du rétro-éclairage tout en augmentant sa durée de vie.

Contrôle de la puissance dissipée par la résistance R18 :

$$P_{D_{R18}} = \frac{{U_{REF}}^2}{R18} = \frac{0.7^2}{39} = \mathbf{13mW}$$

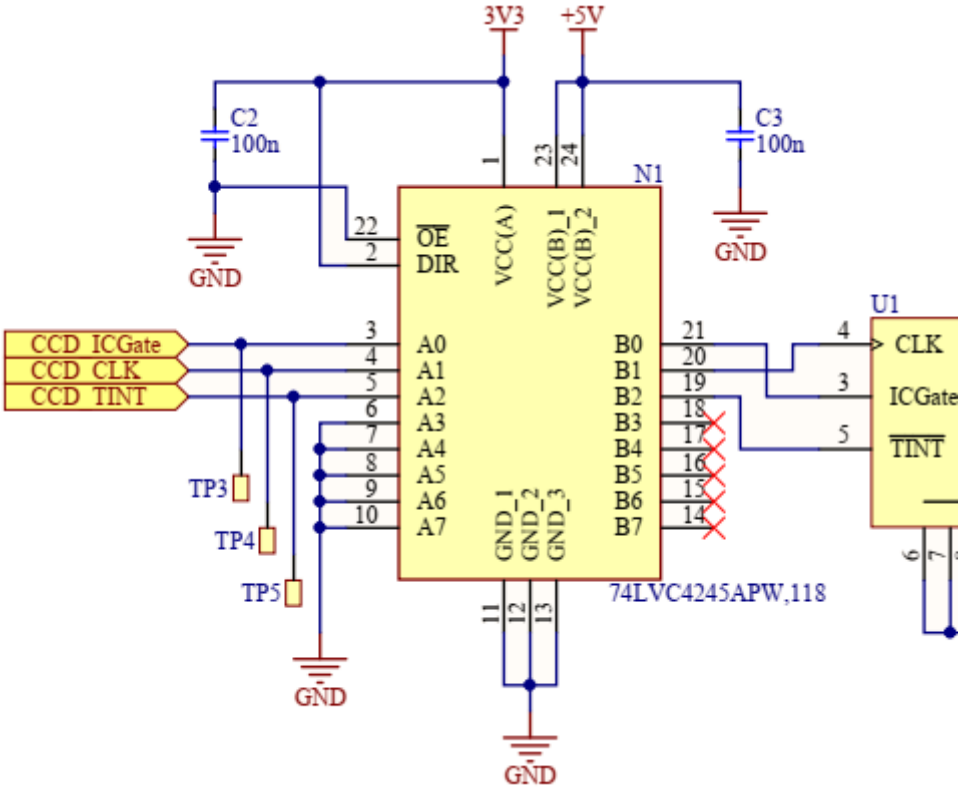
=> Puissance dissipée max pour boitier 0805 => 135mW donc OK.

Capteur CCD

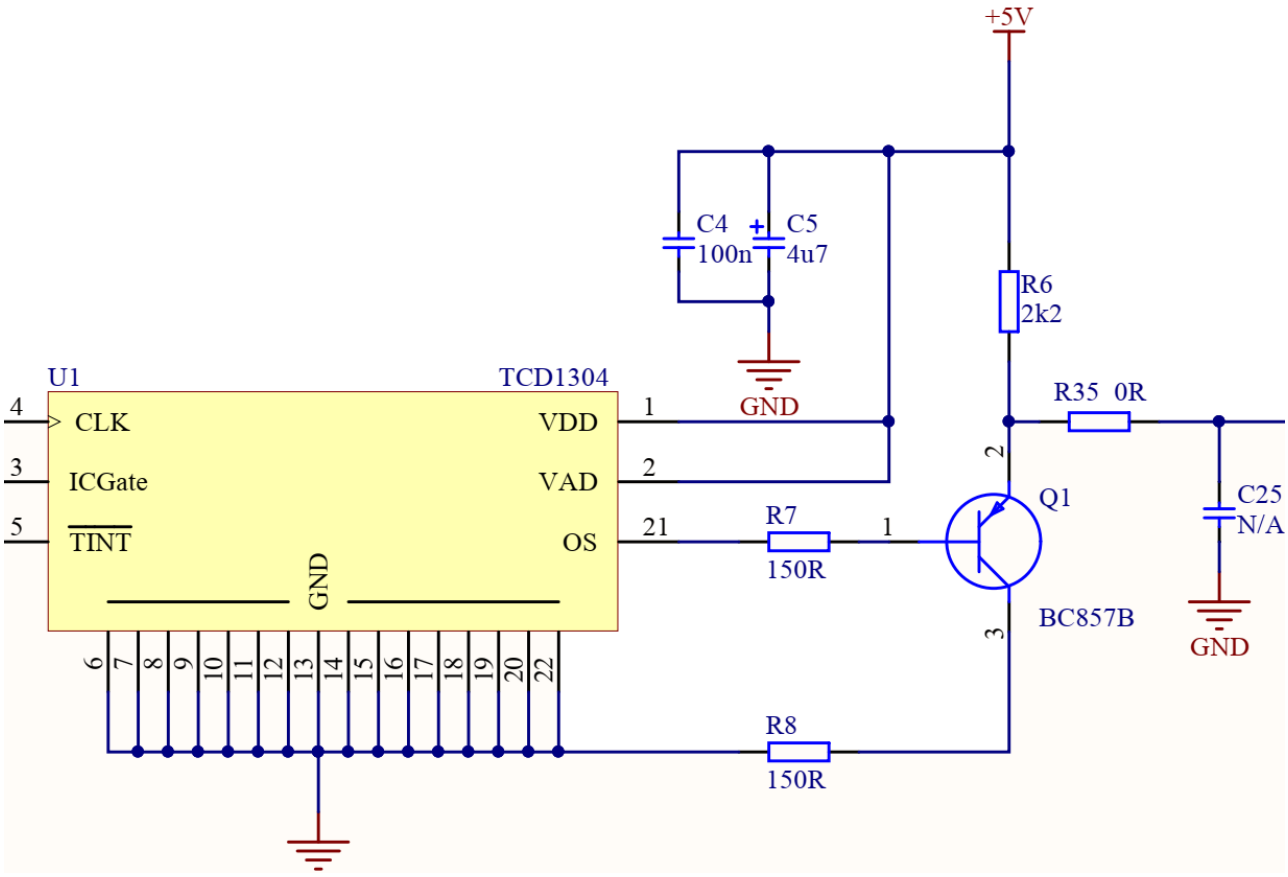
Level-shifter

L'interface logique du CCD est en 5V et peut détecter un état logique haut jusqu'à 3V minimum. Or les GPIOs du microcontrôleur sont à 3.3V mais peuvent descendre jusqu'à 2.4V. Il faut donc prévoir un level-shifter pour faire correspondre les niveaux logiques avec sécurité.

Le 74LVC4245APW remplit cette fonction Voici le schéma qui en découle :



Signal de sortie



La sortie OS fournissant une information analogique codé en courant, le fabricant recommande l'utilisation d'un montage à collecteur commun à base de PNP faisant office d'amplificateur convertisseur courant-tension. Le signal est récupéré sur l'émetteur du transistor.

Ce montage ayant déjà été dimensionné par PHYD Electronics, il a été repris tel quel. Avec le même transistor qui est en stock à l'ES et réutilisé pour les sources de courant.

L'empreinte d'un filtre RC passif passe-bas a été prévue s'il s'avère qu'on ait besoin de filtrer des fréquences en sortie du CCD. Il est pour le moment composé d'une résistance 0Ω et d'un condensateur non-présent.

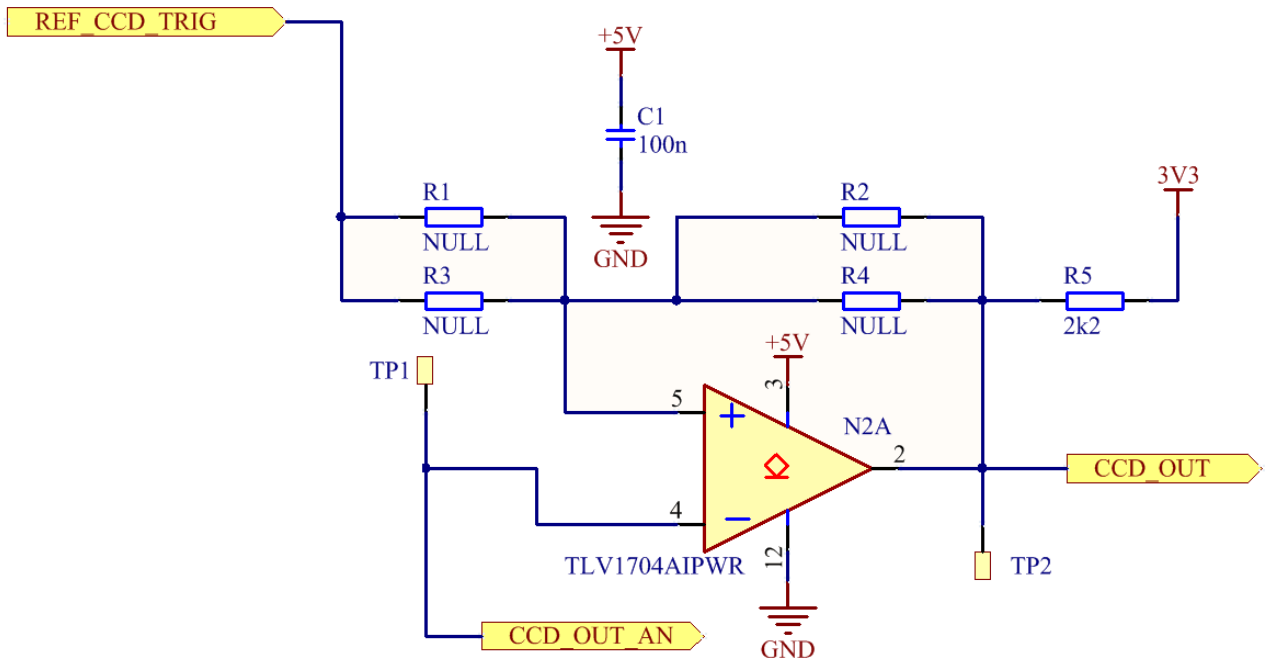
Cette sortie analogique attaque ensuite le trigger de schmitt pour la détection de flancs ainsi qu'une entrée analogique du µC si l'on souhaite effectuer à la place une détection de flancs par traitement numérique.

Trigger de Schmitt réglable

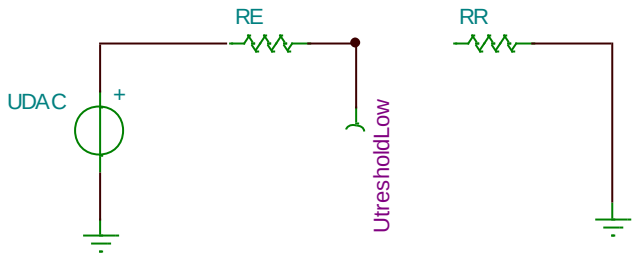
Principe

Le but est de pouvoir générer un flanc numérique (CCD_OUT) avec un seuil de transition souhaité en fonction des flancs du signal analogique généré par le capteur CCD (CCD_OUT_AN).

En l'occurrence, un comparateur à hystérèse peut remplir cette fonction. Le but ici est d'avoir une hystérèse fixe mais de pouvoir ajuster le seuil de basculement à l'aide d'un DAC externe (REF_CCD_TRIG).



Quand le transistor de sortie conduit, la sortie est mise à la masse. Le schéma équivalent du circuit est donc le suivant (R1, R3 = RE et R2, R4 = RR) :

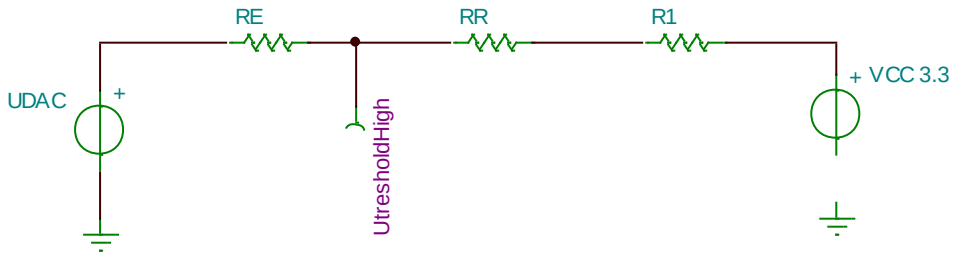


L'équation permettant de sortir le point de basculement inférieur de ce circuit est la suivante :

$$U_{tresh_{low}} = \frac{R_R}{R_R + R_E} U_{DAC}$$

En d'autres termes, la tension aux bornes de RR définit le point de basculement inférieur du comparateur à hystérèse.

En revanche, quand le transistor de sortie du comparateur est en état bloqué, la pull-up de sortie entre dans l'équation, le schéma équivalent devient comme suit :



L'équation déterminant le point de basculement supérieur est donc la suivante :

$$U_{tresh_{high}} = \frac{R_E}{R_E + R_R + R_1}(V_{CC} - U_{DAC}) + U_{DAC}$$

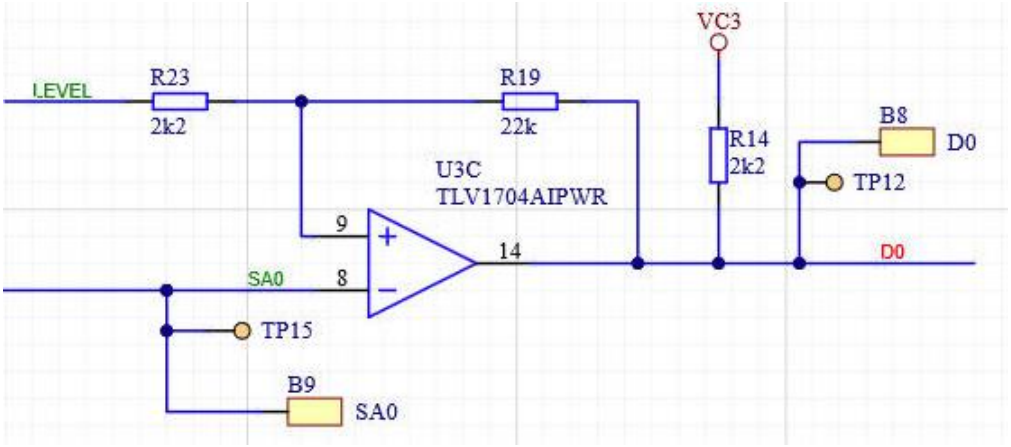
En d'autres termes, nous pouvons dire que la tension de basculement supérieur correspond à la tension aux bornes de RE plus la tension fournie par le DAC.

En se basant sur ces équations, la tension d'hystérèse est déterminée comme suit :

$$U_{hyst} = U_{tresh_{high}} - U_{tresh_{low}}$$

Dimensionnement

Comme il est compliqué de pouvoir déterminer le bon jeu de résistance sans avoir fait des mesures des signaux fournis par le CCD au préalable, les valeurs ont dans un premier temps été reprises d'un schéma fourni par M. Déglon de PHYD Electronics. Voici l'extrait du schéma qu'il a mis à disposition :



A noter que le comparateur TLV1704 a également été repris afin de s'assurer d'obtenir des résultats identiques.

Sur le schéma de la carte, la valeur de R1 sera donc de 2k2 et R2 de 22k. R3 et R4 resteront des empreintes non-montés.

En reprenant les équations déterminées dans la section "principe" de ce chapitre, nous obtenons les valeurs de basculements et d'hystérèses suivantes (par exemple pour U_{DAC} = 1V, 2V et 3V) :

Avec U_{DAC} = 1V

$$U_{tresh_{high}} = \frac{2.2 \times 10^3}{(2.2 + 22 + 2.2) \times 10^3}(3.3 - 1) + 1 = \mathbf{1.19V}$$

$$U_{tresh_{low}} = \frac{22 \times 10^3}{(2.2 + 22) \times 10^3}1 = \mathbf{0.909V}$$

$$U_{hyst} = 1.19 - 0.909 = \mathbf{0.281V}$$

Avec U_{DAC} = 2V

$$U_{tresh_{high}} = \frac{2.2 \times 10^3}{(2.2 + 22 + 2.2) \times 10^3}(3.3 - 2) + 2 = \mathbf{2.11V}$$

$$U_{tresh_{low}} = \frac{22 \times 10^3}{(2.2 + 22) \times 10^3}2 = \mathbf{1.81V}$$

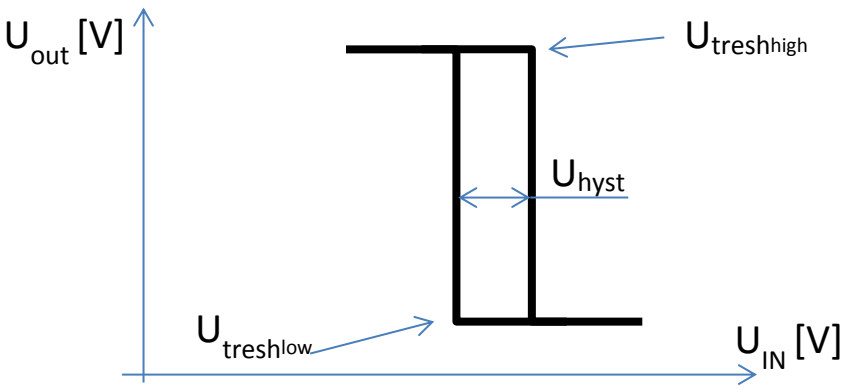
$$U_{hyst} = 2.11 - 1.81 = \mathbf{0.300V}$$

Avec U_{DAC} = 3V

$$U_{tresh_{high}} = \frac{2.2 \times 10^3}{(2.2 + 22 + 2.2) \times 10^3}(3.3 - 3) + 3 = \mathbf{3.02V}$$

$$U_{tresh_{low}} = \frac{22 \times 10^3}{(2.2 + 22) \times 10^3}3 = \mathbf{2.72V}$$

$$U_{hyst} = 3.02 - 2.72 = \mathbf{0.300V}$$



Simulation

Introduction

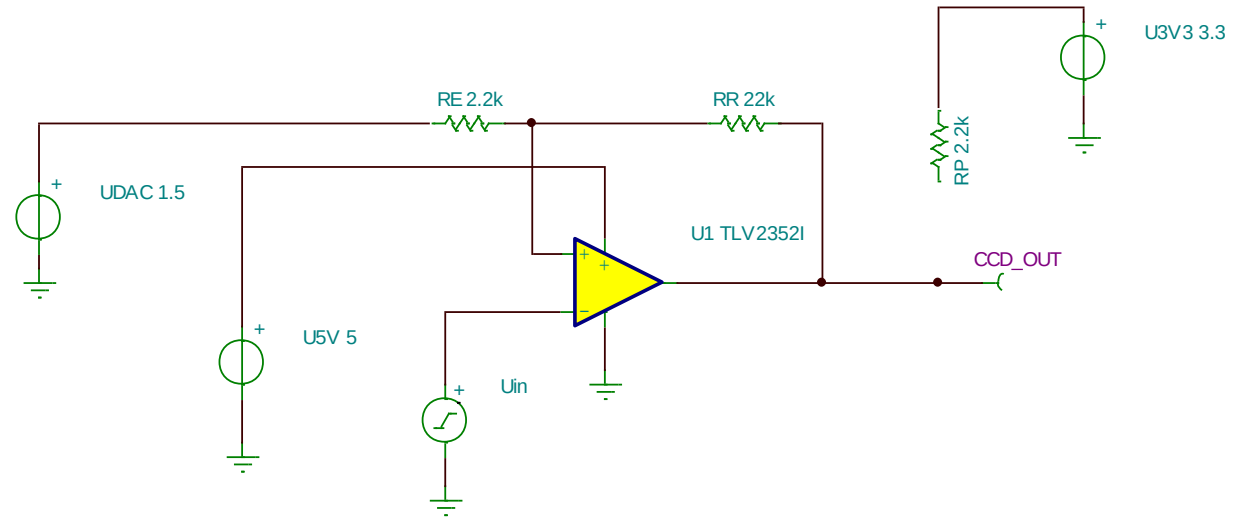
Afin de vérifier le bon fonctionnement du montage à Trigger de Schmitt pour la détection flancs dans les transitions ombre-lumière du capteur CCD, une simulation sous TINA a été réalisée.

Comme le comparateur TLV1704 (utilisé sur le circuit) n'est pas présent dans la base de données par défaut de TINA, il a été remplacé par un circuit équivalent : TLV2352I.

Un générateur de signal triangulaire remplace le signal d'entrée du CCD afin de pouvoir bien observer les points de basculement du comparateur.

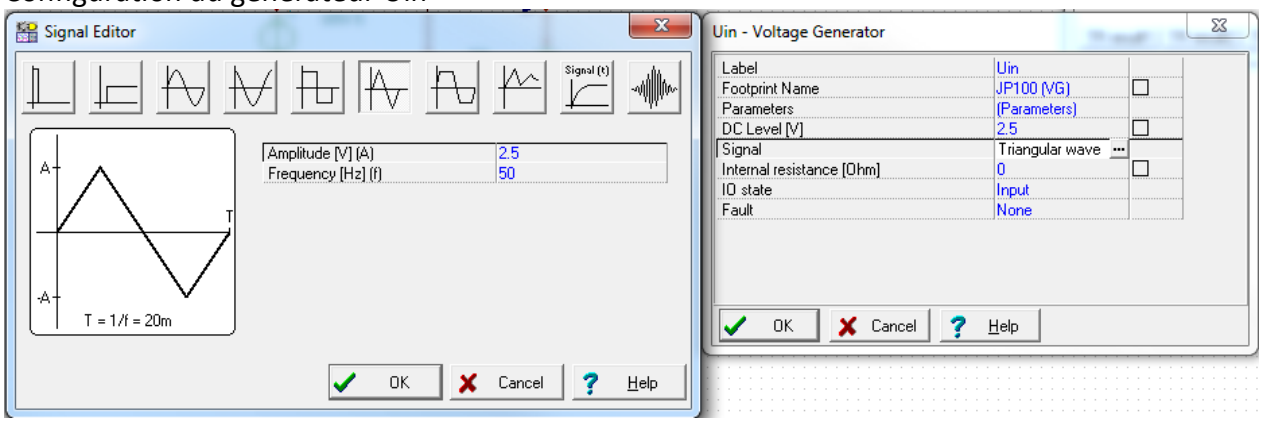
La référence du DAC est simulée par une source de tension fixe.

Schéma de simulation



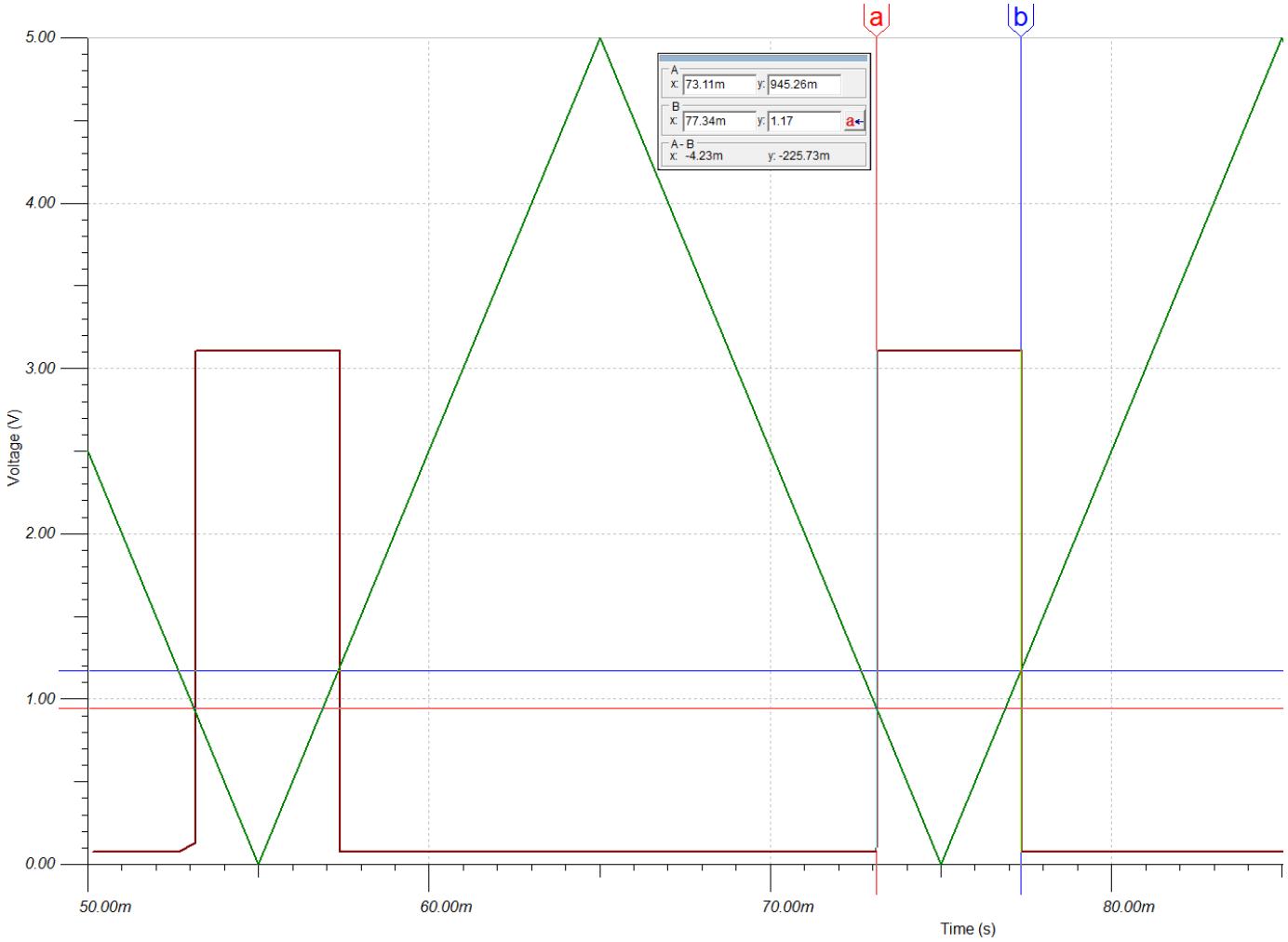
Conditions de simulation

Configuration du générateur Uin



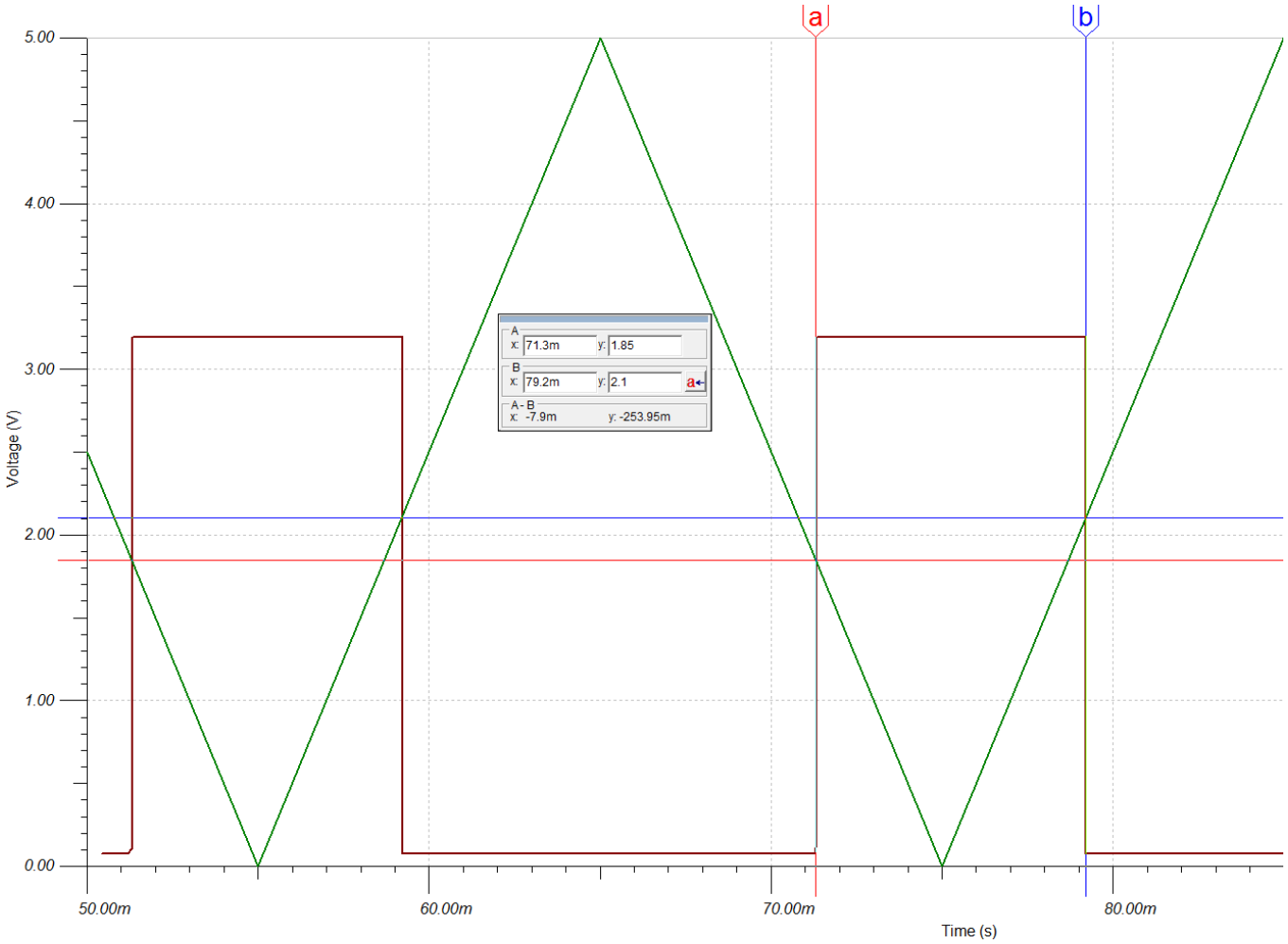
Résultats de simulation

Tension du DAC = 1V



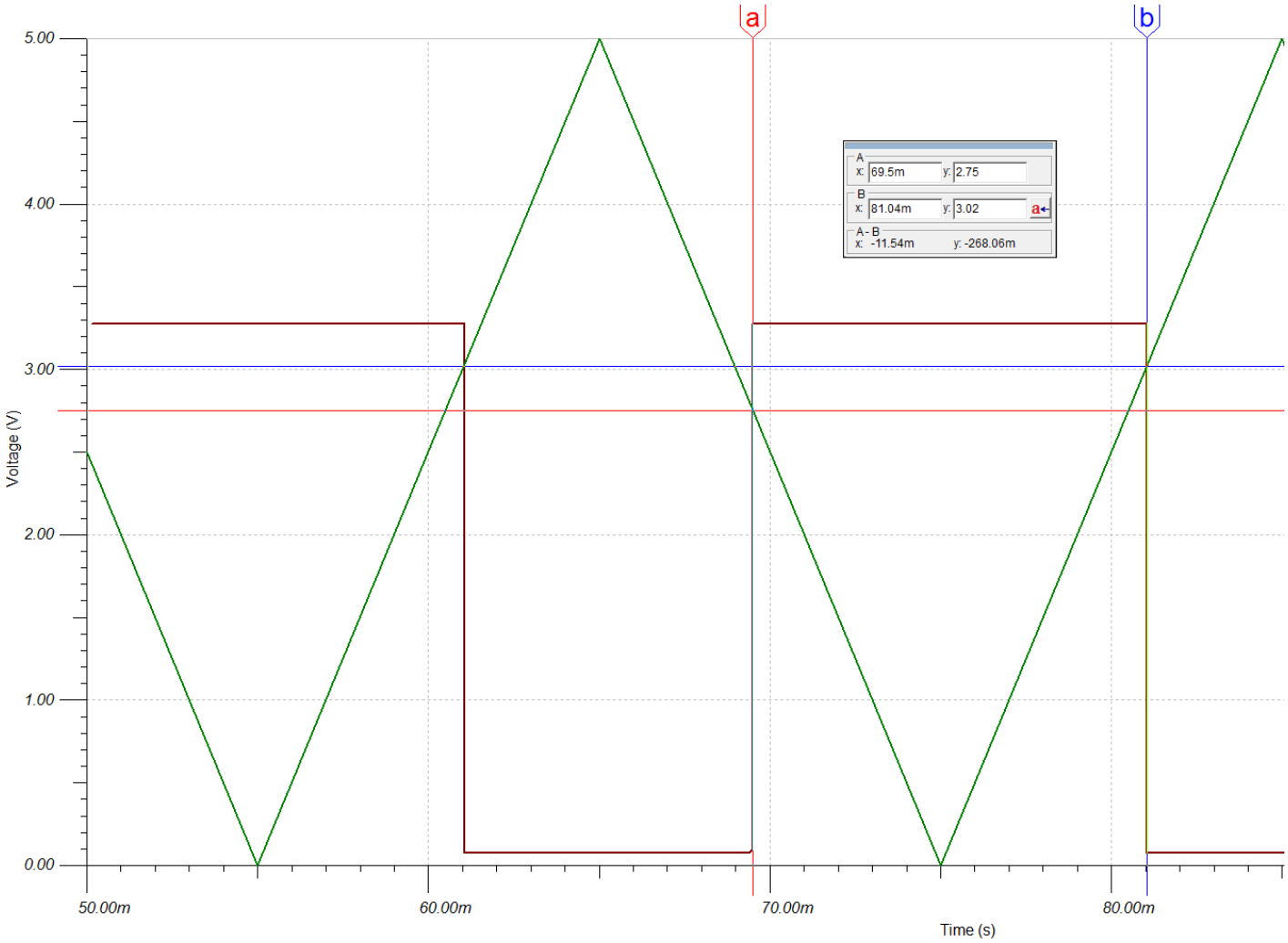
	Inférieur (a) [V]	Supérieur (b) [V]	Hystérèse (b - a) [V]
Point de basculement	0.945	1.17	0.225

Tension du DAC = 2V



	Inférieur (a) [V]	Supérieur (b) [V]	Hystérèse (b - a) [V]
Point de basculement	1.85	2.1	0.25

Tension du DAC = 3V



	Inférieur (a) [V]	Supérieur (b) [V]	Hystérèse (b - a) [V]
Point de basculement	2.75	3.02	0.268

Conclusion

En observant les résultats des simulations avec des tensions de 1V, 2V et 3V, nous pouvons voir que les points de basculements se décalent de tout autant. En faisant la comparaison avec les valeurs calculées, nous nous rendons compte que les résultats sont proches et par conséquent que le raisonnement initial est plausible.

En revanche la tension d'hystérèse reste quasiment fixe. C'est à dire qu'avec la tension du DAC externe, nous pouvons ajuster le point de basculement entre 0 et 3V en fonction du niveau des transitions mesurées souhaités par le CCD. Tout en gardant une hystérèse fixe sur les points de basculement.

En ayant une hystérèse suffisante, nous nous émancipons des basculements indésirables du comparateur provoqué par des sauts de tensions quelconques (bruits, glitchs…etc).

L'hystérèse n'est pas totalement fixe car la tension de référence (DAC) fait indirectement office de gain à travers le jeu de résistance de feed-back du comparateur. Le gap s'élargie donc proportionnellement à la tension de référence.

Alimentation

5V USB

L'alimentation principale du circuit provient du connecteur USB fournissant 5V.

Cette tension de 5V est découplée par un condensateur 100n à proximité du connecteur USB.

Elle est utilisée pour alimenter le régulateur 3.3V, le capteur CCD, les sources de courant, le phototransistor du capteur réflectif, le level shifter, les comparateurs ainsi que la partie numérique de la board d'éclairage IR PHYD externe.

3.3V régulateur linéaire

La partie exclusivement numérique de la carte est alimenté par le régulateur linéaire MCP1700T de Microchip. Il est en boitier SOT-89. Il peut débiter jusqu'à 250mA en sortie.

Déterminer la puissance dissipée par le composant :

TEMPERATURE SPECIFICATIONS

Electrical Characteristics: Unless otherwise specified, all limits are established for V_{IN} = V_R + 1V, I_{LOAD} = 100 µA, C_{OUT} = 1 µF (X7R), C_{IN} = 1 µF (X7R), T_A = +25°C.
Boldface type applies for junction temperatures, T_J (**Note 1**) of -40°C to +125°C.

Parameters	Sym.	Min.	Typ.	Max.	Units	Conditions
Temperature Ranges						
Specified Temperature Range	T_A	-40		+125	°C	
Operating Temperature Range	T_J	-40		+125	°C	
Storage Temperature Range	T_A	-65		+150	°C	
Thermal Package Resistance						
Thermal Resistance, 2x2 DFN	θ_{JA}	—	91	—	°C/W	EIA/JEDEC® JESD51-7 FR-4 4-Layer Board
	$\theta_{JC(Top)}$	—	286	—	°C/W	
	$\theta_{JC(Bottom)}$	—	28.57	—	°C/W	
	Ψ_{JT}	—	8.95	—	°C/W	
Thermal Resistance, SOT-23	θ_{JA}	—	212	—	°C/W	EIA/JEDEC JESD51-7 FR-4 4-Layer Board
	$\theta_{JC(Top)}$	—	139	—	°C/W	
	$\theta_{JC(Bottom)}$	—	11.95	—	°C/W	
	Ψ_{JT}	—	6.15	—	°C/W	
Thermal Resistance, SOT-89	θ_{JA}	—	104	—	°C/W	EIA/JEDEC JESD51-7 FR-4 4-Layer Board
	$\theta_{JC(Top)}$	—	74	—	°C/W	
	Ψ_{JT}	—	30	—	°C/W	

En admettant que la tension maximum pouvant être fournie par l'USB est de 5.25V (selon la norme USB), que la tension de sortie du régulateur est de 3.3V et que le circuit débite le courant maximum (250mA), la puissance dissipée par le composant est la suivante :

$$P_d = (5.25 - 3.3) * 0.25 = \mathbf{487.5mW}$$

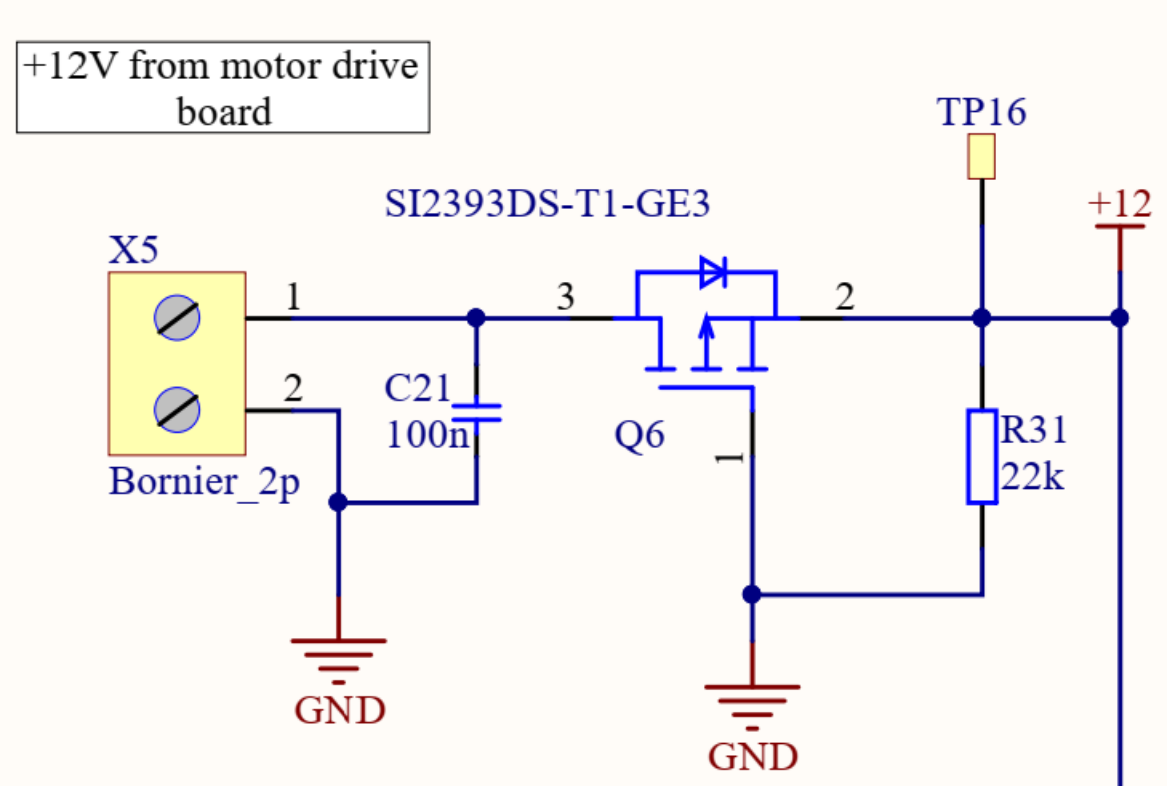
En l'occurence, en admettant que le circuit sera placé dans un environnement dont la température ambiante ne dépasse pas 35°C, la température de jonction pourra atteindre :

$$T_J = P_d * \theta_{JA} + T_A = 0.488 * 104 + 35 = \mathbf{85.7°C} < T_{J_{MAX}}(125°C)$$

Donc même en saturant le courant en sortie du régulateur, il pourra continuer à fonctionner avec une marge de 40°C. Il faudra par contre d'attendre à ce que la surface du composant soit brulante.

12V Carte externe

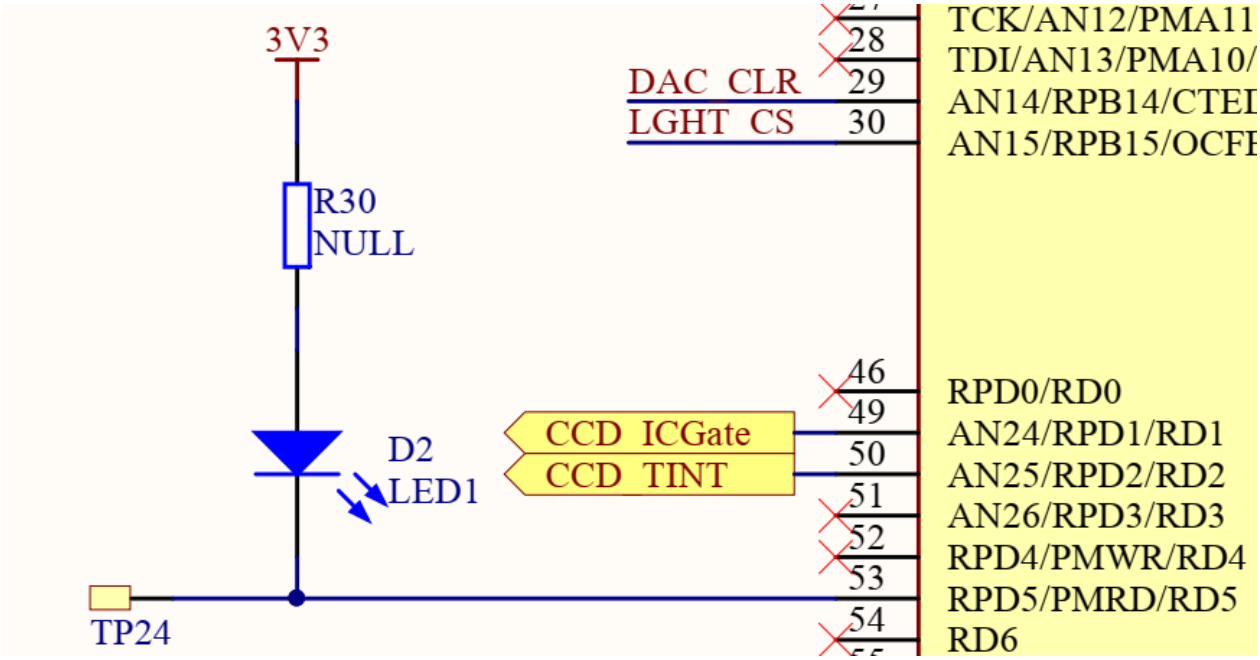
La tension 12V permettant d'alimenter le système d'éclairage IR provient de la carte du projet n°1921_PilotagePrecisionMoteurLin. Elle entre via un bornier à vis protéger contre l'inversion de polarité par un transistor MOSFET canal P et est directement rediriger vers le connecteur pour la carte d'éclairage IR.



En outre, le transistor conduira quand la tension est correctement polarisée puisque la diode parallèle au canal conduira. La tension sur la source sera plus grande que sur la grille donc le transistor sera dans un état de conduction. A l'instar si la polarité est inversée, la diode ne conduira pas, la tension sur la source ne sera pas plus grande que sur la grille et par conséquent le transistor sera dans un état bloqué.

Le transistor Q6 a été sélectionné dans le catalogue du fournisseur Mouser en tenant compte de la préférence d'avoir une RDS(on) (environ 30mΩ) faible tout en pouvant accepter un courant drain source au-delà de 20mA.

Led témoin



La led D2 sert de témoin pour le circuit. En autre indiquer l'état du circuit.

Pendant la phase de développement software, elle permettra également de faire du débug.

Il s'agit d'une led rouge 20mA avec une tension Vf de 1.8V.

Définissons un courant direct de 10mA afin de ne pas utiliser la pleine intensité de la led.

$$R30 = \frac{V_{CC} - V_f}{I_f} = \frac{3.3 - 1.8}{0.01} = 150ohms$$

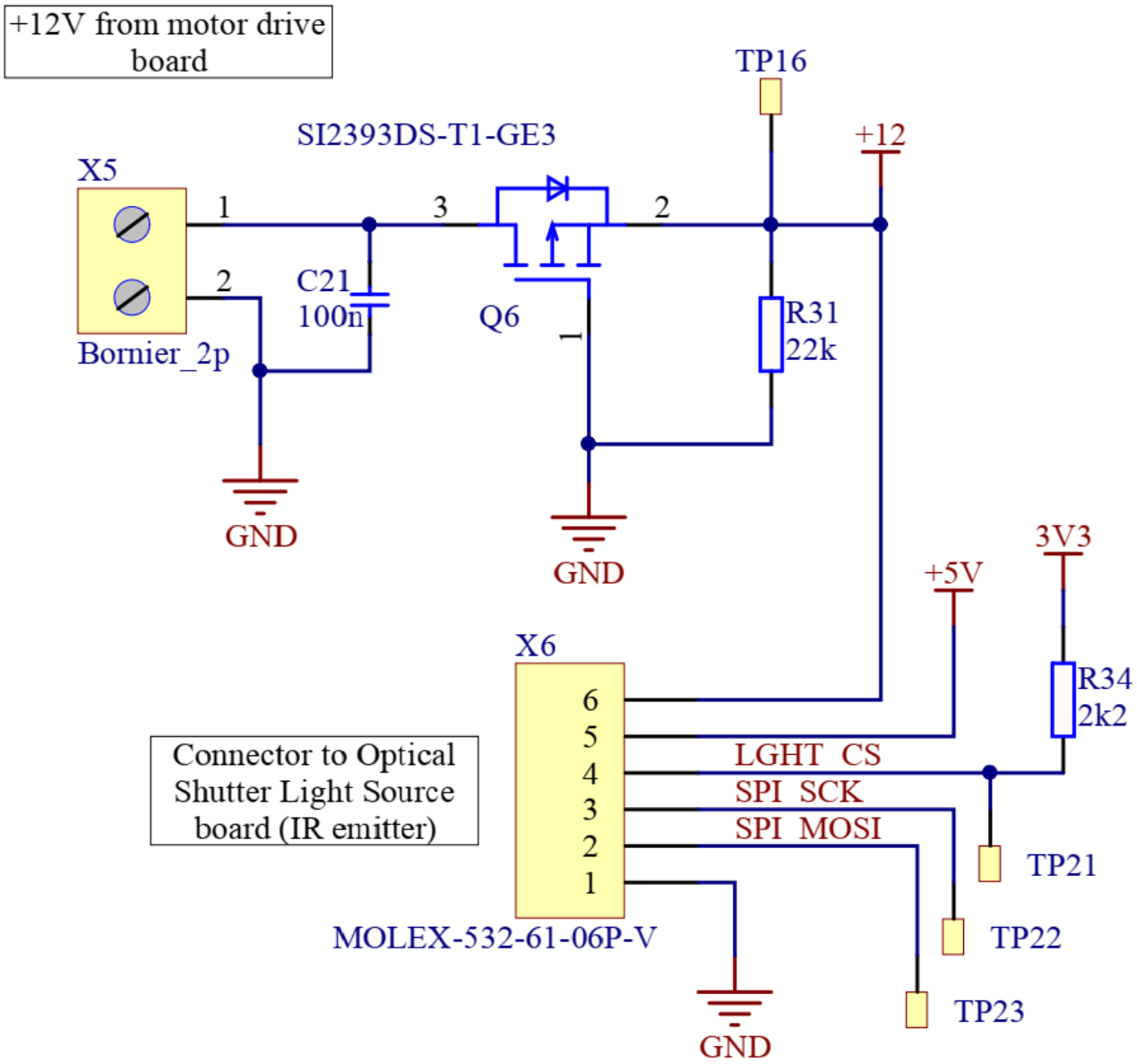
Dans ce cas-ci, la led est pilotée en tension. En toute rigueur, elle devrait être pilotée en courant à l'aide d'une source à courant constant. Ceci car la barrière de potentiel des leds à tendance à s'élargir avec le temps. Par conséquent, la tension de la led augmente ce qui a pour incidence de diminuer le courant traversant la led et sa luminosité.

Interface pour la board d'éclairage IR

Afin de pouvoir interfacer le circuit d'éclairage IR développé par PHYD Electronics, le connecteur X6 a été prévu pour croiser les connexions présentes sur le circuit déjà réalisé.

Le connecteur est un Molex PicoBlade 6 pôle droit.

La résistance de pull-up R34 a été placée afin de garantir un état inactif sur le chip select du SPI de la board lors de la mise sous tension du système.

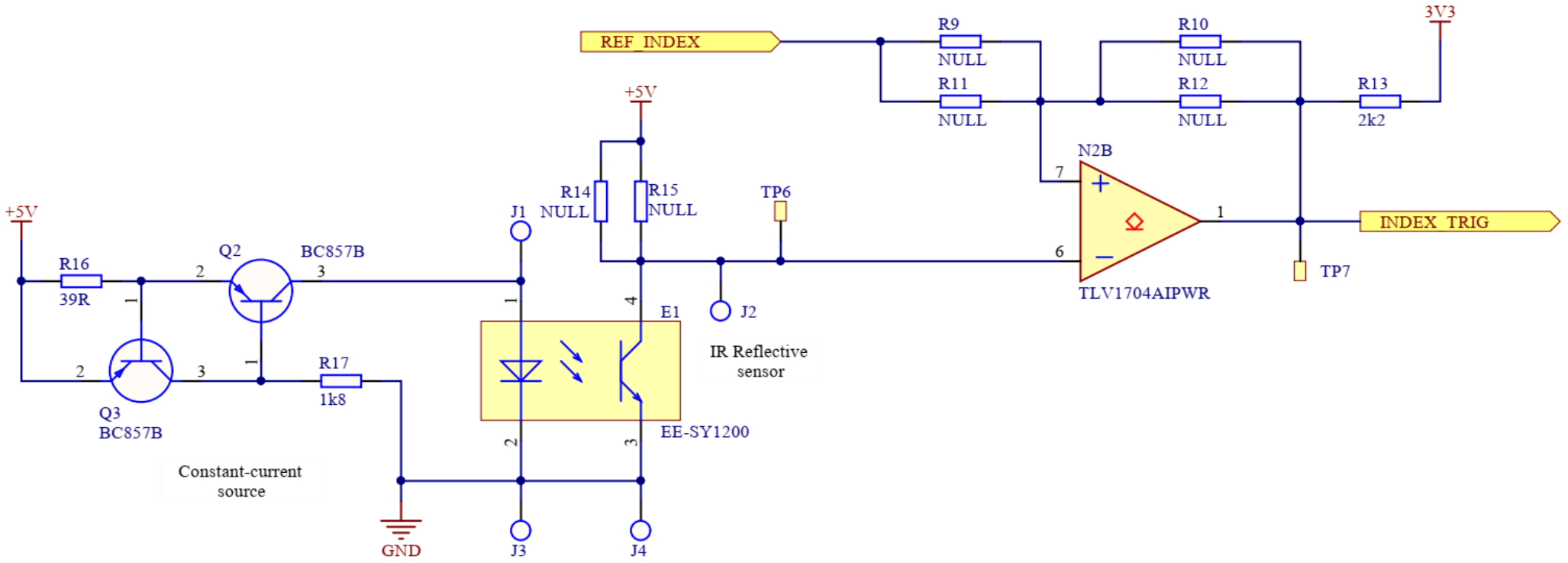


Barrière optique réflective

L'obturateur étant doté d'une fente index, un système de détection par barrière optique réflective a été implémenté en réserve si un jour le besoin d'une détection de retour à 0 se fait ressentir.

Le montage à trigger de schmitt réglable du capteur CCD a été repris afin de pouvoir ajuster le seuil.

Les jeux de résistances n'ont cependant pas été dimensionnés puisque cette fonctionnalité n'est pas nécessaire pour le moment.



Circuit imprimé

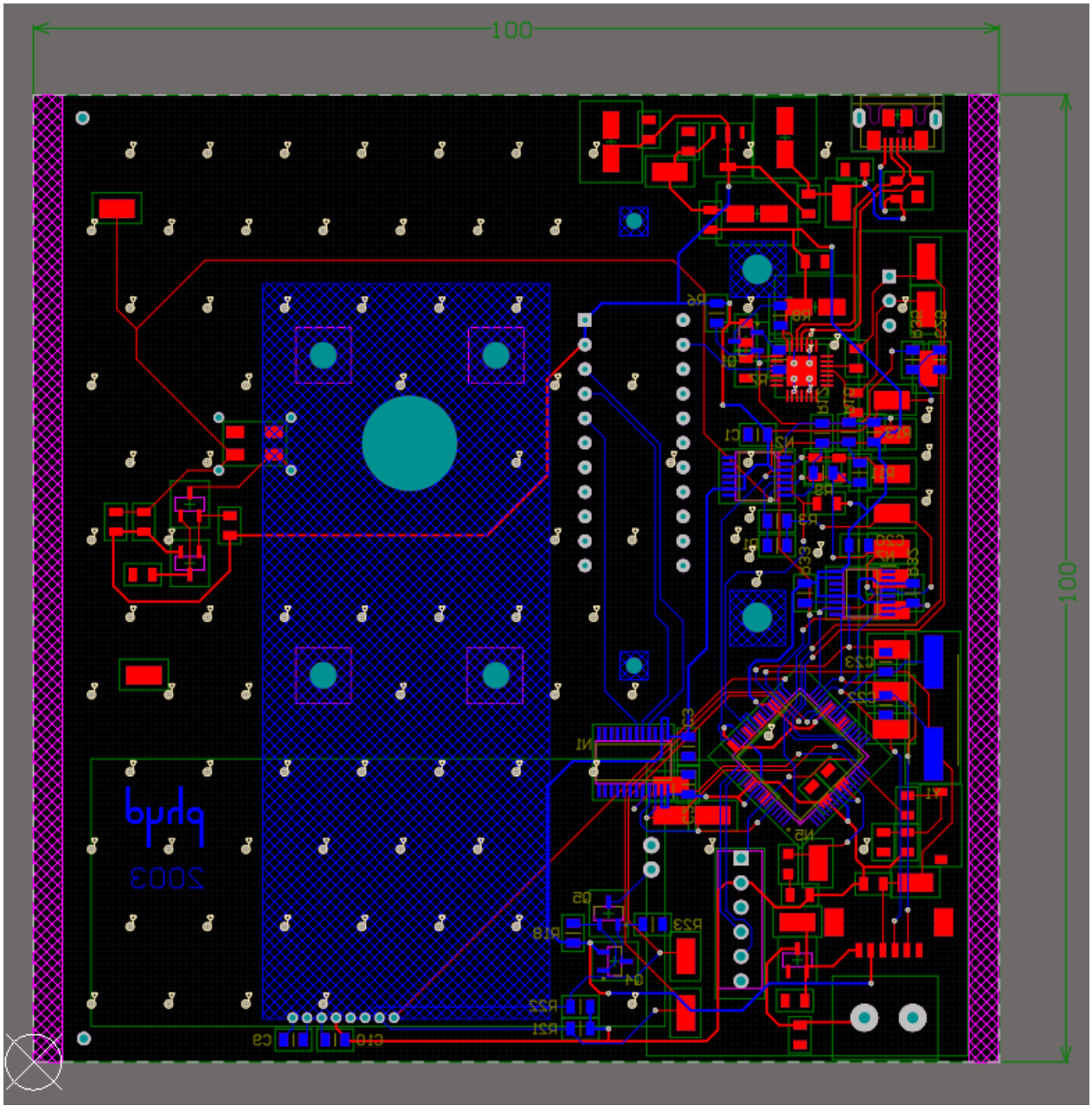
Le circuit a été prévu pour être glissé dans un boitier avec un rail format Europe.

Sa taille est donc de de 100x100mm. Des zones de restrictions ont été placées sur les côtés du PCB afin de n'avoir aucun cuivre, signal ou composant sur la zone recouverte par le rail du boitier.

Il a été routé pour correspondre à la classe de fabrication 6C selon les critères d'Eurocircuit.

Les pistes des signaux ont été routés avec une largeur de 0.2mm et les pistes d'alimentations (+3.3V, 5V, 12V, GND) à 0.4mm.

Il s'agit d'un circuit bi-couche. Il y a un plan de masse (GND) sur chaque couche.



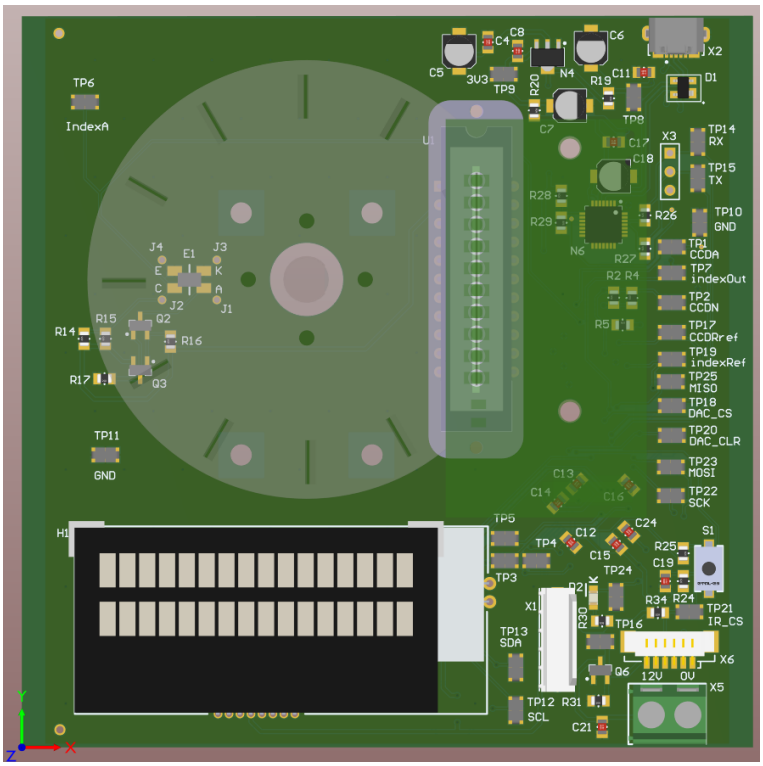
PCB vue TOP

Sur cette vue du circuit, l'obturateur est visible sur la partie centrale gauche. Le capteur CCD ainsi que son capot et l'éclairage sur la partie centrale, légèrement sur la droite.

L'écran LCD se situe en bas à gauche.

Le connecteur USB est en haut à droite.

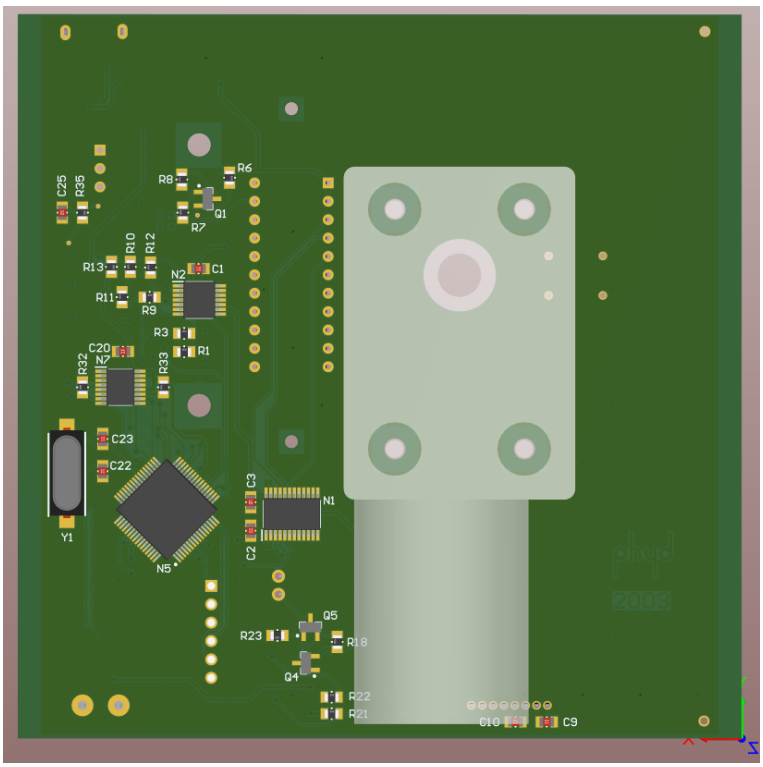
Les connecteurs de programmation, pour la board IR PHYD ainsi que pour l'alimentation 12V se situent en bas à droite.



PCB vue BOTTOM

Le moteur DC et visible sur la partie centrale du circuit.

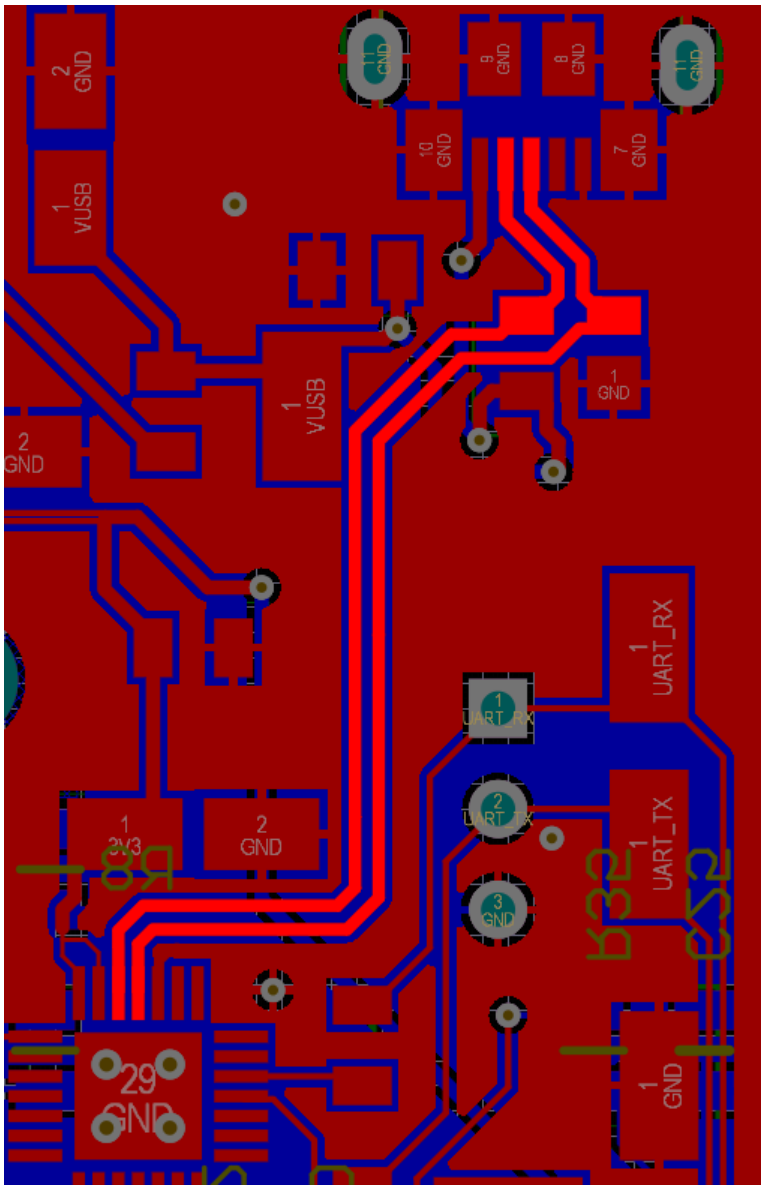
Le µC avec son quartz se trouve sur la partie gauche, légèrement en bas.



Routage en pair différentiel

La partie data USB entre le connecteur USB et le convertisseur USB-UART a été routé en pair différentiel avec des règles de routage permettant d'obtenir une impédance entre 90 et 100Ω.

Le routage a été réalisé de telle manière à ce que la ligne attaque en premier les diodes de protection ESD avant le reste du circuit (alimentation 5V y compris).

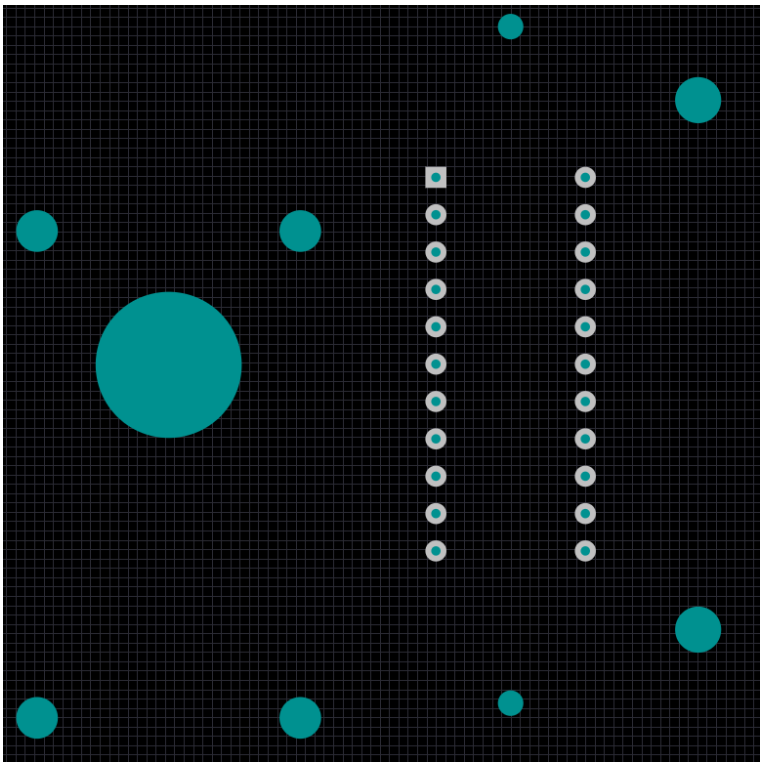


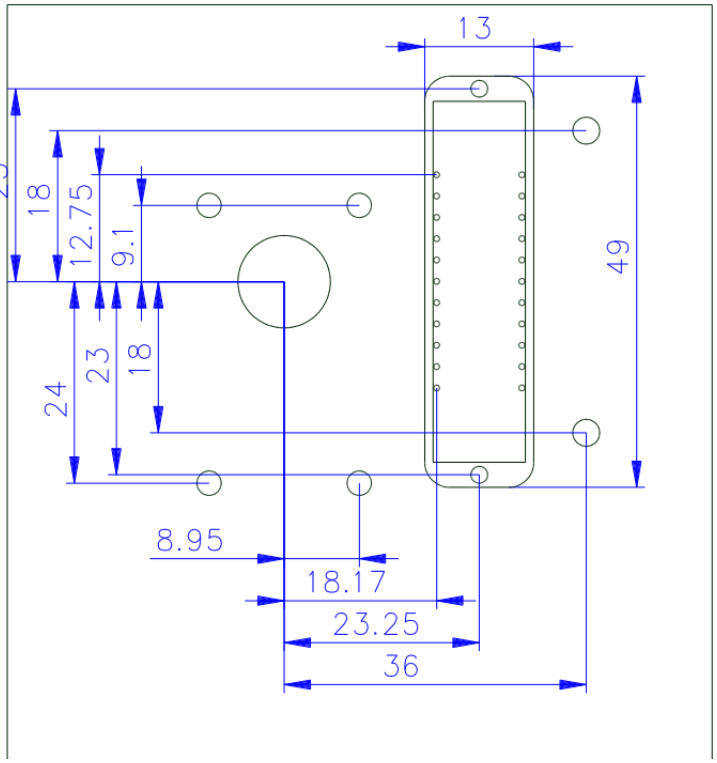
Trous de fixations

Tous les trous ci-contre ont été placés de manière précise à des cotes prédéfinis et fournies par PHYD Electronics.

Les cotes fournis sont indiqués juste en dessous de la vue du PCB.

Le but était de placer l'axe du moteur (couplé à l'obturateur) précisément par rapport aux cellules du CCD et de l'éclairage IR.





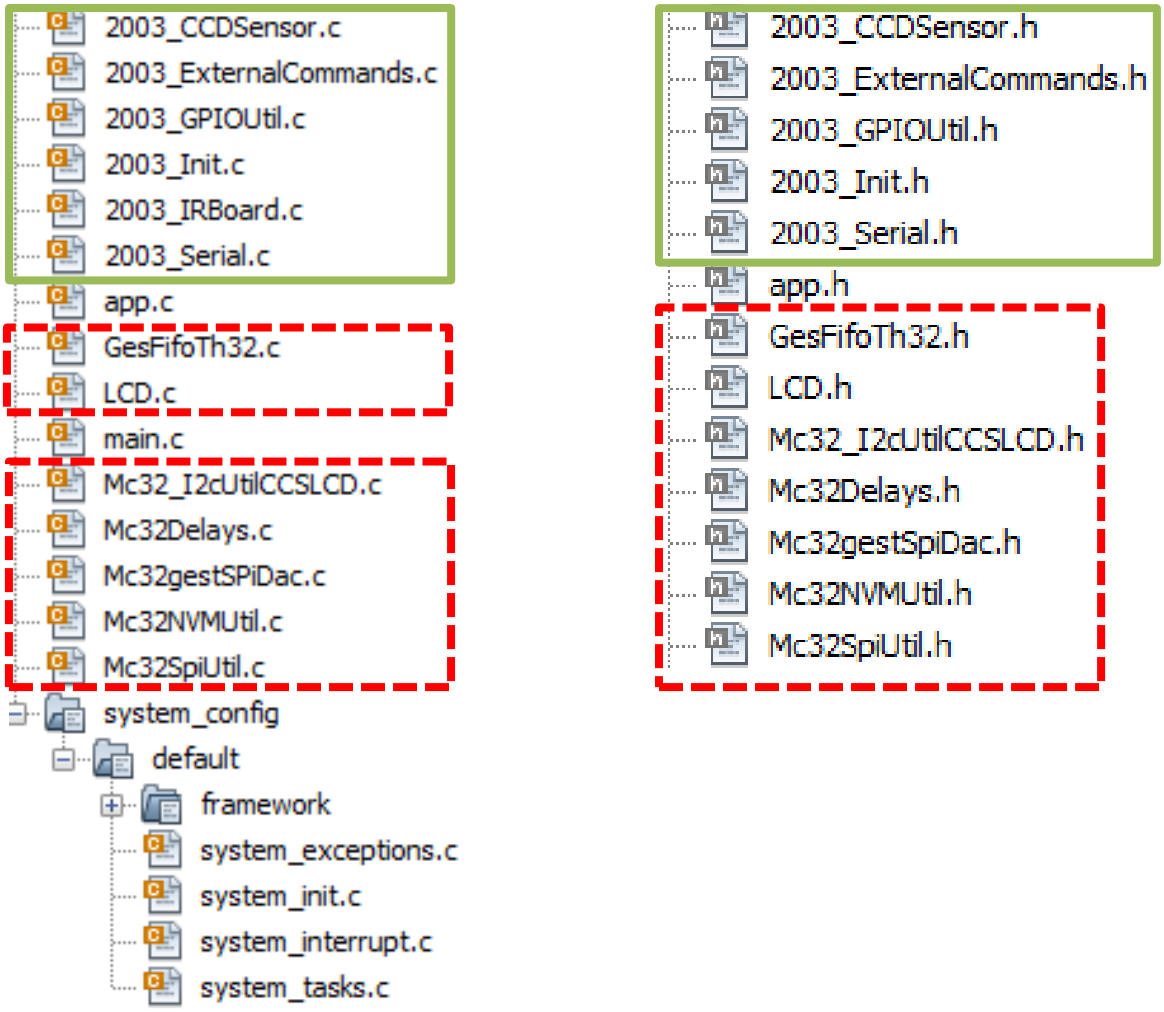
4 Software

Concept

Le firmware, codé en langage C, a été réalisé sous MPLAB X IDE avec le framework Harmony v2.06 de Microchip. Compilé avec le compilateur XC32 v2.15 sans optimisation de compilation.

Le programme consiste en une machine d'état en superloop séquencé par un ordonnanceur sous la forme d'un timer hardware.

Les fichiers utilisés non-généré par le framework Harmony sont les suivants :



Les fichiers encadrés en vert/traits pleins, sont les fichiers qui ont été créé spécifiquement pour ce projet. Ceux encadrés en rouge/traits tillés, sont des fichiers qui ont été repris de la laibrairie standard de l'ETML-ES ou d'autres projets ayant utilisé les même ressources. Les autres sont les fichiers générés par Harmony.

Synthèse des éléments réalisé lors de l'élaboration du programme

Afin de mener à bien la réalisation de ce programme, les éléments suivants ont dû être réalisés :

- Driver SPI pour le DAC LTC2624.
- Driver SPI pour le potentiomètre de la board d'éclairage IR.
- Driver pour piloter le capteur CCD.
- Calcul de la position et de la vitesse angulaire à partir d'un déplacement linéaire tangentielle à un axe de rotation.

Les éléments qui ont été repris mais ont eu besoins d'être ajustés pour ce projet :

- Communication sérielle UART avec Interpréteur de commande (repris du projet 1921).
- Driver de l'écran LCD I^2C (repris d'un ancien projet de diplôme).

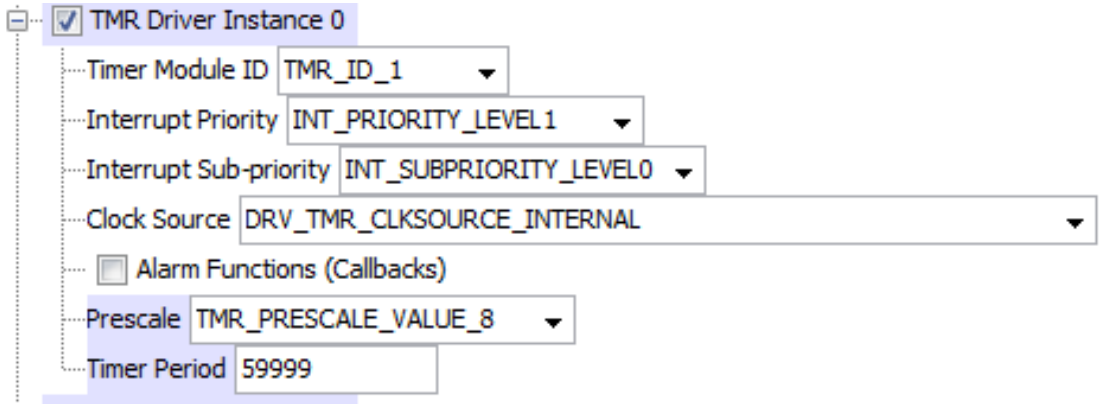
Les éléments repris tels quels :

- Gestion des délais passifs et actifs (librairie de l'ETML-ES).
- Gestion de sauvegarde en flash interne (librairie de l'ETML-ES).

Configuration des drivers sous Harmony

Timer 1

Le timer1 sert essentiellement à passer dans certaines tâches de manière séquentielle et à faire clignoter la led. Il s'agit d'un timer 16 bits donc 65536 pas maximum.



La période cible est 10ms.
Avec un PBCLK de 48MHz, il faut un prescaler de

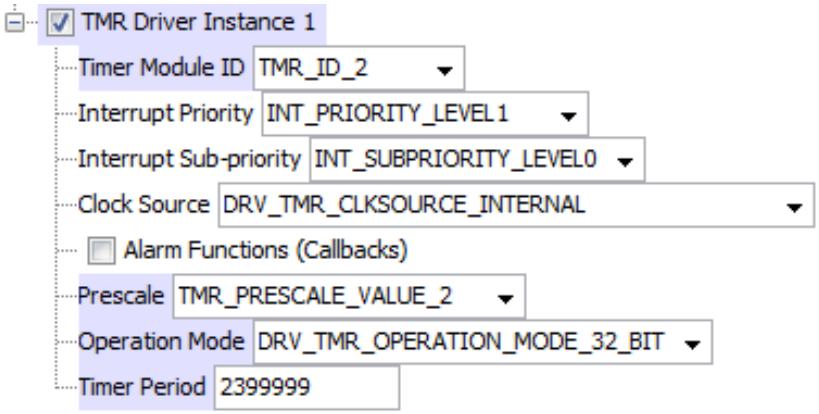
$$Prescaler_{MIN} = \frac{t * PBCLK}{655536} = \frac{10 \times 10^{-3} * 48 * 10^{6}}{65536} = 7.32 => \mathbf{8}$$

La valeur de recharge du timer doit être de

$$Période[tick] = \frac{t * PBCLK}{Prescaler} - 1 = \frac{10 \times 10^{-3} * 48 * 10^{6}}{8} - 1 = 59999$$

Timer 2

Le timer2 permet de générer la pulse pour le signal d'intégration du CCD en accord avec l'Output Compare 3. Il est réglé sur une résolution de 32 bits (couplé avec le timer 3) afin d'avoir une résolution suffisante pour avoir une pulse de durée très petite avec un rapport cyclique grand. Pulse < 1µs avec une période de 100ms.



La période cible est 100ms.
Avec un PBCLK de 48MHz, il faut un prescaler de

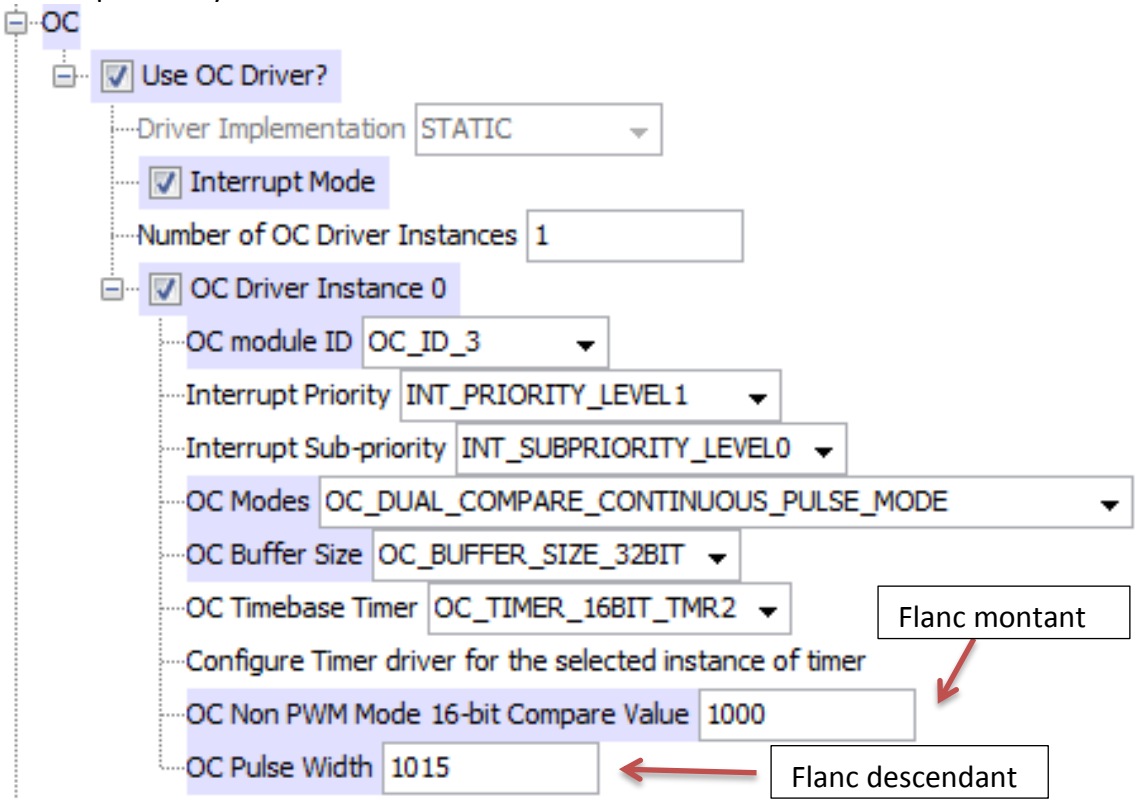
$$Prescaler_{MIN} = \frac{t * PBCLK}{2^{32}} = \frac{100 \times 10^{-3} * 48 * 10^{6}}{2^{32}} = 1.12 => \mathbf{2}$$

La valeur de recharge du timer doit être de

$$Période[tick] = \frac{t*PBCLK}{Prescaler} - 1 = \frac{100\times10^{-3} * 48 * 10^{6}}{2} - 1 = 2399999$$

Output Compare 3 (OC3)

L'OC 3 permet de générer la pulse d'intégration du CCD de manière cyclique, elle est couplée au timer2 qui a un cycle de 100ms.



Il a été réglé en mode OC_DUAL_COMPARE_CONTINUOUS_PULSE_MODE afin de pouvoir générer la pulse de manière cyclique. Le mode DUAL permet de définir deux valeur de comparaisons de l'OC. Une pour le flanc montant et une pour le flanc descendant. Cela permet de pouvoir placer la pulse où lôn souhaite lors de la séquence de comptage du compteur. Ceci car on souhaite mettre la pin de clear gate à 0 (lors de l'interruption du timer2) du CCD avant de faire l'intégration car on veut laisser le temps aux cellules de se décharger avant de les recharger.

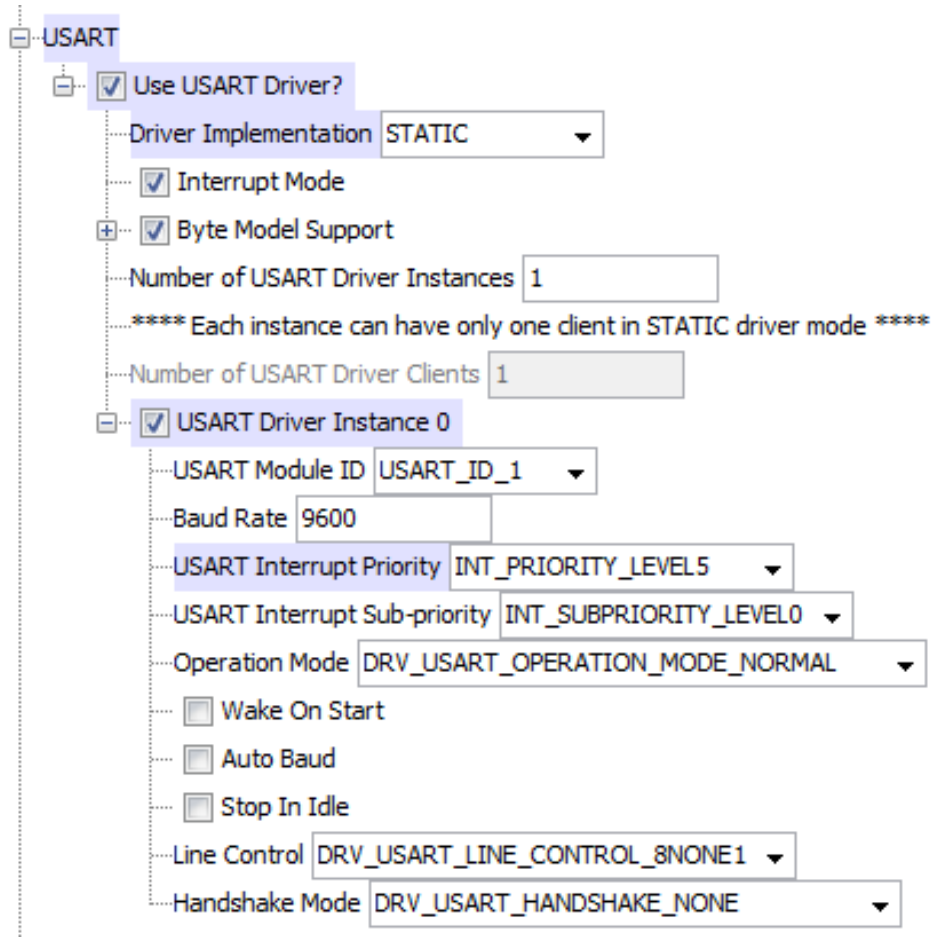
En l'occurence ici, nous avons un flanc montant généré au pas 1000 du timer. Ce qui correspond à un temps t0 de : $t_0 = \frac{1000}{f_{timer2}} = \frac{1000}{24\times10^6} = 41.66\mu s$.

Le flanc descendant est lui généré au pas 1015 du timer2. Cela correspond à un temps t1 de : $t_1 = \frac{1015}{f_{timer2}} = \frac{1015}{24\times10^6} = 42.29\mu s$

La durée de la pulse à l'état haut sera donc de : $t_{high} = t_1 - t_0 = (42.29 - 41.66) \times 10^{-6} = \mathbf{625ns}$

UART 1

Le module USART1 utilisé en UART sert à recevoir les données du convertisseur USB-UART pour le traitement des commandes utilisateur externe.

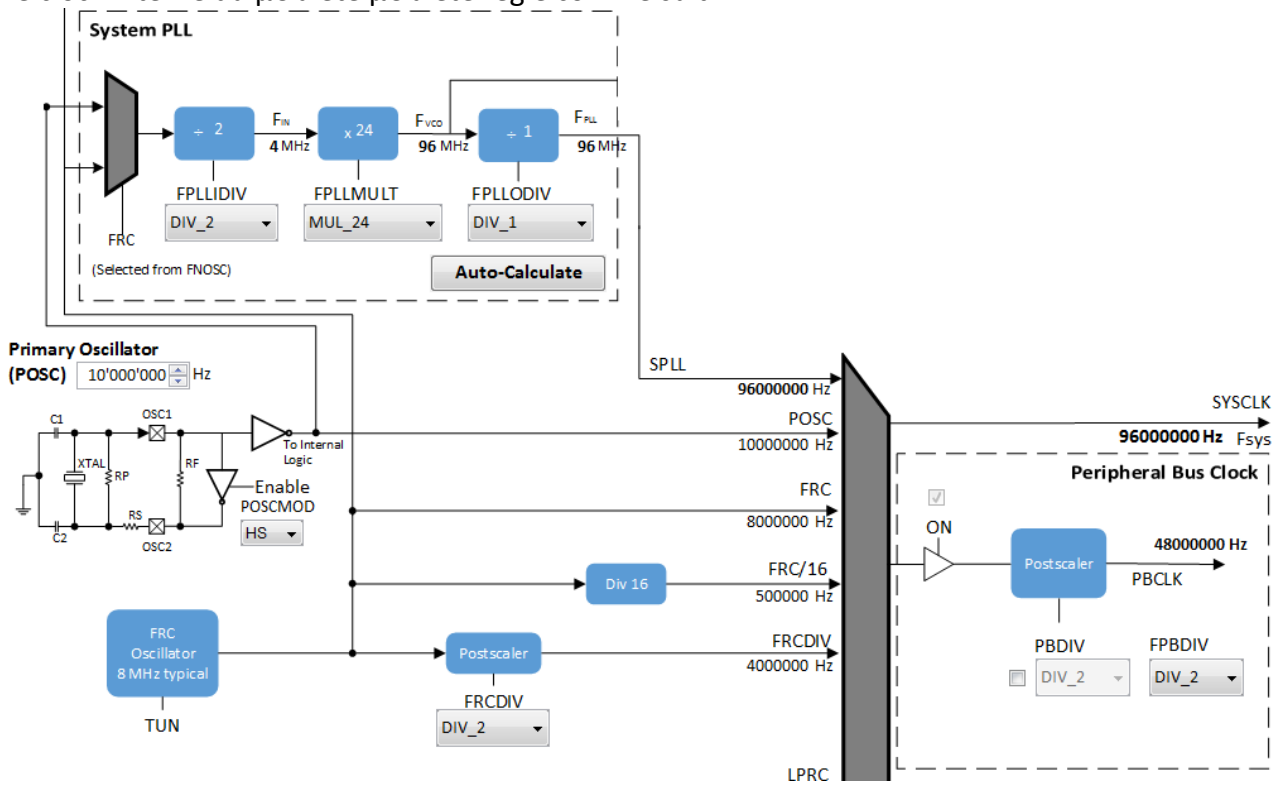


Il a été réglé avec un baudrate standard de 9600 baud. Il s'agit d'un débit bas, suffisant pour cette application mais qui peut être augmenté si le besoin se fait ressentir.

Les trames se composent comme suit : 8 bits de données, pas de parité et un bit de stop. Pas de contrôle de flux hardware.

Configuration du clock

Le clock interne du µC a été µC a été réglé comme suit :



Utilisation de l'oscillateur principale comme source 10MHz qui passe dans la PLL afin de multiplier la fréquence jusqu'à 96MHz. La source de 96MHz alimente le System Clock (CPU et CoreTimer). Le System Clock est divisé par deux avant d'attaquer le Peripheral Bus Clock qui cadence les périphériques du µC.

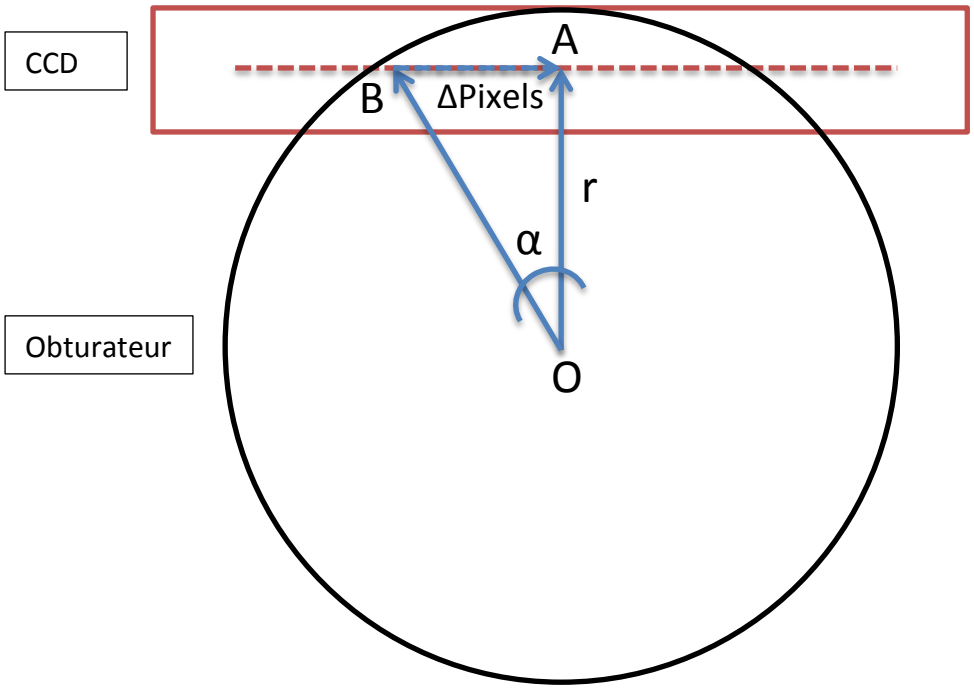
Principes généraux

Le corps du programme consiste à relever toutes les 100ms la position des flancs provoqués par les transitions ombres-lumières du CCD qui apparaissent lors du déplacement des fentes de l'obturateur devant l'éclairage IR. Dans ce cas-ci, avec la configuration du trigger de Schmitt, il s'agit de relever les flancs descendants qui sont les plus précis. Il peut y en avoir maximum 3 et minimum 2 dans le champ du CCD. Il y en a 2 la plupart du temps et 3 quand une ouverture sort du champ et une autre entre.

A chaque flanc descendant, on relève la position du pixel du CCD qui l'a provoqué. A chaque mesure (toutes les 100ms) on compare la nouvelle position du flanc par rapport à la précédente afin de déterminer le déplacement de l'ouverture. Lorsqu'une ouverture s'apprête à sortir du champ, le programme fait en sorte de tenir compte du déplacement de l'ouverture précédente ainsi que de celle qui vient d'entrer dans le champ et ainsi de suite.

Calcul de la position angulaire

Un des aspects dont il a fallu tenir compte dans le programme, était le fait que capteur est placé tangentiellement par rapport au disque de mesure. Il y a donc un rapport tangentiel entre le déplacement mesuré en pixel et la rotation réelle du moteur. Il a donc fallu effectuer dans le programme une conversion de la position en pixel en position angulaire par rapport au centre du capteur. C'est à dire que la référence est prise au centre du capteur, au pixel n° 3648/2 = 1824.



Nous cherchons la position angulaire du pixel par rapport au point de référence A. Nous connaissons la position du pixel au point qui correspond à n pixel/2 (3648/2 = **1824**). Nous connaissons la position du point B puisqu'il s'agit de la position du flanc que le µC a relevé. Avec ces deux valeurs et en sachant la distance entre chaque pixel, nous connaissons donc ΔPixels. Nous connaissons le rayon r puisqu'il s'agit de la distance entre le centre du l'obturateur O et le milieu d'une ouverture A, donc **25mm.**

En sachant que $\tan \alpha = \frac{\Delta Pixels}{r} = \frac{(A-B) * distance_{pixels}}{r} = \frac{(1824 - Position_{Pixel}) * 8\mu m}{25mm}$

L'angle alpha vaut donc $\arctan\left(\frac{(1824 - Position_{Pixel}) * 8\mu m}{25mm}\right)$

La librairie standard <math.h> nous fournit la fonction atan() qui agit comme l'opérateur arctan et retourne une valeur en radian. Il faut donc convertir la valeur en radian avec un facteur 360/2π ou 180/π afin de finalement obtenir la position angulaire du flanc en degré :

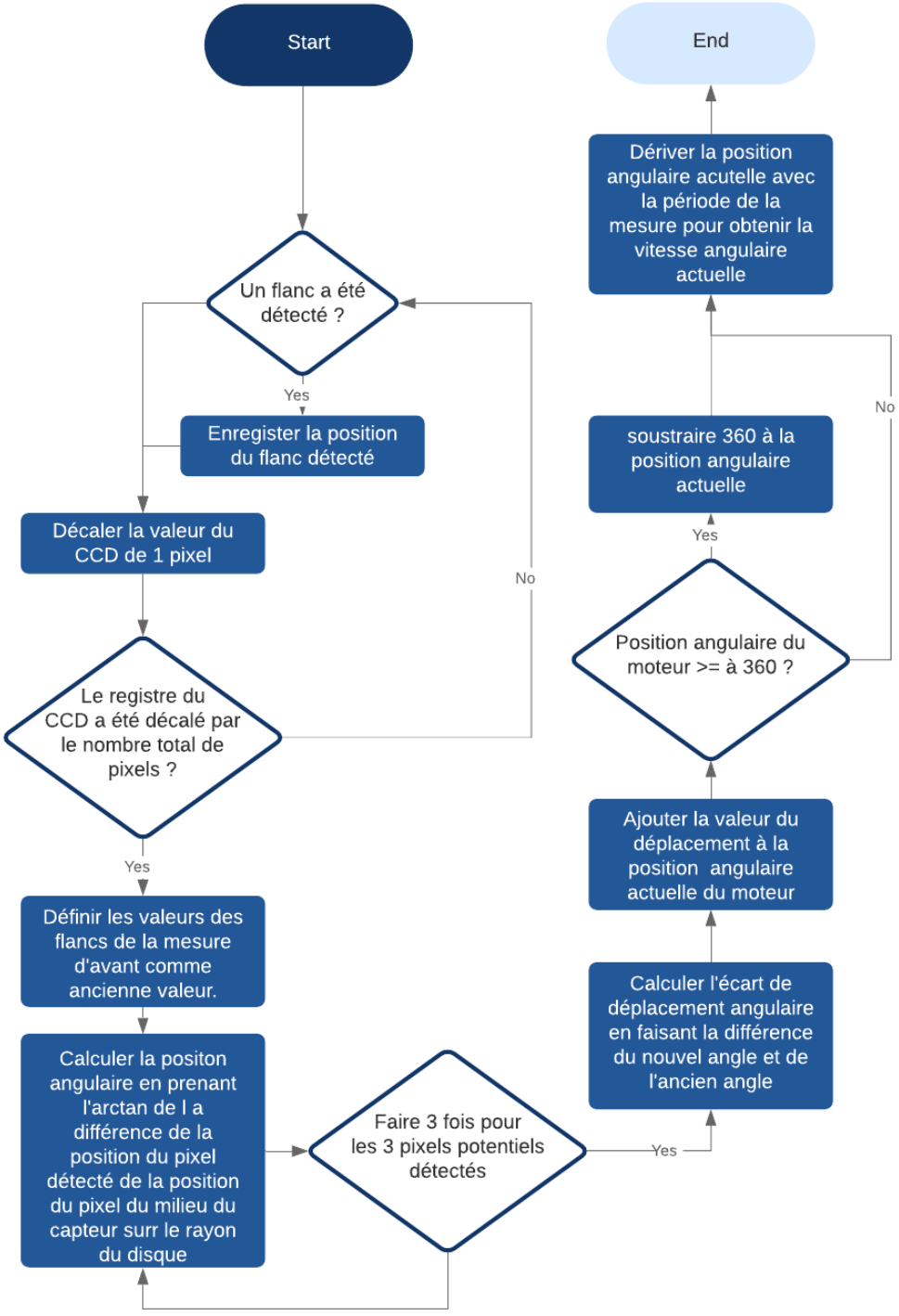
```
//conversion px en deg
for(i = 0; i < 3; i++)
{
    edgePosHistoryDeg[i][1] = (180 * atan((1824-edgePosPx[i]) * CCD_PIXEL_GAP / 25000.0) / M_PI);
}
```

Où CCD_PIXEL_GAP vaut 8 et M_PI vaut π selon la librairie <math.h>.

Une fois que nous avons la position angulaire actuelle du flanc, nous pouvons la soustraire avec la précédente pour déterminer le déplacement entre deux mesures et ainsi incrémenter le compteur de position angulaire avec cette valeur afin de connaitre dynamiquement la position angulaire de l'arbre du moteur.

Organigramme de la tâche APP_STATE_CCD_GET_DATA_TASK

Cette tâche implémentée dans le fichier principal app.c et appelé cycliquement toutes les 100ms par une interruption de timer, se charge d'acquérir les données du CCD et de les traiter pour obtenir la position et la vitesse angulaire :



2003_CCDSensor.c/h

Ce fichier contient les fonctions, variables et constantes directement en lien avec le capteur CCD.

Fonctions

void CCD_Init(void)	Cette fonction est appelée à l'initialisation du programme et à la demande de reset des valeurs par l'utilisateur depuis l'interface série. Elle initialise les valeurs en lien avec le CCD tel que les valeurs par défaut du DAC, vitesses, position…etc. Elle appelle également la fonction d'initialisation de la carte IR PHYD.
void SPI_CCD_CLK_Init(void)	Cette fonction initialise le SPI2 du µC qui permet de cadencer le master clock du capteur CCD.
void SPI_CCD_4pulses(void)	Cette fonction effectue 4 coups de clock sur la ligne master clock du CCD en envoyant al valeur 0xAA (0b10101010) sur la ligne MISO du SPI2. 4 coups de clock correspondant au décalage d'un pixel du CCD.
void Display_values_on_LCD(double pos, double currentSpeed, double minSpeed, double maxSpeed)	Cette fonction permet d'afficher les valeurs mesurées par le CCD sur l'écran LCD. • double pos : position actuelle de l'axe du moteur en degré. • double currentSpeed : vitesse actuelle de l'axe du moteur en degrés par secondes. • double minSpeed : vitesse minimum mesurée par le CCD en degré par secondes. • double maxSpeed : vitesse maximum mesurée par le CCD en degré par secondes.

Variables

S_CCDSensor CCDSensor	Structure globale contenant les paramètres et valeurs principales du système.
_Bool SPI2BusIsFree	Variable de flag globale permettant de savoir s'il est possible de mettre une donnée dans le buffer du SPI2 pour créer le clock du CCD. Cette variable est actualisée dans la routine d'interruption de détection de buffer vide du SPI2.

Constantes

CCD_DEFAULT_TRIG_REF	Valeur par défaut pour la référence du trigger du CCD.
INDEX_DEFAULT_TRIG_REF	Valeur par défaut pour la référence de l'index.

2003_ExternalCommands.c/h

Ce fichier contient les fonctions, variables et constantes permettant de gérer le système de commandes sur le bus série virtuel.

Fonctions

void ExternalCommands_Init(void)	Cette fonction permet d'initialiser les valeurs en lien avec le gestionnaire de commandes externes.
void ExternalCommands_GetFromFifo(void)	Cette routine appelée périodiquement permet d'aller récupérer le contenu de la FIFO software de réception de l'UART et de détecter si une commande y est stockée. Ceci se fait par des fonctions de manipulations de chaines de caractère. Ce mécanisme a été repris de mon projet précédent (projet n°1921).
void Do_ExternalCommand(void)	Routine appelée périodiquement permettant d'exécuter la commande envoyée par l'utilisateur (si commande présente dans le FIFO).
void Command_FirmwareVersion(void)	Fonction permettant d'envoyer la version du firmware actuel sur le port série.
void Command_save(void)	Fonction permettant d'enregistrer les paramètres actuels dans la flash interne.
void Command_reset(void)	Fonction permettant de remettre les valeurs par défaut au démarrage du système.
void Command_NonExistent(void)	Envoi une chaine de caractère indiquant que la commande n'est pas prise en charge.
void Command_SendOk(void)	Envoi "Ok" sur le port série. (Permet d'indiqué quand la commande a été exécutée).
void Commmand_ReadDacValue(void)	Lis les valeurs actuelles des canaux du DAC.
void Command_SetDac(uint8_t dac_ch)	Modifie la valeur du DAC avec la valeur indiquée entre 0 et 4095. • uint8_t dac_ch : numéro du canal du DAC (0 pour le CCD et 1 pour l'index).
void Command_SetIRBoard(void)	Modifie la valeur de l'intensité de l'éclairage IR entre 0 et 255.
void Command_GetPos(void)	Envoi sur le port série la valeur de la position actuelle.
void Command_GetSpeed(void)	Envoi sur le port série la valeur de la vitesse actuelle.
void Command_StreamValues(void)	Active le mode streaming des data (affichage des valeurs en continu sur le port série).

Variables

S_ExternalCommands ExternalCommand	Structure contenant les informations sur le type et la valeur de la commande reçue.

Constantes

Int8_t CommandsPrefixes[EXTERNAL_COMMAND_NB_OF_COMMANDS][EXTERNAL_COMMAND_PREFIX_SIZE]	Tableau contenant les préfixes des commandes codés sur 3 caractères.
EXTERNAL_COMMAND_PREFIX_SIZE	Nombre de caractère des commandes
EXTERNAL_COMMAND_NB_OF_COMMANDS	Nombre de commandes au total
EXTERNAL_COMMAND_VALUE_NB_OF_DIGIT	Nombre de digits de la valeur de la commande.

2003_GPIOUtil.c/h

Ce fichier contient les fonctions, variables et constantes permettant d'intéragir avec les I/Os numériques standard du µC.

Fonctions

inline void Init_GPIO(void)	Initialise les GPIOs utilisées.
void GPIOUtil_SetPinDir(uint8_t Dir, PORTS_CHANNEL portch, PORTS_BIT_POS bitpos)	Permet de définir la direction de la pin souhaitée. • uint8_t Dir : direction de la pin (1 sortie, 0 entrée). • PORTS_CHANNEL portch : port de la pin souhaitée. • PORTS_BIT_POS bitpos : poids binaire sur le port de la pin souhaitée.
inline void GPIOUtil_SetState(PORTS_CHANNEL portch, PORTS_BIT_POS bitpos, _Bool state)	Permet de définir l'état de la pin souhaitée. • PORTS_CHANNEL portch : port de la pin souhaitée. • PORTS_BIT_POS bitpos : poids binaire sur le port de la pin souhaitée. • _Bool state : état souhaité de la pin.
inline void Init_LED1(void)	Initialise la pin de la LED. Sortie et Open-drain.
inline void LED1_On(void)	Allume la LED.
inline void LED1_Off(void)	Eteint la LED.
inline void LED1_Toggle(void)	Bascule l'état de la LED
inline void LED1_Set(_Bool state)	Définit l'état de la LED.

2003_Init.c/h

Ce fichier contient les fonctions, variables et constantes permettant d'initialiser les drivers du circuit.

Fonctions

inline void BoardInit()	Apelle les fonctions d'initialisations de tous les drivers. Est appelée au début du programme.

2003_IRBoard.c/h

Ce fichier contient les fonctions, variables et constantes permettant d'intéragir avec le circuit d'éclairage IR en SPI de PHYD Electronics.

Fonctions

void SPI_InitIRBoard(void)	Initialise le potentiomètre numérique de la carte d'éclairage IR.
void SPI_ConfigureIRBoard(void)	Configure le bus SPI pour la carte IR.
void SPI_WriteIRBoard(int16_t data)	Envoi un data de 16 bits vers le potentiomètre numérique.
void SPI_WriteCfgIRBoard(int16_t data)	Fonction qui mixe la configuration SPI + envoi des data car le bus SPI est partagé avec un autre composant possédant une autre configuration.

Constantes

SPI_LED_HEADER	Masque de l'en-tête du paquet 16 bits pour le potentiomètre numérique.
IR_BOARD_DEFAULT_VALUE	Valeur par défaut du réglage du potentiomètre numérique.

2003_Serial.c/h

Contient les fonctions permettant d'envoyer et recevoir des caractères à partir des FIFOs hardware et software. Cette librairie a été reprise de celle élaborée à l'ETML-ES.

app.c/h

Fichier principale du programme. Elle permet d'appeler toutes les tâches du programme et d'exécuter la partie du code qui mesure et calcule la position et la vitesse angulaire dans la tâche APP_STATE_CCD_GET_DATA_TASK (voir dans le listing du code).

Il y a d'autres tâches telles que APP_STATE_INIT qui appelle la fonction qui initialise les drivers des composants. APP_STATE_WAIT qui est la tâche d'attente quand le programme ne fait rien. APP_STATE_SERVICE_TASK qui ne fait rien mais qui peut être utilisé pour effectuer des actions de tests par exemple. APP_STATE_EXTERNAL_COMMANDS_HANDLE qui permet de traiter les commandes reçues. APP_STATE_CCD_GET_DATA_TASK est la tâche qui gère l'acquisition des données du CCD. Finalement, APP_STATE_CCD_COMPUTE_TASK qui gère le traitement des données du CCD.

GesFifoTh32.c/h

Le fichier qui contient les fonctions permettant d'initialiser et d'interagir avec les FIFOs software. Ceci dans le but de stocker les caractères d'émissions et de réceptions pour l'UART et par conséquent le système de commandes externes. Cette librairie a été réalisée par l'ETML-ES.

LCD.c/h

Contient les fonctions permettant d'initialiser et d'interagir avec l'écran LCD I^2C avec les bons timings. Cette librairie a été reprise et adaptée à partir d'un ancien travail de diplôme utilisant ce composant.

Mc32_I2cUtilCCSLCD.c/h

Librairie adapté de la lib Mc32_I2cUtilCCS.c/h de l'ETML-ES pour le protocole I2C de l'écran LCD. Permet de d'interagir avec la couche hardware de l'I2C.

Mc32Delays.c/h

Fonctions de base permettant de générer des délais actifs et passifs. Librairie de base à l'ETML-ES.

Mc32gestSPiDac.c/h

Librairie contenant les fonctions permettant d'intéragir avec le DAC externe SPI. Native du projet du kit PIC32 de l'ETML-ES.

Mc32NVMUtil.c/h

Contient les fonctions permettant de stocker et lire les données dans la flash.

Mc32SpiUtil.c/h

Librairie de base de l'ETML-ES regroupant les fonctions de la PLIB qui gère les bus SPI.

system_interrupt.c

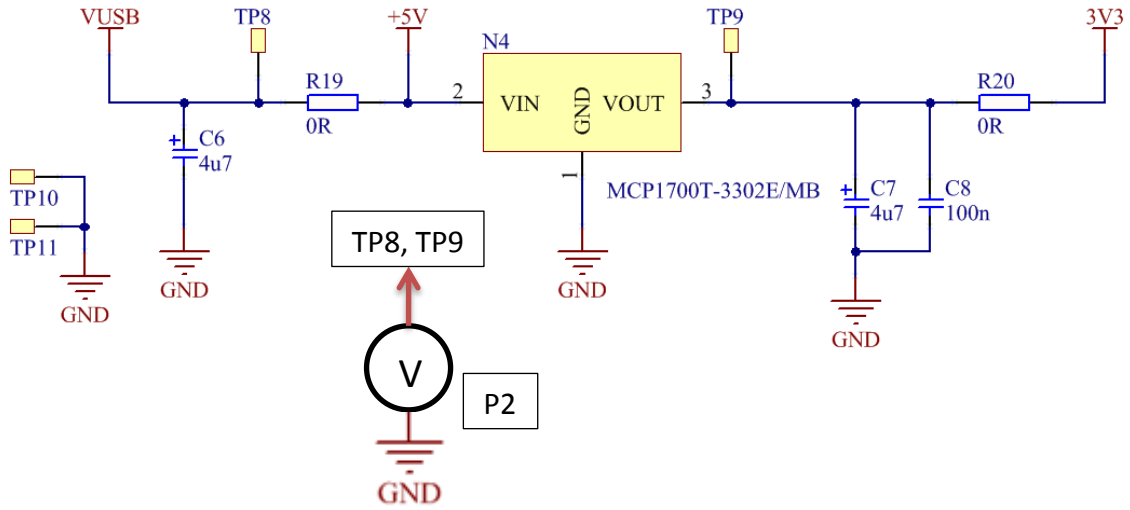
Fichier généré par Harmony regroupant les routines d'interruptions.

5 Tests et mesures

Mise en service

Contrôle des alims

Afin d'éviter d'endommager le circuit en cas de mauvais fonctionnement des alims. Des résistances 0Ω ont été placées entre les alims.



Avant de brancher le câble USB pour la première fois, les résistances R19 et R20 ne sont pas montés.

Brancher le câble USB et mesurer sur TP8 si la tension est bien de 5V (tolérance selon norme USB). Si tel est le cas, monter R19 et mesurer TP9. S'assurer que la tension est bien de 3.3V±0.05V.

Si tel est le cas, monter la résistance R20 pour alimenter le reste du circuit.

Mise en place des drivers des composants

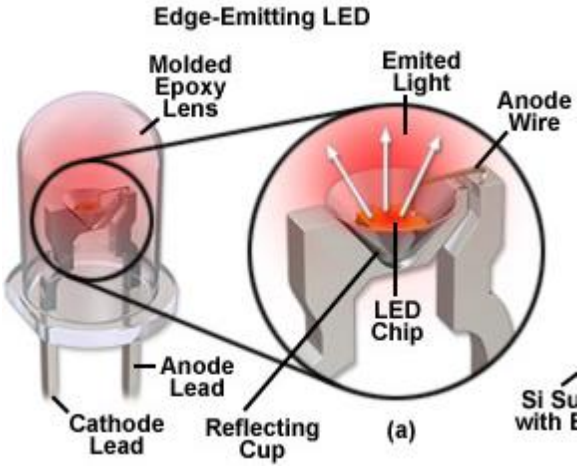
Afin de pouvoir tester si les composants fonctionnent, il a fallu coder les drivers de chacun dans le firmware du µC.

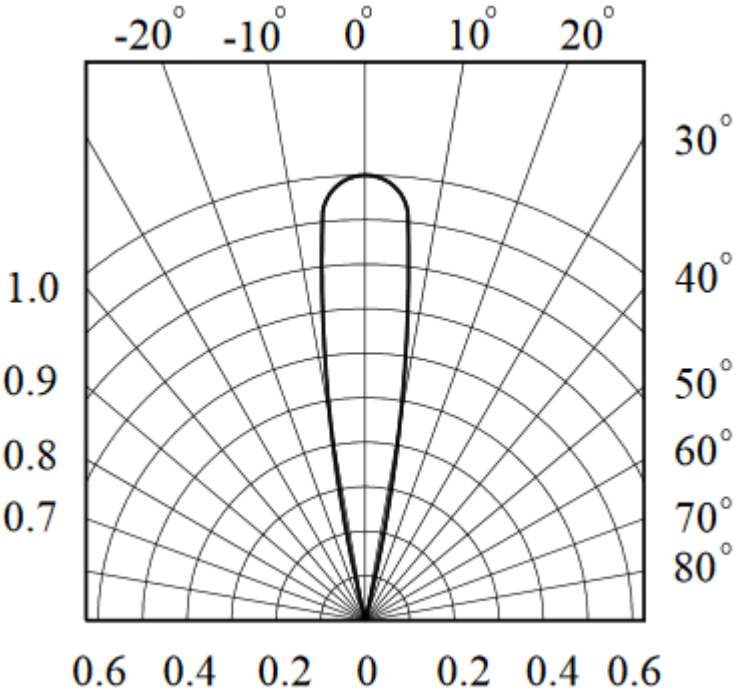
A priori, tous les composants fonctionnent puisqu'ils ont été testés dans le programme. Seul le capteur réflectif n'a pas été testé car il n'est pas utilisé pour cette application.

Effet spot des leds

L'amplitude des traces de l'ouverture provoquée par l'obturateur étant dans un premier temps trop variable en fonction de la position de l'ouverture entre le CCD et les leds, il a fallu déterminer la source de ce problème et le cas échéant une solution.

Selon la mise en garde de Monsieur Déglon sur l'effet spot des leds, l'attention a été portée la dessus. En effet, les leds ont un rayon de diffusion définit par la coupe réflective ("Reflective cup", voir la figure ci-contre).

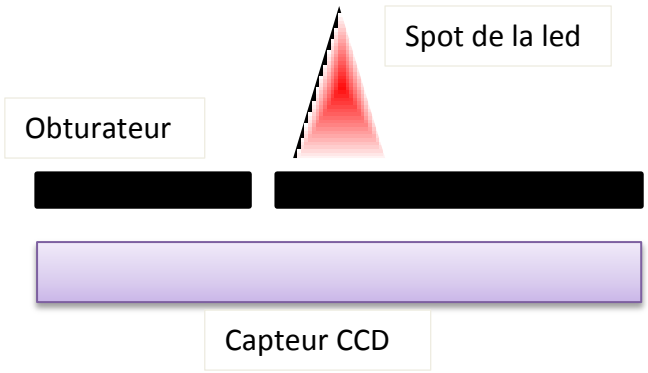




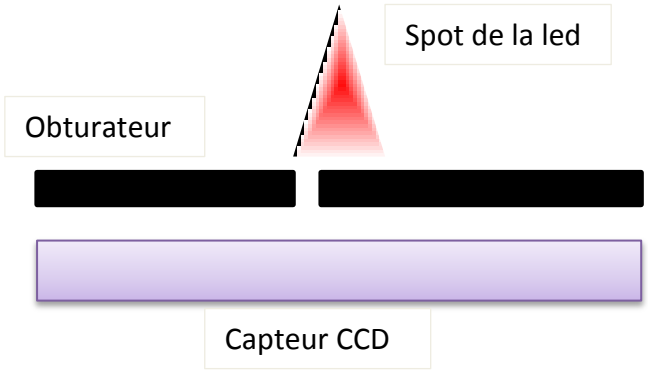
En l'occurrence, ces leds IR (IR26-21C-L110-TR8) ont un rayon d'émission de maximum ± 10° (Voir graphique extrait du datasheet ci-contre). Des leds avec un rayon d'émission plus élevé auraient pu permettre de mieux répartir l'éclairage sur la longueur du CCD. Et également de faciliter la diffusion de la lumière.

L'hypothèse de départ est donc la suivante : Lorsqu'une ouverture passe directement sous le point le plus direct d'une led, le capteur CCD capte une amplitude de pleine lumière. Hors quand une ouverture entre ou sors du champ d'émission d'une led, la lumière captée est bien moins intense qu'en exposition direct.

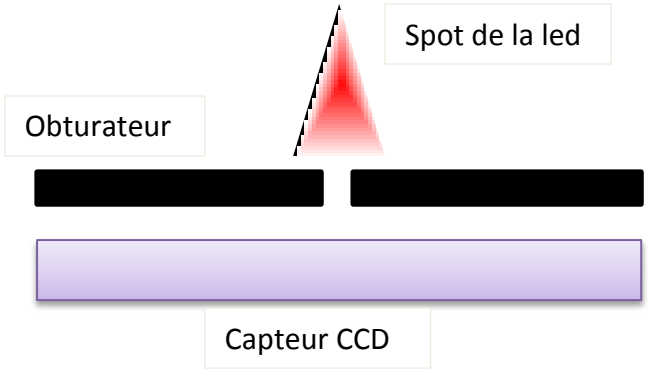
Représentation schématique de l'hypothèse en vue en coupe étape par étape :



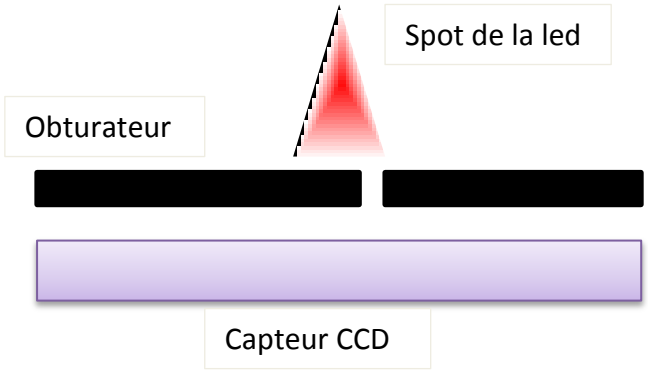
L'ouverture est hors du champ de la led la plus proche, l'intensité lumineuse mesurée par le CCD est théoriquement nulle.



L'ouverture entre dans le champ du spot de la led. L'intensité mesurée par le CCD à cet instant est minime.



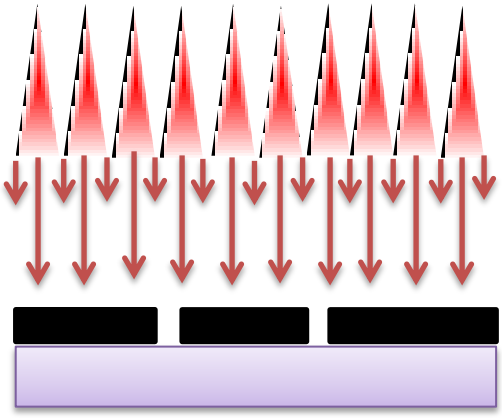
L'ouverture est centrée sur le spot de la led. A cet instant, l'intensité lumineuse mesurée par le CCD est maximale.



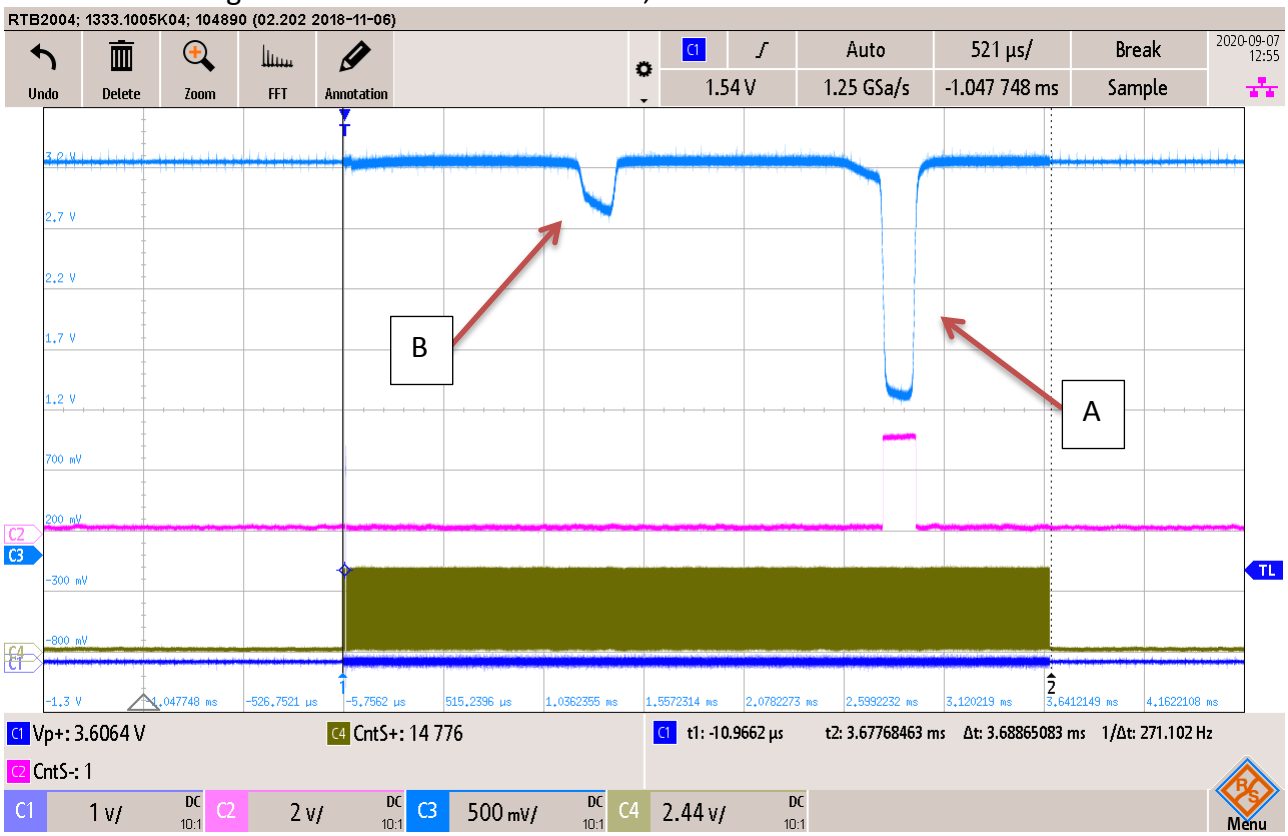
L'ouverture s'apprête à sortir de nouveau du rayon d'émission de la led L'intensité mesurée est de nouveau minime et s'apprête à redevenir théoriquement nulle.

En résumé, en faisant une représentation schématique complète du système complet, nous obtenons le résultat suivant :

Dans ce cas-ci, les cellules du CCD sont exposées à la lumière directe. L'intensité lumineuse n'est pas répartie uniformément sur la longueur du capteur CCD.

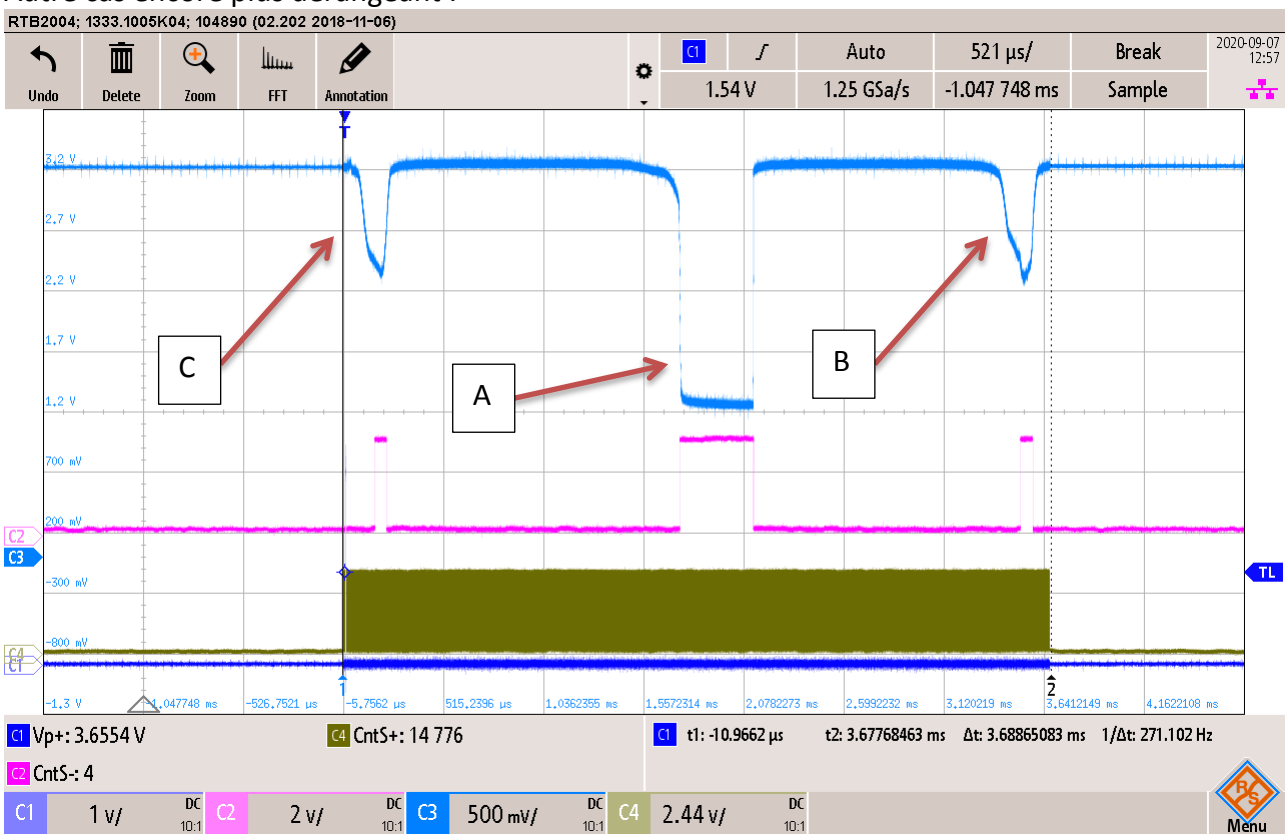


En mesurant le signal du CCD dans ces conditions, nous obtenons le résultat suivant :



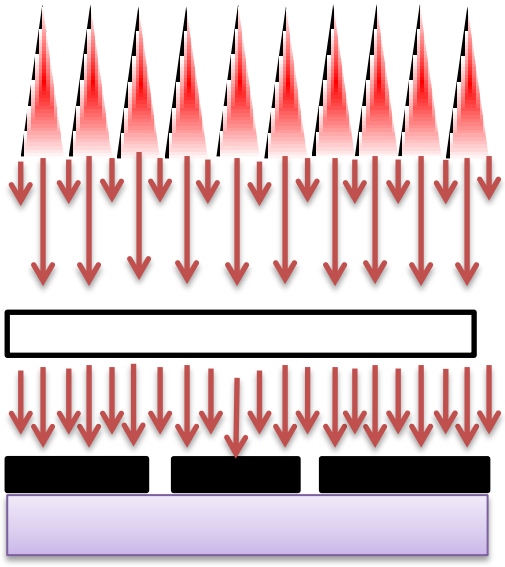
Nous voyons sur le signal du CCD (C3) qu'il peut y avoir un déséquilibre entre l'intensité mesurée entre une ouverture située directement sous le champ plein d'une led (A) et une autre située entre deux intervalles (B). Dans ces conditions, il n'est pas possible de pouvoir récupérer les flancs à partir du trigger de Schmitt car le niveau de commutation n'est pas fixe. Il faudrait pouvoir mesurer de manière analogique le signal et effectuer du processing numérique afin de détecter les flancs de transitions.

Autre cas encore plus dérangeant :



Ici nous pouvoir voir qu'une des ouvertures peut être totalement saturée (B) alors que d'autres ne sont même pas en pleine intensité (B et C).

Afin de ne pas recourir à l'utilisation d'un traitement analogique-numérique qui encombrerait des cycles machines et du temps de développement, la solution serait de placer un diffuseur de lumière entre les leds et l'obturateur. De cette manière, une portion de l'intensité lumineuse sera réfléchie ou absorbée par le diffuseur et donc perdu. En contrepartie, il y aura moins de déséquilibre entre les amplitudes des ouvertures mesurées.



Rangée de leds

Représentation vectorielle simplifiée de l'intensité lumineuse à chaque intervalle

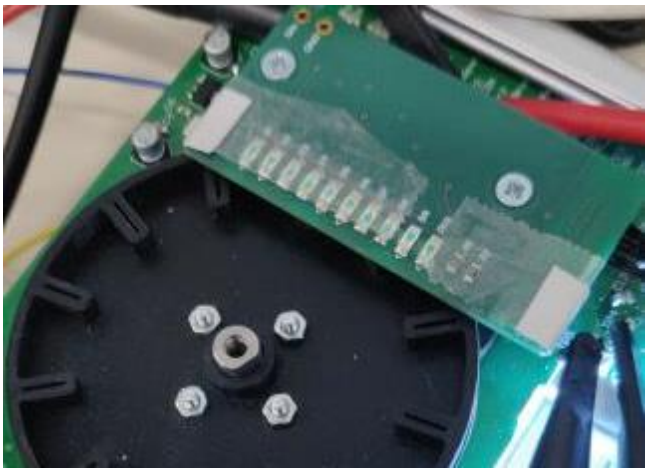
Diffuseur

Vecteurs résultants sortant du

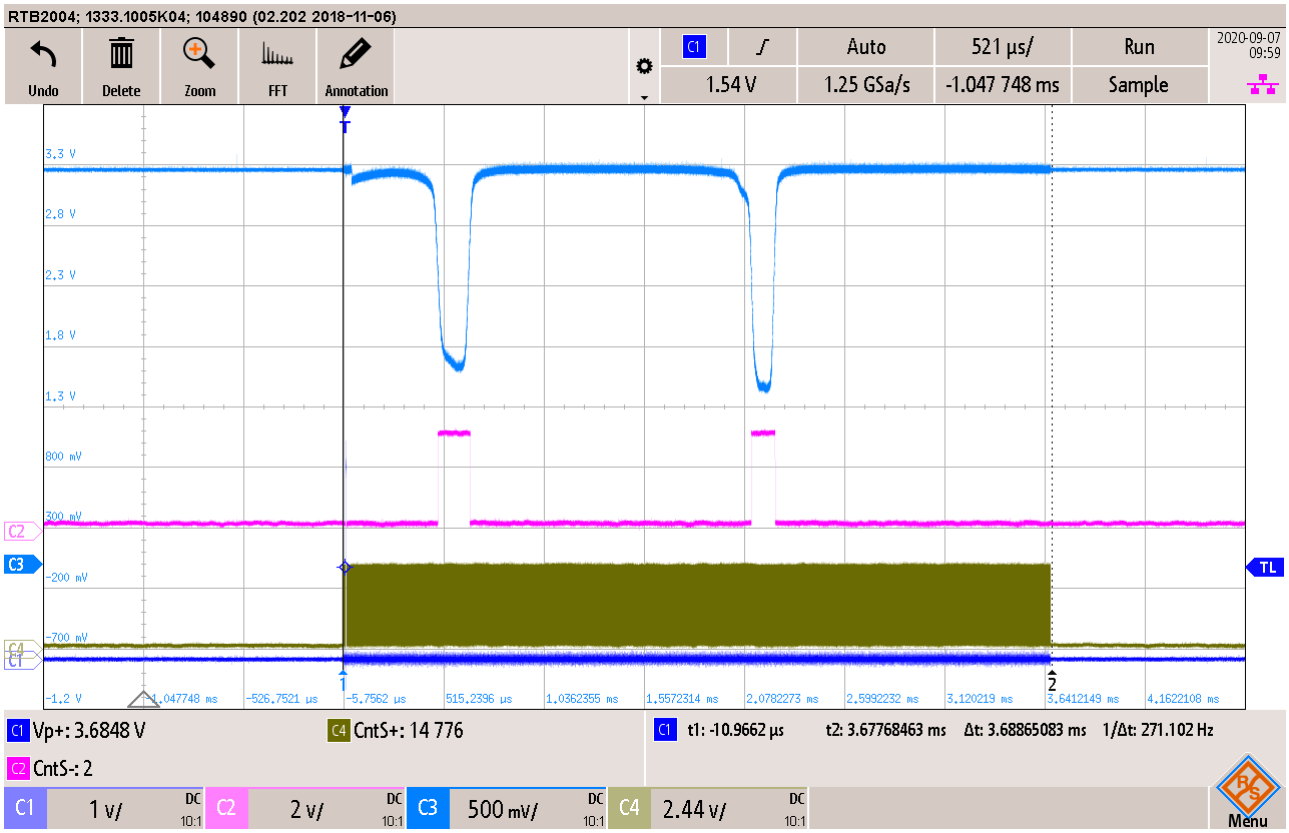
Obturateur

Capteur CCD

A des fins de tests et en attendant d'avoir une solution plus robuste, une simple feuille d'impression blanche 80g/m^2, plié en deux, a été collé à la sortie de la diffusion des leds. Comme ceci :

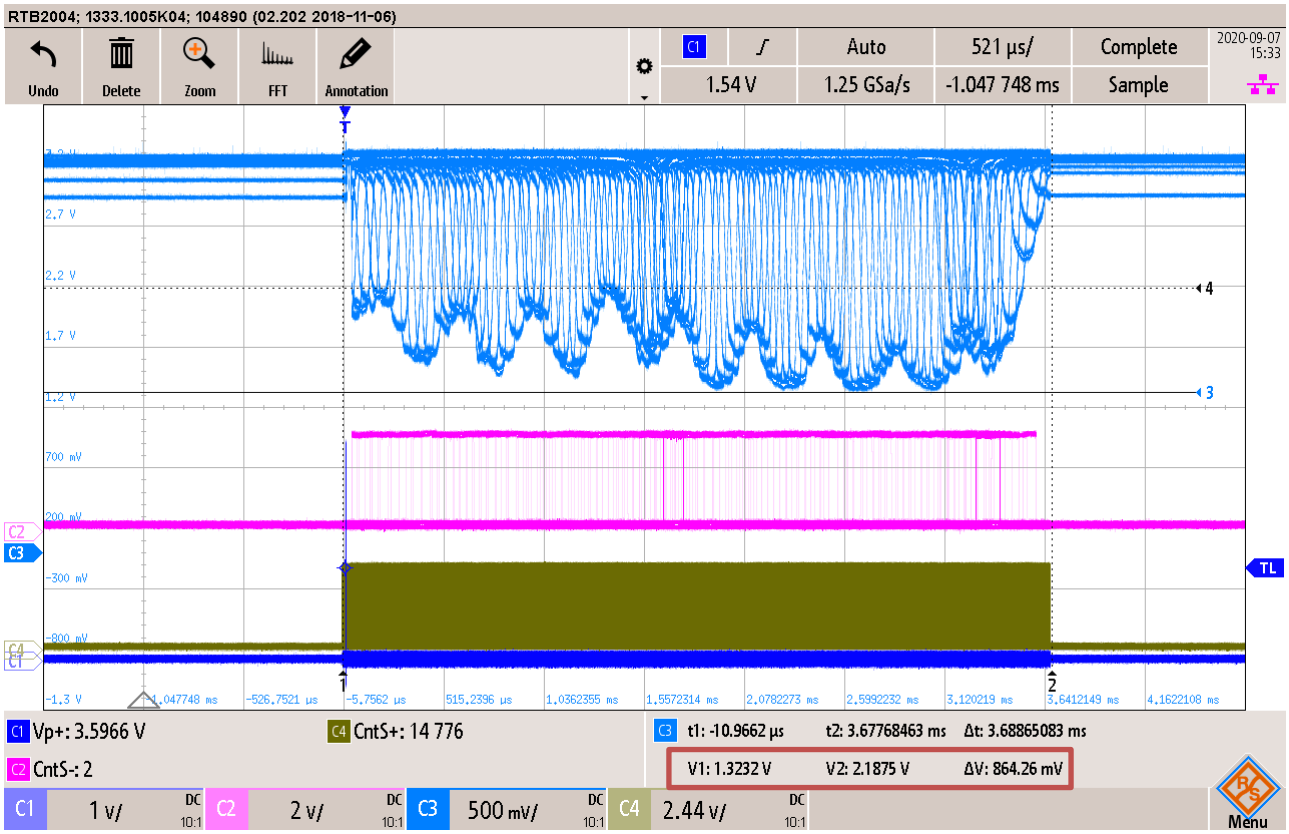


Le résultat obtenu est le suivant :



Nous voyons que nous obtenons l'effet escompté, l'amplitude de pleine luminosité ne varie pas entre deux ouvertures.

En appliquant une persistance d'affichage, nous pouvons relever la variation maximale de l'amplitude du signal mesuré sur le CCD :



L'amplitude peut donc varier de 1.32V à 2.19V ce qui fait un écart de 824mV. Cet écart est donc acceptable pour voir exploiter les informations de flancs capturés par le trigger de Schmitt.

En conclusion, il faudra prévoir à long terme un moyen de **diffuser la lumière à la sortie des leds** en faisant un compromis de chercher une variation de l'intensité quasi-nulle avec une transmittivité de la lumière suffisante par rapport à l'intensité max que peuvent fournir les leds (avoir des amplitudes facilement détectables par les triggers de Schmitt).

Ajuster l'intensité des leds IR

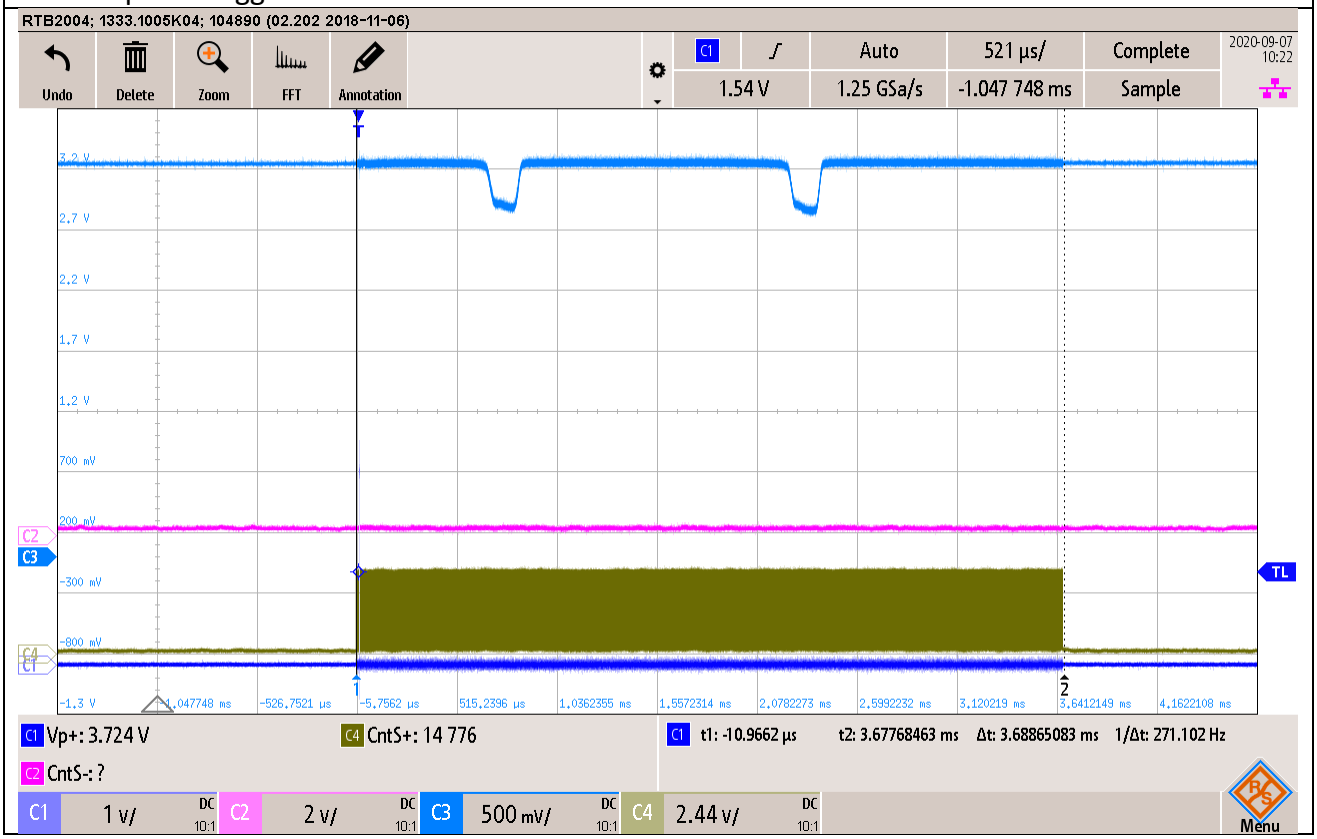
Afin de pouvoir ajuster l'intensité des leds, il faut avoir au préalable connecté le circuit à un PC sur le port série d'un terminal via le protocole du **Mode d'emploi de l'interface série** (en annexe).

Une fois le port série connecté sur un terminal type PuTTY, il faut mesurer à l'oscilloscope le signal de sortie du CCD en fonctionnement. En tapant dans le terminal la commande <IRBxxx> où xxx correspond à un nombre entre 0 et 255, il y a la possibilité de définir une intensité d'éclairage des leds entre le minimum (0) et le maximum (255).

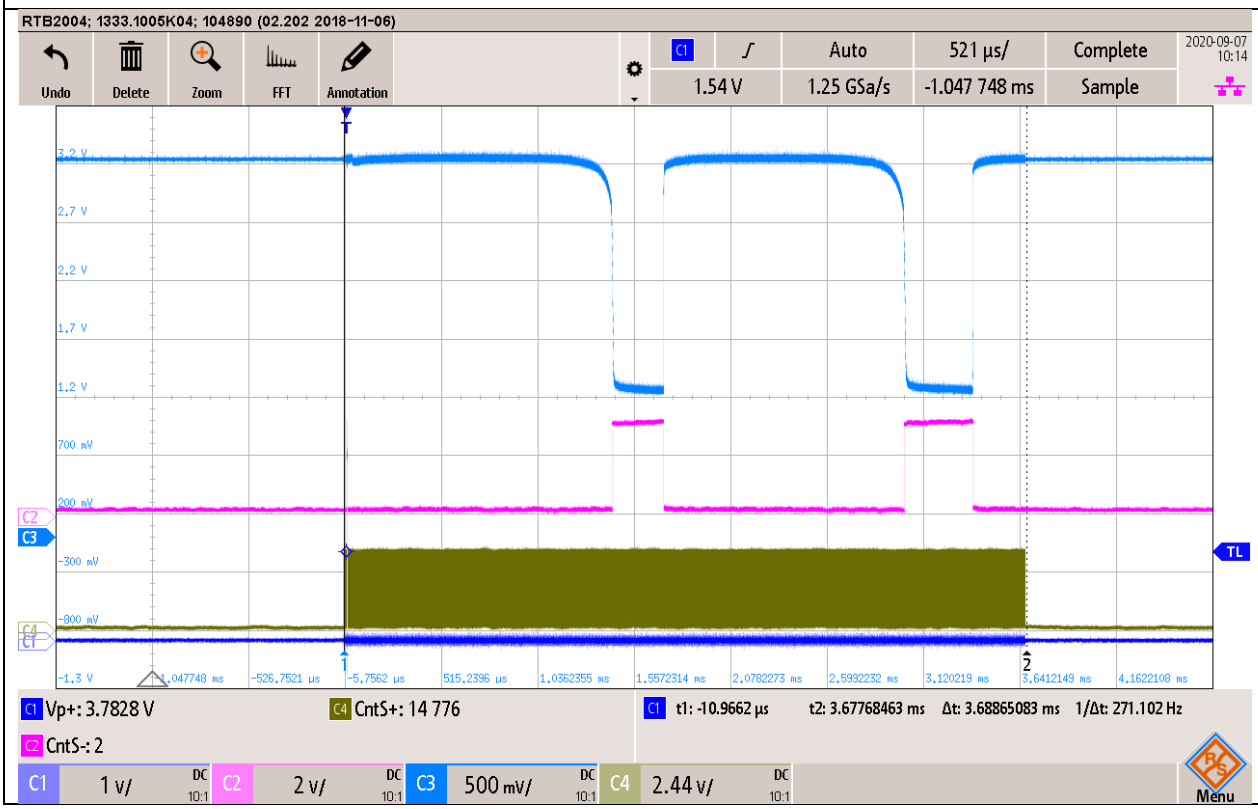
En observant le signal de sortie, il faut ajuster l'intensité en faisant le compromis entre ne pas saturer les cellules du CCD et avoir une amplitude suffisante pouvant être détecté par le trigger de Schmitt.

Exemples :

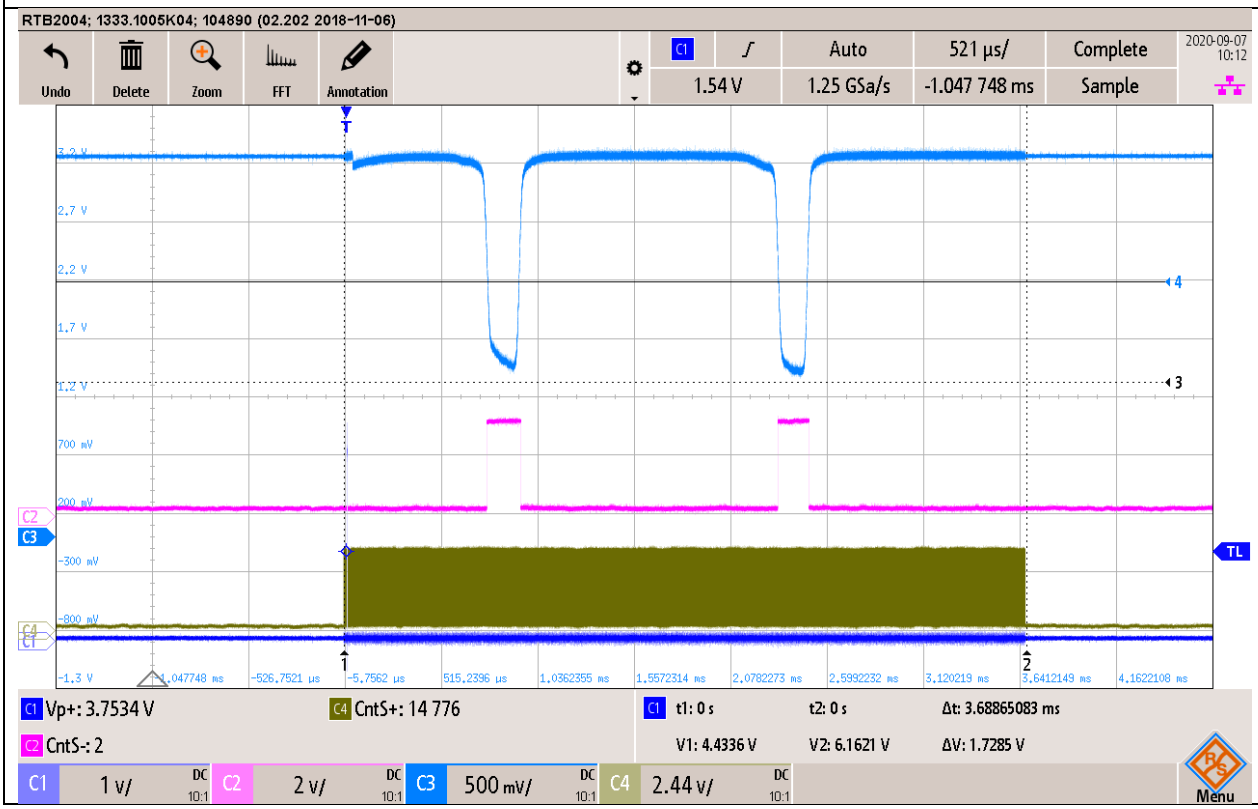
Ici, la commande tapée était <IRB20> avec un diffuseur. Le signal est trop faible pour pouvoir être détecté par le trigger de Schmitt.



Sur cette capture, il est possible de voir que le CCD est saturé. C'est à dire que nous pouvons voir que la durée des traces se prolongent, ce qui amène à retarder les flancs et par conséquent fausser la mesure.



Ici, un exemple de réglage correct, l'amplitude exploitée est maximum avant d'atteindre le seuil de saturation du CCD.



Ajuster le niveau du trigger de schmitt réglable

Afin de pouvoir ajuster le niveau de commutation du trigger de Schmitt, il faut avoir au préalable connecté le circuit à un PC sur le port série d'un terminal via le protocole du **Mode d'emploi de l'interface série** (en annexe).

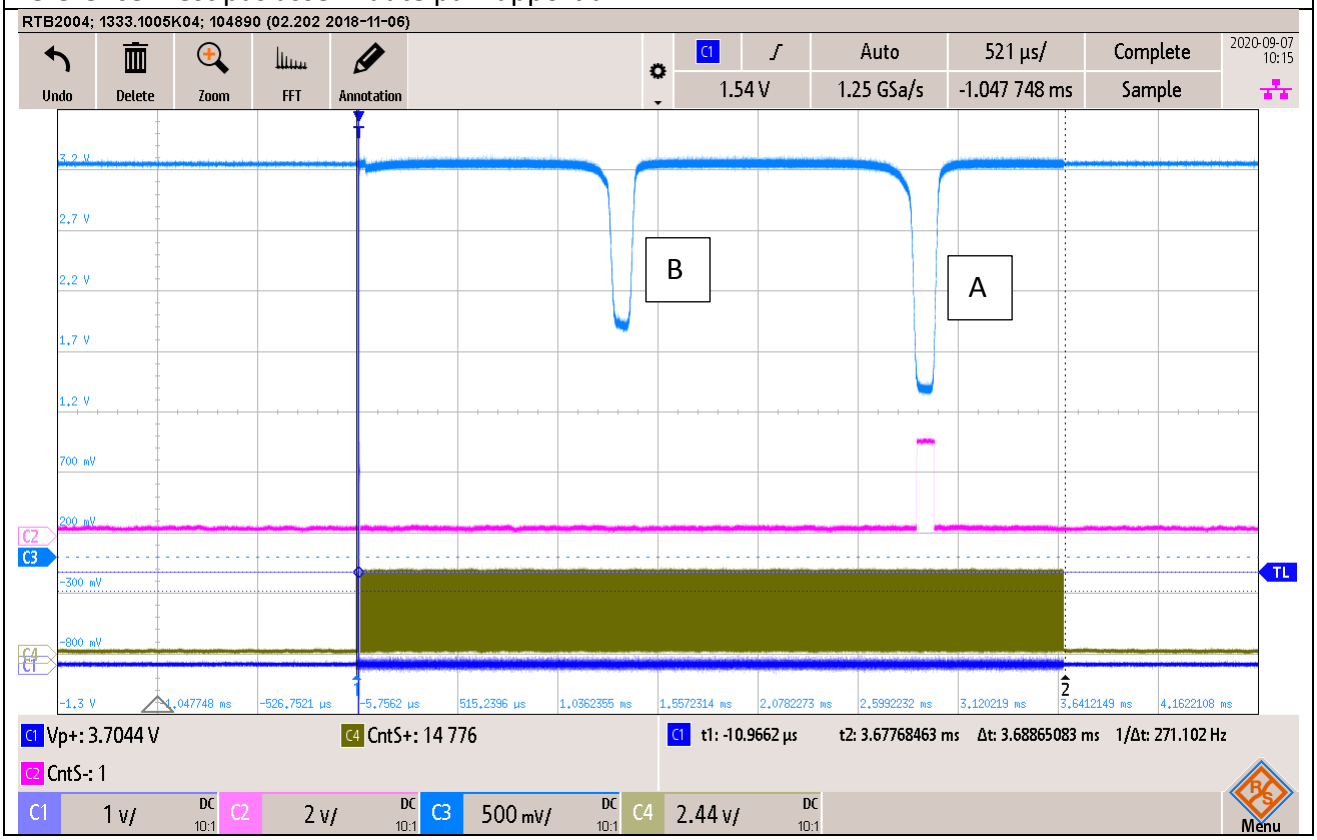
Une fois le port série connecté sur un terminal type PuTTY, il faut mesurer à l'oscilloscope le signal de sortie du CCD en fonctionnement. En tapant dans le terminal la commande <CCDxxxx> où xxx correspond à un nombre entre 0 et 4095, il y a la possibilité de définir un seuil de commutation entre environ 0V (0) et approximativement 3.3V (4095). Les bornes max et min n'ont pas lieu d'être utilisé pour une utilisation fonctionnelle.

En observant le signal de sortie, il faut ajuster le seuil de commutation du CCD en déterminant à partir de quel niveau il n'aura pas de variation de l'amplitude des traces provoqués par les ouvertures de l'obturateur. Ajuster la valeur en mesurant en parallèle la tension sur le point de test TP17 (sortie analogique du DAC).

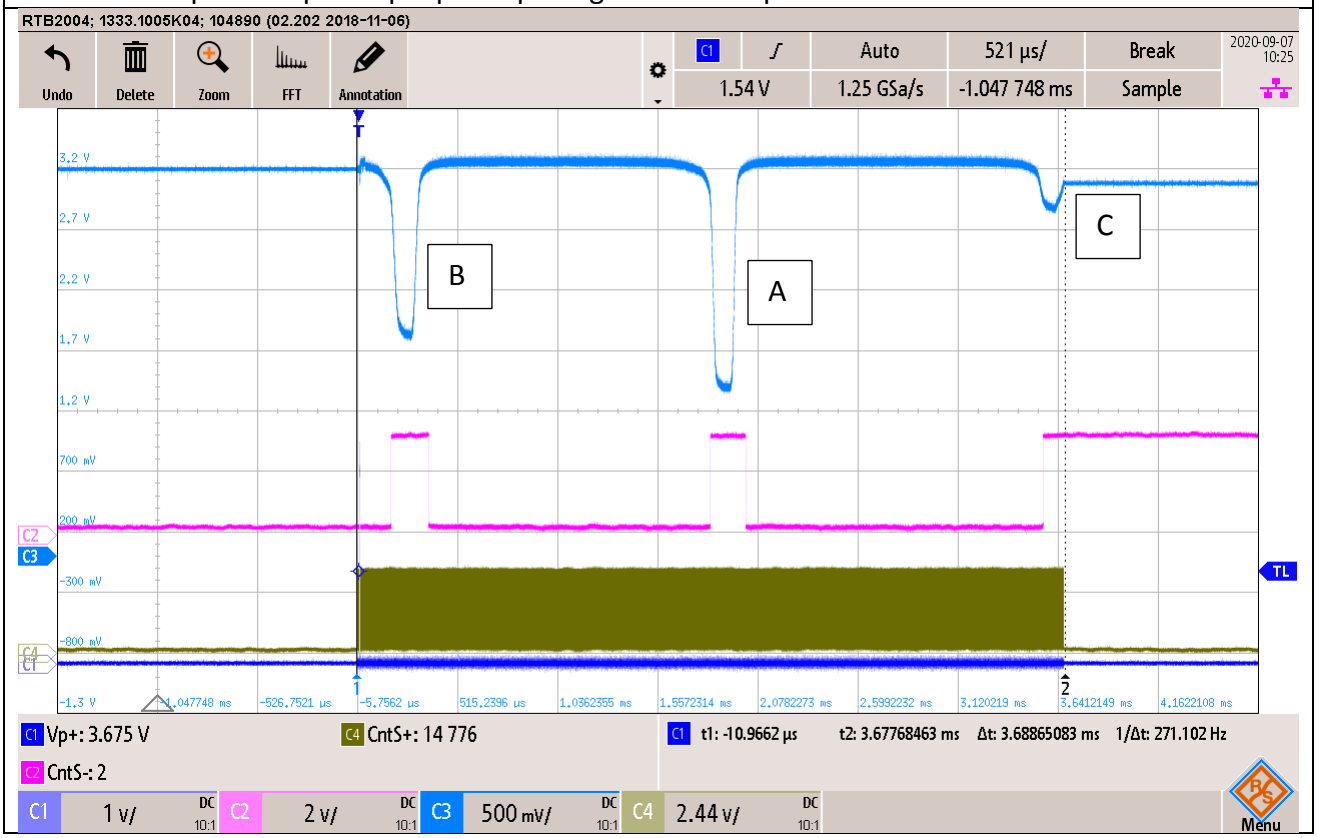
Par exemple, si à l'oscilloscope on relève que le CCD ne fournit pas des traces avec des niveaux au-dessus de 1.5V, il pourrait être intéressant de définir un seuil avec une marge. Par exemple entre 0.2V et 0.4V de plus soit 1.9V. Le plus haut possible serait idéal. Tout en tenant compte du fait que quand le signal du CCD remonte à l'état « sombre », le niveau de tension à tendance à rebondir. Il faut donc trouver le bon compromis.

Exemples :

Ici, avec la sortie du trigger (C2), nous voyons qu'il a pu relever l'ouverture A mais pas la B car la référence n'est pas assez haute par rapport à B.



Dans ce cas, le niveau du trigger est trop haut. Il relève bien les deux traces A et B mais détecte aussi le flanc parasite provoqué par le passage hors champs de l'ouverture C



Cette fois, le niveau est correctement réglé, il détecte correctement les deux traces et ne capte pas de flancs indésirables.

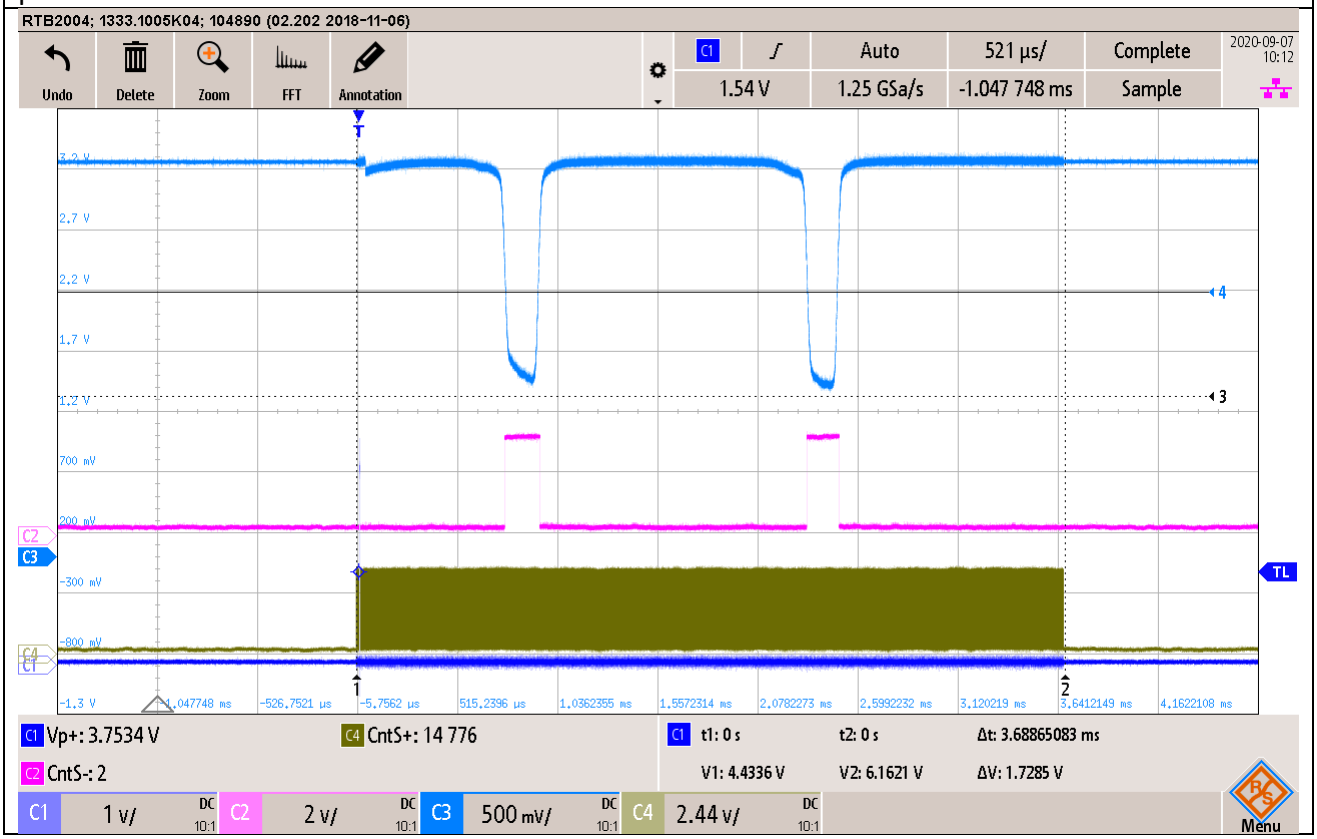
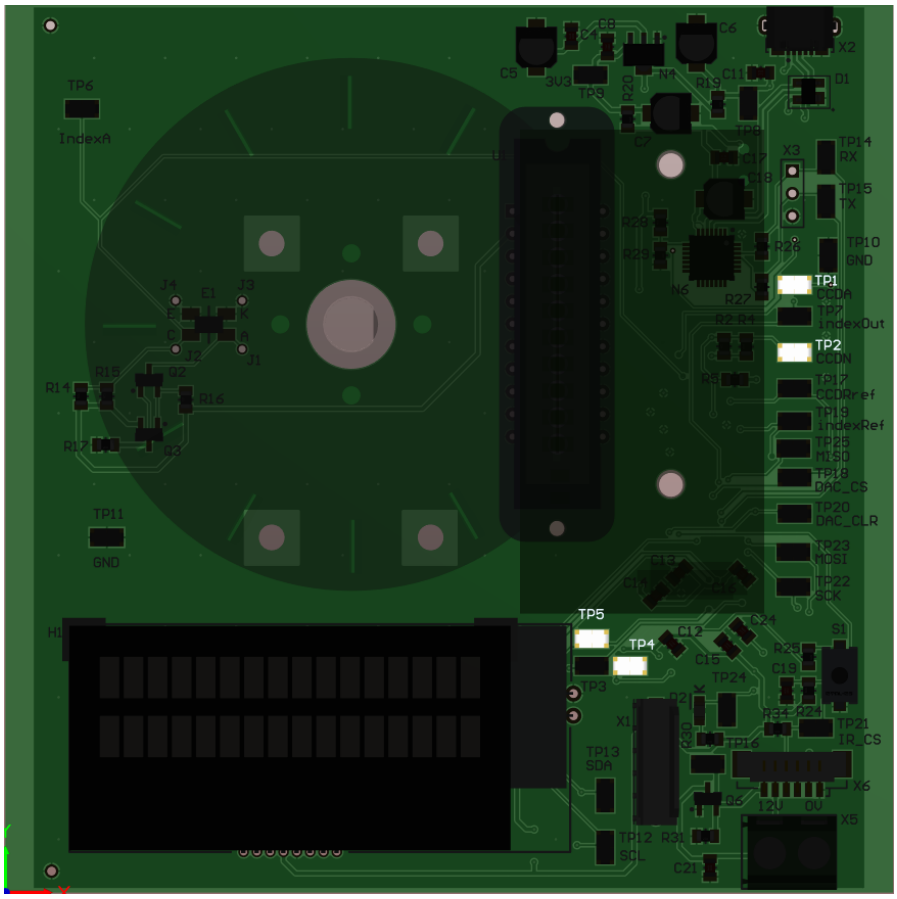
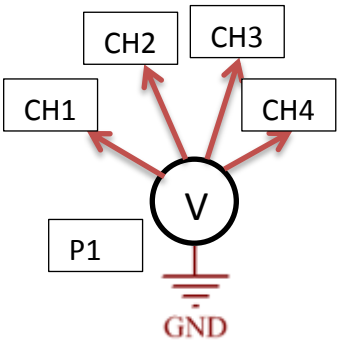


Schéma de mesure





CH1	CH2	CH3	CH4
TP5	TP2	TP1	TP4
Temps d'intégration du CCD	Sortie détection de flanc du CCD	Sortie analogique du CCD	Entrée clock du CCD

Liste du matériel

Désignation	Type	Modèle	Marque	N°ETML-ES
P1	Oscilloscope 4x2.5GS/s	RTB2004	Rhode&Schwarz	ES.SLO2.05.01.08
P2	Multimètre	GDM-396	GwInstek	ES.SLO2.00.00.90
G1	Alimentation DC de laboratoire 0-30V/0-3A	GPS-3303	GwInstek	ES.SLO2.00.00.40

6 Conclusion

Ce travail consistant essentiellement en une analyse sur la fiabilité de l'utilisation d'un capteur CCD dans la mesure de déplacement angulaire, il est intéressant de relever les propriétés observées de cette solution. Premièrement, il a été possible de détecter avec précision les flancs provoqués par les transitions ombre-lumière et de les exploiter pour calculer la différence entre chaque échantillonnage. D'un autre côté, avec un échantillonnage entre 10 et 20Hz, il y un écart type entre 15 et 40 pixels, ce qui laisse peu de place à la précision du calcul de la vitesse angulaire. Idéalement, il faudrait pouvoir augmenter la résolution du capteur CCD. C'est-à-dire augmenter le nombre de cellules de mesure du CCD. Un autre compromis serait de diminuer la fréquence d'échantillonnage en-dessous de 10Hz afin d'avoir un écart de pixels plus conséquent entre mesures et ainsi avoir un échantillonnage plus précis.

Des problèmes ont été également relevés lors de la mesure de la vitesse. La valeur de la vitesse affichée a une excursion entre environ 5 et 7°/s. Dans a version du firmware actuelle, nous faisons déjà la mesure des deux variations de flancs et ainsi faire la moyenne. Une solution serait éventuellement d'ajouter un deuxième capteur de l'autre côté du disque afin de faire une moyenne supplémentaire des valeurs mesurées entre les deux capteurs et ainsi encore doubler la précision de la valeur mesurée.

Lors de ce travail, j'ai été confronté à différents problèmes :

- Dû à la mécanique mise en place, la surface de placement et de layout des composants était limitée à une petite portion de la surface totale du PCB.
- Une pastille du footprint du µC a été décollée durant le montage de ce dernier car elle n'était reliée physiquement et électriquement à rien. L'apport de chaleur du fer à braser était trop important.
- Au niveau software, selon certaines fonctions du driver utilisé pour piloter l'écran LCD, le uC à tendance à planter. Il y a peut-être des ajustements à faire au niveau de la couche I^2C de l'ETML-ES.
- Dans un premier temps, les cellules du CCD étaient exposées directement à la lumière des LEDs de la board. Le rayon d'éclairage des leds étant sous formes de spots, la lumière n'est pas répartie uniformément sur la longueur du capteur. Il a donc fallu placer un diffuseur sur les leds pour mieux répartir le rayonnement sur les cellules. Il a également fallu augmenter l'intensité lumineuse des leds. (Voir section mesure>effet spot des leds).
- J'ai rencontré certaines difficultés de compréhension lors de la réalisation de la partie du code qui s'occupe d'acquérir les données du capteur CCD et de faire les calculs nécessaires permettant d'obtenir les mesures souhaitées.

Pour la suite de ce travail, il faudrait faire des essais afin de trouver un bon moyen de répartir la luminosité des leds IR sur toute la longueur du capteur CCD. Également essayer de trouver des leds IR qui ont un rayon de diffusion plus large. Refaire des essais en couplant mécaniquement un système de mesure de précision telle qu'un encodeur précis avec suffisamment de pas afin de déterminer si le problème provient du moteur ou du système en lui-même. Eventuellement trouver un moyen d'adapter mécaniquement le système pour pouvoir faire des essais sur d'autres types de moteurs.

Ce travail m'a fait découvrir les capteurs CCD et leurs principes de fonctionnement. Par la suite, je serais en mesure de pouvoir en intégrer plus facilement dans un projet qui en requerrait un. J'ai pu me rendre compte que l'utilisation d'un capteur CCD pouvait être intéressante pour des systèmes de mesure optique précise.

Lausanne, le 14 septembre 2020

Erwann Chancerel

7 Annexes

Annexe : Cahier des charges

>2003_CdC_CapteurMvtCcd_Phyd.pdf

Annexe : Schéma électrique

> 2003_CapteurMvtCcd-SchemaElectrique.PDF

Annexe : BOM

> 2003_CapteurMvtCcd-BOM.pdf

Annexe : Mode d'emploi

>2003_CapteurMvtCcd-ModeEmploi.pdf

Annexe : Journal de travail

> logbook_proj_erwannchancerel.pdf

Annexe : Planning

> planning_proj_erwannchancerel.pdf

Annexe : Listing du code

>2003_listing_code.pdf

Annexe : Procès-verbaux

Datasheets

Les datasheets sont mis à disposition sur le répertoire officiel du projet sur le disque réseau K => Projet 2003. Ceci afin d'éviter des impressions superflues.