



Système embarqué & Robotique - Hardware

Année 2019/2020 – Master 2

Auteurs

*David Chauvet*

*Samuel Huet*

Encadrants

*Cyril Sagonero*

*Lucas Testa*

**Rapport de projet**

Conception d’une carte d’évaluation STM32

[Introduction 5](#_Toc29326771)

[Schémas fonctionnel 6](#_Toc29326772)

[Niveau 1 6](#_Toc29326773)

[Fonctions 7](#_Toc29326774)

[Signaux 7](#_Toc29326775)

[Niveau 2 8](#_Toc29326776)

[Contrôler 8](#_Toc29326777)

[Communiquer via Bluetooth 9](#_Toc29326778)

[Communiquer via Ethernet 11](#_Toc29326779)

[Communiquer via USB 12](#_Toc29326780)

[Communiquer avec les modules MikroElectronica 13](#_Toc29326781)

[Communiquer avec une caméra 14](#_Toc29326782)

[Alimenter 14](#_Toc29326783)

[Programmer 15](#_Toc29326784)

[Analyse structurel 17](#_Toc29326785)

[Lan 17](#_Toc29326786)

[Fonctions 17](#_Toc29326787)

[Phy et Environnement 18](#_Toc29326788)

[Connecteur 18](#_Toc29326789)

[Filtrage 18](#_Toc29326790)

[Reset 18](#_Toc29326791)

[Horloge 19](#_Toc29326792)

[Protection 19](#_Toc29326793)

[Wan 20](#_Toc29326794)

[Fonctions 21](#_Toc29326795)

[Filtrage 21](#_Toc29326796)

[Connection 21](#_Toc29326797)

[Programmation 21](#_Toc29326798)

[Reset 21](#_Toc29326799)

[Cadencement 21](#_Toc29326800)

[Adaptation 22](#_Toc29326801)

[Interface 22](#_Toc29326802)

[Fonctions 23](#_Toc29326803)

[Protection 23](#_Toc29326804)

[Horloge 23](#_Toc29326805)

[USB & Alimentation 24](#_Toc29326806)

[Fonctions 25](#_Toc29326807)

[USB 25](#_Toc29326808)

[Alimentations 25](#_Toc29326809)

[Led 26](#_Toc29326810)

[Debug 27](#_Toc29326811)

[Fonctions 28](#_Toc29326812)

[Configuration 29](#_Toc29326813)

[Connection 29](#_Toc29326814)

[Reset 29](#_Toc29326815)

[Multiplexeur 29](#_Toc29326816)

[Micro 29](#_Toc29326817)

[Fonctions 31](#_Toc29326818)

[Cube mx 31](#_Toc29326819)

[Nand Flash 32](#_Toc29326820)

[Programmation 32](#_Toc29326821)

[Cadencement 33](#_Toc29326822)

[Reset 33](#_Toc29326823)

[Filtrage 34](#_Toc29326824)

[SRAM 34](#_Toc29326825)

[Routage 34](#_Toc29326826)

[Stack-up 34](#_Toc29326827)

[Règles 35](#_Toc29326828)

[DCMI 35](#_Toc29326829)

[Ethernet 35](#_Toc29326830)

[Pair différentiels 35](#_Toc29326831)

[QSPI 36](#_Toc29326832)

[RF 36](#_Toc29326833)

[Alimentation 37](#_Toc29326834)

[Encombrement 37](#_Toc29326835)

[Placement 38](#_Toc29326836)

[Top layer 38](#_Toc29326837)

[Alimentation 39](#_Toc29326838)

[Masse 39](#_Toc29326839)

[Bottom Layer 40](#_Toc29326840)

[Ajouts 41](#_Toc29326841)

[Conclusion 41](#_Toc29326842)

[Chauvet David 41](#_Toc29326843)

[Huet Samuel 41](#_Toc29326844)

[Annexes 42](#_Toc29326845)

[GitHub 42](#_Toc29326846)

[Sources 42](#_Toc29326847)

[Figure 1 - Diagramme fonctionnel de niveau 1 6](#_Toc29474676)

[Figure 2 - Diagramme fonctionnel niveau 2 : Contrôler 8](#_Toc29474677)

[Figure 3 - Diagramme fonctionnel niveau 2 : Bluetooth 9](#_Toc29474678)

[Figure 4 - Diagramme fonctionnel niveau 2 : Ethernet 11](#_Toc29474679)

[Figure 5 - Diagramme fonctionnel niveau 2 : USB 12](#_Toc29474680)

[Figure 6 - Diagramme fonctionnel niveau 2 : MikroElectronica 13](#_Toc29474681)

[Figure 7 - Diagramme fonctionnel niveau 2 : Caméra 14](#_Toc29474682)

[Figure 8 - Diagramme fonctionnel niveau 2 : Alimentation 14](#_Toc29474683)

[Figure 9 - Diagramme fonctionnel niveau 2 : Programmateur 15](#_Toc29474684)

[Figure 10 - Diagramme structurel : LAN 17](#_Toc29474685)

[Figure 11 - Reset : LAN 18](file:///C:\Users\Cocasticox\Documents\Projets\Devboard\Rapport\Rapport.docx#_Toc29474686)

[Figure 12 - Diagramme structurel : WAN 20](#_Toc29474687)

[Figure 13 - Diagramme structurel : Interface 22](#_Toc29474688)

[Figure 14 - Diagramme structurel : USB/Alimentation 24](#_Toc29474689)

[Figure 15 - Application typique buck/boost 26](file:///C:\Users\Cocasticox\Documents\Projets\Devboard\Rapport\Rapport.docx#_Toc29474690)

[Figure 16 - Diagramme structurel : Debug 27](#_Toc29474691)

[Figure 17 Diagramme structurel : Debug USB 27](#_Toc29474692)

[Figure 18 - Diagramme structurel Debug Multiplex 28](#_Toc29474693)

[Figure 19 - Diagramme structurel : Nand Flash 29](#_Toc29474694)

[Figure 20 - Diagramme structurel : Clock 30](#_Toc29474695)

[Figure 21 - Diagramme structurel : Micro 30](#_Toc29474696)

[Figure 22 - Diagramme structurel Micro BLE 30](#_Toc29474697)

[Figure 23 - Fonctions CubeMX 31](#_Toc29474698)

[Figure 24 - Pinout CubeMX 32](#_Toc29474699)

[Figure 25 - Clock CubeMX 33](#_Toc29474700)

[Figure 26 - Filtrage Micro 34](file:///C:\Users\Cocasticox\Documents\Projets\Devboard\Rapport\Rapport.docx#_Toc29474701)

[Figure 27 - Stack-up 34](file:///C:\Users\Cocasticox\Documents\Projets\Devboard\Rapport\Rapport.docx#_Toc29474702)

[Figure 28 - Paire différentiel Ethernet 35](file:///C:\Users\Cocasticox\Documents\Projets\Devboard\Rapport\Rapport.docx#_Toc29474703)

[Figure 29 - Paire différentiel USB 36](file:///C:\Users\Cocasticox\Documents\Projets\Devboard\Rapport\Rapport.docx#_Toc29474704)

[Figure 30 - Adaptation RF 36](file:///C:\Users\Cocasticox\Documents\Projets\Devboard\Rapport\Rapport.docx#_Toc29474705)

[Figure 31 - Piste d'alimentation externe 37](#_Toc29474706)

[Figure 32 - Piste d'alimentation interne 37](#_Toc29474707)

[Figure 33 - Top Overlay 1:1 38](#_Toc29474708)

[Figure 34 - Top Layer 1:1 38](#_Toc29474709)

[Figure 35 - Couche d'alimentation 1:1 39](#_Toc29474710)

[Figure 36 - Couche de masse 1:1 39](#_Toc29474711)

[Figure 37 - Bottom Overlay 1:1 40](#_Toc29474712)

[Figure 38 - Bottom Layer 1:1 40](#_Toc29474713)

[Figure 39 - GitHub 42](file:///C:\Users\Cocasticox\Documents\Projets\Devboard\Rapport\Rapport.docx#_Toc29474714)

[Tableau 1 - Fonctions principales 7](#_Toc29474650)

[Tableau 2 - Signaux principaux 7](#_Toc29474651)

[Tableau 3 - Fonctions secondaire : Contrôler 8](#_Toc29474652)

[Tableau 4 - Signaux secondaire : Contrôler 9](#_Toc29474653)

[Tableau 5 - Fonctions secondaires : Bluetooth 10](#_Toc29474654)

[Tableau 6 - Signaux secondaire : Bluetooth 10](#_Toc29474655)

[Tableau 7 - Fonctions secondaires : Ethernet 11](#_Toc29474656)

[Tableau 8 - Signaux secondaires : Ethernet 12](#_Toc29474657)

[Tableau 9 - Fonctions secondaires : USB 12](#_Toc29474658)

[Tableau 10 - Signaux secondaires : USB 13](#_Toc29474659)

[Tableau 11 - Fonctions secondaires : MikroElectronica 13](#_Toc29474660)

[Tableau 12 - Signaux secondaires : MikroElectronica 13](#_Toc29474661)

[Tableau 13 - Fonctions secondaires : Caméra 14](#_Toc29474662)

[Tableau 14 - Signaux secondaires : Caméra 14](#_Toc29474663)

[Tableau 15 - Fonctions secondaires : Alimentation 15](#_Toc29474664)

[Tableau 16 - Signaux secondaires : Alimentation 15](#_Toc29474665)

[Tableau 17 - Fonctions secondaires : Programmateur 16](#_Toc29474666)

[Tableau 18 - Signaux secondaires : Programmateur 16](#_Toc29474667)

[Tableau 19 - Fonctions structurelles : LAN 17](#_Toc29474668)

[Tableau 20 - Filtrage : LAN 18](#_Toc29474669)

[Tableau 21 - Fonctions structurelles : WAN 21](#_Toc29474670)

[Tableau 22 - Fonctions structurelles : Interface 23](#_Toc29474671)

[Tableau 23 - Fonctions structurelles : USB/Alimentation 25](#_Toc29474672)

[Tableau 24 - Consommation 25](#_Toc29474673)

[Tableau 25 - Fonctions structurelles : Debug 28](#_Toc29474674)

[Tableau 26 - Fonctions structurelles : Micro 31](#_Toc29474675)

# Introduction

Dans le cadre de notre cours de conception électronique avec Cyril Sagonero et Lucas Testa, il nous a été demandé de réaliser une carte de développement STM32. Ce projet a démarré le 14 novembre 2019 et se termine le 10 Janvier. L’objectif est de nous faire concevoir de A à Z une carte de développement STM32 répondant du mieux possible aux normes CEM. Les tests de validité de la carte ne seront pas effectués mais ce projet nous responsabilisera à ces normes.

Afin de nous faire gagner du temps certains composants nous ont étés imposés, on peut donc établir une liste des connectivités imposés.

* Microcontrôleur (Composant : STM32H750VBT6)
* Ethernet (Composant : KSZ8081RNA)
* Flash NAND (Composant : W25N01GVZEIT)
* Bluetooth (Composant : BLUENRG-232)
* Port Caméra (Module : OV7670)

Il nous a également été imposé de respecter une dimension de la carte de 8cm par 8cm. De plus, en bonus, nous avons le choix d’ajouter ou non une SRAM. Nous avons décidé par nous-même, que dans le cadre une carte de développement, ajouter un programmateur interne serait judicieux. Cela permettrait d’avoir tout en un. Nous ajoutons donc le programmateur interne à notre cahier des charges.

# Schémas fonctionnel

## Niveau 1

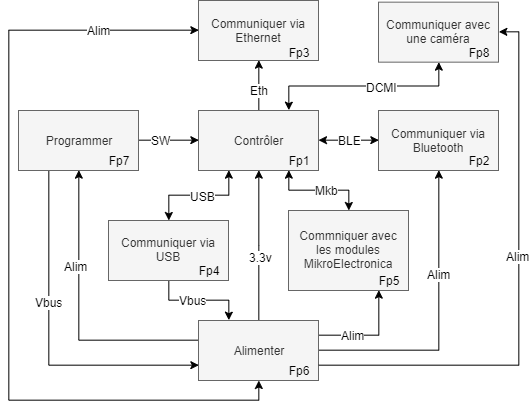


Figure 1 - Diagramme fonctionnel de niveau 1

### Fonctions

|  |  |  |
| --- | --- | --- |
| Numéro | Nom | Description |
| Fp1 | Contrôler | Assurer la coordination entre les modules via un algorithme. |
| Fp2 | Communiquer via Bluetooth | Assurer une communication Bluetooth avec un autre appareil. Permettre de plus la programmation de la fonction principale 1 |
| Fp3 | Communiquer via Ethernet | Assurer une communication via la protocole Ethernet |
| Fp4 | Communiquer via USB | Assurer une communication via le protocole USB |
| Fp5 | Communiquer avec les modules MikroElectronica | Assurer une compatibilité avec les modules MikroElectronica via le socket Mikrobus |
| Fp6 | Alimenter | Assurer l’alimentation des différentes fonctions. |
| Fp7 | Programmer | Permettre la programmation et le Debug de l’algorithme envoyé au la fonction principale 1 |
| Fp8 | Communiquer avec une caméra | Etablir le paramétrage et l’acquisition des données d’une caméra |

Tableau 1 - Fonctions principales

### Signaux

|  |  |  |
| --- | --- | --- |
| Nom | Type | Description |
| Vbus | Signal continue | Alimentation brute non filtrée |
| Alim | Signal continue | Alimentation régulé et filtrée |
| USB | Signal numérique différentiel | Signal numérique de communication USB |
| Mkb | Ensemble de signaux numérique et continue | Ensemble de signaux numérique : SPI, UART, PWM, I2C, mais aussi un signal analogique, un signal d’interruption, et un de reset |
| BLE | Signal numérique | Signal numérique de communication UART |
| Eth | Signal numérique | Signal numérique de communication Ethernet |
| SW | Signal numérique | Signal numérique de programmation SW |
| DCMI | Signal numérique | Signal numérique de communication DCMI |

Tableau 2 - Signaux principaux

## Niveau 2

Les liaisons laissés en l’air représentent une connexion avec un bloque fonctionnel extérieur.

### Contrôler

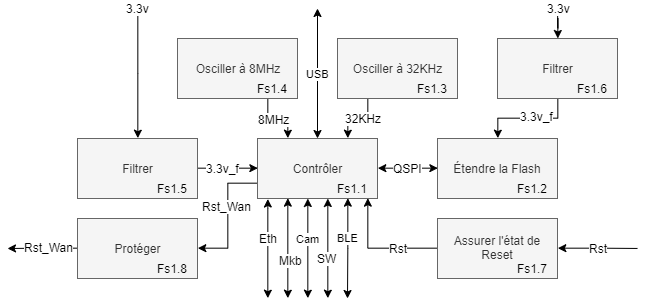


Figure 2 - Diagramme fonctionnel niveau 2 : Contrôler

#### Fonctions

|  |  |  |
| --- | --- | --- |
| Numéro | Nom | Description |
| Fs1.1 | Contrôler | Elément permettant l’exécution d’un programme |
| Fs1.2 | Etendre la Flash | Etendre l’espace de stockage Flash de la fonction Contrôler |
| Fs1.3 | Osciller à 32KHz | Cadencer à 32KHz la fonction Contrôler |
| Fs1.4 | Osciller à 8MHz | Cadencer à 8MHz la fonction Contrôler |
| Fs1.5 | Filtrer | Filtrer l’alimentation de la fonction Contrôler |
| Fs1.6 | Filtrer | Filtrer l’alimentation de la fonction Etendre la Flash |
| Fs1.7 | Assurer l’état de Reset | Maintenir l’état de reset pour une temps minimal |
| Fs1.8 | Protéger | Protéger la ligne contre un courant excessif |

Tableau 3 - Fonctions secondaire : Contrôler

#### Signaux

|  |  |  |
| --- | --- | --- |
| Nom | Type de signal | Description |
| 3.3v | Alimentation continue | Alimentation des fonctions |
| 3.3v\_f | Alimentation continue | Alimentation filtré des fonctions |
| 8MHz | Signal alternatif | Signal de cadencement |
| 32KHz | Signal alternatif | Signal de cadencement |
| USB | Signal numérique | Signal de communication USB |
| Eth | Signaux numérique | Signal de communication Ethernet |
| Mkb | Signaux numérique | Ensemble de signaux de communication Mikrobus |
| Cam | Signaux numérique | Ensemble de signaux de communication Camera |
| SW | Signaux numérique | Signal de programmation |
| BLE | Signaux numérique | Signaux de communication BLE |
| Rst | Signal tout ou rien | Signal Reset de la fonction Contrôler |
| Rst\_Wan | Signal tout ou rien | Signal de Reset de la Wan |

Tableau 4 - Signaux secondaire : Contrôler

### Communiquer via Bluetooth

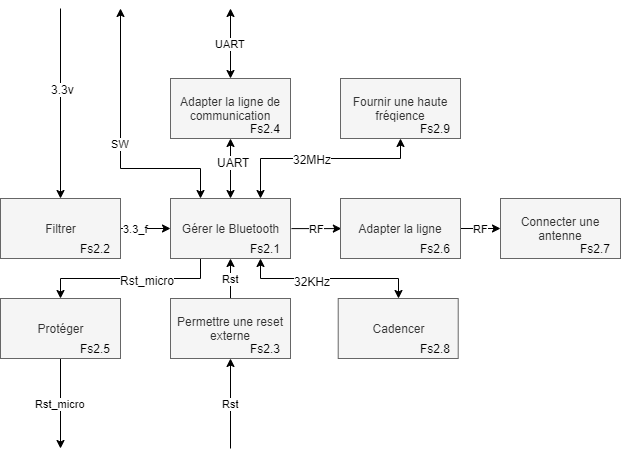


Figure 3 - Diagramme fonctionnel niveau 2 : Bluetooth

#### Fonctions

|  |  |  |
| --- | --- | --- |
| Numéro | Fonction | Description |
| Fs2.1 | Gérer le Bluetooth | Gère à la fois le protocole Bluetooth ainsi que le passage en RF |
| Fs2.2 | Filtrer | Filtrer l’alimentation |
| Fs2.3 | Permettre un reset externe | Maintenir l’état reset suffisamment longtemps pour qu’il soit pris en compte |
| Fs2.4 | Adapter la ligne de communication | Adapter la ligne de communication |
| Fs2.5 | Protéger | Protéger la ligne contre des sur courants |
| Fs2.6 | Adapter la ligne | Adapter la ligne RF sur 50Ω |
| Fs2.7 | Connecter une antenne | Connecter une antenne Bluetooth externe |
| Fs2.8 | Cadencer | Cadencer le contrôleur |
| Fs2.9 | Fournir une haute fréquence | Fournir au contrôleur une fréquence haute qu’il pourra exploiter |

Tableau 5 - Fonctions secondaires : Bluetooth

#### Signaux

|  |  |  |
| --- | --- | --- |
| Nom | Type de signal | Description |
| 3.3v | Signal d’alimentation | Signal d’alimentation 3.3v non filtré |
| 3.3\_f | Signal d’alimentation | Signal d’alimentation 3.3v filtré |
| SW | Signal numérique | Signal de programmation |
| UART | Signal numérique | Signal de communication et de programmation micro/BLE |
| 32MHz | Signal alternatif | Signal servant à générer du 2.4GHz au BLE |
| RF | Signal HF | Signal 2.4GHz modulé |
| 32KHz | Signal alternatif | Signal de cadencement du BLE |
| Rst | Signal tout ou rien | Signal de reset du BLE |
| Rst\_micro | Signal tout ou rien | Signal ordonnant au micro un reset |

Tableau 6 - Signaux secondaire : Bluetooth

### Communiquer via Ethernet

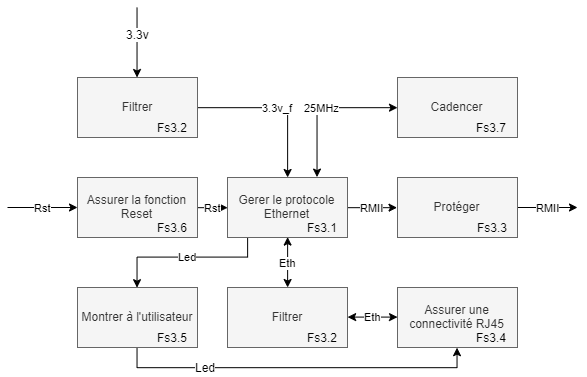


Figure 4 - Diagramme fonctionnel niveau 2 : Ethernet

#### Fonctions

|  |  |  |
| --- | --- | --- |
| Numéro | Fonction | Description |
| Fs3.1 | Gérer le protocole Ethernet | Envoyer et recevoir des trames Ethernet |
| Fs3.2 | Filtrer | Filtrer l’alimentation et éliminer tous les signaux indésirables |
| Fs3.3 | Protéger | Protéger les composants d’une éventuelle différence de potentiel |
| Fs3.4 | Assurer une connectivité RJ45 | Permettre le branchement d’un câble Ethernet RJ45 |
| Fs3.5 | Montrer à l’utilisateur | Montrer à l’utilisateur qu’une communication a bien lieu |
| Fs3.6 | Assurer la fonction Reset | Veiller à maintenir suffisamment longtemps l’état de reset |
| Fs3.7 | Cadencer | Donner une fréquence de fonctionnement au composant principale |

Tableau 7 - Fonctions secondaires : Ethernet

#### Signaux

|  |  |  |
| --- | --- | --- |
| ***Nom*** | **Type de signal** | **Description** |
| *3.3v* | Signal d’alimentation | Signal d’alimentation 3.3v non filtré |
| *3.3v\_f* | Signal d’alimentation | Signal d’alimentation 3.3v filtré |
| *25MHz* | Signal alternatif | Signal de cadencement du phy |
| *RMII* | Signal numérique | Signal de communication micro/phy |
| *Eth* | Signal numérique différentiel | Signal de communication phy/périphérique |
| *Led* | Signal tout ou rien | Signal montrant qu’une communication s’opère |
| *Rst* | Signal tout ou rien | Signal ordonnant le reset du phy |

Tableau 8 - Signaux secondaires : Ethernet

### Communiquer via USB

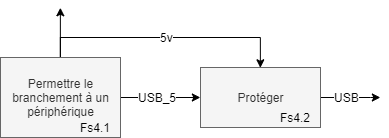


Figure 5 - Diagramme fonctionnel niveau 2 : USB

#### Fonctions

|  |  |  |
| --- | --- | --- |
| Numéro | Nom | Description |
| Fs4.1 | Permettre le branchement à un périphérique | Port standard permettant la connexion avec un Host |
| Fs4.2 | Protéger | Protéger le système d’une puissance excessive |

Tableau 9 - Fonctions secondaires : USB

#### Signaux

|  |  |  |
| --- | --- | --- |
| Nom | Type | Description |
| 5v | Signal continue | Alimentation 5v permettant l’alimentation |
| USB\_5 | Signal numérique | Signal numérique 0-5v différentiel |
| USB | Signal numérique | Signal numérique différentiel |

Tableau 10 - Signaux secondaires : USB

### Communiquer avec les modules MikroElectronica

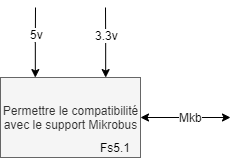


Figure 6 - Diagramme fonctionnel niveau 2 : MikroElectronica

#### Fonctions

|  |  |  |
| --- | --- | --- |
| Numéro | Nom | Description |
| Fs5.1 | Permettre la compatibilité avec le support Mikrobus | Permettre le branchement des différents modules MikroElectronica disposant d’un port mikrobus |

Tableau 11 - Fonctions secondaires : MikroElectronica

#### Signaux

|  |  |  |
| --- | --- | --- |
| Nom | Type | Description |
| 5v | Signal continue | Alimentation 5v permettant l’alimentation |
| 3 .3v | Signal continue | Alimentation 3.3v permettant l’alimentation |
| Mkb | Signal numérique | Ensemble de signaux numérique et de bus de communication synchrone et asynchrone duplex |

Tableau 12 - Signaux secondaires : MikroElectronica

### Communiquer avec une caméra

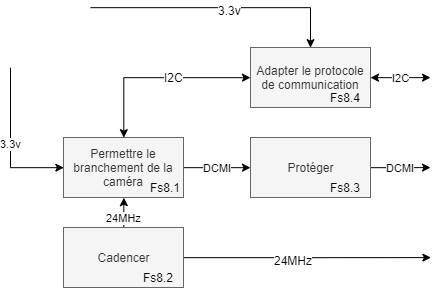


Figure 7 - Diagramme fonctionnel niveau 2 : Caméra

#### Fonctions

|  |  |  |
| --- | --- | --- |
| Numéro | Fonction | Description |
| Fs8.1 | Permettre le branchement de la caméra | Brochage permettant le branchement direct d’un module OV7670 |
| Fs8.2 | Cadencer | Cadencer la caméra et le contrôleur |
| Fs8.3 | Protéger | Protéger la caméra d’un courant excessif |
| Fs8.4 | Adapter le protocole de communication | Maintenir l’état un état haut par défaut en permanence |

Tableau 13 - Fonctions secondaires : Caméra

#### Signaux

|  |  |  |
| --- | --- | --- |
| Nom | Type de signal | Description |
| 3.3v | Signal d’alimentation | Signal d’alimentation 3.3v non filtré |
| I2C | Signal numérique | Signal I2C permettant la configuration de la caméra |
| DCMI | Signal numérique | Signal DCMI permettant l’envois de l’image capturé |
| 24MHz | Signal alternatif | Signal de cadencement partagé entre la caméra et le micro |

Tableau 14 - Signaux secondaires : Caméra

### Alimenter



Figure 8 - Diagramme fonctionnel niveau 2 : Alimentation

#### Fonctions

|  |  |  |
| --- | --- | --- |
| Numéro | Nom | Description |
| Fs6.1 | Permettre la branchement d’une alimentation Jack | Permettre la connexion d’une alimentation externe et standard |
| Fs6.2 | Stabiliser l’alimentation 5v | Convertit et stabilise la tension d’entrée en tension 5v stable |
| Fs6.3 | Stabiliser l’alimentation 3.3v | Convertit et stabilise la tension d’entrée en tension 3.3v stable |
| Fs6.4 | Montrer à l’utilisateur le bon fonctionnement de l’alimentation | Montrer à l’utilisateur, par un moyen rapide à visualiser, le bon fonctionnement de l’alimentation |

Tableau 15 - Fonctions secondaires : Alimentation

#### Signaux

|  |  |  |
| --- | --- | --- |
| Nom | Type | Description |
| 12v | Tension d’alimentation | Alimentation 12v en attente d’être convertie |
| 5v | Tension d’alimentation | Alimentation 5v dédié au Mikrobus |
| 5v\_USB | Tension d’alimentation | Alimentation 3.3v dédié à la quasi-totalité de la carte |

Tableau 16 - Signaux secondaires : Alimentation

### Programmer

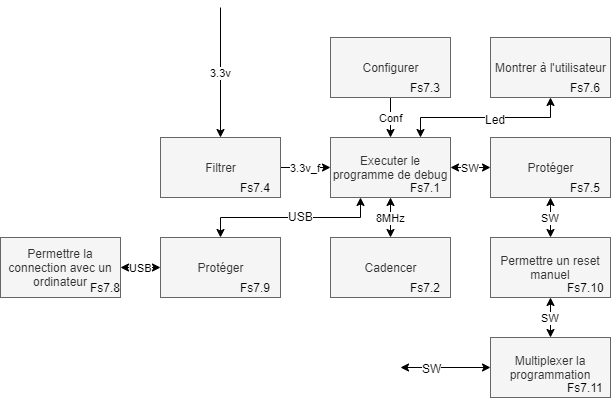


Figure 9 - Diagramme fonctionnel niveau 2 : Programmateur

#### Fonctions

|  |  |  |
| --- | --- | --- |
| Numéro | Fonction | Description |
| Fs7.1 | Exécuter le programme de Debug | Exécute le programme de Debug (STLINK) et sert d’intermédiaire entre PC et Microcontrôleur |
| Fs7.2 | Cadencer | Cadencer le debugger |
| Fs7.3 | Configurer | Configurer le debugger pour qu’il s’accorde avec le programme STLINK |
| Fs7.4 | Filtrer | Filtrer l’alimentation d’éventuels parasites |
| Fs7.5 | Protéger | Protéger le contrôleur contre un sur courant éventuel |
| Fs7.6 | Montrer à l’utilisateur | Montrer à l’utilisateur le bon fonctionnement du Debug |
| Fs7.8 | Permettre la connexion avec un ordinateur | Permettre une connexion débuggeur/ordinateur |
| Fs7.9 | Protéger | Protéger le débuggeur contre les surtensions et sur courants |
| Fs7.10 | Permettre un Reset manuel | Permettre un reset manuel du contrôleur et du module Bluetooth |
| Fs7.11 | Multiplexer la programmation | Permettre la programmation du contrôleur et du BLE via un switch |

Tableau 17 - Fonctions secondaires : Programmateur

#### Signaux

|  |  |  |
| --- | --- | --- |
| Nom | Type de signal | Description |
| 3.3v | Signal d’alimentation | Signal d’alimentation 3.3v non filtré |
| 3.3v\_f | Signal d’alimentation | Signal d’alimentation 3.3v non filtré |
| Conf | Signal tout ou rien | Signal de configuration du contrôleur Debug |
| Led | Signal tout ou rien | Signal indiquant une programmation en cours |
| SW | Signal numérique | Signal de programmation |
| 8MHz | Signal alternatif | Signal de cadencement |
| USB | Signal numérique différentiel | Signal de communication USB interfaçant débuggeur et PC |

Tableau 18 - Signaux secondaires : Programmateur

# Analyse structurel

## Lan

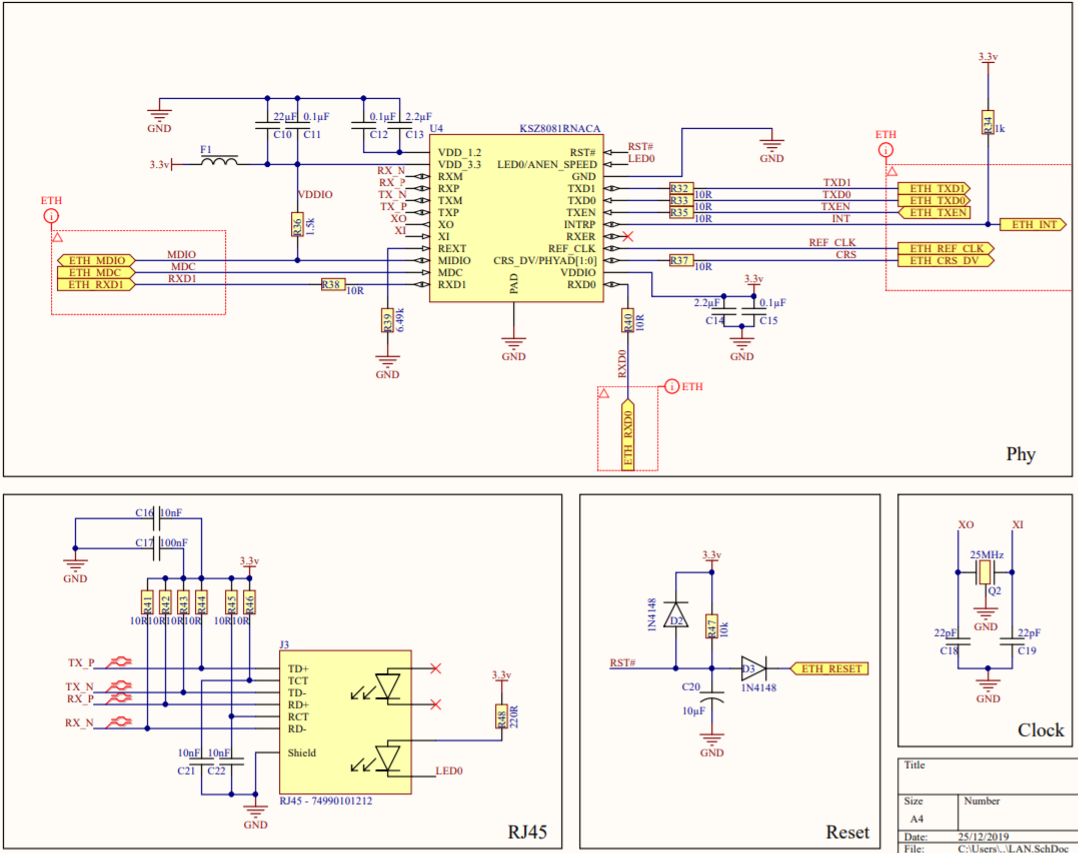


Figure 10 - Diagramme structurel : LAN

### Fonctions

|  |  |  |  |
| --- | --- | --- | --- |
| Numéro | Fonction | Description | Composants |
| Fs3.1 | Gérer le protocole Ethernet | Envoyer et recevoir des trames Ethernet | U4, R39, R36, R34 |
| Fs3.2 | Filtrer | Filtrer l’alimentation et éliminer tous les signaux indésirables | C10, C11, C12, C13, C14, C15, C16, C17, C21, C22, F1 |
| Fs3.3 | Protéger | Protéger les composants d’une éventuelle différence de potentiel | R32, R33, R35, R37, R38, R40 |
| Fs3.4 | Assurer une connectivité RJ45 | Permettre le branchement d’un câble Ethernet RJ45 | R41, R42, R43, R44, R45, R46, J3 |
| Fs3.5 | Montrer à l’utilisateur | Montrer à l’utilisateur qu’une communication a bien lieu | R48 |
| Fs3.6 | Assurer la fonction Reset | Veiller à maintenir suffisamment longtemps l’état de reset | D2, D3, R47, C20 |
| Fs3.7 | Cadencer | Donner une fréquence de fonctionnement au composant principale | Q2, C18, C19 |

Tableau 19 - Fonctions structurelles : LAN

### Phy et Environnement

Comme dit précédemment, le composant principal, à savoir le Phy Ethernet, ou PhyCeiver, nous a été imposé : il s’agit du KSZ8081RNA. Concernant donc le paramétrage de ce composant, la datasheet nous explique de connecter la broche REXT à une résistance de 6.49kΩ, celle-ci permettant au phy lui-même de déterminer son courant de sortie. Les résistances R36 et R34 sont des résistances de tirages afin d’imposer une tension haute sur la ligne, leur valeur importe peu.

### Connecteur

Le connecteur choisis est un connecteur Ethernet 8 contacts. Il intègre deux led qui permet de gagner de la place sur le montage en évitant d’ajouter une led externe. La normalisation de ce connecteur permet une compatibilité maximum avec les équipements équipé pour communiquer en Ethernet. Ce connecteur intègre également un filtrage magnétique, qui nous évite de mettre un magnétique externe qui prendrait une place très importante.

### Filtrage

Le filtrage se fait au travers de condensateur de découplage. Ainsi, en fonction du courant consommé, on pourra en déduire une résistance équivalente, et donc un filtre RC dont la fréquence de coupure se calcule par : . Le condensateur n’étant pas un composant parfait, on placera en général un deuxième condensateur environ à une décade au-dessus pour filtrer les fréquences plus hautes.

|  |  |
| --- | --- |
| Couple | Fréquence de coupure |
| C16, C17 | 1.5MHz |
| C21, C22 | 1.5MHz |
| C10, C11 | 15.9KHz |
| C12, C13 | NC\* |
| C14, C15 | 5.7KHz |

Tableau 20 - Filtrage : LAN

\*Consommation interne non spécifié, la datasheet conseille de connecter à la broche VDD\_1.2 deux condensateurs 0.1µF et 2.2µF

### Reset

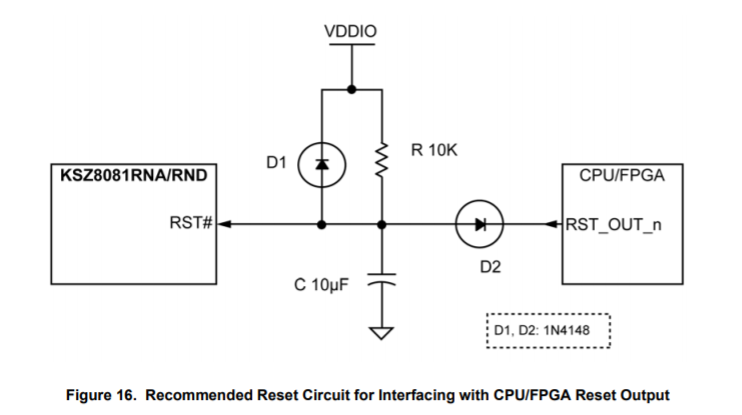


Figure 11 - Reset : LAN

Pour l’interface reset du phy Ethernet, j’ai utilisé la note d’application fournit par la datasheet de ce dernier, expliquant le schéma et le détail des composant à utiliser dans le cadre d’un reset commandé par un composant externe type FPGA ou CPU.

### Horloge

L’horloge est fournie par un quartz à 25MHz, comme préconisé dans la datasheet. Cette option étant la moins contraignante par rapport à un oscillateur actif et plus précis qu’un oscillateur interne. Les condensateurs sous le quartz doivent être adapté à ce dernier. Sur le quartz sélectionné, la capacité de charge est indiquée à 12pF. Ce qui signifie que le calcule doit être égale à la capacité de charge. Ou, pour simplifier les calcule, on peut dire que les capacités doivent être environ égales au double de la capacité de charge du quartz. Donc dans notre cas, 24pF. Idéalement il serait mieux de prendre en compte la capacité parasite de la piste (Environ entre 2pF et 5pF) qui devrait alors s’ajouter au calcul produit sur somme. Cependant, on peut considérer cette capacité parasite comme négligeable.

### Protection

Les résistances en série sur les ports de communications sont une protection. En effet, si la tension de sortie des deux broches diffère, ces résistances limitent le courant. La tension de sortie minimum du phy étant de 2.4v, et la tension de sortie du microcontrôleur étant au maximum de 3.3v, le courant sera, dans le pire des cas de 9mA.

## Wan

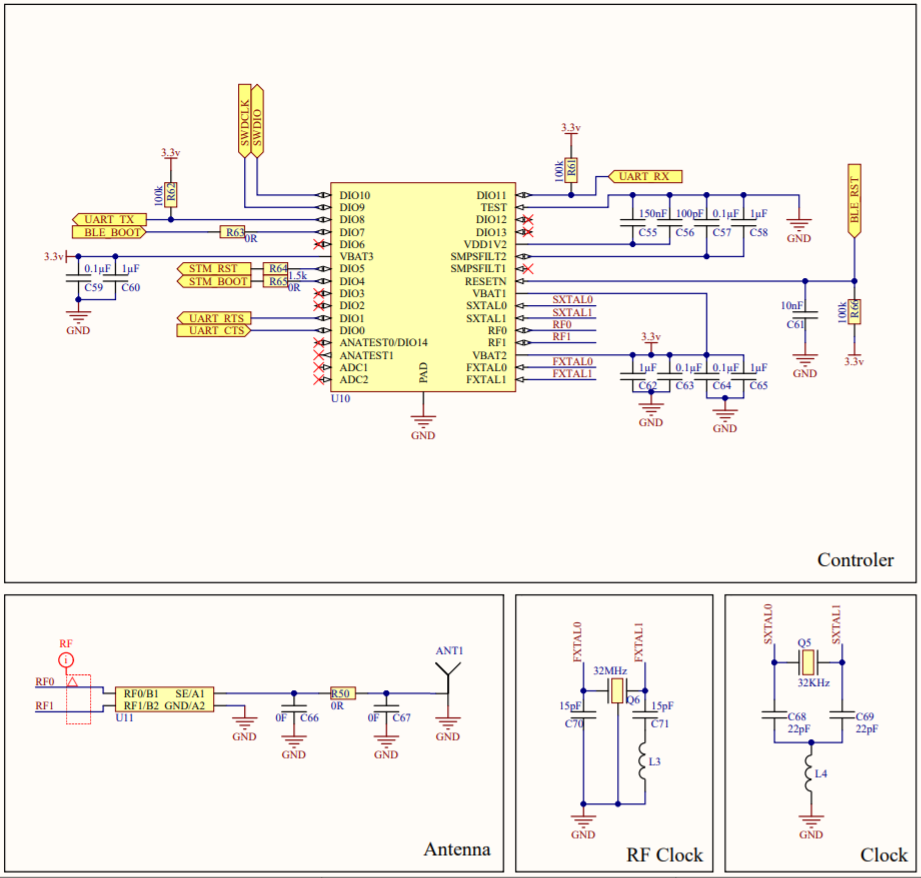


Figure 12 - Diagramme structurel : WAN

### Fonctions

|  |  |  |  |
| --- | --- | --- | --- |
| Numéro | Fonction | Description | Composants |
| Fs2.1 | Gérer le Bluetooth | Gère à la fois le protocole Bluetooth ainsi que le passage en RF | U10 |
| Fs2.2 | Filtrer | Filtrer l’alimentation | C59, C60, C55, C56, C57, C58, C62, C63, C64, C65 |
| Fs2.3 | Permettre un reset externe | Maintenir l’état reset suffisamment longtemps pour qu’il soit pris en compte | R66, C61 |
| Fs2.4 | Adapter la ligne de communication | Adapter la ligne de communication | R61, R62 |
| Fs2.5 | Protéger | Protéger la ligne contre des sur courants | R63, R64, R65 |
| Fs2.6 | Adapter la ligne | Adapter la ligne RF sur 50Ω | U11, C66, C67, R50 |
| Fs2.7 | Connecter une antenne | Connecter une antenne Bluetooth externe | ANT1 |
| Fs2.8 | Cadencer | Cadencer le contrôleur | Q5, C68, C69, L4 |
| Fs2.9 | Fournir une haute fréquence | Fournir au contrôleur une fréquence haute qu’il pourra exploiter | Q6, C70, C71, L3 |

Tableau 21 - Fonctions structurelles : WAN

### Filtrage

Pour une consommation maximale de 30mA, les condensateurs raccordés aux broches VBAT filtrent à la fréquence 15KHz. En revanche, les autres condensateurs sont raccordés à des régulateurs internes et les valeurs de condensateurs sont donnés par la datasheet.

### Connection

Le connecteur d’antenne choisis est un connecteur UFL. Celui-ci présente l’avantage de pouvoir changer d’antenne, mais aussi d’être très peu gourmand en terme de place, comparé à un SMA par exemple. Par contre, il est très peu recommandé dans les appareils soumis à des vibrations ou secousses. On préférera alors utiliser un SMA.

### Programmation

Pour la programmation de ce microcontrôleur, nous avons choisis deux options : D’une part la programmation peut être faite à travers le microcontrôleur principale (STM32). Celui-ci est alors raccordé en UART (qui permet la programmation) et un GPIO est relié au reset du BLE afin de compléter la programmation. Mais dans le cas où on souhaitera de débugger le BLE seul nous avons également ajouté une programmation SW, donc raccordé à un système de Debug STLINK.

### Reset

Le système de reset est exactement similaire à celui présent sur le micro, les calculs y sont détaillés.

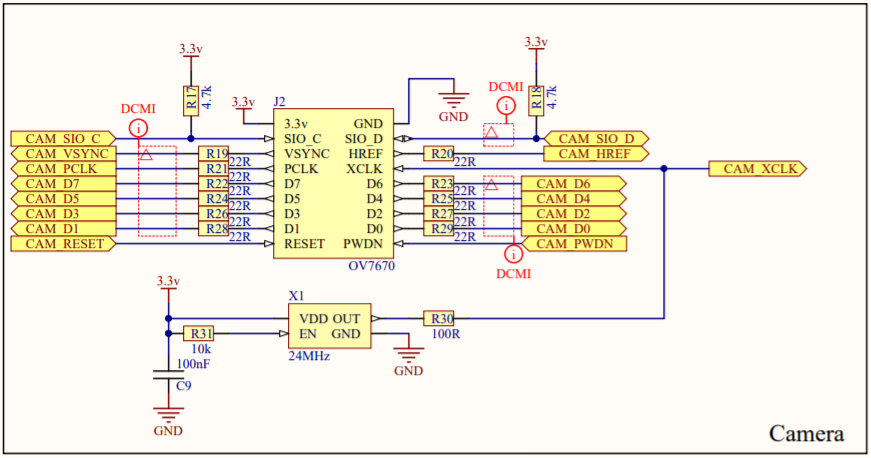
### Cadencement

Le cadencement se fait via 2 quartz. Le premier, basse fréquence, permet de cadencer le microcontrôleur, a 32KHz. Les condensateurs associés sont calculés et expliqué dans la partie LAN, ceux-ci dépendent de la capacité de charge du quartz. La self quant à elle permet de préciser la fréquence et est fourni par la datasheet. Le second quartz, à 32MHz, permet au BLE de créer par PLL une haute fréquence (2.4GHz).

### Adaptation

L’adaptions de la ligne HF est très importante sans quoi une grande partie de l’énergie sera perdu. On place donc une adaptation 50Ω en sortie du BLE, et on place également une adaptation en π avec des composants nuls. On pourra alors modifier la valeur des composants après des tests pour affiner le réglage.

## Interface



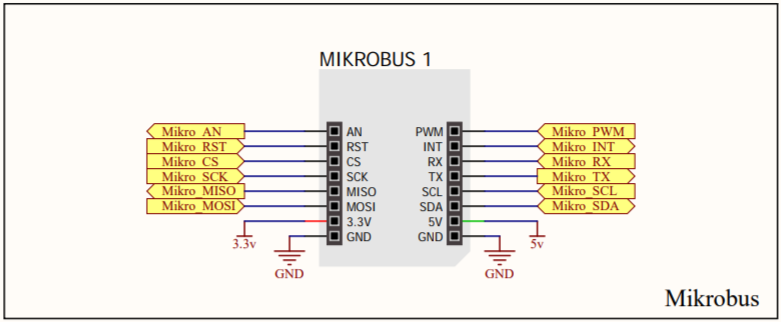


Figure 13 - Diagramme structurel : Interface

### Fonctions

|  |  |  |  |
| --- | --- | --- | --- |
| Numéro | Fonction | Description | Composants |
| Fs8.1 | Permettre le branchement de la caméra | Brochage permettant le branchement direct d’un module OV7670 | J2 |
| Fs8.2 | Cadencer | Cadencer la caméra et le contrôleur | X1, R30, R31, C9 |
| Fs8.3 | Protéger | Protéger la caméra d’un courant excessif | R19, R20, R21, R22, R23, R24, R25, R26, R27, R28, R29 |
| Fs8.4 | Adapter le protocole de communication | Maintenir l’état un état haut par défaut en permanence | R17, R18 |
| Fs5.1 | Communiquer avec les modules MikroElectronica | Permettre le branchement de modules MikroElectronica | Mikrobus1 |

Tableau 22 - Fonctions structurelles : Interface

### Protection

Les résistances de 22Ω font office de protection contre les différences de potentiels. Ainsi la tension minimum de la camera est de 2.97, ce qui fait un courant maximum de 15mA sur ces broches.

### Horloge

Pour cadencer la caméra, nous avons choisis un oscillateur externe, plus précis qu’un quart, mais légèrement plus contraignant car il faut l’alimenter. Cependant il était nécessaire car il était impossible d’atteindre la fréquence de 24MHz avec les quartz que nous avons choisis.

## USB & Alimentation

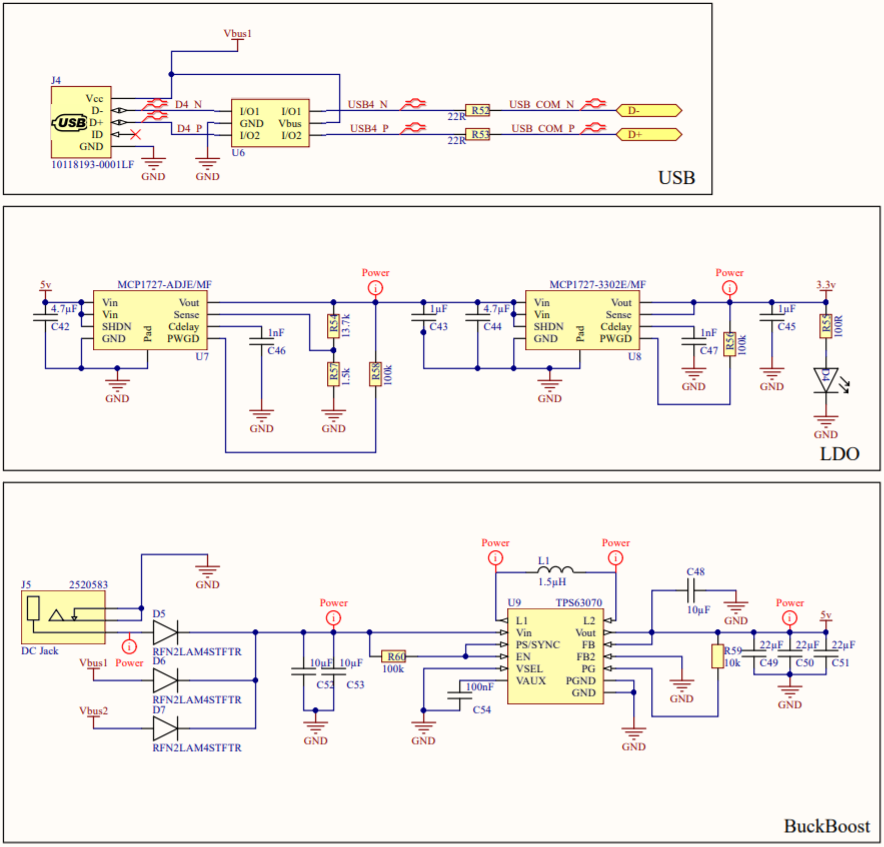


Figure 14 - Diagramme structurel : USB/Alimentation

### Fonctions

|  |  |  |  |
| --- | --- | --- | --- |
| Numéro | Fonction | Description | Composants |
| Fs4.1 | Permettre le branchement d’un périphérique USB | Permettre le branchement d’un périphérique USB en respectant les normes au niveau des connectiques. | J4 |
| Fs4.2 | Protéger | Eviter les surtensions ainsi que les courants excessifs | U6, R52, R53 |
| Fs6.1 | Permettre le branchement d’une alimentation Jack | Permettre le branchement d’une alimentation jack en respectant les normes au niveau des connectiques | D5 |
| Fs6.2 | Réguler une tension 5v | Réguler une tension de 5v que la tension d’entrée soit de 5v ou 12v. | D4, D6, D7, C52, C53, R60, C54, U9, L1, C48, R59, C49, C50, C51 |
| Fs6.3 | Réguler une tension de 3.3v | Réguler une tension à 3.3v en limitant au mieux les parasites | C42, U7, C46, R57, R54, R58, C43, C44, U8, C47, R56, C45 |
| Fs6.4 | Montrer à l’utilisateur le bon fonctionnement de l’alimentation | Montrer à l’utilisateur, par un moyen rapide à visualiser, le bon fonctionnement de l’alimentation | R55, D4 |

Tableau 23 - Fonctions structurelles : USB/Alimentation

### USB

Le module de protection USBLC6-2P6 protège le microcontrôleur des éventuelles surtensions, en effet, il peut supporter jusqu’à 15kV en pic. Les deux résistances sont là, comme dit précédemment, pour éviter les courant excessifs. On limite donc le courant à une vingtaine de mA. Le connecteur quant à lui sera de type Mini-USB femelle, celui-ci étant largement rependu pour les Devices. La broche ID du connecteur servira au second périphérique, pour déterminer si notre carte est un Host ou un Device. Câblé à la masse, il s’agit d’un Device.

### Alimentations

#### DC/DC

L’alimentation est une partie à ne pas négliger. Elle doit effet permettre de délivrer, en plus de la tension, le courant ainsi que la dissipation nécessaire. Pour commencer j’ai mis en commun toutes les tensions par le biais de diodes afin d’éviter le retour de courants.

Par la suite, j’ai opté pour un régulateur à découpage, permettant d’abaisser la tension à 5v. Ceux-ci permettent une chute de tension plus importantes pour un même courant consommé sans que la montée en température ne devienne un gros inconvénient.

|  |  |
| --- | --- |
| Partie | Consommation |
| Microcontrôleur | 620mA |
| Mikrobus | 630mA |
| Debug | 150mA |
| Camera | 18mA |
| Lan | 47mA |
| Wan | 30mA |
| Total | 1.5A |

Tableau 24 - Consommation

Le tableau ci-dessus représente la consommation de chacune des parties. La partie Mikrobus étant composé de modules de constructeur différents, il est difficile d’estimer une consommation. J’ai donc arrondi afin que la consommation totale de la carte soit d’1.5A à plein régime. Nous devons donc nous assurer que les alimentations soient capables de délivrer ce courant.

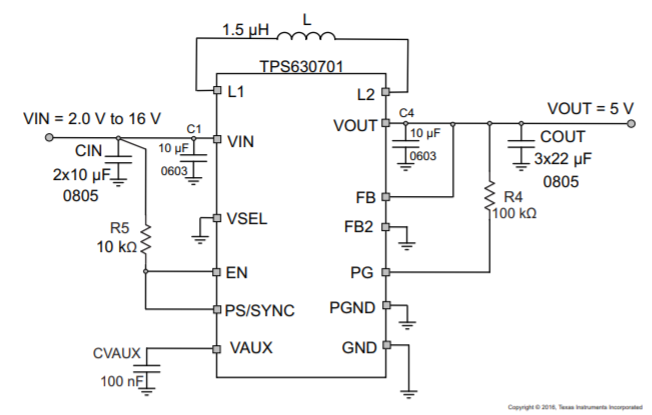
La première partie est donc composé d’un buck-boost à sortie fixe 5v qui se chargera de réguler une tension avec une entrée variant de 3.8v à 12v. Celui-ci est capable de délivrer jusqu’à 2A. Pour sa mise en place, j’ai utilisé principalement l’application typique fournit par le constructeur pour les buck-boost à sortie fixe 5v. On peut tout de même calculer la dissipation du boitier.

Figure 15 - Application typique buck/boost

Pour se faire, il va d’abord falloir calculer la puissance dissipée par l’alimentation. On utilisera donc cette formule : . A raison d’une dissipation de 63°C/W, l’élévation de température sera de 53°C. Même avec une température ambiante élevé (40°C), cela reste largement raisonnable par rapport à la température de fonctionnement maximum du composant (125°C)

#### LDO

Pour ensuite abaisser la tension à 3.3v, il aurait été intéressant d’utiliser un DC/DC, cependant, ce dernier engendre un bruit sur l’alimentation non négligeable provoqué par son mode de fonctionnement. Ceci n’est pas souhaitable et on utilise généralement un LDO pour filtrer ce bruit. J’ai donc décidé de mettre en place un LDO. Mais le courant étant relativement important, il est difficile de mettre en place un LDO capable de dissiper Il était alors possible de mettre en place un DC/DC suivis d’un LDO, mais l’environnement du DC/DC étant assez imposant, il est plus judicieux économiquement de placer deux LDO en cascade, ceux-ci se partageant alors la dissipation.

J’ai donc dans un premier temps divisé par deux le drop-out nécessaire : et ai donc décidé de dimensionner un LDO pour réguler une tension de . J’ai utilisé pour ce faire des MCP1726 qui peuvent se décliner en deux versions : ajustable ou non. Pour le premier on dimensionne donc le pont diviseur lié à la broche Sense, celle-ci réglant la tension de sortie par cette formule : . On peut alors calculer la dissipation, qui s’élève à d’élévation de température. En prenant en compte le fait que le composant fonctionne jusqu’à 125°C, c’est tout à fait acceptable.

Pour le second, j’ai utilisé le même régulateur, mais afin de limiter les composants externes, j’ai choisis la version à sortie fixe à 3.3v. Il a donc suffi de condensateurs filtrage et c’est tout.

### Led

J’ai également placé une led afin de montrer à l’utilisateur que l’alimentation fonctionne. Celle-ci fonctionnant sous 20mA, et la chute de tension de la diode étant de 1.6v, une résistance de 100Ω permet un courant de 17mA.

## Debug

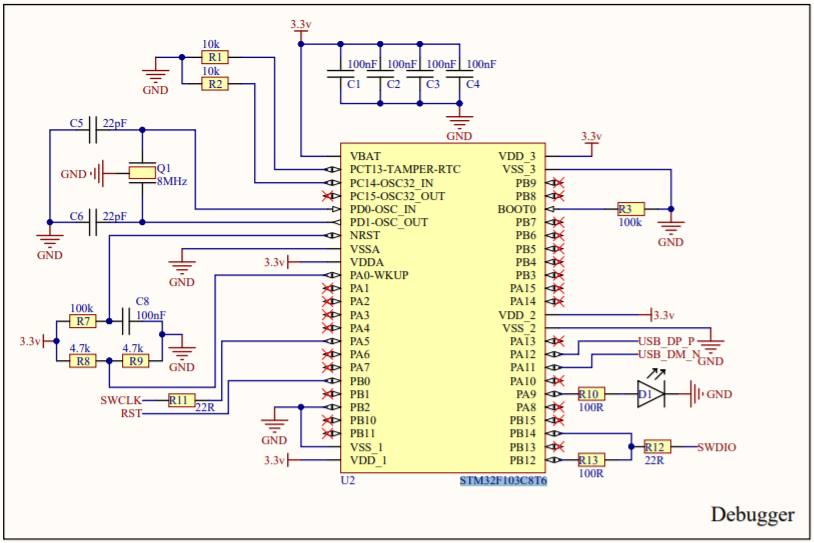


Figure 16 - Diagramme structurel : Debug

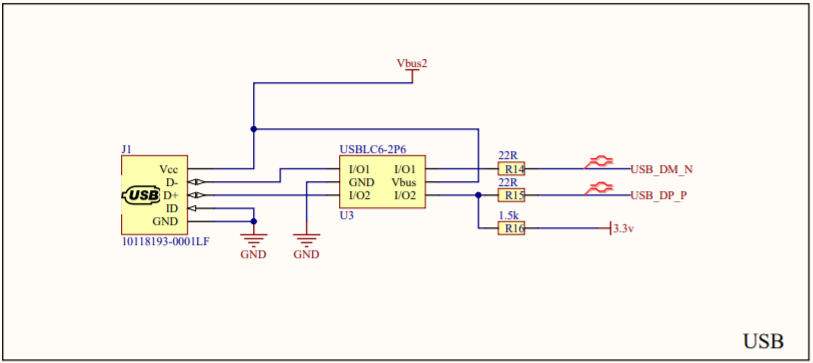


Figure 17 Diagramme structurel : Debug USB

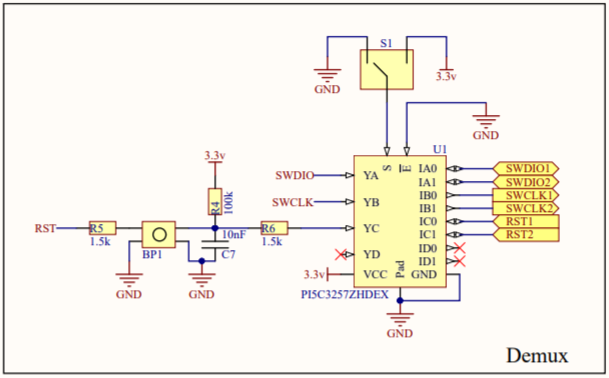


Figure 18 - Diagramme structurel Debug Multiplex

### Fonctions

|  |  |  |  |
| --- | --- | --- | --- |
| Numéro | Fonction | Description | Composants |
| Fs7.1 | Exécuter le programme de Debug | Exécute le programme de Debug (STLINK) et sert d’intermédiaire entre PC et Microcontrôleur | U2 |
| Fs7.2 | Cadencer | Cadencer le debugger | Q1, C5, C6 |
| Fs7.3 | Configurer | Configurer le debugger pour qu’il s’accorde avec le programme STLINK | R1, R2, R3, R7, R8, R9 |
| Fs7.4 | Filtrer | Filtrer l’alimentation d’éventuels parasites | C1, C2, C3, C4 |
| Fs7.5 | Protéger | Protéger le contrôleur contre un sur courant éventuel | R11, R12, R13 |
| Fs7.6 | Montrer à l’utilisateur | Montrer à l’utilisateur le bon fonctionnement du Debug | R10, D1 |
| Fs7.8 | Permettre la connexion avec un ordinateur | Permettre une connexion débuggeur/ordinateur | J1 |
| Fs7.9 | Protéger | Protéger le débuggeur contre les surtensions et sur courants | U3, R14, R15, R16 |
| Fs7.10 | Permettre un Reset manuel | Permettre un reset manuel du contrôleur et du module Bluetooth | R4, R5, R6, BP1, C7 |
| Fs7.11 | Multiplexer la programmation | Permettre la programmation du contrôleur et du BLE via un switch | U1, S1 |

Tableau 25 - Fonctions structurelles : Debug

### Configuration

Pour s’assurer de la compatibilité du programme STLINK V2 avec notre débuggeur, j’ai utilisé la schématique de la Dev-Board STM34F4 Discovery. Sur ce dernier, j’ai pu déterminer les connexion SWDIO, SWCLK et Reset ainsi que le cadencement. Cette schématique m’a également fourni les résistances de filtrage.

### Connection

La connexion entre le débuggeur et l’ordinateur se fera par le biais d’un micro-USB, largement rependu pour les Devices. On n’oubliera pas le circuit de protection pour éviter les surtensions et les courants excessifs. Cette partie est exactement la même que la connexion qui se fera du contrôleur à l’ordinateur.

### Reset

J’ai choisis de placer le reset en amont du multiplexeur afin d’éviter d’en placer deux. Le bouton force donc simplement la broche à l’état bas. La résistance de tirage et la capa sont là pour s’assurer que l’état bas reste tel quel suffisamment longtemps pour qu’il soit pris en compte. Le calcul de ce temps est détaillé dans la partie Micro. Les deux résistances en série sont là pour éviter un retour de courant trop important.

### Multiplexeur

Le multiplexeur évite de devoir mettre en œuvre deux débuggeur. On choisira donc grâce à un micro switch quel contrôleur (BLE ou Micro) programmer.

## Micro

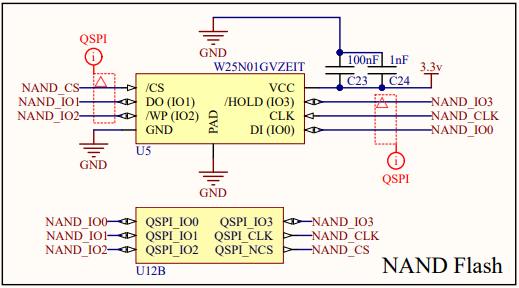


Figure 19 - Diagramme structurel : Nand Flash

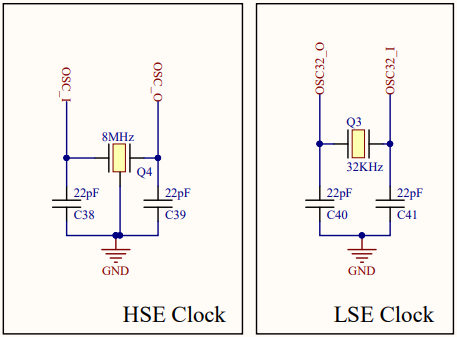


Figure 20 - Diagramme structurel : Clock

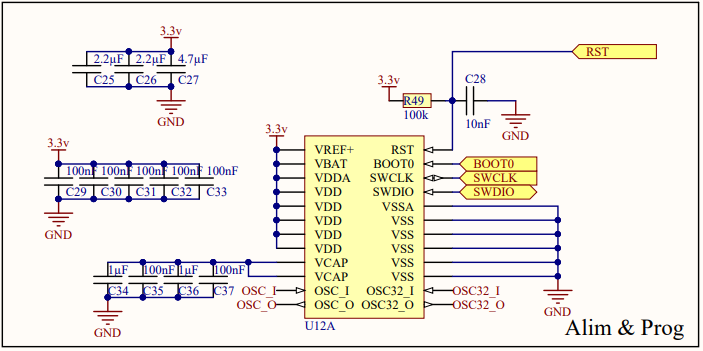


Figure 21 - Diagramme structurel : Micro

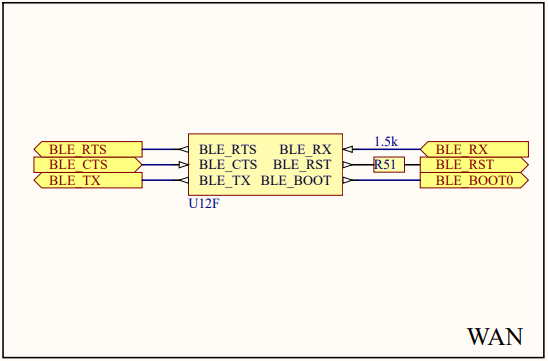


Figure 22 - Diagramme structurel Micro BLE

### Fonctions

|  |  |  |  |
| --- | --- | --- | --- |
| Numéro | Fonction | Description | Composants |
| Fs1.1 | Contrôler | Elément permettant l’exécution d’un programme | U12 |
| Fs1.2 | Etendre la Flash | Etendre l’espace de stockage Flash de la fonction Contrôler | U5 |
| Fs1.3 | Osciller à 32KHz | Cadencer à 32KHz la fonction Contrôler | Q3, C40, C41 |
| Fs1.4 | Osciller à 8MHz | Cadencer à 8MHz la fonction Contrôler | Q4, C38, C39 |
| Fs1.5 | Filtrer | Filtrer l’alimentation de la fonction Contrôler | C25, C26, C27, C29, C30, C31, C32, C33, C34, C35, C36, C37 |
| Fs1.6 | Filtrer | Filtrer l’alimentation de la fonction Etendre la Flash | C23, C24 |
| Fs1.7 | Assurer l’état de Reset | S’assurer que l’état bas de Reset soit maintenu suffisamment longtemps pour être compris par la fonction Contrôler | R49, C28 |
| Fs1.8 | Protéger | Protéger la broche Reset du Wan contre un courant excessif | R51 |

Tableau 26 - Fonctions structurelles : Micro

### **Cube** mx

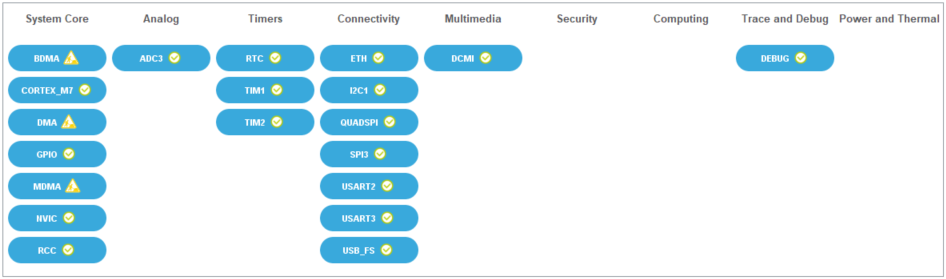


Figure 23 - Fonctions CubeMX

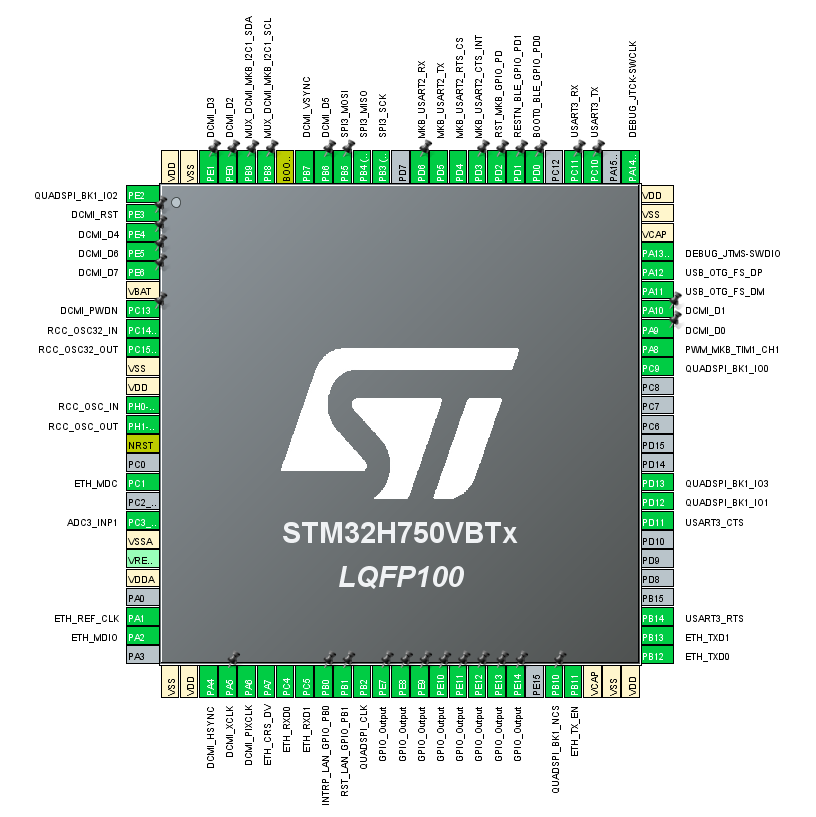


Figure 24 - Pinout CubeMX

### Nand Flash

La Nand flash communique en QSPI, qui est une SPI composé de 4 fils. Pour la mettre en œuvre, il nous a donc simplement fallu activer la QSPI sur notre microcontrôleur.

### Programmation

La programmation pourra se faire de deux façons. D’une part par le protocole SW permettant le Debug via un débuggeur STLINK. Mais d’autre part on pourra programmer le micro via Bluetooth. En effet, le BLE est relié au micro via une UART permettant la programmation, et un GPIO du BLE est dédié au reset du micro.

### Cadencement

Le cadencement du microcontrôleur se fera par des quartz pour les raisons citées précédemment. Pour exploiter tout le potentiel de notre microcontrôleur, j’ai choisis d’intégrer sur notre carte 2 quartz. Un haute fréquence, et un basse fréquence.

Afin d’atteindre la fréquence maximale de fonctionnement de 480MHZ, il est nécessaire que l’oscillation externe soit de 8MHz comme le montre la capture d’écran de CubeMX ci-dessous.  
De même, pour profiter d’un OS temps réel sur notre carte, une oscillation basse fréquence est indispensable. Un quartz à 32.768KHz est donc indispensable. Le détail des calculs pour les capacités de charges sont exprimés dans la sous-partie Horloge de la LAN.

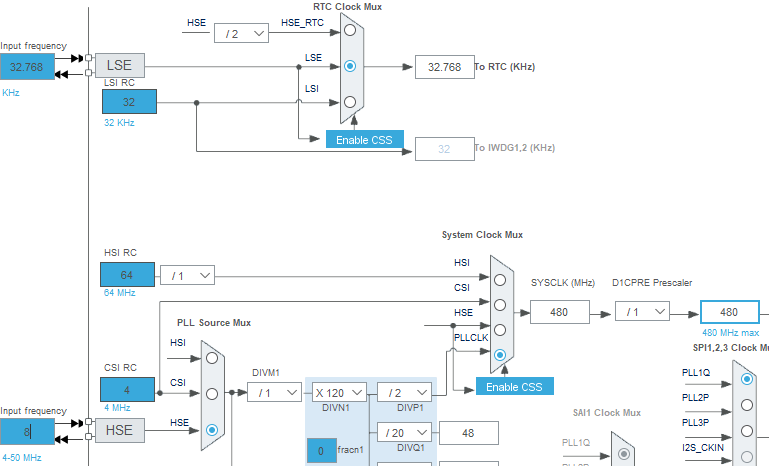


Figure 25 - Clock CubeMX

### Reset

Le Reset s’effectue sur le microcontrôleur par la présence d’un état bas sur la broche Reset. Cependant, cet état doit être maintenu suffisamment de temps pour s’assurer de la bonne compréhension du Reset. Ainsi, grâce à ce montage, lorsque la broche reset est forcée à l’état bas, le condensateur se décharge instantanément. Mais il mettra un certain temps à se charger, temps caractérisé par la valeur du couple . Or, d’après la datasheet de notre microcontrôleur, l’état bas doit être maintenu pendant 377µs.

La charge d’un condensateur est caractérisé par la formule suivante . On peut donc calculer la tension à ses bornes après 377µs : . Or la technologie CMOS indique qu’un état bas sera lu si la broche est au maximum à .

### Filtrage

Pour satisfaire le filtrage du microcontrôleur, j’ai utilisé la note d’application fournit par la datasheet du constructeur. Il est donc inscrit sur celle-ci la valeur de chaque condensateur ainsi que la broche à laquelle la rattacher.

Concernant la flash externe, pour un courant de fonctionnement maximum de 35mA, la résistance équivalente est donc de 100Ω. Les condensateurs de découplages coupent donc à la fréquence de 15KHz.

Figure 26 - Filtrage Micro

### SRAM

La SRAM, fonctionnant sur un port parallèle, pour une vitesse de transfert maximum, était un composant facultatif proposé. Cependant, en vue de l’encombrement de notre microcontrôleur, il nous a été impossible de trouver un port parallèle complet, disponible, et compatible avec une SRAM. Nous avons donc dû nous résoudre à ne pas l’inclure.

# Routage

## Stack-up

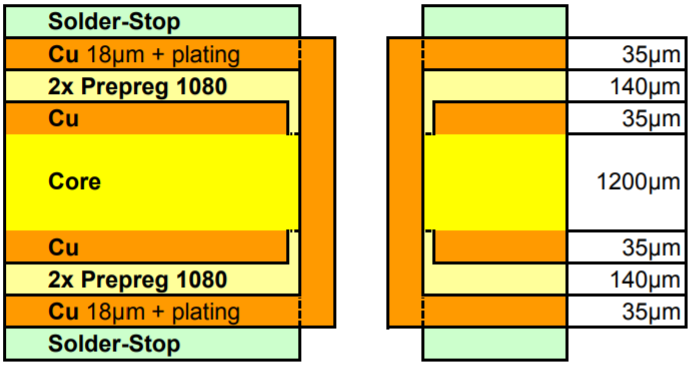
Le but de notre carte est de respecter au mieux les normes CEM. Notre carte comportera donc 4 couches, deux couches de signaux, une couche de masse et une couche d’alim. De plus, un se basera sur un stack-up FR4, qui définit l’épaisseur du cuivre ainsi l’épaisseur et la nature du diélectrique. Voici donc un aperçu de l’épaisseur du cuivre ainsi que celui du diélectrique. On peut également ajouter la constante diélectrique qui se mesure généralement à 1MHz et qui vaut, pour du FR4, à une valeur compris entre 4.2–4.8.

Figure 27 - Stack-up

## Règles

### DCMI

Le DCMI est un protocole de communication dédié aux équipement de capture d’image ou de vidéo. La fréquence de ce dernier dépend de la fréquence de rafraichissement de notre module de capture. Dans notre cas 24MHz. Il est composé de différentes horloges et d’un port parallèle donnant la valeur des pixels. La rapidité de ce protocole de communication nous impose des contraintes lors du routage. Ici, les lignes du DCMI doivent toutes faire environ la même longueur. La datasheet nous explique qu’une tolérance est possible à 7mm près, mais je l’ai personnellement abaissé à 2.

### Ethernet

L’Ethernet se décompose en deux parties, d’une part les pairs différentiels qui proviennent du connecteur Ethernet et d’autre part du protocole RMII entre le phy et le micro. Ce protocole dont la vitesse maximum est de 100MHz, subit les mêmes contraintes que le DCMI. Un écart de longueur au niveau des pistes impliquerait un déphasage, j’ai donc limité l’écart de longueur entre les pistes à 2mm.

### Pair différentiels

#### Ethernet

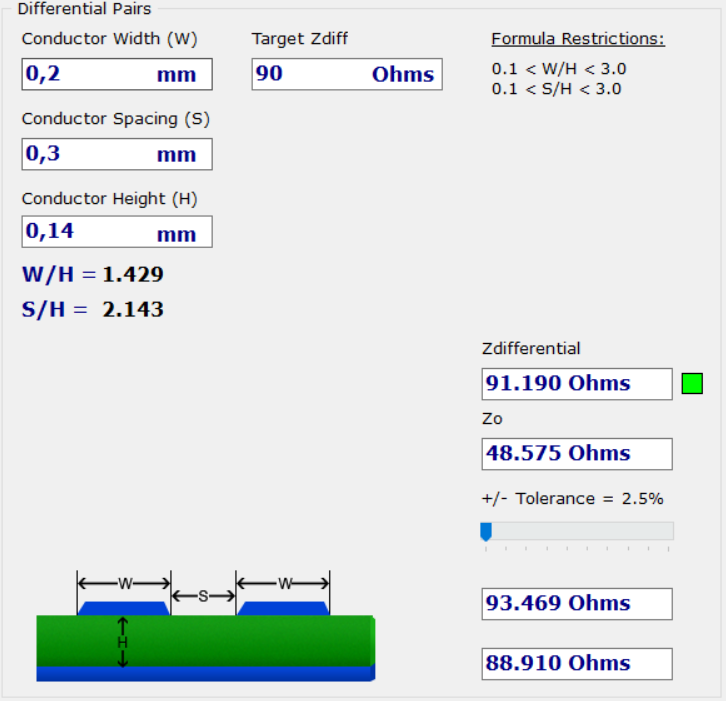
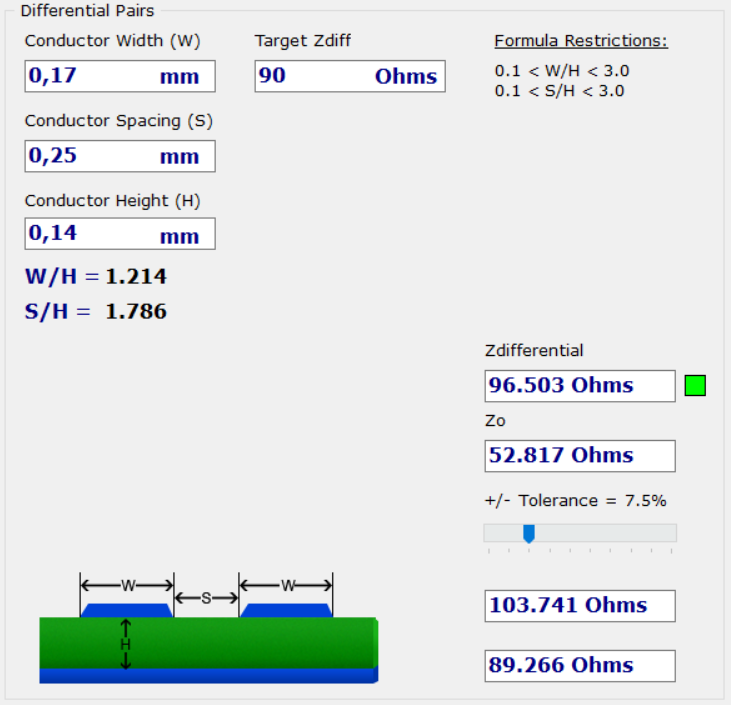
 Entre le connecteur Ethernet et le phy, le signal se propage à travers deux pairs différentiels. En connaissant la distance séparant les pistes de la couche internet (voir le stack-up), et en faisant varier l’espacement des lignes et la largeur des pistes, on peut arriver à trouver une impédance différentielle de 90Ω et une impédance caractéristique à 50Ω. Pour se faire, nous avons utilisé le logiciel PCB Saturn. Celui-ci intègre les formules afin de calculer les impédances caractéristiques.

Figure 28 - Paire différentiel Ethernet

#### USB

De même pour l’USB, bien que les valeurs soient légèrement différentes, le résultat est sensiblement le même.

Figure 29 - Paire différentiel USB



### QSPI

Le signal QSPI correspond au signal entre la Flash externe est le micro. Il se doit d’être très rapide. On peut lire sur la datasheet de la Nand Flash que la vitesse de transfert QSPI peut atteindre 416MHz. Lors du routage, nous devons donc faire attention à ce que les lignes aient la même longueur (j’ai choisis 2mm). Exception faite du signal d’horloge, qui risque de rayonner de façon importante si l’adaptation en longueur est trop importante. On fera donc également attention à ne pas positionner la flash trop loin du micro.

### RF

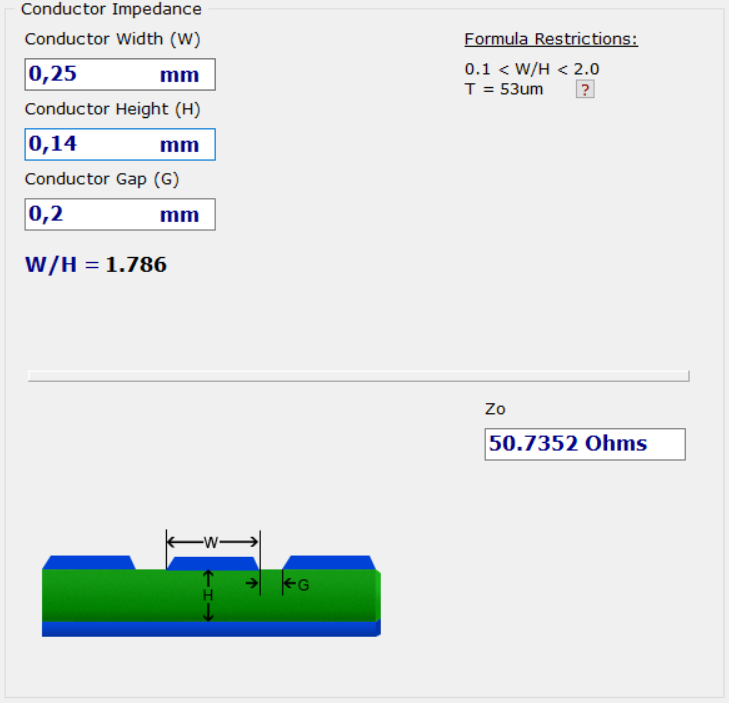
 La ligne RF, directement sortie du BLE se doit d’être adapté à 50Ω afin que l’énergie apporté ne soit pas perdue. C’est pourquoi la datasheet du BLE préconise de placer un adaptateur 50Ω en sortie du BLE. Très précisément à 800µm de la sortie. Par la suite c’est à nous d’adapter la ligne grâce à PCB Saturn. Encore une fois en se servant du stack-up, on peut définir la largeur de la ligne ainsi que sa clearance pour obtenir une ligne se rapprochant le plus de 50Ω jusqu’au connecteur d’antenne. Le montage en π est toujours là pour parfaire notre adaptation une fois celle-ci réalisé.

Figure 30 - Adaptation RF

### Alimentation

Pour l’alimentation nous devons calculer la largeur des pistes. Celle-ci doit être suffisante pour laisser passer un courant important. Nous avons déterminé 1.5A. Pour se faire, j’ai utilisé [le blog de Nononux](http://nononux.free.fr/index.php?page=elec-brico-outils#!elec-brico-outil-largeur-piste-pcb) qui propose un calculateur en ligne. On peut donc faire le calcule pour la couche externe et la couche interne :

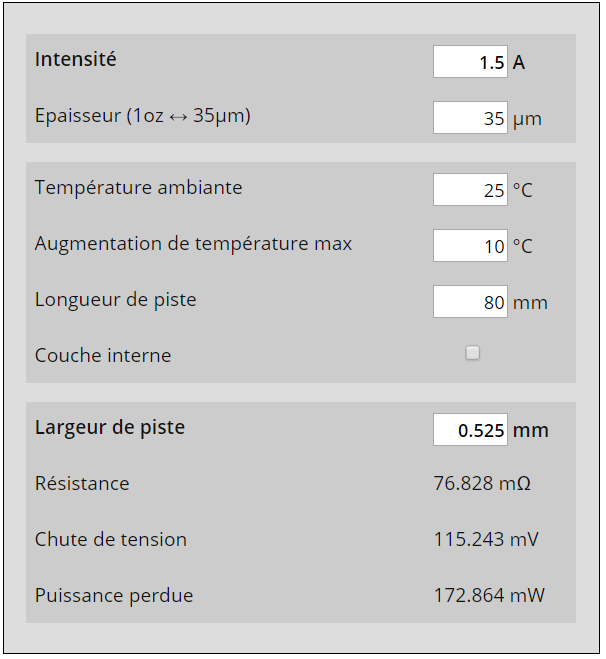


Figure 31 - Piste d'alimentation externe

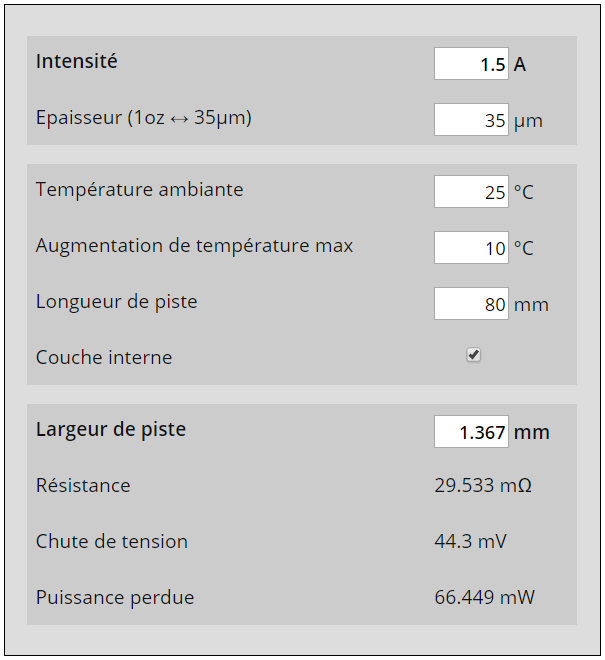


Figure 32 - Piste d'alimentation interne

### Encombrement

La taille de la carte nous a été imposé lors du cahier des charges. La carte ne doit pas faire plus de 8cm par 8cm. Cependant, c’est l’unique contrainte de taille qui nous a été donné. Aucun trou de perçage n’a été prévu et aucun encombrement vertical non plus.

## Placement

### Top layer

#### Overlay

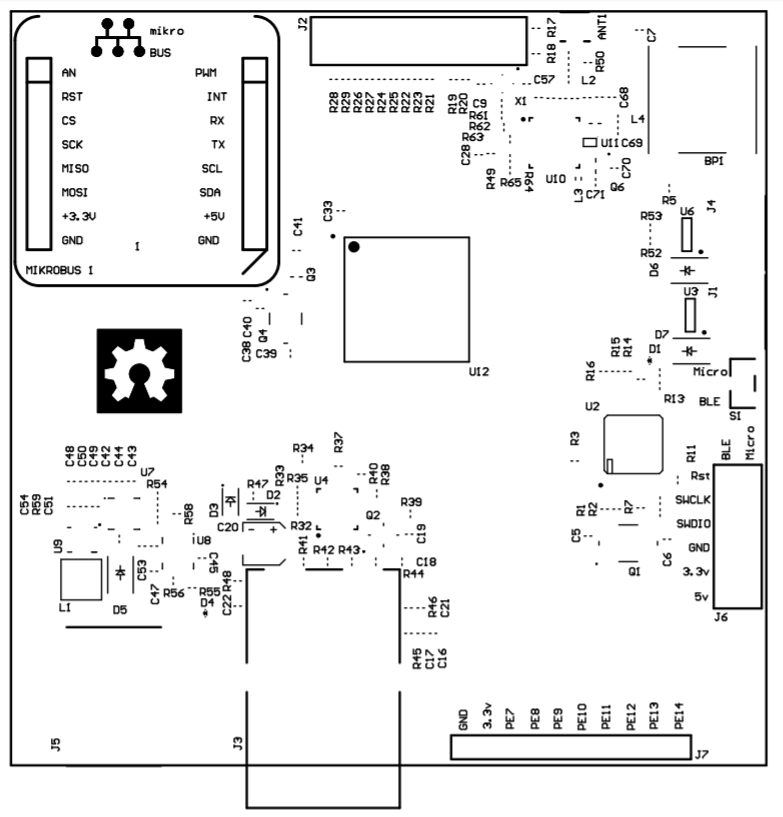


Figure 33 - Top Overlay 1:1

#### Typon

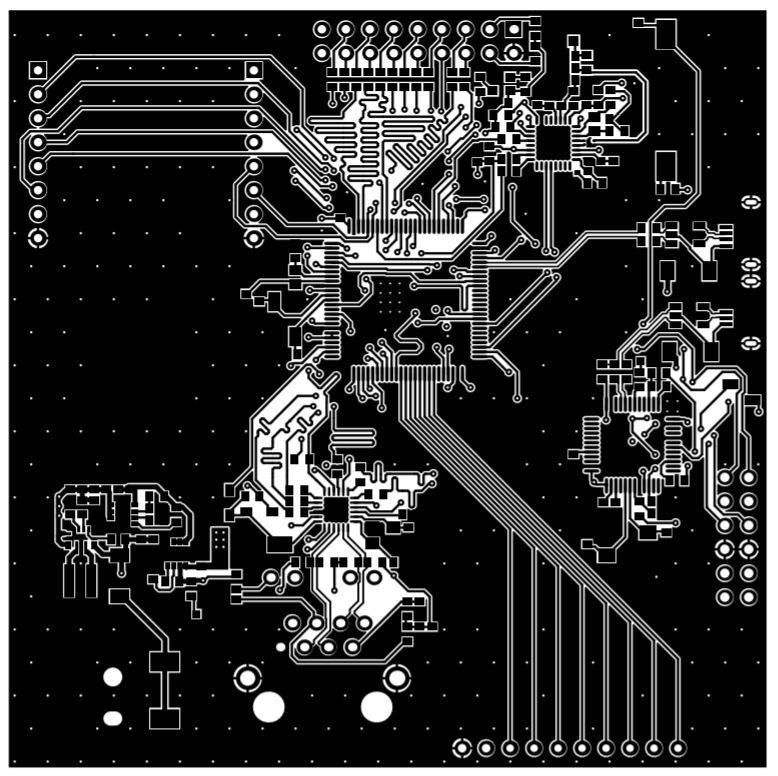


Figure 34 - Top Layer 1:1

### Alimentation

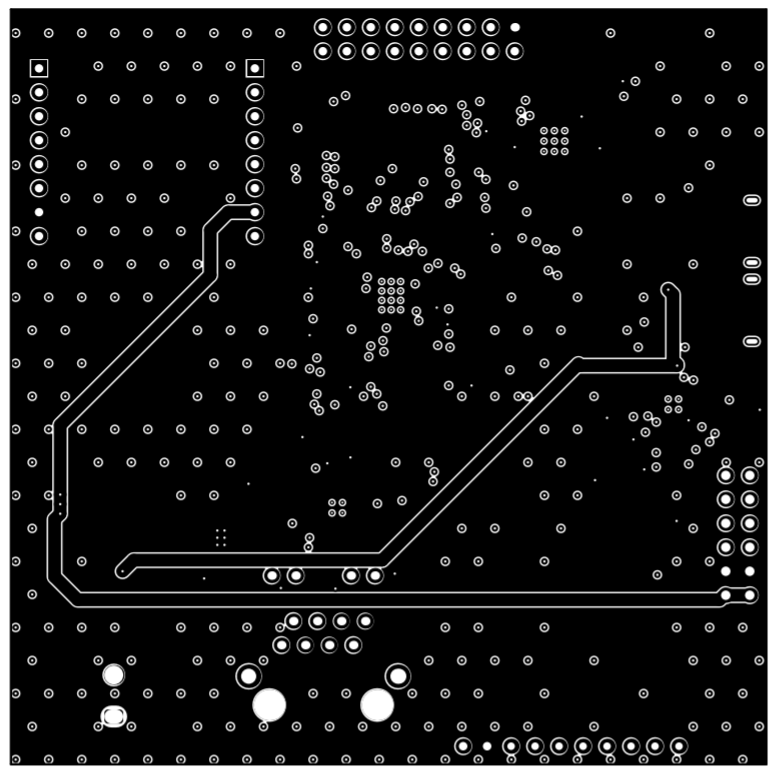


Figure 35 - Couche d'alimentation 1:1

### Masse

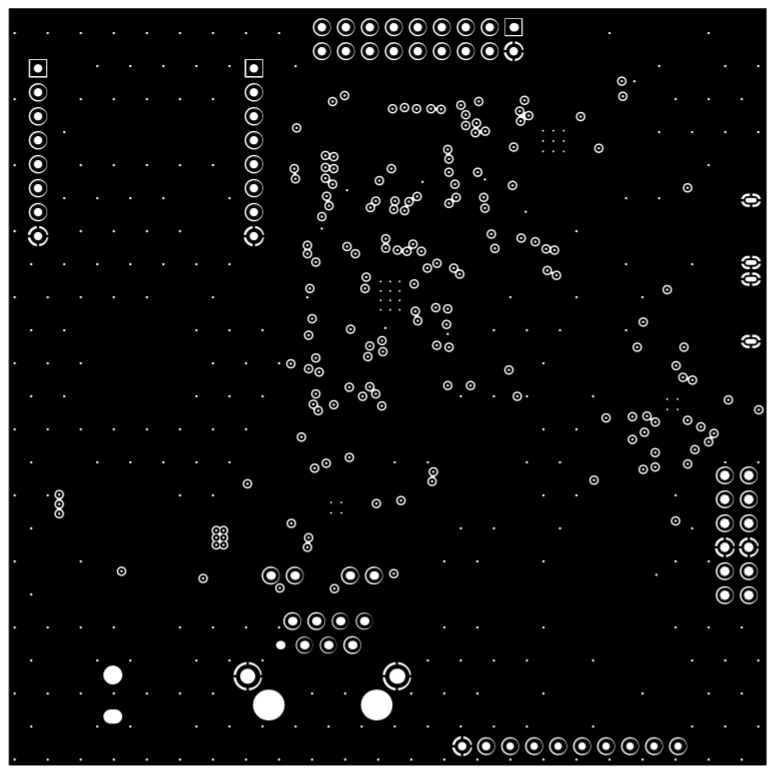


Figure 36 - Couche de masse 1:1

### Bottom Layer

#### Overlay

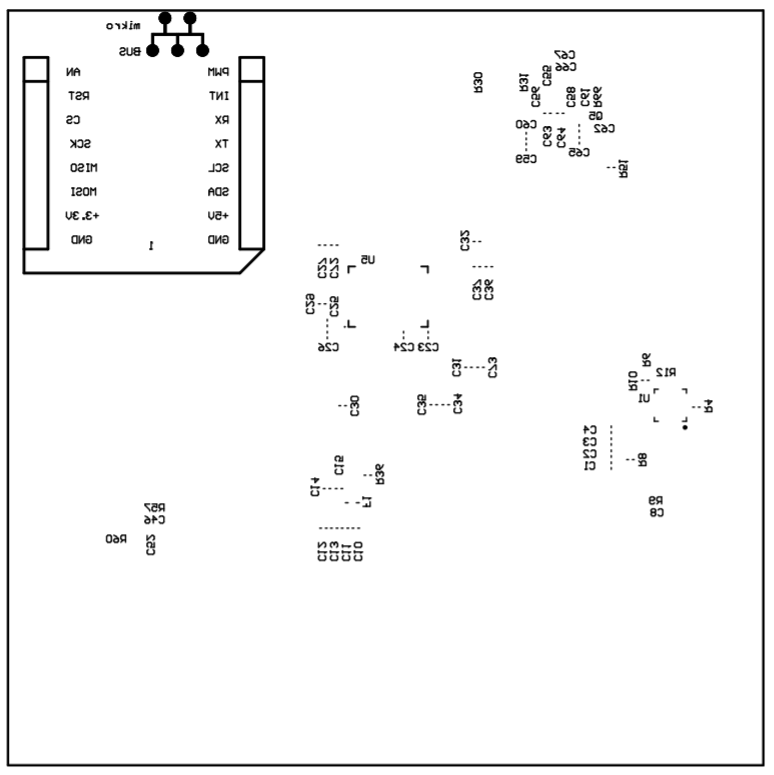


Figure 37 - Bottom Overlay 1:1

#### Typon

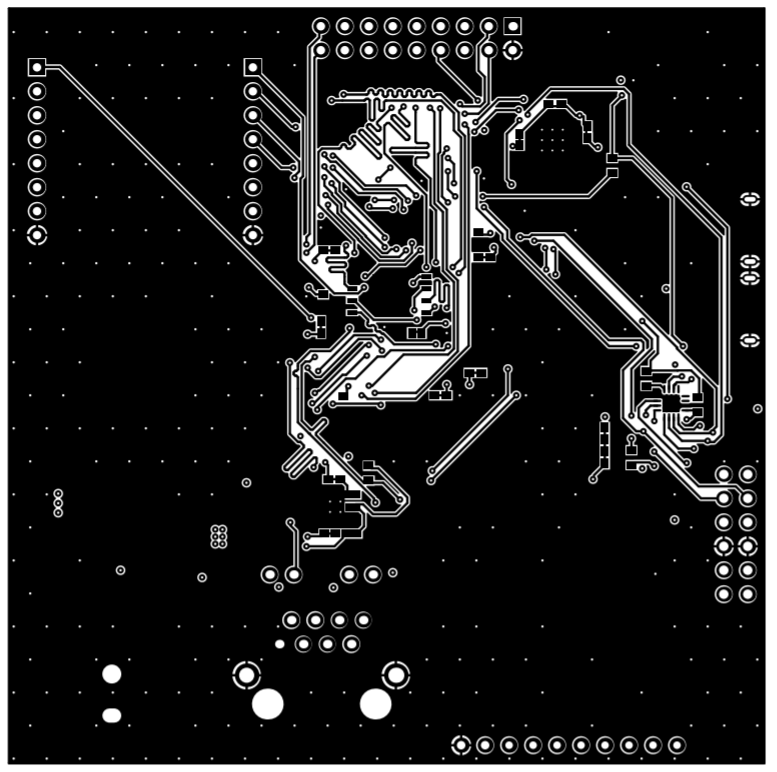


Figure 38 - Bottom Layer 1:1

## Ajouts

Vous aurez sans doute constaté que quelques éléments se sont vu rajouté sur la carte et qui n’ont pas été détaillés dans le présent rapport. Il existe deux changements majeurs. D’une part, étant donné la large place ainsi que la non connexion de la quasi-totalité d’un port du micro, j’ai décidé d’ajouter un port GPIO en bas de la carte. De plus, pour éviter toute mauvaise surprise, j’ai ajouté des headers de programmation, en sortie du multiplexeur, permettant d’utiliser un programmateur externe. Celui-ci serai utilisé si on découvre que notre programmateur ne fonctionne pas ou comporte une erreur.

Ces quelques ajouts font grimper la BOM de quelques euros mais apportent un confort certain et une sécurité pour les futurs utilisateurs de la carte.

# Conclusion

## Chauvet David

Ce fut une expérience toute nouvelle et très enrichissante pour ma part. Je n’avais jamais utilisé le logiciel Altium avant ce projet de conception de carte électronique. Il s’agissait donc entièrement d’une découverte.

J’avais pour mission de créer la schématique du BLE, puis de commencer le routage du PCB pour mieux comprendre et appréhender Altium. J’ai également pu compter sur mon binôme qui m’a donné des conseils et des précisons sur le fonctionnement du logiciel. Il est parfois repassé derrière mon travail pour que celui-ci soit le plus exhaustif possible.

Je serai à nouveau amené à concevoir ce genre de carte, voire d’effectuer un travail plus poussé, j’ai donc des progrès à faire si je veux pouvoir être plus autonome sur ce type de projet.

## Huet Samuel

Les délais courts nous ont obligés à faire des compromis aussi bien sur la schématique que sur le routage. Certaines valeurs ont été extraites directement de la datasheet sans pour autant avoir été calculés. Cela peut entrainer des erreurs d’inattentions ou d’incompréhension. On retrouve cela par exemple sur le programmateur, dont il m’a été difficile de comprendre exactement le fonctionnement. Concernant le routage, j’ai négligé certains points. Notamment les vias qui ne sont pas du tout adapté selon le signal, mais aussi l’alimentation arrivant sur les composants arrive par un seul via, ce qui est peu souhaitable car cela entraine une inductance parasite. Mais par manque de temps je n’ai pas pu intégrer cela.

Cependant, dans l’ensemble nous avons réussi à respecter les contraintes les plus importantes, et j’ai personnellement beaucoup appris grâce à ce projet. Les différentes contraintes qui régissent la CEM ou les fonctionnalités d’Altium en sont un exemple. On pourra citer par exemple le via stitching, ou la possibilité de lier les composants d’une librairie avec un fournisseur. Je compte bien utiliser ces connaissances et les approfondir dans mes futurs projets pour être plus efficace.

# Annexes

## GitHub

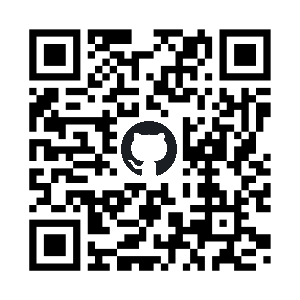
[](https://github.com/SamuelHuet/DevBoard_STM32) Afin de centraliser toutes les informations nécessaires à la compréhension du projet ainsi qu’à sa conception, toutes les données ont été importé au sein d’un même repository Github. Vous trouverez donc en scannant ce QRcode (ou en cliquant dessus) le projet Altium, les différents fichiers générés, la BoM ainsi que la dernière mise à jour de ce présent rapport. Le projet est sans licence et il est ouvert à tous de participer à son amélioration sous toutes ses formes.

Figure 39 - GitHub

## Sources

Phy (référence)  
[www.farnell.com/datasheets/2310338.pdf?src-supplier=Element14](http://www.farnell.com/datasheets/2310338.pdf?src-supplier=Element14)

Phy (aide)  
<https://www.renesas.com/eu/en/doc/DocumentServer/012/R19AN0014ED0102_ASSP.pdf>

RJ45  
<http://www.farnell.com/datasheets/1736687.pdf?_ga=2.102175939.333202117.1574096918-1106813667.1574096918>

Quartz phi  
<http://www.farnell.com/datasheets/2048793.pdf?_ga=2.30960893.333202117.1574096918-1106813667.1574096918>

Feritte  
<https://www.mouser.fr/datasheet/2/281/QNFA9102-1485076.pdf>

Diode 1N4148  
<https://datasheet.ciiva.com/26921/getdatasheetpartid-764103-26921186.pdf?src-supplier=Verical>

Empreinte mikrobus  
<https://www.mikroe.com/mikrobus>

Cablage caméra  
<https://www.lucidar.me/fr/arduino/camera-ov7670-and-arduino/>

Datasheet caméra  
<http://web.mit.edu/6.111/www/f2016/tools/OV7670_2006.pdf>

Debuger micro  
<http://e.pavlin.si/2016/02/22/st-link-v2-on-a-single-sided-diy-pcb/>

ESD microUSB debuger  
<https://fr.farnell.com/stmicroelectronics/usblc6-2p6/reseau-de-diode-tvs-usb2/dp/1295310>

BOOTLOADER Micro  
<https://www.st.com/content/ccc/resource/technical/document/application_note/b9/9b/16/3a/12/1e/40/0c/CD00167594.pdf/files/CD00167594.pdf/jcr:content/translations/en.CD00167594.pdf>

BOOTLOADER\_BLE (choix de l'uart prioritaire) <https://www.st.com/content/ccc/resource/technical/document/application_note/group0/ed/6f/b7/ec/16/1f/41/1c/DM00294188/files/DM00294188.pdf/jcr:content/translations/en.DM00294188.pdf>

DCMI sur STM32  
<https://www.st.com/content/ccc/resource/technical/document/application_note/group0/c0/ef/15/38/d1/d6/49/88/DM00373474/files/DM00373474.pdf/jcr:content/translations/en.DM00373474.pdf>