

# Guide pas-à-pas pour la conception du PCB du projet « Robots chercheur/cacheur »

**Objectif :** Aider les futurs élèves à concevoir un PCB fonctionnel avec KiCad, en évitant les erreurs fréquentes et en suivant une méthodologie claire.

## 1. Préparer le terrain : Avant de commencer

- **Lister tous les composants** nécessaires avant d'ouvrir KiCad.
- **Vérifier leur disponibilité** à l'école ou en stock perso (ou demander au prof).
- **Télécharger ou créer les bons modèles** pour symboles et empreintes (footprints).

Lors de la création de circuits électroniques, vous allez devoir chercher plein d'informations dans différents documents (datasheets des composants). Vous devez avoir également une connexion internet tout au long du projet pour aller chercher diverses informations, datasheets, packages, etc...

**Piège 1** : Utiliser des composants non disponibles, ce qui oblige à tout refaire.

## 2. Créer le schéma électronique (Schematic Editor)

- Ouvrir **KiCad**, créer un nouveau projet (ex : robot\_labyrinthe\_PCB), puis aller dans "Editeur de schématique" (Schematic Editor).



### Configuration du microcontrôleur :

Broches utilisées pour communiquer avec les différents composants de votre circuit (à l'aide du logiciel STM32CubeIDE) :

- Les broches PA10 et PA11 seront utilisées pour la liaison **UART**, vers l'émetteur, le récepteur IR et le connecteur SWD
- La broche PA8 est utilisée pour contrôler une LED de status LED de statut (affiche si la carte est active),
- Les broches PA13 et PA14 sont utilisées lors de la programmation du microcontrôleur (SWDIO, SWCLK)
- Les alimentations sont repérées par VDD, VDDA (Alimentation logique et analogique) et VSS (masse du circuit),
- La broche NRST (non-reset) permet de redémarrer le processeur à l'état bas,

- Tous les autres signaux (STEP, DIR, ENABLE, capteurs, boutons) sont connectés sur d'autres GPIOs libres (ex. : PBx, PCx...).

Broche STM32L476	Signal dans le schéma	Fonction réelle
PA10	USART2_RX	Réception UART (vers IR RX)
PA11	USART2_TX	Émission UART (vers IR TX)
PA8	LED_STATUS	LED de statut allumée quand active
PA13	SWDIO	Programmation (debug)
PA14	SWCLK	Programmation (horloge debug)
NRST	NRST	Bouton de reset connecté (SW1)
PB1, PB2 et PB3	MS1, MS2, MS3	Sélection microstepping moteurs
PB11, PB12 et PB13	STEP, DIR, ENABLE	Contrôle des moteurs A4988
PC10	LED_MODE, SW3	LED ou boutons utilisateur
PC15	SW2	Alimenter la carte
PC1	TRIG	Émission ultrason
PC2	ECHO	Réception ultrason
VDD, VDDA, VBAT	VDD, VDDA, VBAT	Alimentation logique + analogique (3.3 V)
VSS, VSSA	VSS, VSSA	Masse du système : Toutes les masses sont reliées ensemble

Tout le reste du projet s'articulera autour de ce microcontrôleur.

- **Organiser les blocs fonctionnels** : alimentation, microcontrôleur, moteurs, capteurs, LED...

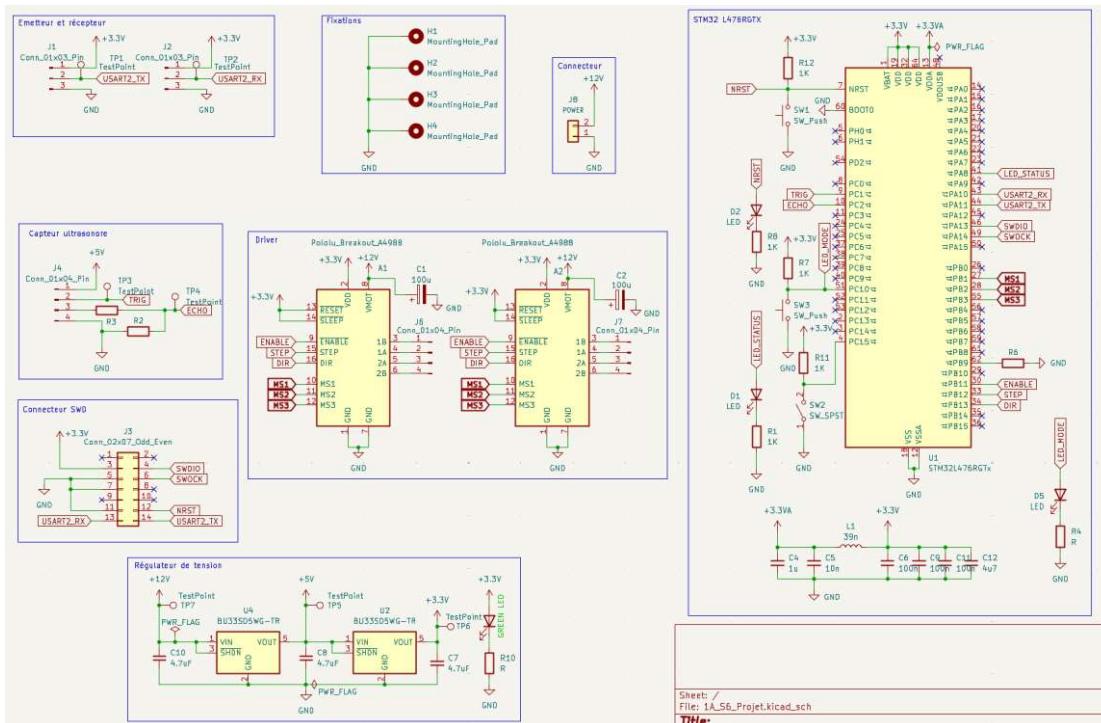
Blocs	Composants
<b>Microcontrôleur</b>	<ul style="list-style-type: none"> <li>○ STM32L476RG</li> <li>○ Connecteur SWD (SWDIO, SWCLK, NRST) Résistances</li> <li>○ Condensateurs de découplage</li> <li>○ Labels</li> </ul>
<b>Alimentation</b>	<ul style="list-style-type: none"> <li>○ Connecteur d'alim 2 broches (batterie 12V)</li> <li>○ Un régulateur BU33SD5WG-TR pour 5V</li> <li>○ Un régulateur BU33SD5WG-TR pour 3.3V</li> <li>○ Condensateurs de découplage</li> </ul>
<b>Drivers moteurs</b>	<ul style="list-style-type: none"> <li>○ 2 × A4988 avec signaux STEP, DIR, ENABLE, MS1–3, VMOT (12V), VDD (3.3V)</li> <li>○ Moteurs : Connecteurs 4 broches reliés à 1A–2B</li> </ul>
<b>Capteur ultrasonore</b>	<ul style="list-style-type: none"> <li>○ HC-SR04 (TRIG, ECHO),</li> <li>○ Connecteur 4 broches</li> <li>○ Résistances pour adaptation de niveau</li> </ul>
<b>Emetteur et récepteur</b>	<ul style="list-style-type: none"> <li>○ Deux connecteurs 3 broches</li> <li>○ IR TX/RX (USART2_TX / RX) au STM32</li> </ul>
<b>Tests &amp; interaction</b>	<ul style="list-style-type: none"> <li>○ 3 boutons (SW1–SW3)</li> <li>○ 3 LED (statut, mode et reset)</li> <li>○ Points tests 12V/5V/3.3V</li> <li>○ GND</li> <li>○ Pastilles de fixation à la masse</li> </ul>

### Segmentation de votre projet :

Dans tous les projets de la réalisation de carte électronique, vous aurez différents composants qui réaliseront différentes tâches, essayer dès le début de lister ces composants et de leur dédier une zone. Avec cet outil  délimiter 8 zones pour les différents composants :

1. Microcontrôleur STM32L476,
2. Connecteur SWD pour la programmation du microcontrôleur,
3. Capteur ultrasonore
4. Régulateurs de tension BU33SD5WG-TR,
5. Connecteurs de l'émetteur et du récepteur,
6. Drivers
7. Connecteur alimentation
8. Trous de fixation.

Rajouter un titre avec l'outil de texte T ou la touche raccourci T à chacune de ces sections. Une fois que vous aurez rempli ces sections (dans la suite du projet), cela devrait ressembler à quelque chose de ce genre :



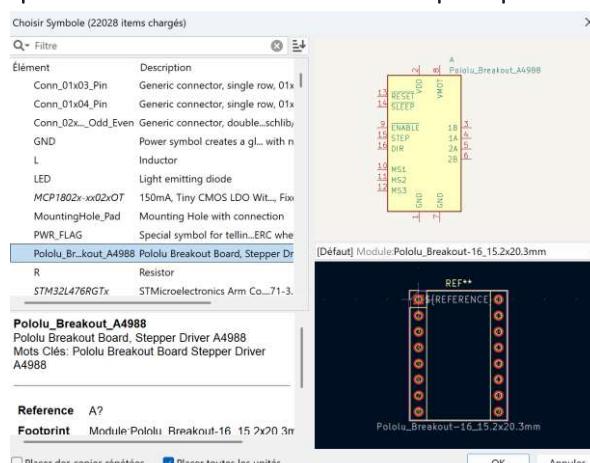
Vue d'ensemble du schéma électrique

## Ajout des symboles des composants :

Pour rajouter un symbole dans votre schéma, vous avez trois solutions :

- Depuis le menu supérieur : Place > Add symbol
- Depuis le menu latéral droit :
- Avec le raccourci clavier a

Une fenêtre s'ouvre, on peut effectuer une recherche rapide pour le composant souhaité.



Dans la zone préparée à cet effet, importer le symbole du microcontrôleur.

## Ajout des entrées/sorties, VDD, VDDA et VSS :

+12V	Power symbol creates a global label with name "+12V"
+3.3V	Power symbol creates a global label with name "+3.3V"
+3.3VA	Power symbol creates a global label with name "+3.3VA"
+5V	Power symbol creates a global label with name "+5V"

On choisira une tension de 3.3V fournie par le régulateur de tension. Rajouter sur le schéma les alimentations et la masse avec l'outil . Placer les condensateurs de découplage des alimentations. On utilisera le symbole C\_small des librairies. Saisir les valeurs de chaque condensateur dans le champ value (à la place de C\_Small). Placer cette inductance (symbole L\_small). On choisira une valeur de 39nH pour cette inductance. \_small veut juste dire que le symbole est "petit" pour éviter de prendre trop de place sur le schéma.

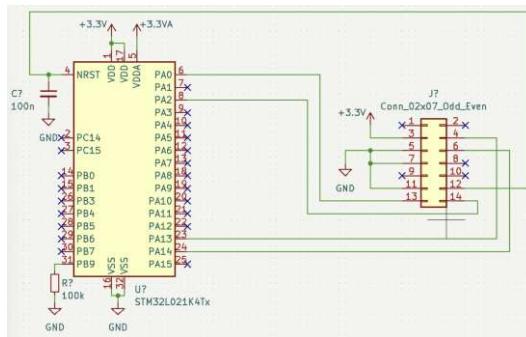
## Label & Global label :

Les labels permettent de relier électriquement des broches des différents composants sans devoir réaliser des traits d'une branche à l'autre :

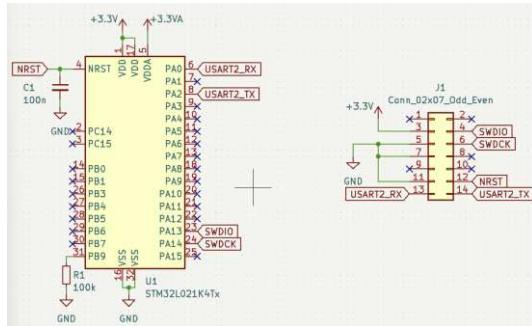
- La lisibilité du schéma est grandement augmentée,
- Lorsque vous allez passer à la phase de routage, les pistes auront le nom du label (et non pas un nom générique qui ne voudra pas dire grand-chose).



Deux types de label coexistent, mais ce qu'on va utiliser pour ce projet sont les labels globaux.



Exemple de câblage d'un microcontrôleur sans label



Exemple de câblage d'un microcontrôleur avec label

## NRST :

Le signal NRST (non-Reset) est un signal particulier. Si la tension de l'entrée NRST est nulle, alors le microcontrôleur redémarre. En interne, le microcontrôleur possède une résistance de pull-up qui maintient sauf signal contraire la broche à l'état haut.

Pour éviter tout phénomène parasite et redémarrage non contrôlé, on rajoute un condensateur de 100nF entre NRST et GND.

**Remarque :** Il faut garder le label NRST car il faut connecter ce signal au connecteur qui sert à programmer le microcontrôleur. Durant la phase de programmation, la sonde externe a besoin de faire un reset du microcontrôleur, il est donc indispensable de pouvoir récupérer ce signal ailleurs dans le schéma.

## LED :

Rajouter la LED de status sur votre schéma avec une résistance en série pour limiter le courant. On choisira une Led verte au format 0603.

Broches inutilisées

Après avoir câblée toutes les broches non grisées du schéma de microcontrôleur Rajouter des flags de non-connexion ✗ sur chaque broche restée libre. L'outil de contrôle électrique vous génère une erreur si vous ne spécifiez pas qu'une broche n'est pas utilisée.

Le schéma du microcontrôleur devrait ressembler à quelque chose de similaire à ça :

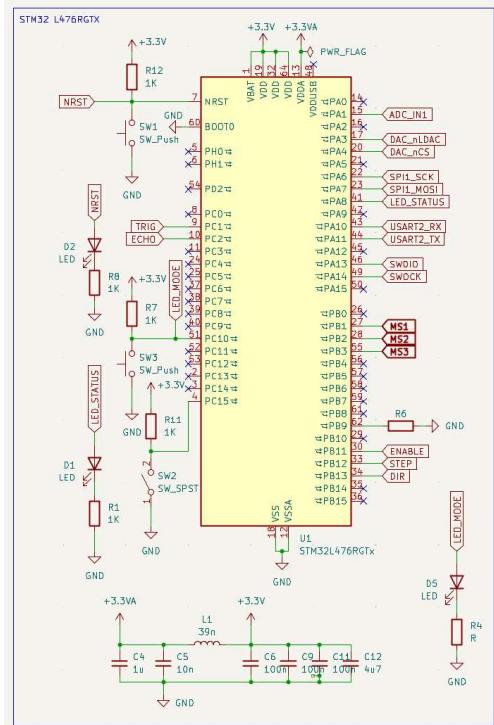


Schéma électrique du microcontrôleur STM32L476

## Régulateur de tension :

Pour générer les tensions de 3.3V et 5V à partir du 12V, il faut utiliser le régulateur linéaire LDO BU33SD5WG-TR. Ajouter le composant MCP1802x-xx02x0T. Et le renommer en BU33SD5WG-TR. Lire les 3 premières pages de la datasheet du régulateur et compléter le schéma selon les recommandations constructeurs. Rajouter une LED sur le bus de tension 3.3V afin qu'elle s'allume quand la tension de sortie du régulateur est bien à 3.3V. On prendra une Led SMD identique à celle utilisé pour le signal status du microcontrôleur.

**Power Flag :** On ne le voit pas, mais les broches des composants sont de plusieurs types :

- Input
- Output
- Power Input
- Power Output
- Input/Output

C'est utile lors de la vérification automatique du schéma : 2 sorties ne peuvent pas être câblées ensemble, cela risque de produire un court-circuit et endommager les composants. L'entrée du régulateur est de type *Power Input* et la sortie *Power Output*.

Tout entrée Power Input doit être câblé à un Power Output, cependant, nous n'avons pas de Power Output pour le 12V car il provient d'une alimentation externe du schéma via un connecteur.

Pour éviter les erreurs lors de la vérification, il faut rajouter un *PWR\_FLAG* en parallèle du +12V. Ce flag n'a aucun effet sur votre schéma, il sert juste à dire à KiCad que vous avez bien conscience qu'il faudra amener cette alimentation de l'extérieur. Rajouter le flag sur la tension +5V et sur le signal GND  *PWR\_FLAG*. Le résultat de cette partie devrait ressembler à cela :

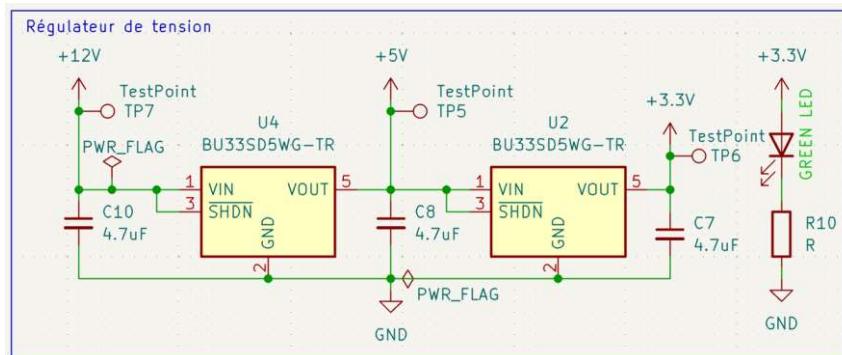


Schéma électrique des régulateurs de tension BU33SD5WG-TR

## Drivers moteurs :

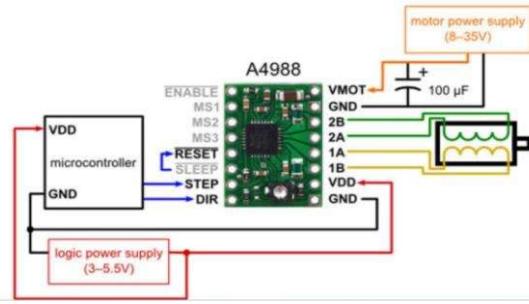
Pour piloter les moteurs pas-à-pas à partir des signaux logiques générés par le microcontrôleur, on utilise deux modules **Pololu Breakout A4988**. Chaque driver est chargé de recevoir les signaux de contrôle (STEP, DIR, ENABLE) et d'alimenter les bobines du moteur via les sorties 1A, 1B, 2A, 2B.

Le composant utilisé dans le schéma est le **Pololu\_Breakout\_A4988**, à placer depuis la bibliothèque correspondante dans KiCad. Il est alimenté en **12 V sur la broche VMOT** pour fournir la puissance nécessaire au moteur, et en **3.3 V sur VDD** pour l'alimentation logique du circuit interne.

Les broches MS1, MS2, MS3 permettent de configurer le mode de microstepping (1/1, 1/2, 1/4, 1/8, 1/16). Elles sont ici reliées à des broches du microcontrôleur pour permettre une configuration logicielle.

Il est **indispensable** d'ajouter un **condensateur électrolytique de 100 µF** entre **VMOT et GND**, proche des driver, afin d'**absorber les pointes de courant** et protéger le circuit.

Enfin, chaque sortie du driver (1A, 1B, 2A, 2B) est reliée à un **connecteur 4 broches** destiné à recevoir un moteur pas-à-pas bipolaire.



Datasheets du driver A4988

Le résultat de cette partie devrait ressembler à cela :

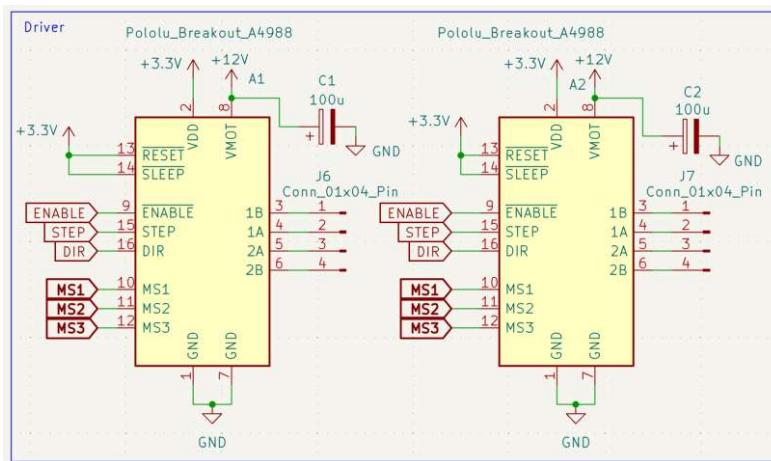


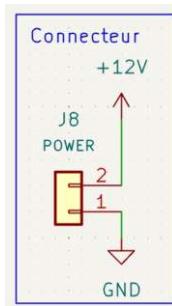
Schéma électrique des drivers A4988

## Connecteurs :

Rajouter les connecteurs pour interfaçer votre carte avec l'extérieur :

Un connecteur 2 pins (Conn\_01x02 dans la librairie Connector\_Generic) d'alimentation :

- GND
- +12V



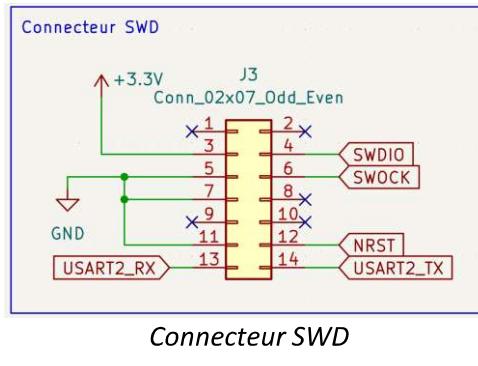
Connecteur d'alimentation

Un connecteur 14 pins (Conn\_02x07\_Odd\_Even dans la librairie Connector\_Generic) pour programmer le microcontrôleur :

- GND
- +3.3V

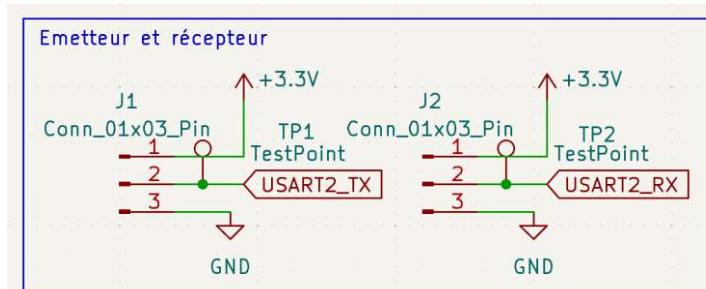
- USART2\_RX et USART2\_TX
- SWDIO et SWDCK
- NRST

A partir de la [datasheet](#) de la sonde de programmation STLINK-V3, câbler correctement les pins du connecteur (référence STDC14 dans la documentation). Repérer les broches non utilisées par ✗



Deux connecteurs 3 pins (Conn\_01x03 dans la librairie Connector\_Generic) pour l'émetteur et le récepteur :

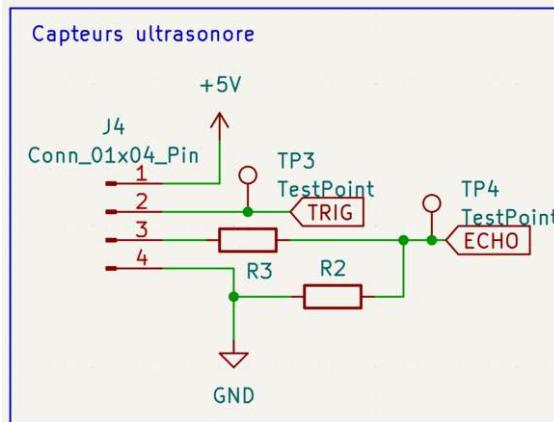
- GND
- +3.3V
- USART2\_TX / USART2\_RX



Connecteurs pour l'émetteur et le récepteur infrarouges

Un connecteur (Conn\_01x04 dans la librairie Connector\_Generic) pour le capteur ultrasonore :

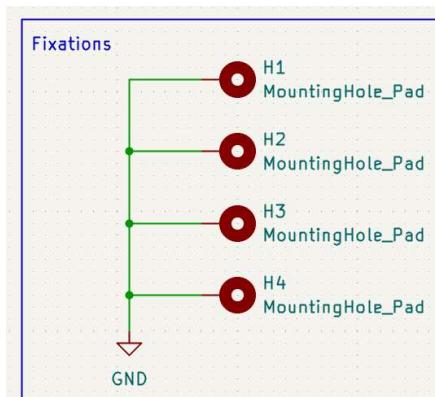
- GND
- +5V
- Trigger
- Echo



Connecteur capteur ultrasonore

### Fixations :

Un circuit électronique se fixe toujours quelque part. Il faut donc ajouter 4 trous de fixations (MountingHole\_Pad) reliés à la masse.



**Piège 2** : Oublier de relier certaines masses ou alimentations → bugs invisibles.

### Astuce :

- ✓ Utilise des labels (noms des signaux) clairs pour chaque net au lieu de tout relier avec des fils → plus lisible.
- ✓ Ajouter les labels (global ou non ) sur l'ensemble des broches utilisées.
- ✓ Ajouter **les alimentations avec les symboles "Power"** et vérifier les masses.

## 3. Analyse de la qualité du schéma : Vérification du schéma (ERC)

- Lancer l'**ERC (Electronic Rules Check)** en cliquant sur le symbole . L'ensemble des erreurs électriques y sont listées.
- En cliquant sur les erreurs/warnings, le logiciel vous montre très exactement où sont les problèmes.

- Modifier le schéma et corriger tous les **warnings** : pins non connectées, alimentations manquantes, etc. pour que toutes les erreurs soient supprimées.



**Piège 3 :** Ignorer les avertissements ERC "parce que ça a l'air bon visuellement". L'ERC peut sauver votre projet. Corrigez tout et si vous n'y arrivez pas demandez de l'aide au professeur.

Astuces :

- ✓ Posez-vous la question de la lisibilité de votre schéma et des commentaires éventuels
- ✓ Justifiez vos choix de valeurs de composants,
- ✓ Organisez au mieux vos différentes parties,
- ✓ Rajoutez des labels si nécessaire pour supprimer des liens trop long.

## 4. Choisir les empreintes (footprints)

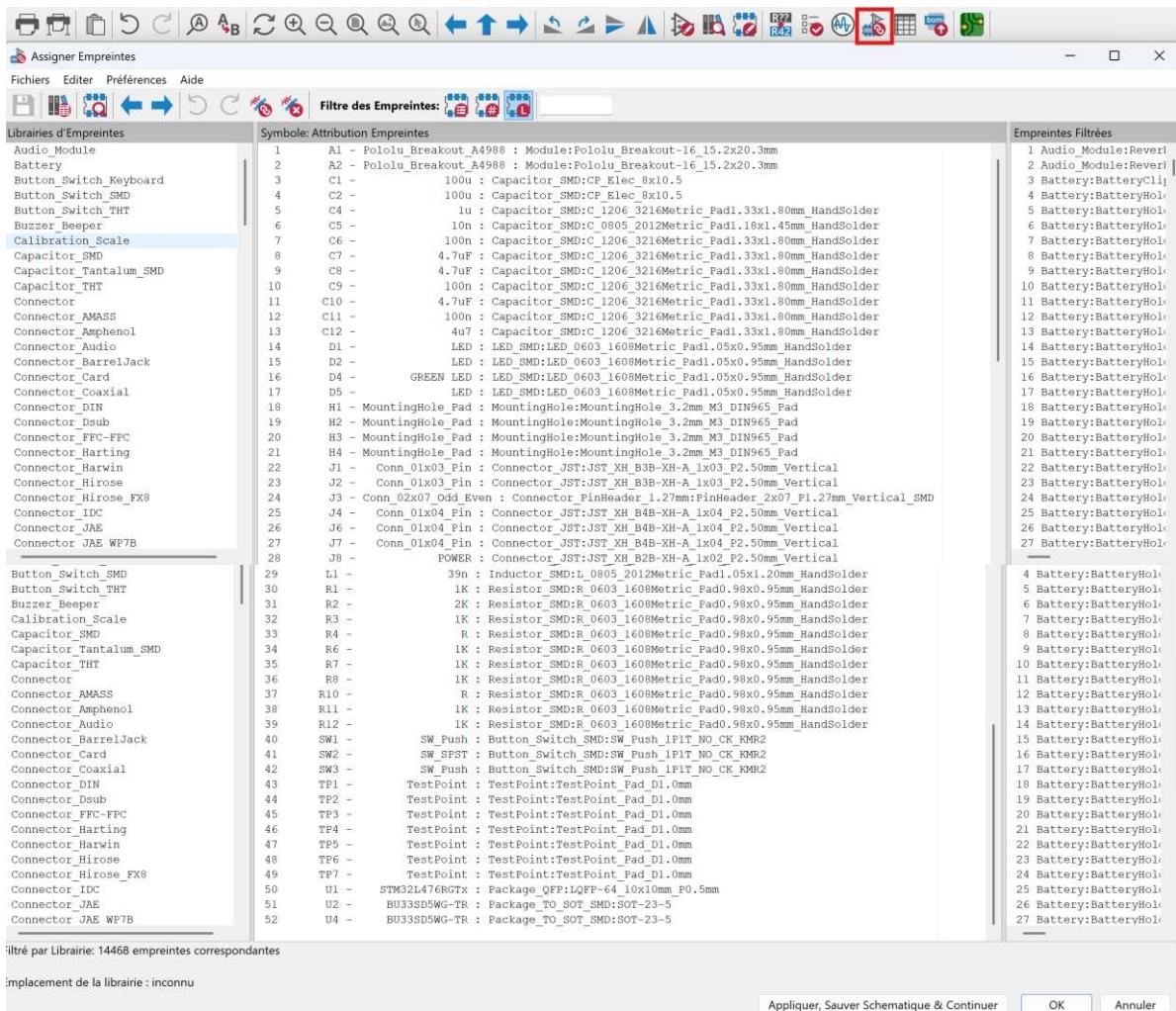
Jusqu'à présent, vous n'avez réalisé qu'un schéma de principe de votre carte électronique. Chaque composant a une empreinte qui représente les dimensions du composant. Un même composant peut exister dans différents "packages" et donc avoir une empreinte (footprint) différente. Ouvrir l'outil d'affectation des empreintes

- Associer **chaque composant à une empreinte réelle**, compatible avec le stock de l'école ou un composant qu'on peut commander.
- Pour les connecteurs, choisir les modèles avec **pas standard (2.50 mm)**.
- Pour les LED, résistances, boutons : utiliser les modèles traversants sauf si soudure CMS maîtrisée.

La figure ci-dessous est divisée en 3 colonnes :

1. Les librairies d'empreinte disponibles
2. Les symboles de votre schéma et leurs empreintes associés (association non réalisée)

### 3. Les empreintes possibles pour chaque composant



#### Choix et attribution des empreintes

Le terme "HandSolder" agrandie légèrement les empreintes pour que ce soit plus facile à souder à la main.

- Pour les condensateurs choisir (on peut sélectionner plusieurs condensateurs en même temps, pour leurs assigner les empreintes tous en même temps) : Capacitor\_SMD:C\_0603\_1608Metric\_PAD1.08x0.95mm\_HandSolder.
- Faire de même pour les résistances (en 0603), l'inductance (en 0805, n'est pas en stock en 0603 à l'école) et les leds (en 0603) : en changeant bien évidemment la librairie.
- Pour le connecteur de 2 broches, il faut choisir un connecteur JST\_XH\_B2B-XH-A vertical avec un pas de 2.5mm.
- Pour les 2 connecteurs de 3 broches, il faut prendre un connecteur JST\_XH\_B3B-XH-A vertical avec un pas de 2.5mm.
- Pour les 3 connecteurs de 4 broches, il faut prendre un connecteur JST\_XH\_B4B-XH-A vertical avec un pas de 2.5mm.

- Pour le connecteur de la sonde de programmation (SWD), il faut choisir un pinheader SMD vertical avec un pas de 1.27mm.
- Trous de fixation : Pour les trous de fixation, les PCB sont fixés par des vis M3 (diamètre de 3mm). Pour garder un peu de jeu, on réalise des perçages de 3.2mm.

#### Piège 4 : Mélanger composants CMS (SMD) et traversants sans le vouloir.

- ✓ **Astuce :** Faire un tableau avec composants → empreinte → boîtier réel pour vérifier.

#### Génération de la BOM (Bill of Material) :

La BOM est la liste exhaustive de vos composants. Vous pouvez directement l'exporter et vous en servir pour :

- Demander un chiffre et un envoi de tous les composants nécessaires,
- L'imprimer et l'avoir avec vous quand vous allez aller chercher les composants dans les bureaux des différents techniciens

Cliquer sur l'icône dans le menu supérieur  puis sur Generate. On a alors un simple fichier .csv dans le dossier de votre projet :

Table Champs de Symboles	
Editer	Exporter
Délimiteur de champ:	,
Délimiteur de texte:	"
Délimiteur de référence:	,
Délimiteur de plage:	
<input type="checkbox"/> Garder Tabulations	
<input type="checkbox"/> Conserver les sauts de ligne	
Préréglage de format:	
CSV	
Fichier de sortie: ..\1A_S6_Projet.csv	
Prévisualisation:	
<pre>"Reference","Value","Footprint","Datasheet","Qty","DNB","Exclude from BOM","Exclude from Board" "A1,A2","Pololu_Breakout_A4988","Module:Pololu_Breakout-16_15.2x20.3mm","https://www.pololu.com/product/29 "C1,C2","100u","Capacitor_SMD:CP_Elec_8x10.5","~","2","","" "C4","1u","Capacitor_SMD:C_1206_3216Metric_Pad1.33x1.80mm_HandSolder","~","~","~","~","~","~" "C5","10n","Capacitor_SMD:C_0805_2012Metric_Pad1.18x1.45mm_HandSolder","~","~","~","~","~","~" "C6,C9,C11","100n","Capacitor_SMD:C_1206_3216Metric_Pad1.33x1.80mm_HandSolder","~","~","~","~","~","~" "C7,C8,C10","4.7uF","Capacitor_SMD:C_1206_3216Metric_Pad1.33x1.80mm_HandSolder","~","~","~","~","~","~" "C12","4uF","Capacitor_SMD:C_1206_3216Metric_Pad1.33x1.80mm_HandSolder","~","~","~","~","~","~" "D1,D2,D5","LED","LED_SMD:LED_0603_1608Metric_Pad1.05x0.95mm_HandSolder","~","~","~","~","~","~" "D4","GREEN LED","LED_SMD:LED_0603_1608Metric_Pad1.05x0.95mm_HandSolder","~","~","~","~","~","~" "H1,H2,H3,H4","MountingHole_Fad","MountingHoleMountingHole_3.2mm_M3_DIN965_Pad","~","~","~","~","~","~" "J1,J2","Conn_01x03_Pin","Connector_JST:JST_XH_B3B-XH-A_1x03_P2.50mm_Vertical","~","~","~","~","~","~" "J3","Conn_02x07_Odd_Even","Connector_PinHeader_1.27mm:PinHeader_2x07_P1.27mm_Vertical_SMD","~","~","~","~","~","~" "J4,J6,J7","Conn_01x04_Pin","Connector_JST:JST_XH_B4B-XH-A_1x04_P2.50mm_Vertical","~","~","~","~","~","~" "J8","POWER","Connector_JST:JST_XH_B2B-XH-A_1x02_P2.50mm_Vertical","~","~","~","~","~","~" "R1","Resistor_SMD:R_0603_1608Metric_Pad0.98x0.95mm_HandSolder","~","~","~","~","~","~" "R2","2K","Resistor_SMD:R_0603_1608Metric_Pad0.98x0.95mm_HandSolder","~","~","~","~","~","~" "R4,R10","R","Resistor_SMD:R_0603_1608Metric_Pad0.98x0.95mm_HandSolder","~","~","~","~","~","~" "SW1,SW3","SW_Push","Button_Switch_SMD:SW_Push_1P1T_NO_CK_KMR2","~","~","~","~","~","~" "SW2","SW_SPT","Button_Switch_SMD:SW_Push_1P1T_NO_CK_KMR2","~","~","~","~","~","~" "TP1,TP2,TP3,TP4,TP5,TP6,TP7","TestPoint","TestPoint_Pad_D1.0mm","~","~","~","~","~","~" "U1","STM32L476RGTX","Package_QFP:LQFP-64_10x10mm_P0.5mm","https://www.st.com/resource/en/datasheet/stm32l "U2,U4","BU33SD5WG-TR","Package_TO_SOT_SMD:SOT-23-5","http://ww1.microchip.com/downloads/en/DeviceDoc/2205</pre>	
Exporter	
Appliquer, Sauver Schématique & Continuer	
OK	
Annuler	

Après avoir réalisé le schéma de la carte, il faut à présent réaliser le routage de celle-ci. Cette partie se décompose en 12 étapes :

1. Import des composants,
2. Choix de la grille de travail,
3. Placement des trous de fixation et création du contour de la carte,

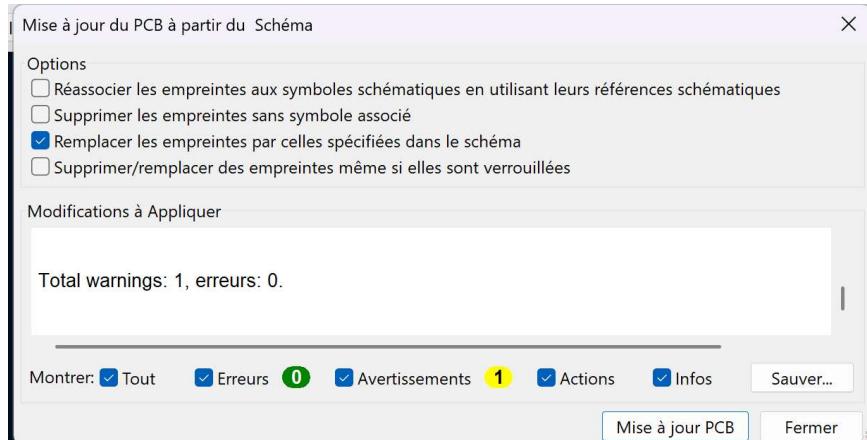
4. Création du plan de masse,
5. Placement des connecteurs, et des composants volumineux,
6. Placement des composants passifs (RLC),
7. Etablissement des classes d'équipotentielles,
8. Routage,
9. Analyse du routage (vérification ERC)
10. Finalisation du PCB pour une impression à l'ENSEA

## 5. Passer au PCB (PCB Editor)

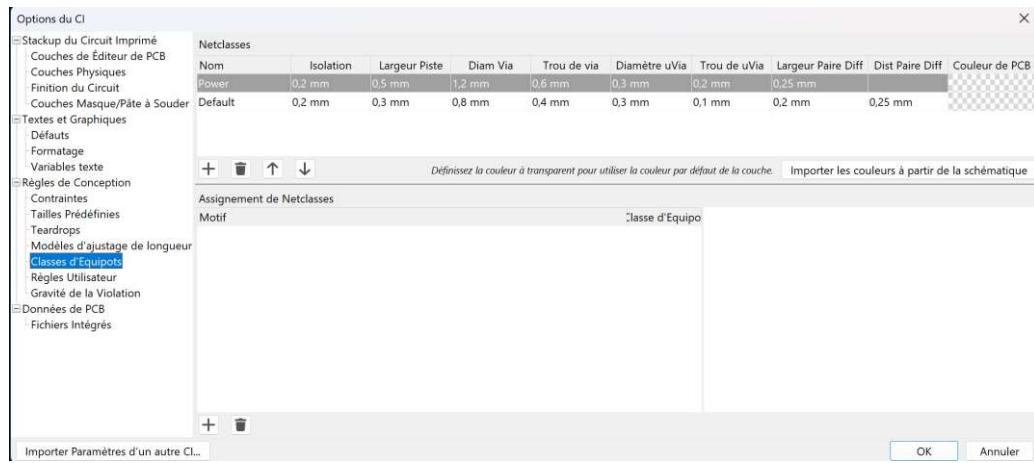
- Lancer le PCB Editor : Soit depuis le schéma avec l'icône  dans le menu supérieur ou soit depuis le gestionnaire de projet avec



- Importer les composants de depuis le schéma avec le bouton  se trouvant dans le menu supérieur. Puis cliquer sur Update PCB et Close.



- Délimiter l'espace de travail et la taille de votre PCB.
- **Placer les composants par blocs logiques** : alimentation, microcontrôleur, moteurs, capteurs.
- **Respecter les distances minimales** et orienter les composants de manière logique.

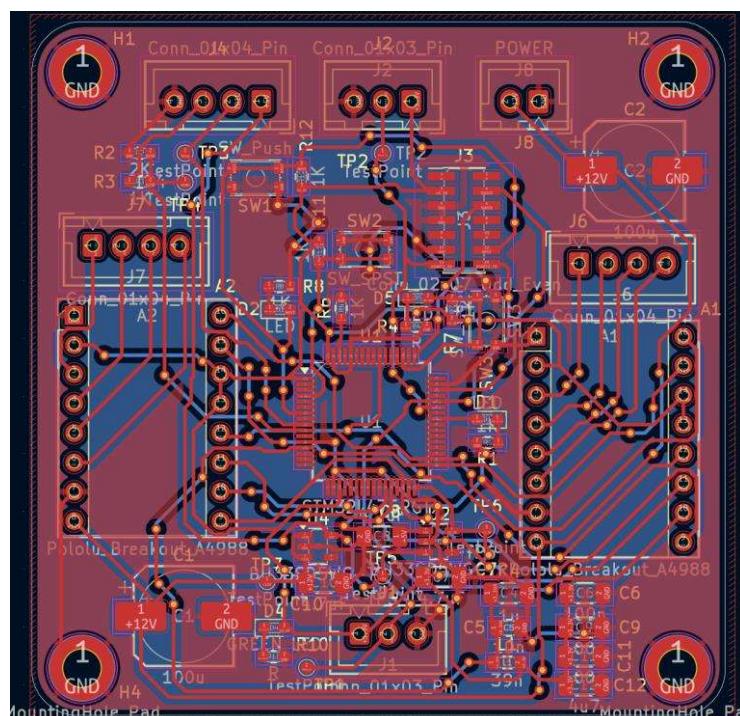


**Piège 5 :** Mettre tous les composants au hasard → pistes trop longues et croisement partout.

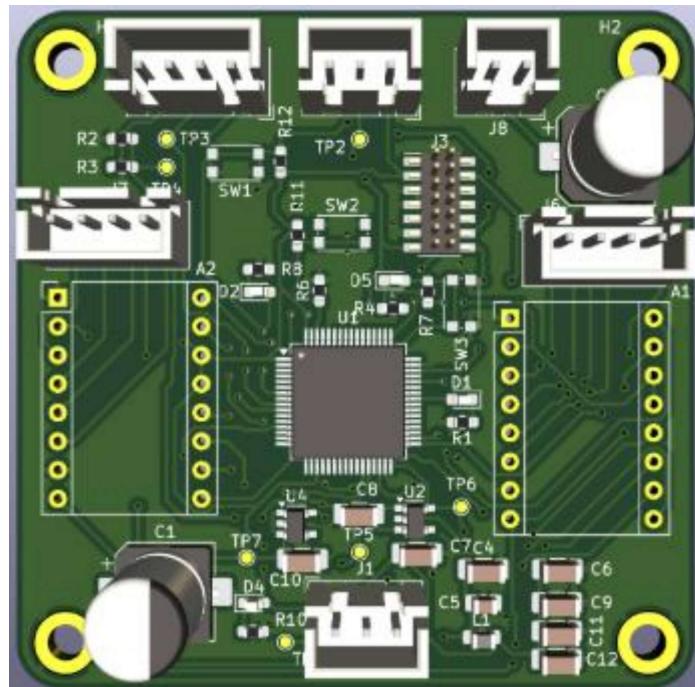
- ✓ **Astuce :** Regarder votre schéma fonctionnel avant de placer les composants pour être cohérent dans le placement.

## 6. Faire le routage (les pistes)

- Utiliser **deux couches** de cuivre (F.Cu : dessus/B.Cu : dessous (rouge et bleu))
- Garder les pistes **courtes, droites mais éviter les angles droits et larges**.
- Utiliser des **vias intelligemment**, pas partout.
- Ajouter un **plan de masse** pour éviter les perturbations.
- Vérifier les **tailles de vias et pistes** selon les capacités de l'atelier ou du Fablab.
- Ne pas oublier d'ajouter les **pastilles de fixation**, si la carte est vissée.



Routage du PCB (avec l'éditeur PCB de KiCad)

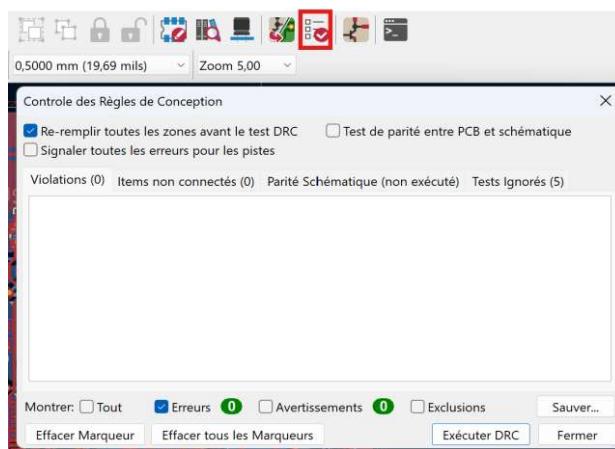


Vue de haut du PCB en 3D (avec l'éditeur PCB de KiCad)

**Piège 6** : Trop de vias, ou des pistes trop fines → carte non imprimable ou plus de temps pour l'imprimer.

## 7. Vérification finale (DRC)

- Lancer le **DRC (Design Rule Check)** pour vérifier :
  - Espace minimum respecté
  - Pas de court-circuit
  - Toutes les connexions faites



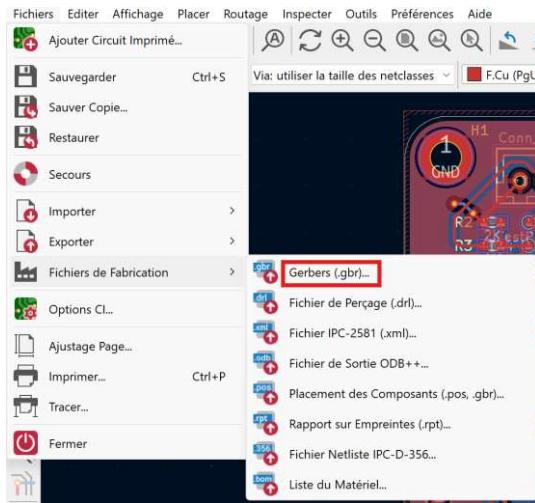
- Corriger **absolument tout**.

**Piège 7** : Négliger le DRC ou corriger "à moitié".

- ✓ **Astuce** : Faire valider par un camarade ou un prof → second regard utile.

## 8. Export pour fabrication

- Générer les fichiers Gerber et fichiers de perçage (drill).



**Piège 8** : Trop de vias, ou fichiers mal nommés → impression impossible.

- ✓ **Astuce** : Toujours tester avec un viewer Gerber avant d'envoyer à l'impression.

## Conseils bonus

- ✓ Ajouter des points de test (TestPoints) sur les alimentations et les signaux importants.
- ✓ Ajouter une LED d'alim (3.3V) pour vérifier si la carte est bien alimentée.
- ✓ Penser aux trous de fixation dès le début, pas à la fin.
- ✓ Toujours vérifier les dimensions physiques des connecteurs.
- ✓ Minimiser la longueur des pistes critiques, notamment celles transportant des signaux rapides (par exemple : STEP, DIR, USART, SPI).
- ✓ Utiliser un plan de masse complet (VSS, GND) sur les deux couches du PCB afin d'améliorer la stabilité, la compatibilité électromagnétique et la réduction du bruit.
- ✓ Respecter les largeurs minimales de pistes et les espacements définis par les règles de fabrication :
  - Pistes de puissance (VMOT 12 V)  $\geq 0,8 \text{ mm}$
  - Pistes logiques (3,3 V)  $\geq 0,3 \text{ mm}$
- ✓ Placer les Vias au plus court chemin entre les points à relier, et éviter les vias inutiles.
- ✓ Placer les condensateurs de découplage au plus près des broches VDD/VDDA des composants critiques (STM32, régulateurs).
- ✓ Optimiser l'orientation des composants pour limiter les croisements de pistes et simplifier le routage.