Une image contenant Police, logo, Graphique, texte

Description générée automatiquement

Félix-Antoine Guimont

Marc-Étienne Gendron-Fontaine

Groupe : 2317

**CANVision**

Manuel technique présenté à :

Benoit Beaulieu

Simon-Pierre Gagnon

Louis-Philippe Gauthier

Université de Sherbrooke (FUSE)

Département du génie électrique

Pour le cours

Projet de fin d’études

Cégep de Sherbrooke

30 mai 2024

Table des matières

[**1.** **Description générale** 3](#_Toc167618428)

[1.1 À propos 3](#_Toc167618429)

[1.2 Explication générale du projet 4](#_Toc167618430)

[**2** **Fonctionnement** 5](#_Toc167618431)

[**3** **Procédure d’installation et d’opération** 7](#_Toc167618432)

[3.1 Installation du système 7](#_Toc167618433)

[3.2 Comment opérer l’écran 9](#_Toc167618434)

[**4** **Contenu matériel** 10](#_Toc167618435)

[4.1 Régulateur de tension 10](#_Toc167618436)

[4.2 Connecteur CAN Bus et Transmetteur 10](#_Toc167618437)

[4.3 Connecteur de l’écran 11](#_Toc167618438)

[**5** **Contenu logiciel** 12](#_Toc167618439)

[5.1 Logiciel Utilisé 12](#_Toc167618440)

[5.2 Librairie utilisé 12](#_Toc167618441)

[5.3 Compréhension du code principale 13](#_Toc167618442)

[**6** **Procédure de développement** 14](#_Toc167618443)

[6.1 Téléverser dans l’écran 14](#_Toc167618444)

[6.2 Installation des logiciels et des librairies 14](#_Toc167618445)

[6.3 Modifier l’affichage du projet 17](#_Toc167618446)

[6.4 Modifier le code CAN 18](#_Toc167618447)

[**7** **Listes de matériel et coûts** 20](#_Toc167618448)

[**8** **Modifications et améliorations** 23](#_Toc167618449)

[**9** **Annexe** 24](#_Toc167618450)

[9.1 Annexe 1 : Schémas électrique 24](#_Toc167618451)

# **Description générale**

## **À propos**

Le projet consiste à afficher des informations dans le but de diagnostiquer une voiture de Formule SAE durant la compétition aux États-Unis. Mais c’est quoi une Formule SAE ? Ce sont des épreuves de voiture de course monoplace. Cette compétition a pour objectif la formation de futurs professionnels à l’ingénierie de la mobilité. Il y a quatre catégories de véhicule pour cette compétition : véhicule thermique, hybride, électrique et autonome.

Une image contenant Course de voitures, roue, sport, Voiture de course

Description générée automatiquement

Figure 1: Voiture de Formule SAE

Ce projet a démarré à l’automne 2023 et nous sommes la première équipe à faire ce projet. Le but est de créer une interface de diagnostic pour les mécaniciens de la monoplace. Les clients sont des étudiants de l’Université de Sherbrooke en Génie Électrique. Notre projet est d’installer un écran dans la partie arrière de la voiture pour aider les mécaniciens durant la compétition. Cet écran est connecté au réseau CAN de la monoplace.

Une image contenant croquis, voiture, illustration

Description générée automatiquement

Figure 2: Illustration du Réseau CAN

## **Explication générale du projet**

Le but du projet est d’afficher les données des capteurs pour pouvoir faire des diagnostics de la monoplace. Présentement, nous utilisons un affichage de sept pouces de la compagnie 4D system. Aussi, nous avons réalisé un PCB qui réceptionne le réseau CAN. Ce PCB est alimenté avec du 12V de la Drive du véhicule. Les données reçues sont mises à jour en temps réel (0,2 seconde) sur l’écran. Sur cette dernière, nous avons trois pages de diagnostics : HV (accumulateur), Temp (Température), LV (Batterie). Ces trois pages contiennent environ 3 ou 4 widgets. Par exemple, la page HV contient 4 widgets qui vérifient sa santé, sa charge, sa température et sa tension.

A person standing next to a computer screen

Description automatically generated

Figure 3: Illustration du projet

# **Fonctionnement**

Sur le schéma ci-dessous, on remarque que le projet est constitué de deux sections principales. Le PCB et l’écran.

* Le PCB s’occupe de s’interfacer au CAN bus pour convertir les signaux électriques vers les signaux CAN RX et CAN TX qui seront lus par l’ESP32 de l’affichage.
* L’écran contient un ESP32-S3R8. Il reçoit les données en série CAN.

A diagram of a computer

Description automatically generated

Figure 4: Schémas conceptuel

Le réseau CAN en différentiel et l’alimentation de 12V sont fourni dans le connecteur 4 pin vers le PCB.

Le régulateur (LM2675M-5.0), qui est une alimentation à découpage, prend l’alimentation de 12V provenant de la batterie et l’abaisse à 5V.

Le transmetteur (TJA1051T) reçoit les trames CAN. Ces trames contiennent des données de l’état de certaines composantes du véhicule telles que : l’accumulateur (HV), la batterie (LV) ainsi que des températures. Il reçoit une tension de 3v3 pour que le ESP32-S3R8 se communique bien ensemble. Le transmetteur envoie les trames reçues au ESP32-S3R8 via une communication série CAN.

Le flex de 30 pins fait la connexion entre le PCB et l’écran. Il alimente l’écran avec une tension de 5V.

Le ESP32-S3R8 reçoit une trame CAN. Il les traite les données pour les convertir en valeur entière pour que sur l’écran les données soient lisibles. Les trames sont reconnues en fonction des identifiants des capteurs. Ensuite, le ESP32-S3R8 met à jour les valeurs des widgets pour qu’ils soient visibles sur l’affichage.

# **Procédure d’installation et d’opération**

## **Installation du système**

1. Connecter l’une des extrémités du flex sur le PCB et l’autre sur l’écran comme montré ci-dessous.



Figure 5: connections interne

1. Insérer l'écran sur le boitier. L'écran devrait s’emboiter sur le boitier comme montré ci-dessous.



Figure 6: Vue extérieur

1. Voici la disposition des broches du connecteur ATM13-4P : broche 1 GND, broche 2 CAN Low, broche 3 CAN High et broche 4 12V

* Broche 1 : GND
* Broche 2 : CAN Low
* Broche 3 : CAN High
* Broche 4 : 12V

1 2 3 4

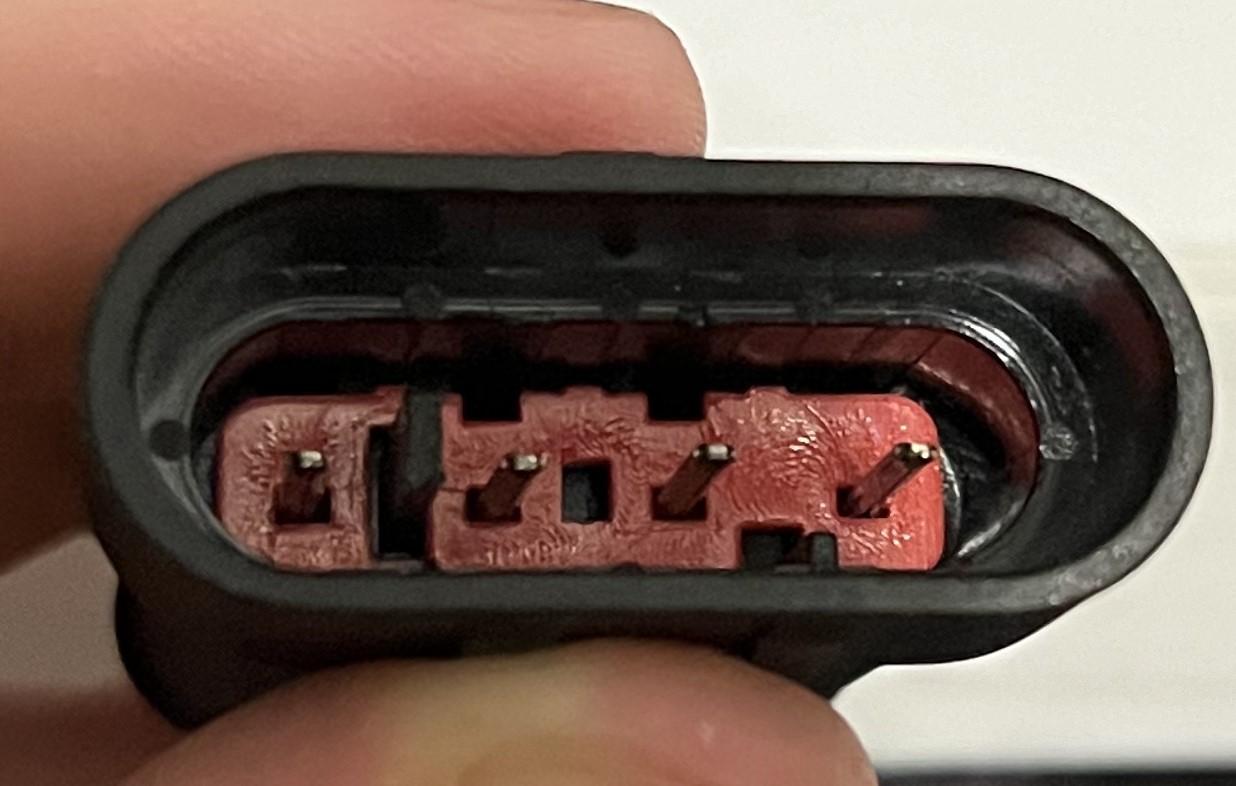


Figure 7: Connecteur

## **Comment opérer l’écran**

Une fois l’écran allumé, il affichera la page par défaut qui est « Temp ». Cependant, si l’écran ne détecte pas de réseau CAN, il y aura un message d’erreur qui affichera « NO Data CAN Recieved ». Pour changer de page, il faut simplement toucher le bouton correspondant. Les valeurs des widgets se mettent à jour de manière automatique.

Une image contenant texte, Instrument de mesure, jauge, nombre

Description générée automatiquement

Une image contenant texte, Instrument de mesure, jauge, capture d’écran

Description générée automatiquement

Une image contenant texte, Instrument de mesure, jauge, compteur

Description générée automatiquement

# **Contenu matériel**

Le schéma électrique complet est dans l’annexe. **Annexe 1 : Schémas électrique**

## **Régulateur de tension**

Une image contenant texte, diagramme, Plan, capture d’écran

Description générée automatiquement

Le circuit d’alimentation est composé d’un LM2675M-5.0 qui est un régulateur de découpage. Ce régulateur prend une tension de 12V en entrée et donne une tension de 5V en sortie et fournie 1A. Une DEL verte sert à indiquer qu’il y a une tension de 5V en sortie.

## **Connecteur CAN Bus et Transmetteur**

Une image contenant texte, diagramme, Police, ligne

Description générée automatiquement

Ce circuit est la section du réseau CAN et du transmetteur. L’alimentation et le CAN arrivent du connecteur 4 pins. Le réseau CAN va au transmetteur (TJA1051T/3, 118). Ce transmetteur reçoit une tension de 3v3 sur la patte 5 pour aider la communication entre l’écran et le TJA1051T. La patte 8 défini si le transmetteur transmet ou reçoit. Les pattes 1 et 4 sont du série CAN. Elle envoie à l’écran la trame CAN.

## **Connecteur de l’écran**

Une image contenant texte, capture d’écran, diagramme, nombre

Description générée automatiquement

Le connecteur F32Q-1A7H1-11030 fait la liaison entre le PCB et l’écran de 4D Systems. Il alimente en 5V l’écran sur 4 pins. Elle fournit aussi le 3v3 au TJA1051T sur la patte Vio. La patte 5 est le GPIO1. Elle envoie un niveau haut ou bas pour modifier le TJA1051T en émetteur ou receveur. Nous la mettons à un niveau bas pour que le transmetteur soit en receveur.

# **Contenu logiciel**

## **Logiciel Utilisé**

Le logiciel que nous utilisons est Workshop4 IDE. Ce logiciel est celui de la compagnie 4D System. Voici le lien pour l’installation : <https://4dsystems.com.au/software/>.

A computer and a box

Description automatically generated

Workshop4 est un logiciel qui fonctionne avec Arduino, cela signifie que nous devons coder en C++. Il nous permet de mettre des widgets sur l’affichage.

## **Librairie utilisé**

Il y a trois librairies que nous utilisons.

#include "gfx4desp32\_%%displaynm%%.h"

#include "Code\_Final\_CANVision\_V1\_FAConst.h"

#include <ESP32-TWAI-CAN.hpp>

La première librairie est seulement pour définir la dimension de l’écran.

La deuxième librairie est pour définir les widgets du code au complet.

La troisième librairie est pour le CAN, soit la plus importante. Pour l’utilisation du projet, nous utilisons seulement la partie de lecture de la trame CAN. Voici le créateur de cette librairie : <https://github.com/handmade0octopus/ESP32-TWAI-CAN>.

## **Compréhension du code principale**

Tout le contenu logiciel pour utiliser le projet peut être retrouvé sur le GitHub du projet. <https://github.com/FelixDuck15/projet_Formule_SAE/tree/main>

Essentiellement, tout le code réside dans le dossier projet\_Formule\_SAE/H24/LOGICIEL/AFFICHAGE\_SAE/Code\_Final\_CANVision\_V1\_FA.

Le code de la boucle principale est le suivant :

void loop()

{

btn(); // verifie les boutons s'il sont appuyer

readCAN(); // lis le BUS CAN

switch(pageValue){

case 1:

hv\_page(false); // appel la page HV

break;

case 2:

temp\_page(false); // appel la page Temp

break;

case 3:

lv\_page(false); // appel la page LV

break;

}

}

Ce code vérifie tout le temps si un bouton de l’écran a été touché. Si un de ces boutons a été touché, il va nous diriger vers la page demandée. La fonction « readCAN() » va lire les trames reçues et afficher les bonnes valeurs sur la page active.

# **Procédure de développement**

## **Téléverser dans l’écran**

1. Allez sur l’onglet « Comms » et vérifier s’il s’agit du bon port COM.
2. Allez sur l’onglet « Home » et appuyer sur « Complie ». Il va automatiquement vérifier tous les widgets et avoir une barre de progression qui va créer les fichiers des widgets qui sont des extensions « dat » et « gci ». Cliquer sur « No Thanks ».
3. Vérifiez qu’il n’y a pas d’erreur.
4. Quand tout est compilé et sans erreur, cliquez sur « Download ».
5. Prenez la carte SD à l’arrière de l’écran et mettez les fichier « dat » et « gci » qui ont été créés à l’étape 2. Remettez la carte SD dans l’écran.
6. Alimentez l’écran et il devrait faire le reste automatiquement.

## **Installation des logiciels et des librairies**

1. Installez Arduino IDE 2.0 <https://www.arduino.cc/en/software>

A screenshot of a computer program

Description automatically generated

1. Installez Workshope4 IDE (seulement sur Windows) <https://4dsystems.com.au/software/>
2. Dans « Fichier/Préférence » d’Arduino, collez la référence suivante pour avoir les cartes ESP32 dans « URL de gestionnaire de cartes supplémentaires :

<https://github.com/espressif/arduino-esp32/releases/download/3.0.0-alpha3/package_esp32_dev_index.json>

A screenshot of a computer

Description automatically generated

1. Dans « Gestionnaire de cartes », installez « esp32 » version 3.0.0-a par Espressif Systems. Le nom de la carte est « 4D Systems gen4-ESP32 16MB Modules (ESP32-S3R8n16) ».

A screenshot of a computer

Description automatically generatedA screenshot of a computer program

Description automatically generated

1. Installez la version la plus récente de la librairie « sdFat ».

A screenshot of a computer

Description automatically generated

1. Téléchargez puis installez la librairie « GFX4dESP32 » via ce lien : <https://github.com/4dsystems/GFX4dESP32/tree/arduino-esp32-v3>

A screenshot of a computer program

Description automatically generated

1. Installez la librairie « ESP32-TWAI-CAN ».

A screenshot of a computer program

Description automatically generated

Vous êtes maintenant prêts à utiliser Workshope4 IDE. Le fichier de code utilisé par Workshope4 IDE est celui avec l’extension « 4dArdUsd ».

## **Modifier l’affichage du projet**

1. Le code de 4D system est le fichier avec l’extension « 4dArdUsd ».
2. Pour modifier un widget, il faut cliquer dessus et vérifier le nom dans « object » s’il s’agit du bon widget. Changer les propriétés dans « Properties » ou dans le code. S’il est changé dans « Properties », il faut appuyer sur « Paste Static Code ».
3. Pour ajouter un widget ou une page, il faut aller dans l’onglet « Widget ». Il faut choisir entre : «Background », « Buttons », « Digits », « Gauges », « Inputs », « Labels », « Primitives », « System/Media ».

## **Modifier le code CAN**

Il y a deux parties du code CAN: void setup () et void readCAN (). Pour la partie « void setup () », le code ressemble à ceci :

// CAN

ESP32Can.setPins(CAN\_TX, CAN\_RX); // initialise les pin série CAN

ESP32Can.setSpeed(ESP32Can.convertSpeed(1000)); // initialise la vitesse de CAN

ESP32Can.setRxQueueSize(1); // Initialisation du queue size

ESP32Can.setTxQueueSize(1);

[…]

// Debut du CAN

if(ESP32Can.begin()){ // si le CAN est bien initaliser

gfx.println("CAN bus started!");

delay(1000);

}

else // si pas bien initialiser

{

noDataRecieved();

while(1);

}

Ce code contient les pins du série CAN, la vitesse de transmission ainsi que les queues de la réception et de la transmission du série CAN. Aussi, le programme vérifie si le CAN est actif ou non. Si le CAN est actif, un message de confirmation s’affiche sur l’écran. Sinon, une fonction affiche que le programme ne reçoit rien et il entre dans une boucle infinie.

Pour la partie « void readCAN () » :

void readCAN(){

if(ESP32Can.readFrame(rxFrame, 50)) // lecture à chaque 50ms

{

idCAN1 = rxFrame.identifier; // ID

canTest1 = rxFrame.data[0]; // centaine des données

canTest2 = rxFrame.data[1]; // unites des données

totalCAN = (canTest1\*100) + canTest2; // faire un chiffre entre 92 et 425

if(totalCAN == 92){ // si on reçoit le minimum du capteur

dataArray[1] = 0;

}

else if(totalCAN > 92 && totalCAN < 425){

dataArray[1] = map(totalCAN, 92, 425, 0, 100);

}

}

else if(totalCAN == 425){ // si on reçoit le maximum du capteur

dataArray[1] = 100;

}

}

Ce code est encore en développement, car il faudrait 10 parties au total. Soit, une partie pour chaque widget. Cependant, ce code fait fonctionner le widget « SOH » de la page HV. Elle vérifie l’identifiant envoyé et prend les deux données. La donnée rxFrame.data[0] est une valeur de 0 à 9 et rxFrame.data[1] est une valeur entre 0 et 99. La fonction « map » sert à prendre les valeurs reçues et les mettre dans les limites du widget.

# **Listes de matériel et coûts**

Figure 8: liste de matériaux

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Nom et numéro de la pièce** | **Description** | **Lien Web chez le fournisseur** | **Prix unitaire** | **Quantité** | **Transport** | **Total** | Sans écran |
| gen4-ESP32-70CT Starter Kit | L'écran de 4D System | <https://4dsystems.com.au/products/gen4-esp32-70ct/#Description> | 220,05 $ | 1 |  | 220,05 $ |  |
| ATM13-4P | Gros connecteur CAN 4Pin | <https://www.digikey.ca/en/products/detail/amphenol-sine-systems-corp/ATM13-4P-BM02/9836883> | 14,74 $ | 1 |  | 14,74 $ | 14,74 $ |
| PCB | PCB | <https://cart.jlcpcb.com/quote?orderType=1&homeUploadNum=07dbbcc42247433e96adc198a3f9c333&businessType=example&fileNameGerber_file_AffichageCAN_F.A.GUIMONT.zip> | 2,00 $ | 1 | 50 | 52,00 $ | 2,00 $ |
| TJA1051T/3,118 | transmetteur de CAN | <https://www.digikey.ca/fr/products/detail/microchip-technology/MCP2515T-I-SO/593681/> | 3,78 $ | 1 |  | 3,78 $ | 3,78 $ |
| LM2675M-5.0/NOPB | Régulateur de tension 12V à 5V-1A | <https://www.digikey.ca/en/products/detail/texas-instruments/LM2675M-5-0-NOPB/271162> | 8,75 $ | 1 |  | 8,75 $ | 8,75 $ |
| 2073-05-30-D-0030-A-4-06-4-T-ND | CABLE FFC/FPC 30 pin 0.5mm | <https://www.digikey.ca/en/products/detail/gct/05-30-D-0030-A-4-06-4-T/21266133> | 1,98 $ | 1 |  | 1,98 $ | 1,98 $ |
| boitier | boitier impirmer en 3D | <https://cad.onshape.com/documents/03c94472e16741a177019661/w/f4f8831b2ce90ca943e7f51e/e/c7fe7dde4c521339c3673f8e> | 1,00 $ | 1 |  | 1,00 $ | 1,00 $ |
| Pieces discrètes | Les pieces discrètes | Lien dans la page "Pieces discrètes" | |  |  | 10,31 $ | 10,31 $ |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  | Sous-total | | | 312,61 $ | 42,56 $ |
|  |  |  | Taxes | | | 46,89 $ | 6,38 $ |
|  |  |  | Grand Total | | | 359,50 $ | 48,94 $ |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Nom et numéro de la pièce** | **Description** | **Lien Web chez le fournisseur** | **Prix unitaire** | **Quantité** | **Total** |
| CAP ALUM 100UF 20% 50V SMD / EEE-FH1H101L | 100 µF 50 V Aluminum Electrolytic Capacitors Radial, Can - SMD 500mOhm @ 100kHz 10000 Hrs @ 105°C | <https://www.digikey.ca/en/products/detail/panasonic-electronic-components/EEE-FH1H101L/20372560?s=N4IgTCBcDaIKIIGIAkCMaAMqAyIC6AvkA> | 1,63 | 3 | 4,89 $ |
| condensateur 68uF / EEE-FN1H680XP | CAP ALUM 68UF 20% 50V SMD | <https://www.digikey.ca/en/products/detail/panasonic-electronic-components/EEE-FN1H680XP/11656970> | 1,03 | 1 | 1,03 $ |
| F32Q-1A7H1-11030 | Flex Connector, 0.50mm Pitch, Height 2.00mm, Right angle, Slider type, ZIF, Upper contact, 30 position , Without MYLAR | <https://www.digikey.ca/en/products/detail/amphenol-cs-fci/F32Q-1A7H1-11030/11564712> | 1,38 | 1 | 1,38 $ |
| condensateur 10 nF / 80-C0603C154Z3V | Chip Capacitor, 10 nF, +/- 10%, 50 V, -55 to 125 degC, 0603 (1608 Metric), RoHS, Tape and Reel | <https://www.mouser.ca/ProductDetail/KEMET/C0603C154Z3VACTU?qs=dlXU2iuYu4SgHNv%2Fd0eG3Q%3D%3D> | 0,41 | 1 | 0,41 $ |
| condensateur 100 nF / CL21B104KCFNNNE | Automotive Ceramic Capacitor, 0805, 100nF, 10%, X7R, 15%, 100V | <https://www.digikey.ca/en/products/detail/samsung-electro-mechanics/CL21B104KCFNNNE/5961324> | 0,15 | 2 | 0,30 $ |
| condensateur 100pF / C0805C101J5GAC7800 | CAP CER 100PF 50V C0G/NPO 0805 | <https://www.digikey.ca/en/products/detail/kemet/C0805C101J5GAC7800/411121> | 0,15 | 2 | 0,30 $ |
| LED verte SMD / LG R971-KN-1-0-20-R18 | LED GREEN CLEAR 0805 SMD | <https://www.digikey.ca/en/products/detail/ams-osram-usa-inc/LG-R971-KN-1-0-20-R18/1227925> | 0,42 | 1 | 0,42 $ |
| SS14FL | DIODE SCHOTTKY 40V 1A SOD123F | <https://www.digikey.ca/en/products/detail/onsemi/SS14FL/5305065?0=%2Frectifiers%2Fsingle-diodes%2Fsod-123f&s=N4IgjCBcpgbFoDGUBmBDANgZwKYBoQA3AOygBcAnAV3xAHsoBtEAFgAYWAOAZgCYQAugQAOZKCADKlAJbEA5iAC%2BBAKwB2NghDJI6bLQaRm7NtzWdBIsZEkz5SggE4wLLTr24Ch5mDCO1KppCIKLiUhSyCsogvGxqENDaqJie9EysbAAEAGqWIdYgAKrE0mQA8igAsjhoWFQUOErR-EaSEi4AYgAygopAA> | 0,38 | 1 | 0,38 $ |
| résistence 1K / RC0805FR-071KL | Chip Resistor, 1 KOhm, +/- 1%, 125 mW, -55 to 155 degC, 0805 (2012 Metric) | <https://www.digikey.ca/en/products/detail/yageo/RC0805FR-071KL/727444> | 0,15 | 1 | 0,15 $ |
| Bobine 47uH / NPI31W470MTRF | Inductor Power Unshielded Wirewound 47uH 20% 100KHz 1.8A 0.14Ohm DCR T/R | <https://www.arrow.com/en/products/npi31w470mtrf/nic-components> | 0,45 | 1 | 0,45 $ |
| résistence 270 hom SMD / RC0805JR-07270RL | RES 270 OHM 5% 1/8W 0805 | <https://www.digikey.ca/en/products/detail/yageo/RC0805JR-07270RL/728291> | 0,15 | 1 | 0,15 $ |
| résistence 120 hom SMD / RC0805JR-07120RL | RES 120 OHM 5% 1/8W 0805 | <https://www.digikey.ca/en/products/detail/yageo/RC0805JR-07120RL/728247> | 0,15 | 1 | 0,15 $ |
| résistence 10 hom SMD / RMCF0805JG10R0 | RES 10 OHM 5% 1/8W 0805 | <https://www.digikey.ca/en/products/detail/stackpole-electronics-inc/RMCF0805JG10R0/1711843> | 0,15 | 2 | 0,30 $ |
|  |  |  |  |  |  |
|  |  |  |  | Sous-total | 10,31 $ |
|  |  |  |  | Taxes | 1,55 $ |
|  |  |  |  | Grand Total | 11,86 $ |

# **Modifications et améliorations**

1. Il serait préférable de sceller le boitier pour ne pas avoir aucune entrée d’eau. Peut-être utiliser une gaine en caoutchouc qui se colle sur le devant du boitier pour qu’il y aille cet aspect de sceller.
2. Il faudrait avoir un plus petit boitier, car il est très épais à l’arrière. Peut-être, le réduire de 3 ou 4 centimètres.
3. Utiliser un autre connecteur que ATM13-4P parce qu’il est trop gros. Peut-être utiliser un connecteur fait en long donc les pins une par-dessus l’autre.

# **Annexe**

## **Annexe 1 : Schémas électriques**

A diagram of a bus and transceiver

Description automatically generated

Figure 9: Schémas électriques complets