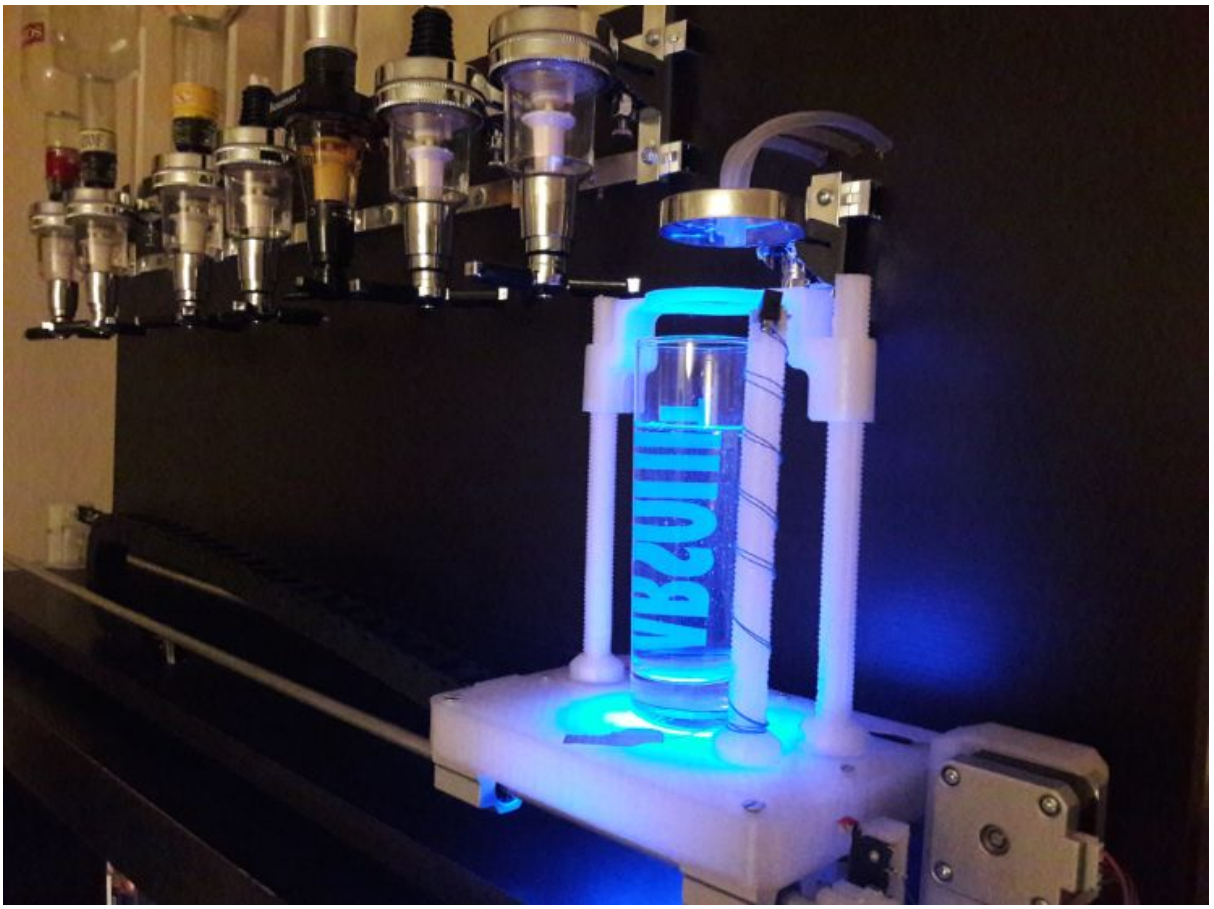
Réalisation d’une machine à cocktail connectée avec un wroom32 espressif



1) Introduction

2) Diagramme sagittal
- 2.1) Description des liaisons

3) Schéma fonctionnel
- 3.1) FS1.0 : Alimentation
- 3.2) FS1.1 : Moteurs pas à pas
- 3.3) FS1.2 : LED RGB
- 3.4) FS1.3 : uC wifi
- 3.5) FS1.4 : Capteurs
- 3.6) FS1.5 : Pompes
- 3.7) FS1.6 : Connecteur d'extension

4) Schéma structurel
- 4.1) FS1.0 : Alimentation
  - 4.1.1) Calcul d'un radiateur
- 4.2) FS1.1 : Moteurs pas à pas
- 4.3) FS1.2 : LED RGB
- 4.4) FS1.3 : uC wifi
  - 4.4.1) Définition des entrées, sorties du microcontrôleur.
- 4.5) FS1.4 : Capteurs
- 4.6) FS1.5 : Pompes
- 4.7) FS1.6 : Connecteur d'extension

5) Le PCB et liste des composants
- 5.1) PCB
- 5.2) PCB liste des composants
- 5.3) Meca

6) Le programme
- 6.1) Partition de la flash
- 6.2) L'architecture
  - 6.2.1) Le SDK
  - 6.2.2) Les pilotes
  - 6.2.3) La HAL
  - 6.2.4) l'application
- 6.3) Les Fonctionnalités du programme
  - 6.3.1) Mode wi-fi connecté
  - 6.3.2) Mode smartconfig
  - 6.3.3) Mise à jour (OTA)
  - 6.3.4) Bouton poussoir
  - 6.3.5) Système de fichiers
  - 6.3.6) Détection de fin de course

6.3.7) Page web pour la cmd des cocktails
6.3.8) Commande moteurs
6.3.9) Commande moteurs Pas à Pas
6.3.10) LED RGB
6.3.11) cJson
6.13.12) Add-on
6.4) Le JSON
6.4.1) Le JSON de l'emplacement des bouteilles
6.4.2) Le JSON de la liste est ingrédient des cocktails.
6.5) La page web
6.5.1) Le CSS
6.5.1) Le HTML
7) Mécanique
7.1) Le bois
7.2) Le plateau
8) Mise en route et fonctionnement
8.1) Configurer le programme
8.2) Compiler et programmer le code
8.3) Premier démarrage
8.4) Vidéo
8.5) Hygiène de la machine à cocktail
9) L'avenir du projet
9.1) Modifier
9.2) Ajouter

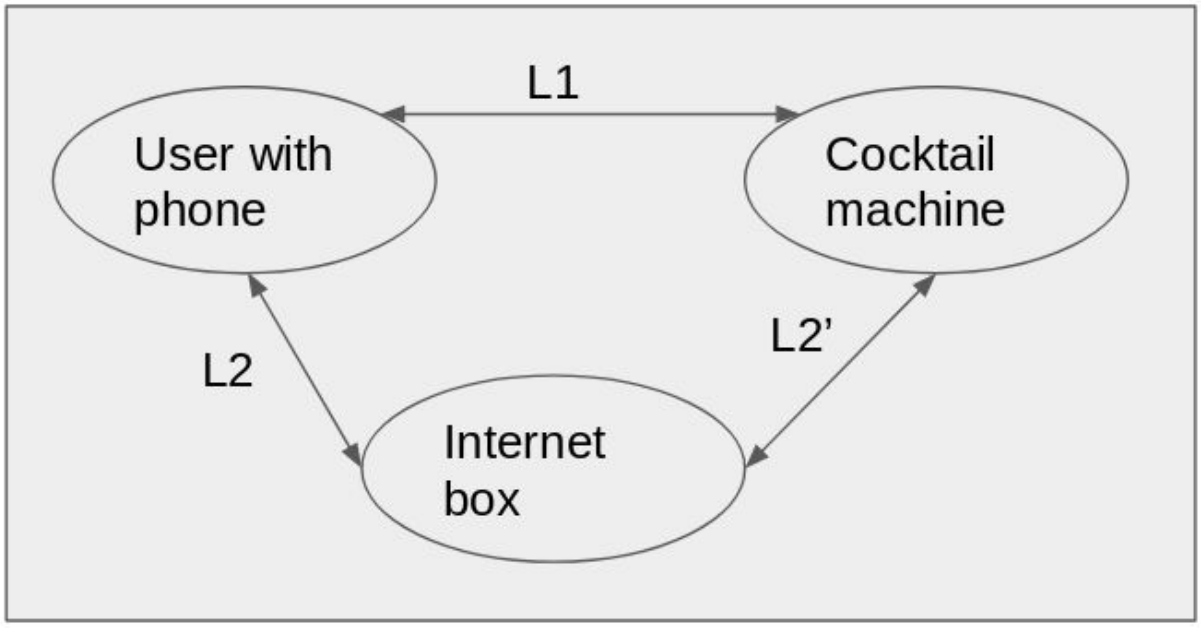
Révision	Date	Auteur	Modifications
R01	17/01/2018	Q.THEROND	Création

1) Introduction

A défaut de réaliser une fonctionnalitée intéressante pour le SDK espressif comme le wi-fi mesh je vais vous présenter un projet fun. La réalisation d'une machine à cocktail connectée. Le but est de pouvoir commander un cocktail (ou un biberon en pleine nuit) via un téléphone ou PC. La machine à cocktail expose une page web accessible via une IP fixe sur le réseau internet domestique. (On pourra imaginer une évolution dans un second temps, NFC ou autre par exemple)

Le coeur du système utilise un module Wi-fi/Bleutooth WROOM32. Ce module présente de nombreux avantages comme un prix attractif (pour un module wi-fi), une belle capacité mémoire, un SDK simple et complet.
Les inconvénients du module résident en la gestion de la protection de la flash qui demande 30 secondes (au premier démarrage) pour chiffrer les 4Mb. Il manque également d'une réelle accélération hardware pour le handshake ssl (entre 1 et 5 secondes pour du RSA256 en 2048).

2) Diagramme sagittal



2.1) Description des liaisons

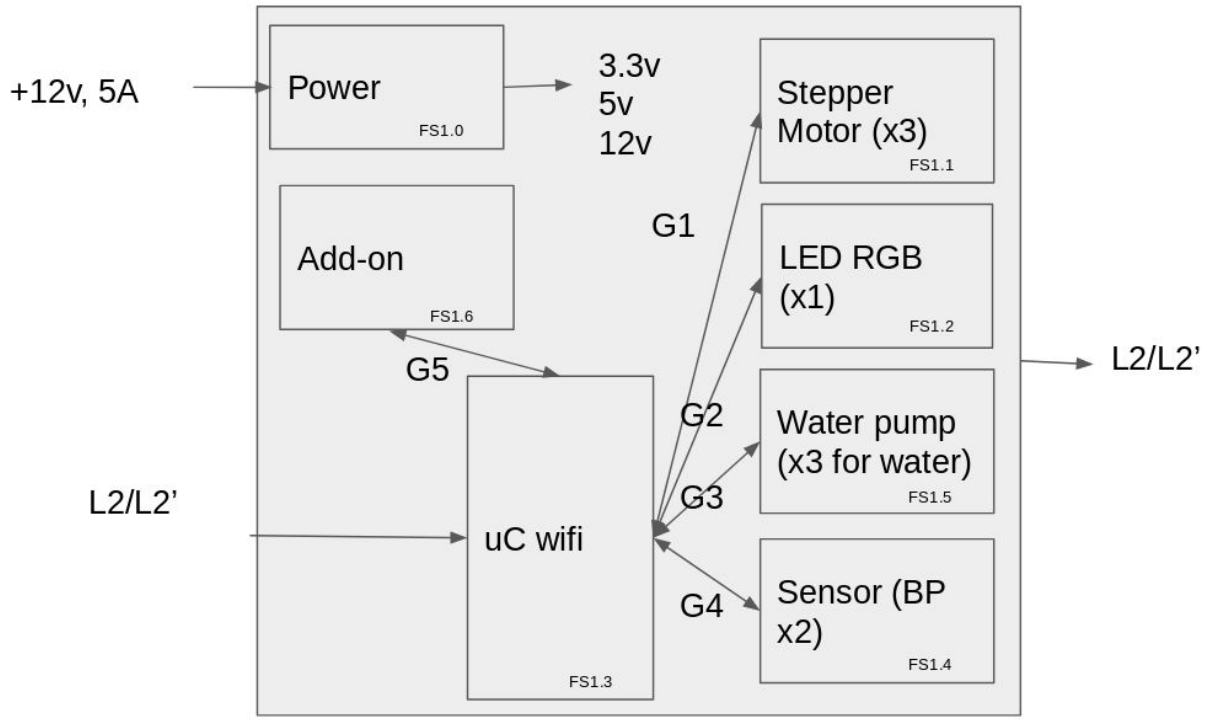
L2/L2' :

- Permet à l'utilisateur de sélectionner le cocktail via une page web.
- D'envoyer son SSID et mot de passe avec son smartphone pour connecter le système à internet.

L1 :

- Permet à l'utilisateur de poser et récupérer son verre
- Information lumineuse sur l'état du système

3) Schéma fonctionnel



3.1) FS1.0 : Alimentation

La fonction alimentation permet de créer, grâce à une tension d'entrée de 12V 5A, du +5V, +3,3V ,0V.

Entrée
12V, 5ADC.

Sorties
+5V, +3,3V ,0V

3.2) FS1.1 : Moteurs pas à pas

La fonction moteur permet de piloter le plateau avec le verre et de servir le cocktail via les doseurs.

Entrée
G1 : Signaux numériques (dir et clk) pour la commande des moteurs pas à pas.

Sortie

Mouvement physique des moteurs

3.3) FS1.2 : LED RGB

La LED RGB permet d'informer l'utilisateur de l'état du système.

Entrée
G2 : Signaux numériques pour commander la LED RGB.

Sortie
L2 : Signaux lumineux pour informer l'utilisateur

3.4) FS1.3 : uC wifi

Assure grâce à un traitement programmé (logiciel) l'acquisition, le traitement et la restitution des informations. Le mode smartconfig du wroom32 permet de récupérer le SSID et mot de passe de la box internet. Le wifi permet de commander le système via un smartphone ou un ordinateur grâce à une page web sur une adresse IP fixe. Il communique avec les moteurs, la LED RGB. les capteurs, et propose des entrées/sorties pour ajouter des fonctionnalitées supplémentaires.

Entrées
L2/L2' : Connection wifi en wpa2, wep, ... Mode smartconfig,
G4 : Signaux numériques pour la détection de fin de course

Sorties
L2/L2' : Page web http sur IP fix
G1 : Signaux numériques pour la commande des moteurs pas à pas
G2 : Signaux numériques pour la commande de la LED RBG
G3 : Signaux numériques pour la commande des pompes

3.5) FS1.4 : Capteurs

Permet la détection des signaux de fin de course et de supprimer le SSID et mot de passe du wi-fi.

Entrée
L1 : Grandeur physique

Sortie
G4 : Signaux numériques on/off

3.6) FS1.5 : Pompes

La fonction "pompes" permet de piloter des moteurs à courant continu.

Entrée
G3 : Signaux numériques pour la commande des pompes

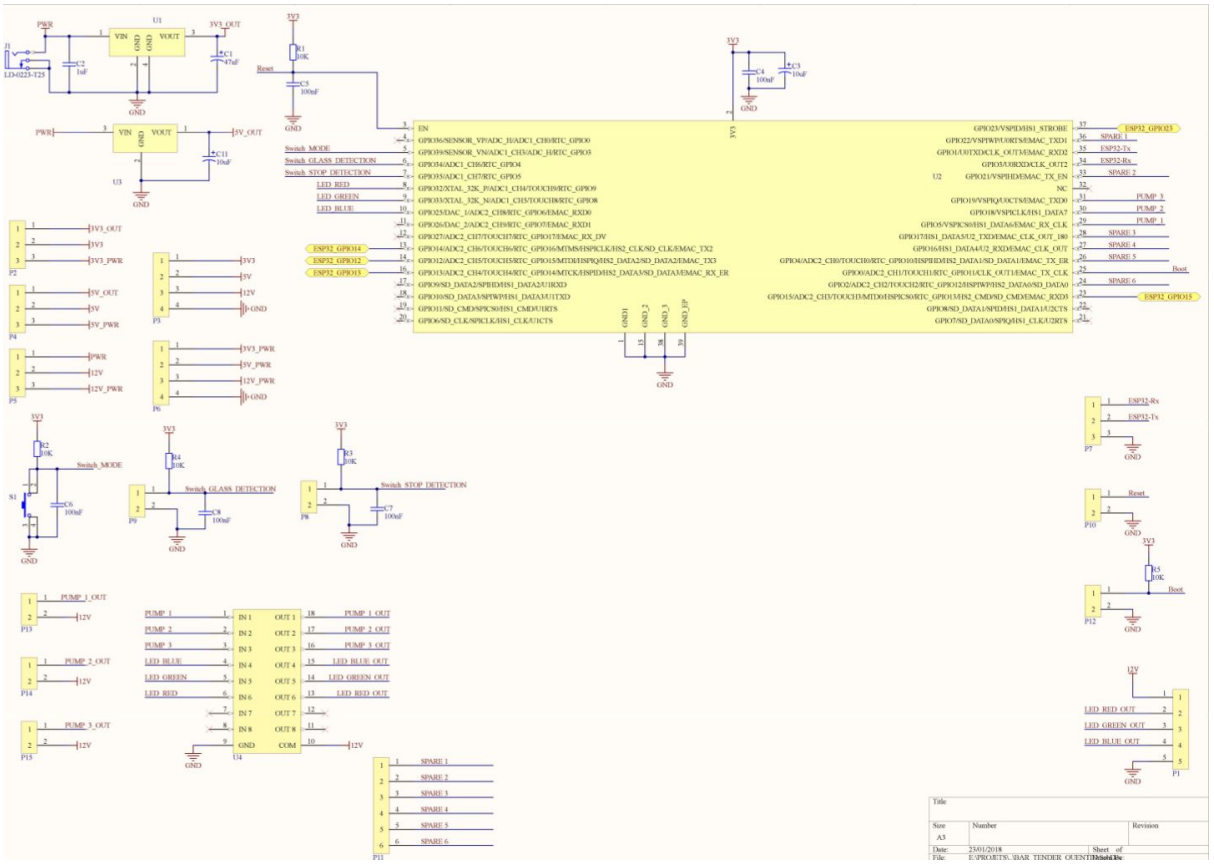
Sortie
Mouvement physique.

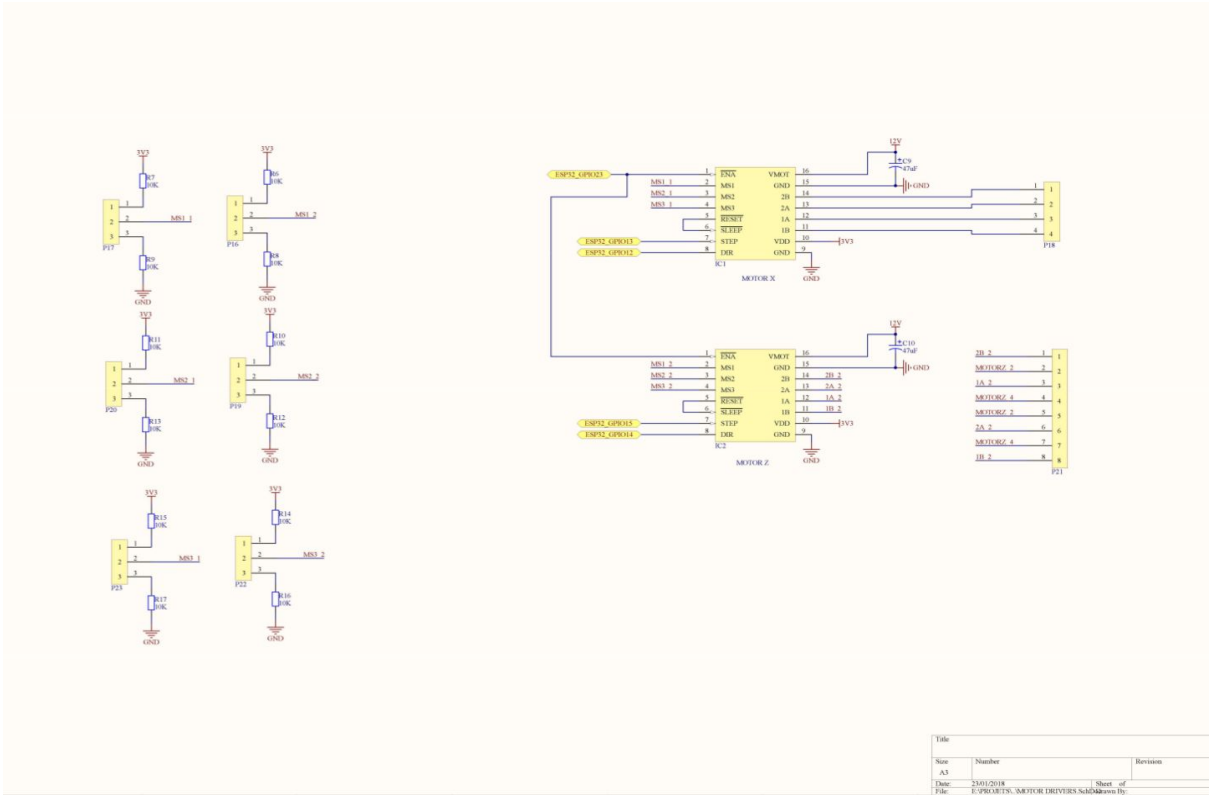
3.7) FS1.6 : Connecteur d'extension

Permet d’ajouter des fonctionnalitées à la carte électronique.
Entrée
G5 : Signaux numériques

Sortie
G5 : Signaux numériques

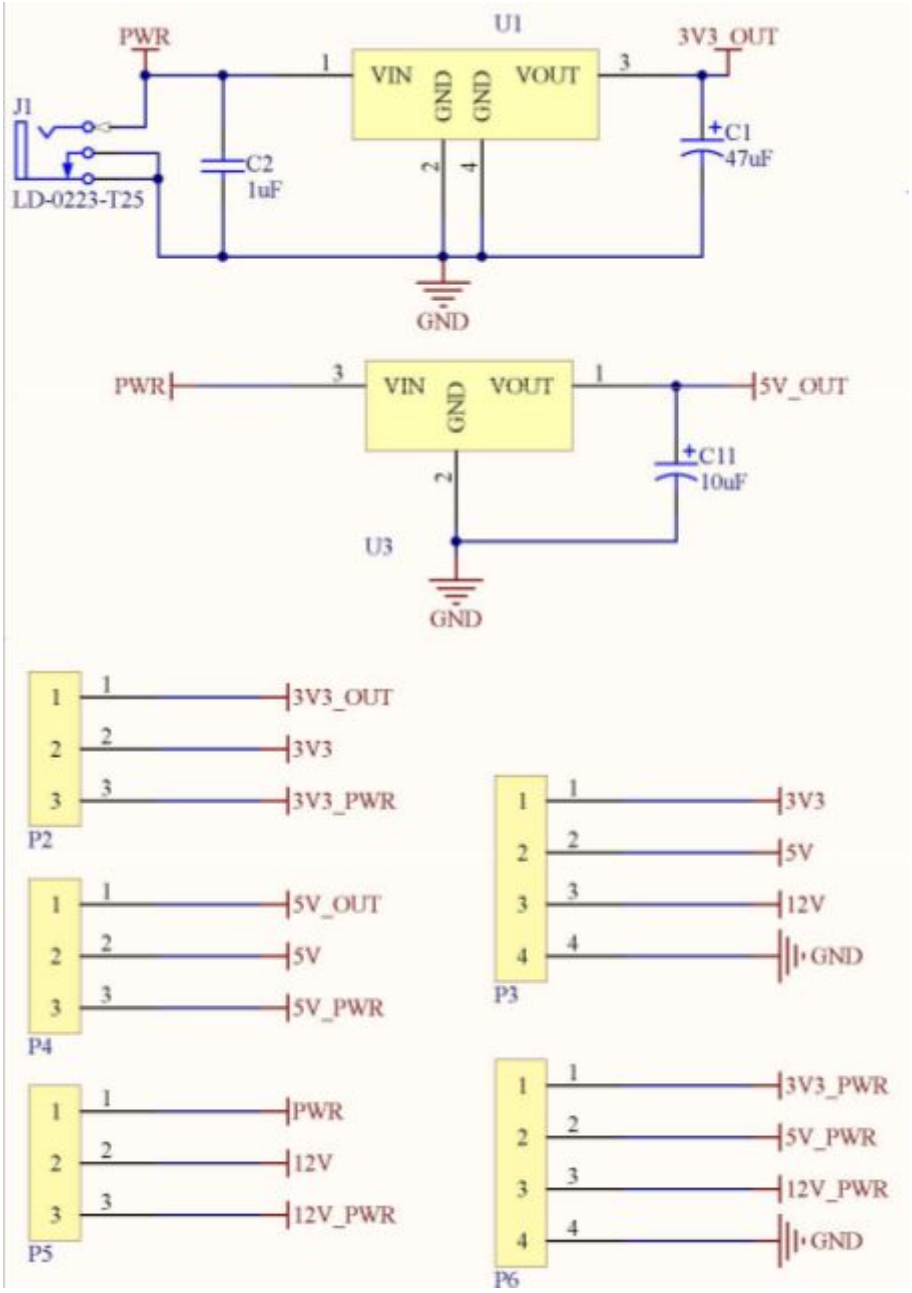
4) Schéma structurel





4.1) FS1.0 : Alimentation

La fonction alimentation permet de créer, grâce à une tension d'entrée de 12V 5A, du +5V et du +3,3V.



J1: Entrée 12v

C1, C11 : Condensateurs chimiques polarisés. Ils réalisent le filtrage, et permettent un découplage en cas de microcoupure d'alimentation

C2 : Condensateur plastique, sert d'anti-parasite pour supprimer les hautes

fréquences (il est conseillé dans la documentation technique).

U3: régulateur 5V, il permet de réguler la tension d'entrée 12V DC en une tension de 5V DC. D'après la documentation constructeur du 78XX il faut une tension d'entrée minimum de la tension de sortie plus Vdrop (2V), soit Ve = Vs + Vdrop = 5 + 2 = 7V minimum, pour un bon fonctionnement.

U1: LDO 3.3V, il permet de réguler la tension d'entrée 12V DC en une tension de 3.3V DC.

P2, P4, P5: Permet de choisir la source d'alimentation (interne ou externe) en fonction du besoin en courant. Il s'agit physiquement, de barrettes de connexion sur lesquelles on vient connecter un "jumper" (ou cavalier) afin de sélectionner la tension souhaitée.

P3: Connecteur pour tester les tensions de sortie.

P6: Permet de brancher une alimentation externe pour ne pas utiliser les régulateurs (si besoin de plus de courant). Attention P2, P4, P5 doivent être relié correctement.

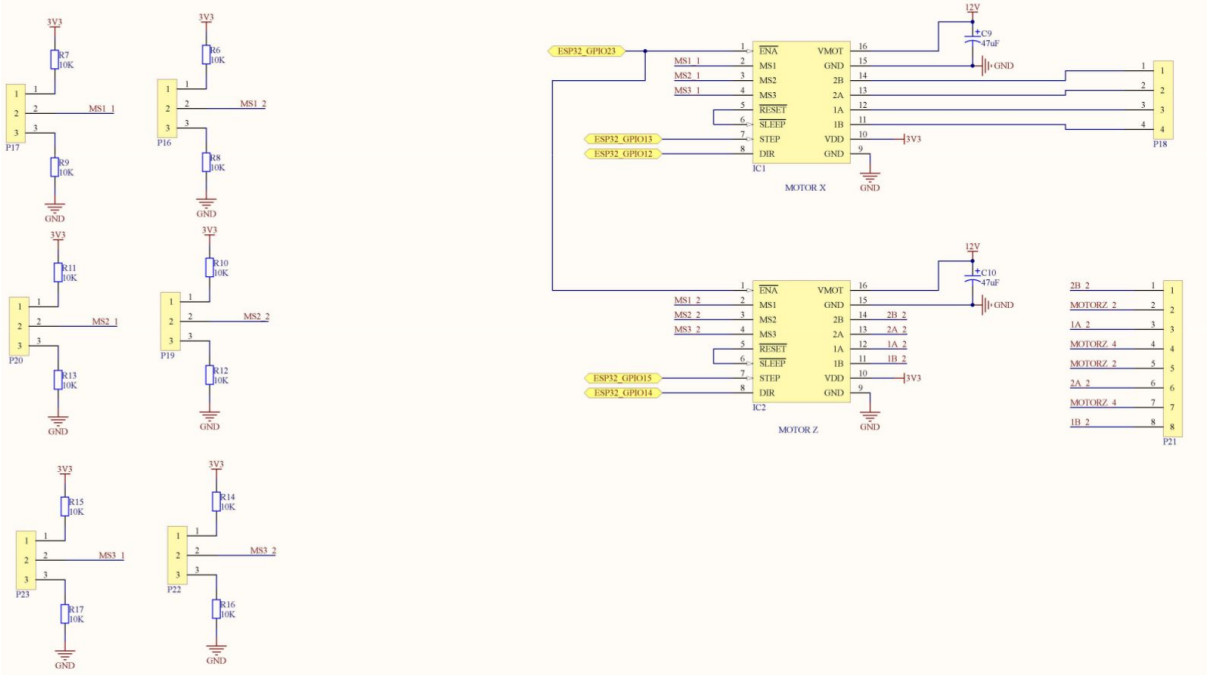
4.1.1) Calcul d'un radiateur

hypothèse: esp32=60mA, autre 50mA max
Isystéme = 60+50 = 110mA
Pmax = (Tj – Ta)/RTHja = (125-25)/65 = 1,5W
Putil(3.3V) = Isystéme*(Ve-Vs) = 110*10^-3*(12-3.3) = 0,957W
Putil(5V) = le 5 n'est pas utilisé sur la carte
Putil<Pmax.

L'utilité d'un radiateur pour les régulateurs n'est pas indispensable à température ambiante.

4.2) FS1.1 : Moteurs pas à pas

La fonction moteur permet de piloter le plateau avec le verre et de servir le cocktail via les doseurs.



IC1, IC2: sont des modules à base de A4988 pour la commande des moteurs pas à pas alimentés en 3.3v. les entrées MSI_x permettent de configurer la résolution des pas. l'entrée STEP est une horloge qui déclenche un pas par période d'horloge. l'entrée DIR permet de choisir le sense de rotation du moteur.

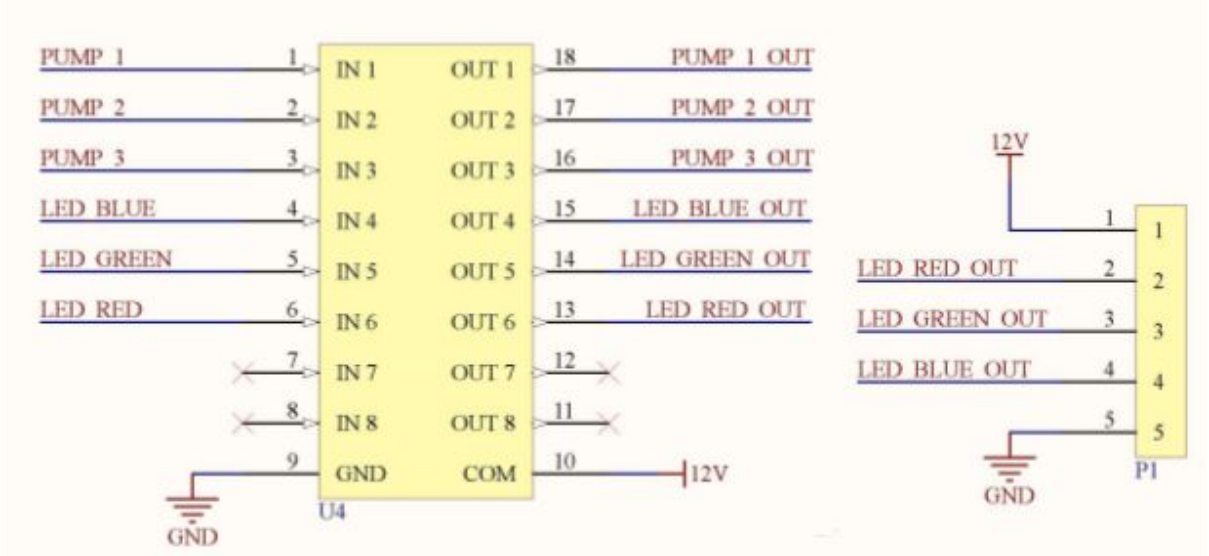
P16, P17, P19, P20, P22, P32 : permettent de sélectionner l'une des cinq résolutions d'état selon le tableau de vérité ci-dessus.

MS1	MS2	MS3	Microstep Resolution
Low	Low	Low	Full step
High	Low	Low	Half step
Low	High	Low	Quarter step
High	High	Low	Eighth step
High	High	High	Sixteenth step

C9, C10 : Condensateurs de découplage chimiques polarisés.
P18 : Permet de connecter le moteur du plateau avec le verre (axe X).
P21 : Permet de connecter les moteurs pour servir le cocktail via les doseurs (axe Y).

4.3) FS1.2 : LED RGB

La LED RGB permet d’informer l’utilisateur de l'état du système.

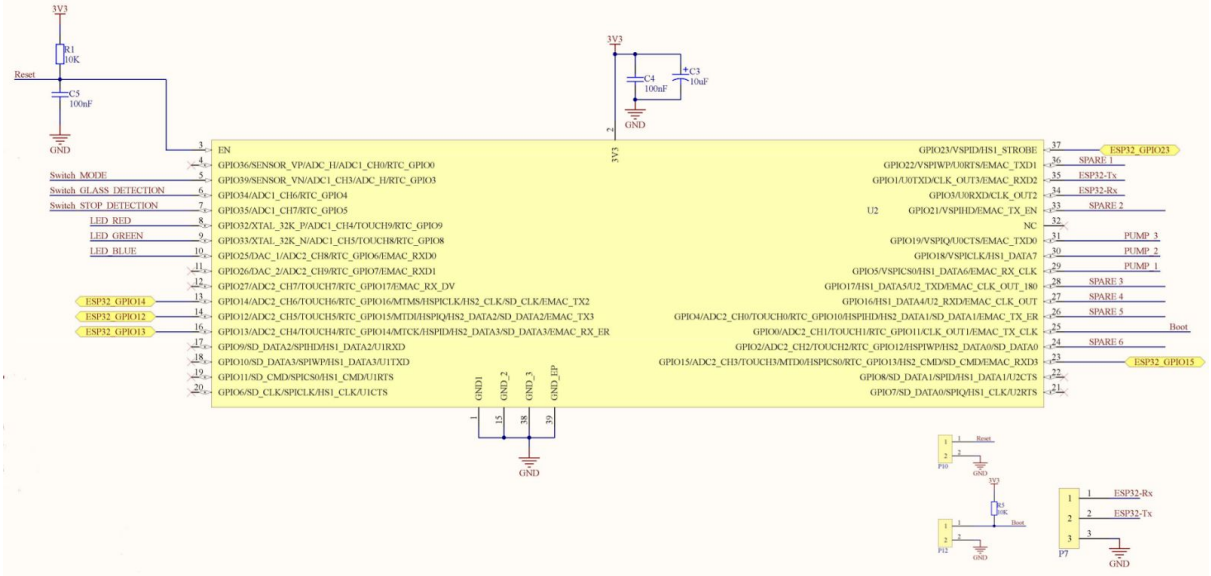


U4 : Interrupteurs commandés (réseau de transistors - ULN2803). Chacun de ces 8 canaux peut être ouvert ou fermé. Il est capable de piloter une charge jusqu'à 500mA.

P1 : Permet de connecter la LED RGB. La PIN 5 n’est pas utilisé.

4.4) FS1.3 : uC wifi

Assure grâce à un traitement programmé (logiciel) l'acquisition le traitement et la restitution des informations. Le mode smartconfig du WROOM32 permet de récupérer le SSID et mot de passe de la box internet. Le wifi permet de commander le système via un smartphone ou un ordinateur grâce à une page web sur une adresse IP fixe. Il communique avec les moteurs, la LED RGB. les capteurs, et propose des entrées/sorties pour ajouter des fonctionnalités supplémentaires.



R1, C5, P10 : Permet de faire un reset sur le wroom32.

C4, C3 : servent à filtrer les hautes et basses fréquences.

P7 : Permet de programmer le wroom32 via une UART.

P12 : Permet de mettre le wroom32 en mode de programmation.

4.4.1) Définition des entrées, sorties du microcontrôleur.

Entrées:
Switch_MODE : Permet de supprimer la configuration du wi-fi (SSID et mot de passe)
Switch_GLASS_DETECTION : Permet de détecter la fin de course du plateau.
Switch_STOP_DETECTION : Permet de détecter la fin de course du système de gestion des doseurs.
Boot : Permet de mettre le WROOM32 en mode de programmation.
Reset : Permet de faire un reset sur le WROOM32.
ESP32-Rx : Permet d'envoyer les binaires à sauvegarder en mémoire du wroom32

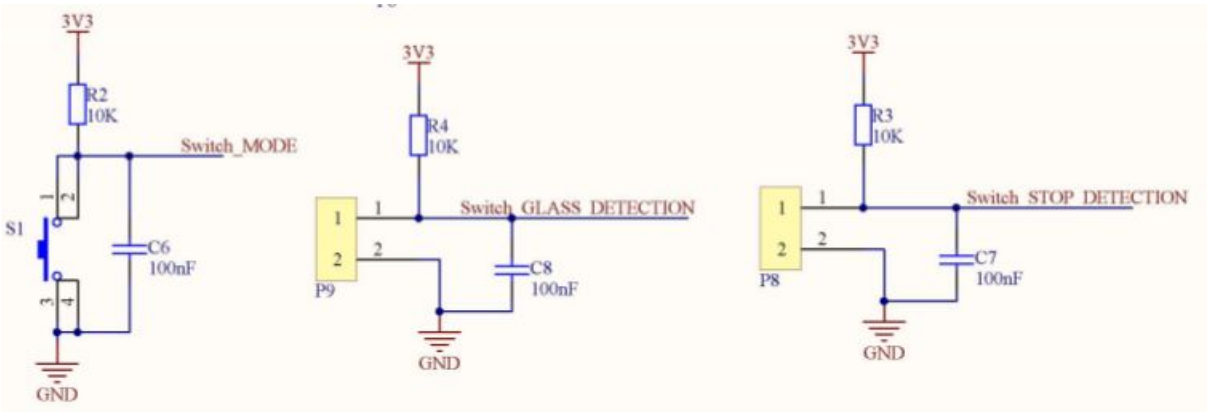
Sorties:

ESP32_GPIO14 , 12, 13, 15 : Permet de commander le sense de rotation des moteurs et la vitesse des pas.
ESP32_GPIO23 : Permet d'activer et désactiver les modules à base de A4988
LED_RED : Permet de commander la LED rouge. Au démarrage du système la LED est rouge tant que le système n'a pas atteint les fins de course
LED_GREEN : Permet de commander la LED verte. Non utilisé pour le moment.
LED_BLEU : Permet de commander la LED BLEU. Le bleu fix indique que le système est prêt à être utilisé. Le bleu clignotant indique que le système est en préparation d'un cocktail.
ESP32-Tx : Permet d'envoyer des informations à l'outil de programmation.
PUMP_1 à PUMP_3 : Permet de commander des pompes.

Autres:
SPARE1 à SPARE6 : Les entrées/sorties SPARE permettent d'ajouter des nouvelles fonctionnalitées à la machine à cocktail comme le dosage de lait en poudre (bien entendu, ceci implique tout un système de conservation et d'hygiène du lait. Ceci n'est donné qu'à titre d'exemple.)

4.5) FS1.4 : Capteurs

Permet la détection des signaux de fin de course et de supprimer le SSID et mot de passe du wi-fi. Le bouton poussoir (switch_MODE) permet de demander la suppression du SSID et mot de passe au module wroom32.



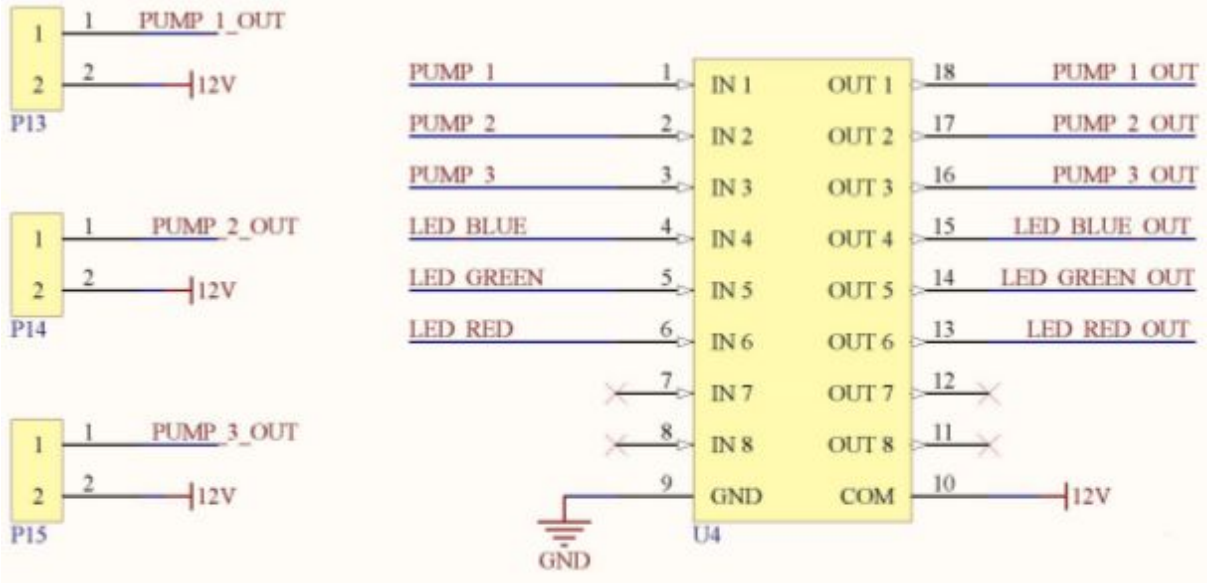
R2, R3, R4 : Résistance de pull-up.

C6, C7, C8 : Condensateur anti-rebond.

P9, P8 : Permet de connecter les capteurs fin de course.

4.6) FS1.5 : Pompes

La fonction pompe permet de piloter des moteurs à courant continu.

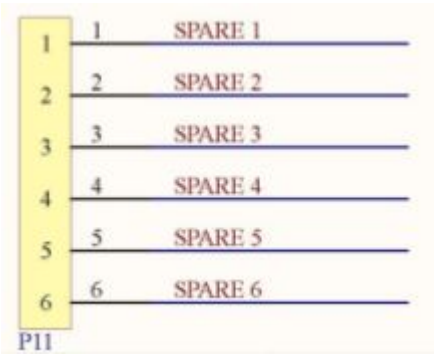


P13, P14, P15 : Permet de connecter les pompes.

U4 : Interrupteurs commandés. Chacun de ces 8 canaux peut être ouvert ou fermé. Il est capable de piloter une charge jusqu'à 500mA. après les tests la pompe choisi consomme plus de 500mA, je vais donc ajouter une carte d'extension avec un relais.

4.7) FS1.6 : Connecteur d'extension

Permet d’ajouter des fonctionnalitées à la carte électronique.

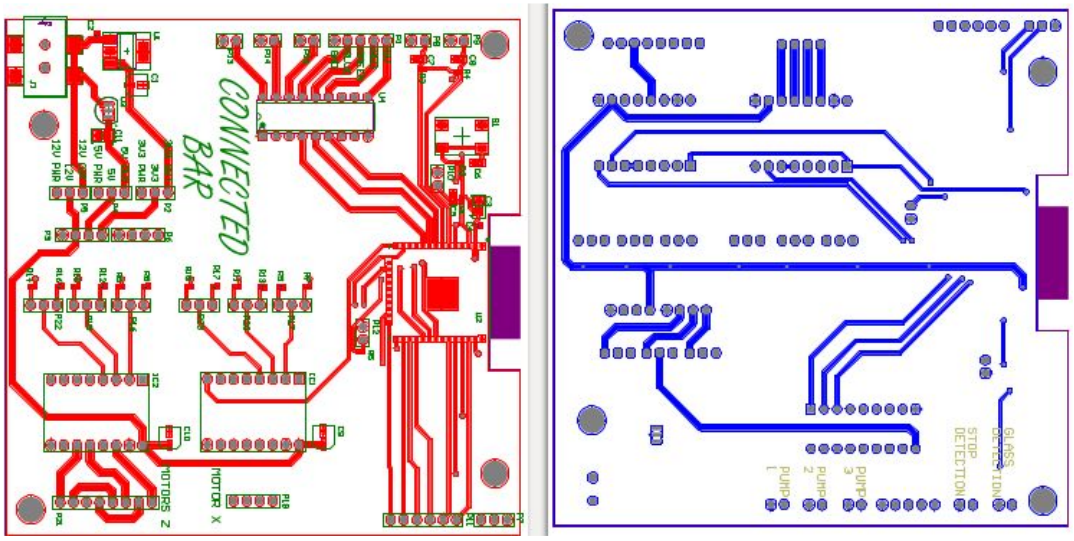


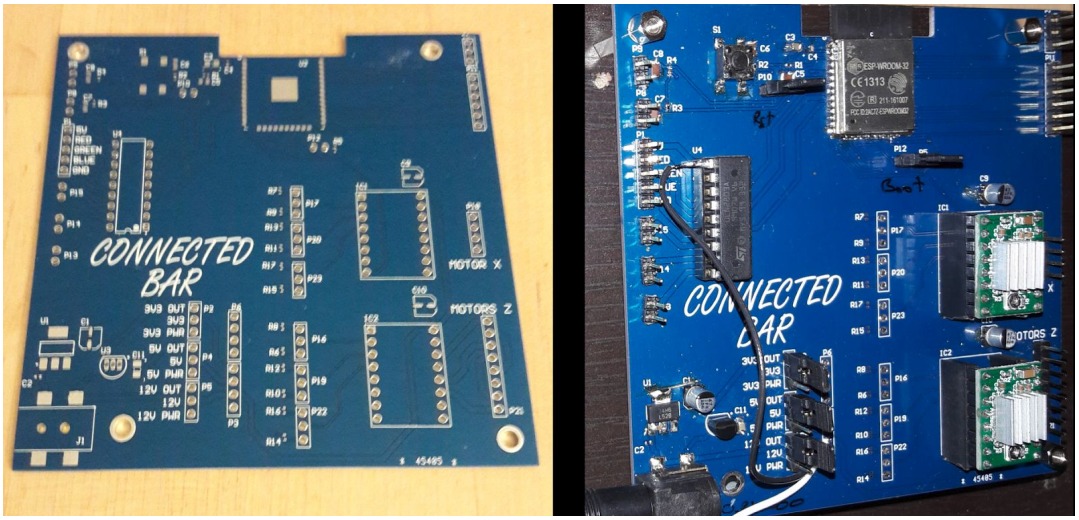
P11 : Permet de relier une autre carte électronique à l’unité centrale.

5) Le PCB et liste des composants

5.1) PCB

Le PCB n’a pas de contrainte particulière si ce n'est de respecter le “reference design” proposé par expressif pour avoir les meilleurs performances radio (les plans des masse ont été mis en transparent ici pour une meilleure représentation).





5.2) PCB liste des composants

Comment	Description	Designator	Footprint	LibRef	Quantity
EEEFK0J470UR	CAP 47uF 6.3V ALU CMS	C1, C9, C10	CAPA CHIMIQUE CMS	EEEFK0J470UR	3
GRM155R60J105KE19D	CAP 1UF 6V3 X5R CER 0402 CMS	C2	0402	GRM155R60J105KE19D	1
0805YD106KAT2A	CAP 10uF 16V X5R CER 0805 CMS	C3, C11	0805 POLARISE	0805YD106KAT2A	2
CC0402KRX7R6BB104	CAP 100NF K10V X7R CER 0402 CMS	C4	0402	CC0402KRX7R6BB104	1
MC0603B104K250CT	CAP 100NF 25V X7R CER 0603 CMS	C5, C6, C7, C8	0603	MC0603B104K250CT	4
A4988	MOTOR DRIVER	IC1, IC2	DRIVER MOTEUR	A4988	2
LD-0223-T25	CMS JACK SOCKET PLUG	J1	PJACK-LIHSHENG-LD-0223-25	LD-0223-T25	1
HEADER 1x5	EMBASE MALE 1 RANGEE VERT 5 VOIES	P1	EMBASE 5 VOIES	HEADER 1x5	1
HEADER 1x3	EMBASE MALE 1 RANGEE VERT 3VOIES	P2, P4, P5, P7, P16, P17, P19, P20, P22, P23	EMBASE 3 VOIES	HEADER 1x3	10
HEADER 1x4	EMBASE MALE 1 RANGEE VERT 4 VOIES	P3, P6, P18	EMBASE 4 VOIES	HEADER 1x4	3
HEADER 1x2	EMBASE MALE 1 RANGEE VERT 2 VOIES 2.54mm	P8, P9, P10, P12, P13, P14, P15	EMBASE 2 VOIES - 2.54mm	HEADER 1x2 - 2.54 mm	7
HEADER 1x6	EMBASE MALE 1 RANGEE VERT 6 VOIES	P11	EMBASE 6 VOIES - VERTICAL	HEADER 1x6	1
HEADER 1x8	EMBASE MALE 1 RANGEE VERT 8 VOIES	P21	EMBASE 8 VOIES - VERTICAL	HEADER 1x8	1

MC 0.0625W 0402 1% 10K	RESIST.CMS 0402 1/16W 5% 10K	R1, R2, R3, R4, R5, R6, R7, R8, R9, R10, R11, R12, R13, R14, R15, R16, R17	0402	MC 0.0625W 0402 1% 10K	17
Switch	SWITCH TACTILE	S1	TACT SWITCH 6X6	DTSM-61R-V-*	1
LM3940IMP-3.3/NOPB	REGULATEUR LINEAIRE 3V3	U1	SOT-223	LM3940IMP-3.3/NOPB	1
ESP-WROOM-32	MODULE WIFI ESPRESSIF WROOM 32	U2	MODULE_WIFI ESPRESSIF WROOM 32	ESP WROOM 32	1
LM78L05ACZ/NOPB	REGULATEUR LINEAIRE 5V	U3	TO-92	LM78L05ACZ/NOPB	1
ULN2803A	Transistor en réseau bipolaire	U4	DIP 18	ULN2803A	1

5.3) Meca

BQLZR 500mm Longueur 8mm Diam¨¨tre ext¨¦rieur Axe linéaire horizontal Axe optique Ensemble de palier ¨¤ glissi¨¨re et axe linéaire Ensemble de soutien	https://www.amazon.fr/gp/product/B01K4MX1WW/ref=oh_aui_detailpage_o03_s00?ie=UTF8&psc=1
Myarmor 2 pcs 2 GT 20 dents Timing Poulie de roue + 5 m 2 Gt-6 mm Ouverture en caoutchouc Ceinture	https://www.amazon.fr/gp/product/B01K4MX1WW/ref=oh_aui_detailpage_o03_s00?ie=UTF8&psc=1
XCSOURCE NEMA 17 2 Phase Moteur 4-Wire 1.8A Stepper 42 * 42 * 34mm Pour Imprimante 3D TE225	https://www.amazon.fr/gp/product/B011NRMXYO/ref=oh_aui_detailpage_o02_s00?ie=UTF8&psc=1
Chenxi Shop 15 x 15 mm L1000 mm Plastique câble Drag Fil de chaîne de transport pour CNC Router machine	https://www.amazon.fr/gp/product/B073XFG975/ref=oh_aui_detailpage_o03_s00?ie=UTF8&psc=1
DC 3-12V Mini Pompe à Eau Moteur à Engrenages RS-360SH pour Aquarium, Expérimentation	https://www.amazon.fr/gp/product/B074V2CWFC/ref=oh_aui_detailpage_o04_s00?ie=UTF8&psc=1

SODIAL(R) 10 Pcs Mini Micro Fin de course Levier a galet bras SPDT Declic LOT	https://www.amazon.fr/gp/product/B00U8MPR8U/ref=oh_aui_detailpage_o07_s00?ie=UTF8&psc=1
Transformateur 220Vac vers 12Vdc Chargeur Alimentation à découpage pour guirlande Ruban Led Bande 5m 60W	https://www.amazon.fr/gp/product/B06WGRXCPK/ref=oh_aui_detailpage_o05_s00?ie=UTF8&psc=1
TecTake Boissons cuillère pour 6 bouteilles pour le mur bouteille support bar butler	https://www.amazon.fr/gp/product/B0196P9XR0/ref=oh_aui_detailpage_o07_s00?ie=UTF8&psc=1
Anycubic 20 Dents Alumium Timing Poulie 5mm Alésage Ceinture Roue D'entraînement pour 10mm Largeur GT2 Ceinture 5PCS	https://www.amazon.fr/gp/product/B06XT274LH/ref=oh_aui_detailpage_o08_s00?ie=UTF8&psc=1
Popprint 5 pcs Ramps1.4 A4988 pilote de moteur pas à pas avec dissipateur thermique pour imprimante 3d, Green, 5	https://www.amazon.fr/gp/product/B06Y28H956/ref=oh_aui_detailpage_o00_s00?ie=UTF8&psc=1

6) Le programme

Il est disponible avec le document en zip et bientôt sur github :-).

6.1) Partition de la flash

Pour permettre la mise à jour du système via le wifi nous avons besoin de plusieur partition. Espressif propose pour cela la possibilité de créer un fichier cvs pour définir les adresses. Voici celui de la machine à cocktail :

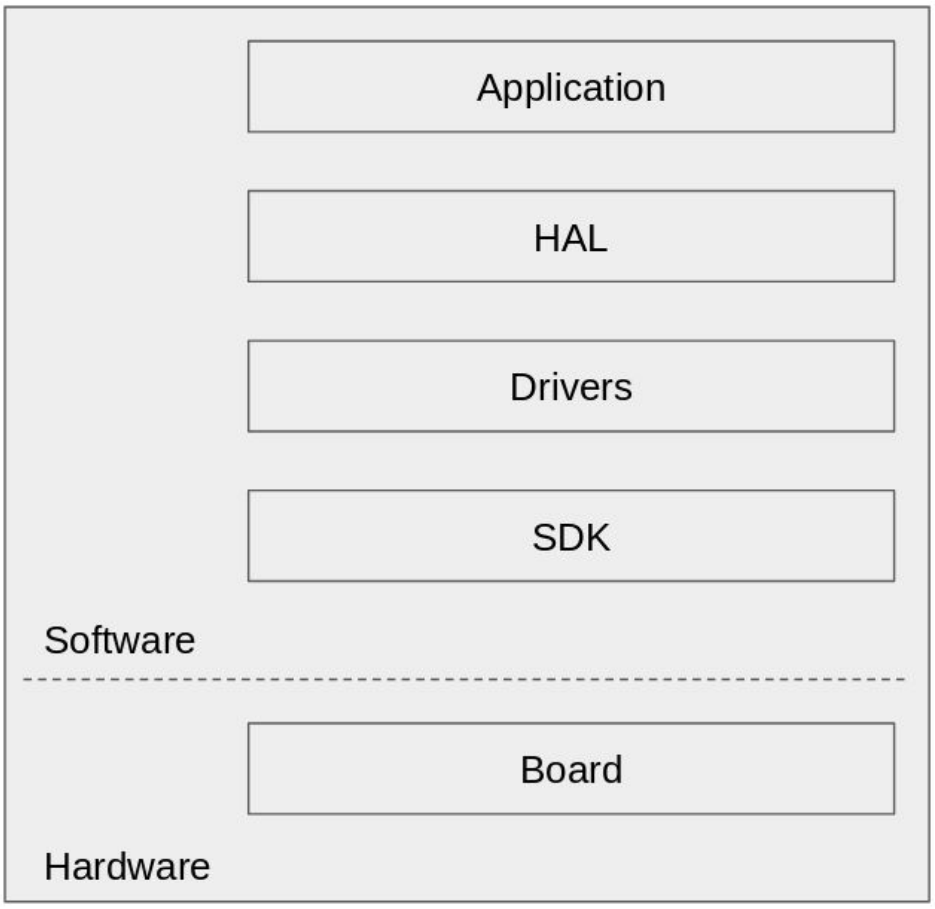
```
# Name, Type, SubType, Offset, Size, Flags
otadata,data,ota,0xd000,8K,
phy_init,data,phy,0xf000,4K,
factory,app,factory,0x10000,1M,
ota_0,app,ota_0,0x110000,1M,
ota_1,app,ota_1,0x210000,1M,
nvs,data,nvs,0x315000,500K,
```

Nous avons donc une mémoire partitionné comme suite:

```
|   Addr      |   Binaries
 ---------------------------------------
| 0x001000 |  Bootloader.bin
| 0x008000 |  Partitions.bin
| 0x010000 |  Factory.bin
| 0x110000 |  OTA_0.bin
| 0x210000 |  OTA_1.bin
| 0x315000 |  NVS (500Ko)
| 0x3F0000 |  Free
```

6.2) L’architecture

Le programme est architecturé comme suite :



6.2.1) Le SDK

Le SDK utilisé est le SDK-IDF sur la branche v2.1 disponible sur github à l’adresse (https://github.com/espressif/esp-idf). C’est le sytème de développement officiel de la puce ESP32.

6.2.2) Les pilotes

Ils permettent l’utilisation des différentes interactions matériels comme les moteurs, le LED RGB, ou les fonctionnalitées du module WROOM32 (Gpio, wifi, smartconfig, OTA, ...).

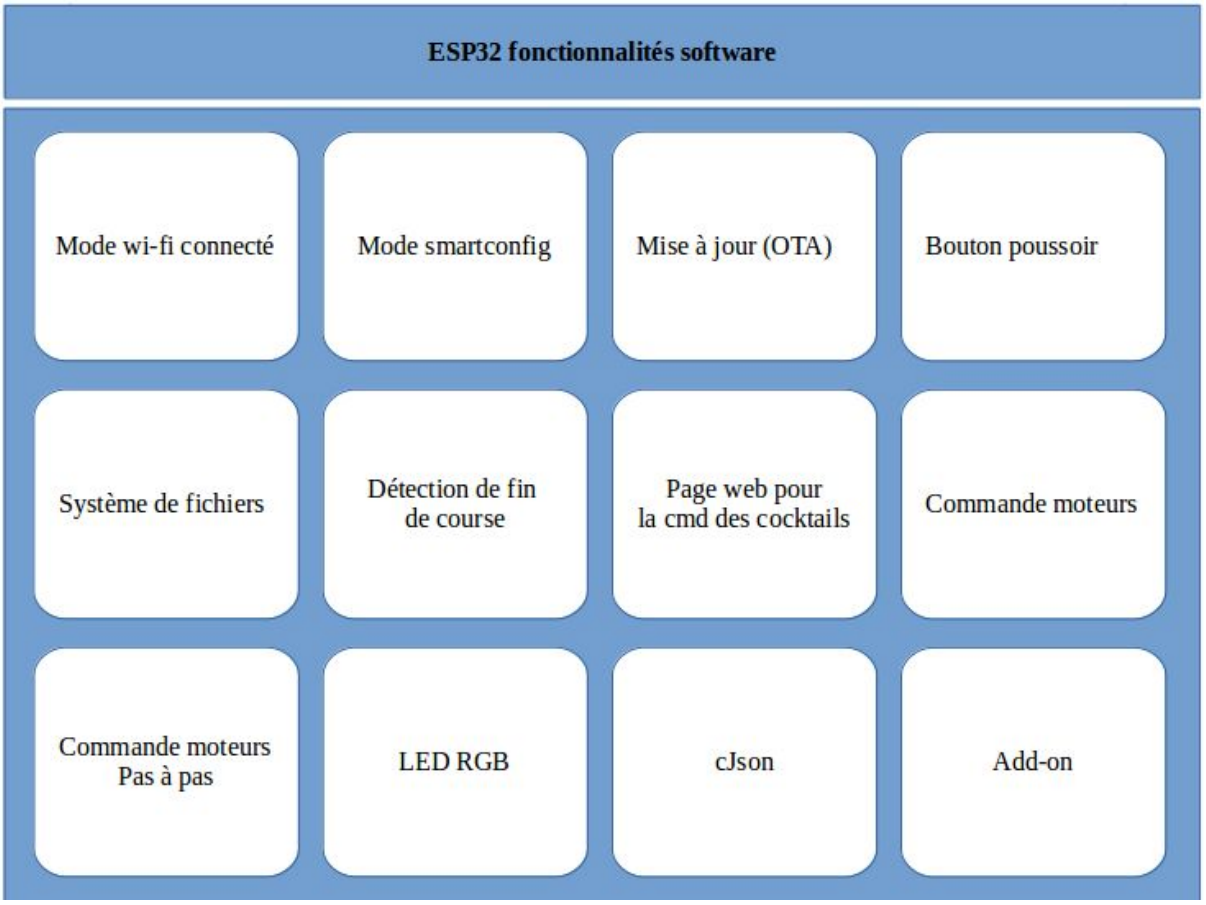
6.2.3) La HAL

La HAL permet un portage simple de machine à cocktail sur un autre SDK, OS, ou une autre plateforme matériel.

6.2.4) l’application

C’est elle qui propose la page web et la gestion des cocktails

6.3) Les Fonctionnalités du programme



6.3.1) Mode wi-fi connecté

Permet de se connecter à une box internet avec une sécuritée WEP, WPA-PSK [TKIP], WPA2-PSK [AES] ou WPA-PSK [TKIP] + WPA2-PSK [AES].
La fonction Esp32wifi_init permet de paramétrer le wi-fi pour votre pays et de remplir la structure “wifi_config_t” avec votre SSID et mot de passe.

```
wifi_init_config_t cfg = WIFI_INIT_CONFIG_DEFAULT();
ESP_ERROR_CHECK(esp_wifi_init(&cfg));
ESP_ERROR_CHECK(esp_wifi_set_storage(WIFI_STORAGE_RAM));
ESP_ERROR_CHECK(esp_wifi_set_mode(WIFI_MODE_STA));
wifi_config_t sta_config = Esp32Wifi_getSSIDAndPass();
ESP_ERROR_CHECK(esp_wifi_set_config(WIFI_IF_STA, &sta_config));
ESP_ERROR_CHECK(esp_wifi_set_country(WIFI_COUNTRY_EU));
ESP_ERROR_CHECK(esp_wifi_start());
```

La fonction Esp32_init permet de configurer une ip static sur le wroom32. Si votre réseau n’est pas 192.168.1.x avec un mask en 255.255.255.0 vous devez modifier le code ci dessous.

```
tcpip_adapter_init();
```

```
ESP_ERROR_CHECK(tcpip_adapter_dhcpc_stop(TCPIP_ADAPTER_IF_STA));
tcpip_adapter_ip_info_t info = { 0,};
IP4_ADDR(&info.ip, 192, 168, 1, 51);
IP4_ADDR(&info.gw, 192, 168, 1, 1);
IP4_ADDR(&info.netmask, 255, 255, 255, 0);
ESP_ERROR_CHECK(tcpip_adapter_set_ip_info(TCPIP_ADAPTER_IF_STA, &info))
```

6.3.2) Mode smartconfig

Permet d'envoyer le SSID et mot de passe en wi-fi broadcast. Quand le système a fini le smartconfig la tâche Esp32SmartConfig_task fait une sauvegarde en mémoire des informations.

```
if((this->pass != NULL) && (this->ssid != NULL))
{
    Wifi_saveSSIDAndPass(this->ssid, this->pass);
    OsFree(this->ssid);
    OsFree(this->pass);
}
```

Si la sauvegarde existe alors le smartconfig est désactivé.

6.3.3) Mise à jour (OTA)

Permet la mise à jour via wifi. Bientôt disponible.

6.3.4) Bouton poussoir

Permet de supprimer le SSID et mot de passe wifi. Le boutons est géré par une tache OS à trois états (Idel, front montant et front descendant).

```
typedef enum eButtonMode_t
{
  IDLE_BUTTON,
  RISING_BUTTON,
  FALLING_BUTTON
} eButtonMode_t;
```

6.3.5) Système de fichiers

Permet de lire ou écrire des données dans un système de fichiers. Exemple pour le SSID et mot de passe wi-fi.

```
if(Fs_write(SSID_WIFI_FILE, ssid, SSID_WIFI_FILE) != ESP_OK)
{
        BarDebug_err("Error to save SSID\n");
}

if(Fs_write(PASSWORD_WIFI_FILE, password, PASSWORD_WIFI_FILE) != ESP_OK)
{
```

```
        BarDebug_err("Error to save password\n");
}
```

6.3.6) Détection de fin de course

Permet de bien initialiser le système grâce à la fonction “MotorHandling_setInitialPosition”. Cette fonction est appelé au démarrage et a chaque demande de cocktail.

```
  uint32_t bp1[1], bp2[1];
  BarDebug_info("Set motor at the initial position on y...\n");
  Gpio_get(DETECTION_AXE_Y, bp2);
  BAR_ERROR_CHECK(Gpio_set(MOTOR_AXE_Y_DIR, BAR_LEVEL_HIGH));
  int end = MOTOR_LIMIT;
  while(bp2[0] && (end != 0))
  {
        Gpio_get(DETECTION_AXE_Y, bp2);
        BAR_ERROR_CHECK(Gpio_set(MOTOR_AXE_Y_CLK, BAR_LEVEL_LOW));
        CpuDelay_ms(1);
        BAR_ERROR_CHECK(Gpio_set(MOTOR_AXE_Y_CLK, BAR_LEVEL_HIGH));
        CpuDelay_ms(1);
        end--;
  }
  BarDebug_info("Motor is now at the initial position on y.\n");
  BarDebug_info("Set motor at the initial position on x...\n");
  Gpio_get(DETECTION_AXE_X, bp1);
  BAR_ERROR_CHECK(Gpio_set(MOTOR_AXE_X_DIR, BAR_LEVEL_HIGH));
  while(bp1[0])
  {
        Gpio_get(DETECTION_AXE_X, bp1);
        BAR_ERROR_CHECK(Gpio_set(MOTOR_AXE_X_CLK, BAR_LEVEL_LOW));
        CpuDelay_ms(1);
        BAR_ERROR_CHECK(Gpio_set(MOTOR_AXE_X_CLK, BAR_LEVEL_HIGH));
        CpuDelay_ms(1);
  }

  BarDebug_info("Motor is now at the initial position on x.\n");
```

6.3.7) Page web pour la cmd des cocktails

Permet de commander un cocktail. Une fois le système connecté au réseau wi-fi on appelle “Cocktail_init” et “QueueCocktail_init”.
La tache “Cocktail” va récupérer les informations des JSON pour les sauvegarder en RAM dans les structures ci-dessous:

```
typedef struct
{
  char name[MAX_NAME_SIZE];
  char note[MAX_NOTE_SIZE_FOR_BOTTLE];
  int position;
```

```
} sBottle;

typedef struct
{
  char name[MAX_NAME_SIZE];
  int measure;
} sIngredient;

typedef struct cocktail
{
  char name[MAX_NAME_SIZE];
  sIngredient ingredient[MAX_INGREDIENT];
} sCocktail;

sCocktail cocktail[MAX_COCKTAIL];
sBottle bottle[MAX_BOTTLE];
```

Quand la tâche “Cocktail” a fini de récupérer les informations on appelle “Html_init”. Qui va créer la page web HTML.

```
  html_indexBegin(http_index_hml);
  html_indexTitle(http_index_hml);
  html_tabBegin(http_index_hml);
  html_tabColumnBegin(http_index_hml, "Selection");
  html_tabColumnMiddle(http_index_hml, "Ajouter");
  html_tabColumnEnd(http_index_hml, "Fourni");
  Cocktail_createHtmlCodeForCocktails(http_index_hml);
  html_tabEnd(http_index_hml);
  html_indexEnd(http_index_hml);
```

La fonction “Cocktail_createHtmlCodeForCocktails” va trier les ingrédients disponible en fonction des bouteilles sur la machine à cocktail. Cela permet d’indiquer à l’utilisateur les ingrédients non disponible.

```
for(i = 0; i < MAX_COCKTAIL; i++)
{
  for(j = 0; j < MAX_INGREDIENT; j++)
  {
     if(cocktail[i].ingredient[j].name[0] != '\0')
     {
       if(Cocktail_isBottleExiste(cocktail[i].ingredient[j].name))
       {
...
```

La tache “QueueCocktail_receivedTask” est appellé quand un utilisateur sélectionne un cocktail sur la page web. Si on demande trois cocktail alors ils seront mis en liste d’attente. La variable “goToPosition” et “currentPosition” permettent de calculer la future position du plateau en fonction de l’état courant.

```
OsQueueReceive(pCtx->xQueueCocktailEventQueue, &QueueCocktail,
OsPortTimingPeriod);
LedRGBHandling_ExecuteLedTaskFromISR(BLUE_LED_FAST_BLINKING);
MotorHandling_setInitialPosition();
int nbIngredients = Cocktail_getDispoIngredients(bottleList.bottle, bottleList.position,
bottleList.measure, QueueCocktail);
int goToPosition = 0;
int currentPosition = 0;
for(int i = 0; i < nbIngredients; i++)
{
	if(currentPosition != bottleList.position[i])
	{
		goToPosition = bottleList.position[i] - currentPosition;
		MotorHandling_setPositionOnX(goToPosition);
		currentPosition += goToPosition;
		CpuDelay_ms(500);
	}
	if(currentPosition != 0)
	{
		MotorHandling_getAMeasureOnY(bottleList.measure[i]);
	}
	else
	{
		MotorHandling_getAMeasureOnPump(bottleList.measure[i]);
	}

}
MotorHandling_setInitialPosition();
LedRGBHandling_ExecuteLedTaskFromISR(BLUE_LED);
```

6.3.8) Commande moteurs

Permet de commander un moteur à courant continue en position 0. Pour le moment un seul moteur est possible.

```
BAR_ERROR_CHECK(Gpio_set(MOTOR_PUMP_1, BAR_LEVEL_HIGH));
for(int j = 0; j < measure; j++)
{
	CpuDelay_ms(1000);
}
BAR_ERROR_CHECK(Gpio_set(MOTOR_PUMP_1, BAR_LEVEL_LOW));
```

6.3.9) Commande moteurs Pas à Pas

Permet de commander les axes X et Y ainsi que leurs senses. Ci-dessous un exemple pour l'axe X. Le "MOTOR_OFFSET" permet de faire avancer le plateau de 10cm (distance entre chaque bouteille). "MOTOR_OFFSET" se calcule en fonction du degré d'un pas et du périmètre de la poulie.

```
if(position > 0)
{
	BAR_ERROR_CHECK(Gpio_set(MOTOR_AXE_X_DIR, BAR_LEVEL_LOW));
	end = position * MOTOR_OFFSET;
}
else
{
	BAR_ERROR_CHECK(Gpio_set(MOTOR_AXE_X_DIR, BAR_LEVEL_HIGH));
	end = (position * (-1)) * MOTOR_OFFSET;
}

for(int i = 0; i < end; i++)
{
	BAR_ERROR_CHECK(Gpio_set(MOTOR_AXE_X_CLK, BAR_LEVEL_LOW));
	CpuDelay_ms(1);
	BAR_ERROR_CHECK(Gpio_set(MOTOR_AXE_X_CLK, BAR_LEVEL_HIGH));
	CpuDelay_ms(1);
}
```

6.3.10) LED RGB

Permet de commander la couleur de la LED pour informer l'utilisateur. C'est une tâche qui gère le cadencement du clignotement. La variable "enableLed" permet de d'activer ou de désactiver la LED. La variable "queueReceiveDelay" permet de gérer la vitesse de clignotement en milliseconde.

```
OsQueueReceive(tsQueueForLed, &eAsyncMsg, queueReceiveDelay);
eAsyncCurrent = eAsyncMsg;

if(IDEL_LED == eAsyncMsg)
{
	LedRGBGpioDriver_SetColor(LED_NOT_DEFINED);
}
else if((BLUE_LED_FAST_BLINKING | enableLed) == eAsyncMsg)
{
	queueReceiveDelay = BLINK_FAST;
	if(!LedRGBGpioDriver_ToggleColor(LED_BLUE))
	{
		BarDebug_info("LED NOT DEFINED");
	}
}
...
```

6.3.11) cJson

Permet de récupérer les champs des JSON grâce à la librairie cJon. Voici un exemple pour récupérer les informations d'une bouteille.

```
cJSON * _name = cJSON_GetObjectItem(_bottle, "name");
if(_name != NULL)
{
```

```
	cJSON * _note = cJSON_GetObjectItem(_bottle, "note");
	if(_note != NULL)
	{
		cJSON * _position = cJSON_GetObjectItem(_bottle, "position");
		if(_position != NULL)
		{
		memcpy(bottle[i].name, _name->valuestring, strlen(_name->valuestring));
		memcpy(bottle[i].note, _note->valuestring, strlen(_note->valuestring));
		bottle[i].position = (int) _position->valuedouble;
...
```

6.13.12) Add-on

Rien pour le moment.

6.4) Le JSON

il y a deux JSON dans le projet. Le premier permet de définir l'emplacement des bouteilles et le deuxième la liste est ingrédient des cocktails.

6.4.1) Le JSON de l'emplacement des bouteilles

La table "bottles" ci dessous est une liste de bouteille avec un nom et une position. Attention la position doit exister physiquement sur la machine à cocktail. La position 0 est la position initial du plateau.

```
{
   "bottles": [{
                  "bottle": {
                        "name": "eau",
                        "note": "0%/vol",
                        "position": 0
                  }
         }, {
                  "bottle": {
                        "name": "jus de pomme",
                        "note": "0%/vol",
                        "position": 1
                  }
         },
         ….
    }]
}
```

6.4.2) Le JSON de la liste est ingrédient des cocktails.

La table "cocktails" ci dessous est une liste de cocktail. Un cocktail doit avoir un nom et une table "ingredients" qui est une liste d'ingrédient. Chaque ingrédient à un nom (qui n'est pas obligatoirement disponible dans la machine à cocktail) et une "measure" qui est la quantité de liquide à servir (1 = 2ml, 2 = 4ml, ...).

```
{
   "cocktails": [{
                  "cocktail": {
                        "name": "Virgin mojito",
                        "ingredients": [{
                                    "ingredient": {
                                          "name": "sirop de mojito",
                                          "measure": 2
                                    }
                              }, {
                                    "ingredient": {
```

```
                                        "name": "menthe verte",
                                        "measure": 1
                                }
                        }, {
                                "ingredient": {
                                        "name": "jus de citron",
                                        "measure": 1
                                }
                        }, {
                                "ingredient": {
                                        "name": "sucre de canne",
                                        "measure": 1
                                }
                        }, {
                                "ingredient": {
                                        "name": "eau gazeuse",
                                        "measure": 2
                                }
                        },
                        {
                                "ingredient": {
                                        "name": "glace pilee",
                                        "measure": 1
                                }
                        }
                ]
        }
    }, {
     ....
  }]
}
```

Lors de l'affichage d'un cocktail sur la page web on peut voir deux colonnes. Une colonne pour les ingrédients disponible dans la machine et une colonne pour les ingrédients à rajouter manuellement.

6.5) La page web

Oui mais pourquoi une page web et pas une application mobile ? La réponse est simple, c'est parce que c'est compatible sur tous les smartphone et PC.

La page web est accessible à l'adresse IP http://192.168.1.51 de votre reseau. C'est du code HTML et CSS généré par le code C du wroom32. Au démarrage du module le programme parcourt les JSON "bottles" et "cocktails" pour créer un tableau HTML dynamique à trois colonnes et N ligne(s). Dans la colonne une nous avons les boutons CSS avec le nom du cocktail (disponible dans le JSON). Dans la colonne deux nous avons les ingrédient(s) à ajouter manuellement (non disponible dans liste de bouteille). Dans la

colonne trois nous avons les ingrédient(s) disponible dans la machine à cocktail (disponible dans liste de bouteille).

6.5.1) Le CSS

Dans le CSS nous avons les boutons, la police, la couleur d'arrière plan...

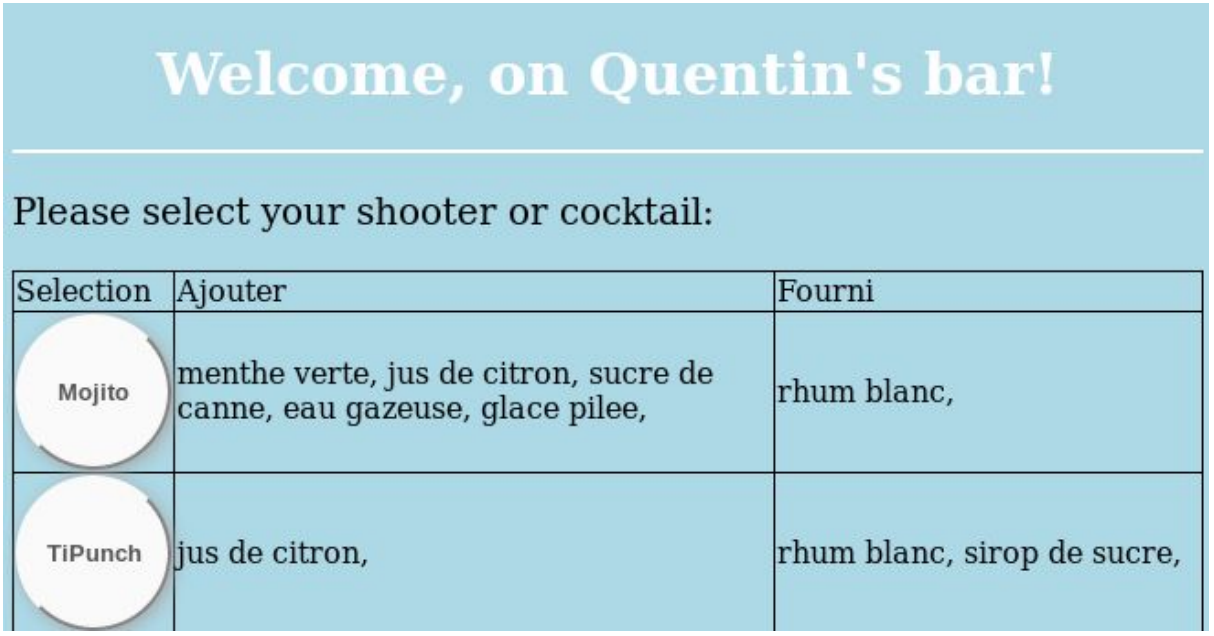
```
<style>
h1 {
color: white;
text-align: center;
}
p {
font-family: verdana;
font-size: 20px;
}
body {
background-color: lightblue;
}
table
{
border-collapse: collapse;
}
td
{
border: 1px solid black;
}
.button {
 width:85px;
 height:85px;
 background:#fafafa;
 box-shadow:2px 2px 8px #aaa;
 font:bold 13px Arial;
 border-radius:50%;
 color:#555;
}
</style>
```

6.5.1) Le HTML

Dans la première ligne du code HTML on trouve le titre et un lien vers une image. Cette ligne Cela permet, quand on ajoute le raccourci de la page web sur un smartphone d’avoir un beau raccourcie avec un nom et logo.

<title>Connected bar</title><link href="https://www.bodipure.com/wp-content/uploads/2016/09/B-favicon.png" rel="icon" type="image/x-icon" />	

La suite du code HTML est le tableau à trois colonnes. Voici un exemple ci-dessous:



Welcome, on Quentin's bar!

Please select your shooter or cocktail:

Selection	Ajouter	Fourni
Mojito	menthe verte, jus de citron, sucre de canne, eau gazeuse, glace pilee,	rhum blanc,
TiPunch	jus de citron,	rhum blanc, sirop de sucre,

7) Mécanique

7.1) Le bois

J’ai acheté le bois et les équerres métalliques dans un magasin de bricolage. Vous avez besoin d’une perceuse, d’une scie circulaire, et un tournevis.

7.2) Le plateau

Vous pouvez facilement réaliser vous même le plateau mécanique cependant Pour éviter de passer trop de temps sur la réalisation de cette partie mécanique j’ai utiliser les plans 3D proposer par “DIY Machines” (https://www.thingiverse.com/thing:2478890/zip).

8) Mise en route et fonctionnement

Vous devez cloner le dépôt git :

8.1) Configurer le programme

Après avoir soudé votre carte et construire la partie mécanique du système vous devez configurer certains paramètres.
1 - Board.h
Permet de définir le mapping des GPIO du système. Exemple pour la LED RGB:

```
#define LED_GPIO_RED        BAR_GPIO_NUM_32
#define LED_GPIO_GREEN      BAR_GPIO_NUM_33
#define LED_GPIO_BLUE       BAR_GPIO_NUM_25
```

2 - les moteurs
vous devez modifier les “#define” de Moteur.c pour calibrer le nombre de pas en fonction de la mécanique de la machine à cocktail.

3 - Le JSON
Vous devez rentrer vos bouteilles et créer votre cocktail.

8.2) Compiler et programmer le code

1 - Dans un terminal GNU/Linux faire un exporte du SDK et de la toolchain

```
$ export IDF_PATH=/YOUR_PATH/esp-idf
$ export PATH=$PATH:/YOUR_PATH/xtensa-esp32-elf/bin/
```

2 - Dans le projet faire un “make”

3 - Brancher la carte avec le jumper Boot. Puis utiliser le logiciel de flashage de espressif.

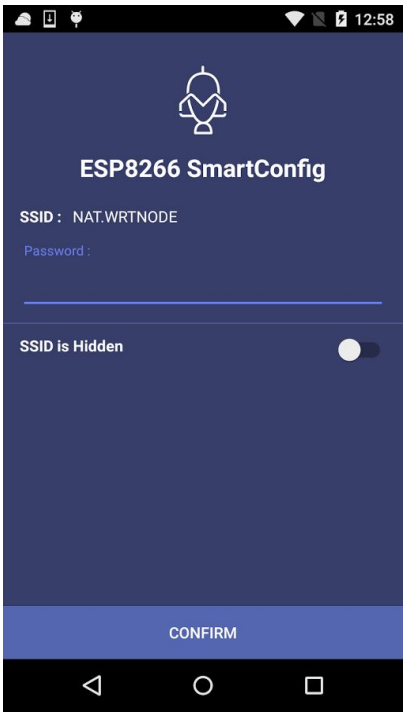
```
$ python /esptool.py --chip esp32 --port /dev/ttyUSB0 --baud 115200 --before
default_reset --after hard_reset write_flash -z --flash_mode dio --flash_freq 40m
--flash_size detect 0x1000 bootloader.bin 0x10000 cocktail-machine.bin 0x8000
customPart.bin
```

8.3) Premier démarrage

1 - Au démarrage la LED est rouge, l’axe Y cherche la position de fin de course, puis l’axe X cherche la position de fin de course. Une fois le système initialisé la LED passe bleu.

2 - Au premier démarrage du système il faut connecter le système à votre box wi-fi. Pour cela télécharger sur le store de votre téléphone l’application “ESP8266 SmartConfig”.

Connecter votre téléphone à votre wifi, et lancer l'application espressif. Vous avez juste à rentrer votre mot de pass wifi et cliquer sur “confirm”. Le module va sauvegarder en flash votre SSID et mot de passe. une fois sauvegarder le mode smartconfig sera désactivé au démarrage.



3 - Allez sur http://192.168.1.51 et choisiser votre cocktail.

4 - Enjoy :-)

8.4) Vidéo

Vous pouvez voir la vidéo sur ma chaine youtube :
https://www.youtube.com/user/MrAS2aN/videos?shelf_id=0&view=0&sort=dd

8.5) Hygiène de la machine à cocktail

Pour une bonne hygiène pensez à bien nettoyer la machine à chaque changement de bouteille.

9) L'avenir du projet

La version présenté est la version 1. Bien des choses restent à modifier et ajouter

9.1) Modifier

-Les vis sans fin de l’axe Y.
-Les doseurs de bouteille (J’ai pris le premier prix, ce n’est pas extraordinaire...).
-La position 0 doit pouvoir gérer N pompe(s).

9.2) Ajouter

-Détection du verre.
-OTA : Les fonctions de mise à jour de la HAL sont fonctionnelles mais il faut les ajouter au programme.
-Add-on de lait en poudre : C’est une fonctionnalitée indispensable pour moi. La préparation d’un biberon automatique.
-Google home et Alexa via IFTTT (pour une commande vocale)