Systèmes Embarqués II

Multitâche

ISAT – EPHEC 2020-2021

Juan Alvarez et Olivier Grabenweger

2 Table des matières

1 Introduction

Ce TP sert d'introduction au multitâche avec l'ESP32 et à nous montrer la puissance de ce dernier après avoir passé tout le reste du quadrimestre avec l'ESP8266.

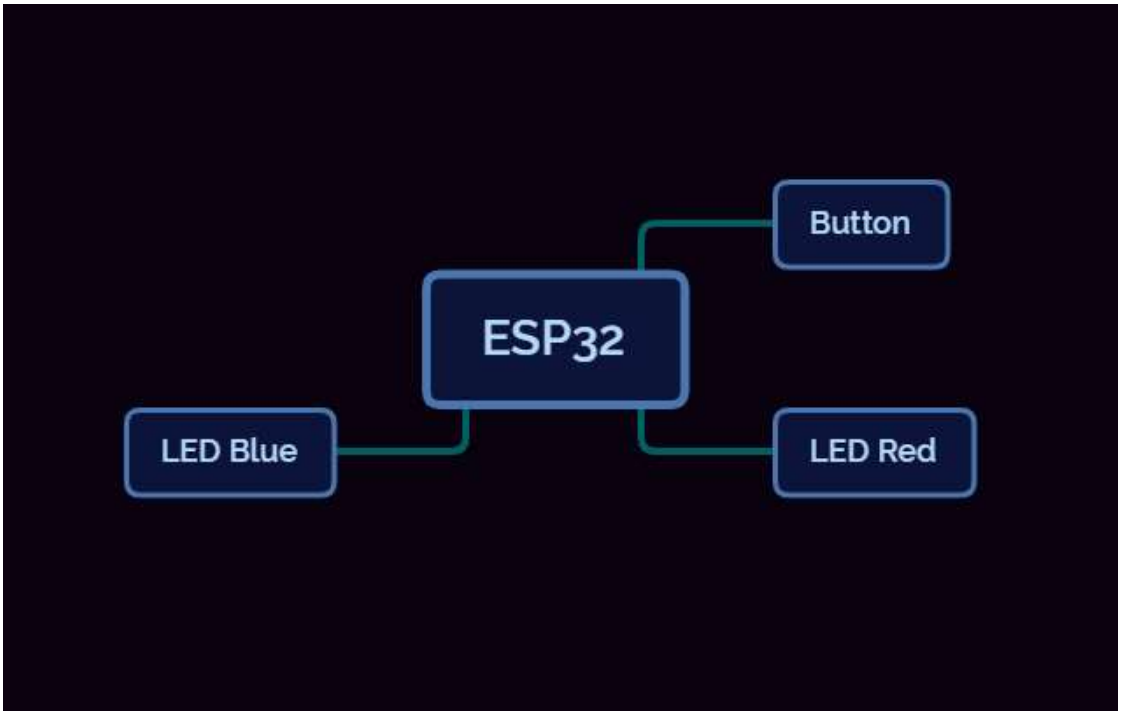
Nous allons utiliser free RTOS, un système d'exploitation temps réel qui permet l'ordonnance de tâches.

L'objectif est d'écrire un programme pour piloter trois périphériques différents : 2 LED et un bouton-poussoir.

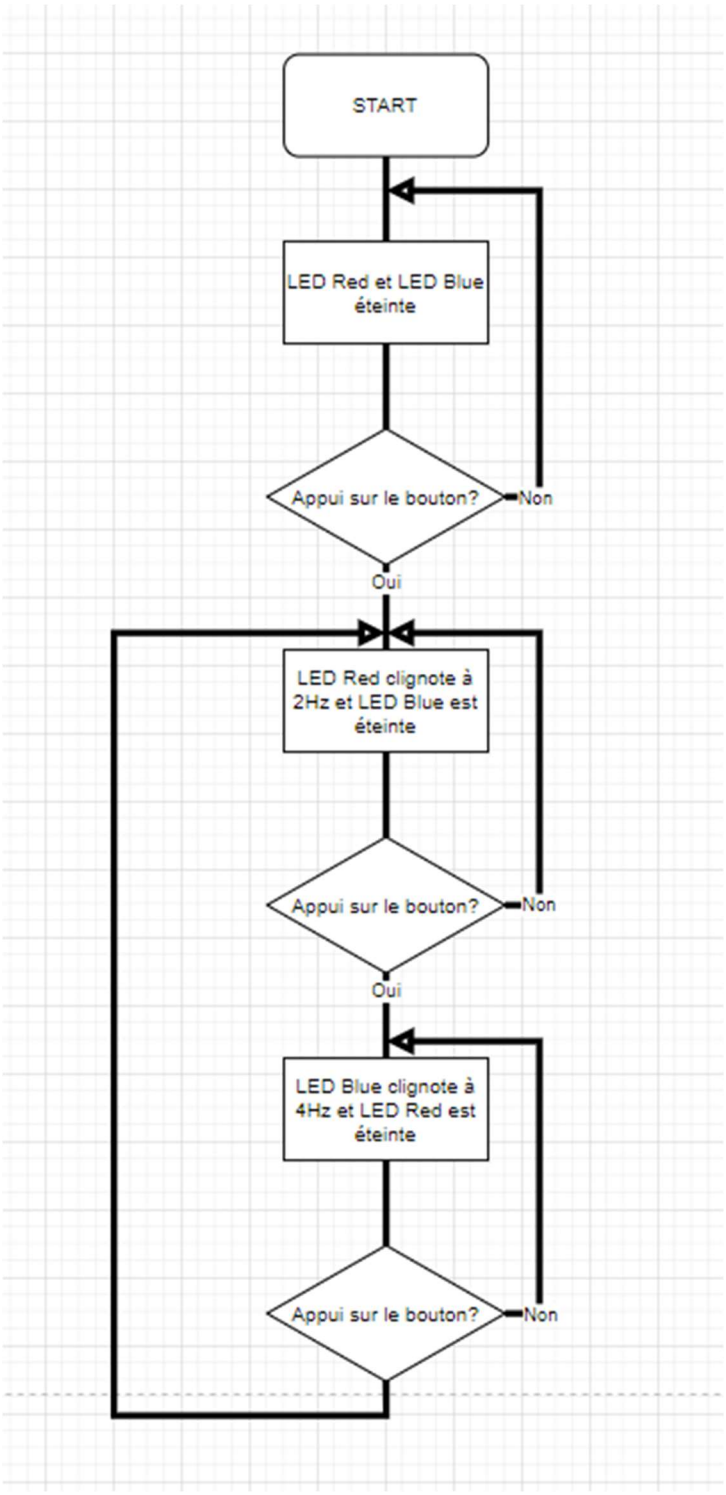
Au démarrage, les LED sont éteintes. Quand on appuie sur le bouton pour la première fois, la LED Red devra clignoter avec une période d'une demi seconde (500ms) tandis que la LED Blue devra rester éteinte. Au deuxième appui, c'est la LED Blue qui devra clignoter avec une période d'un quart de seconde cette fois (250ms) alors que la LED Red sera éteinte. Et le cycle se répètera indéfiniment.

2 Mindmapping - Logigramme

a) Mindmap



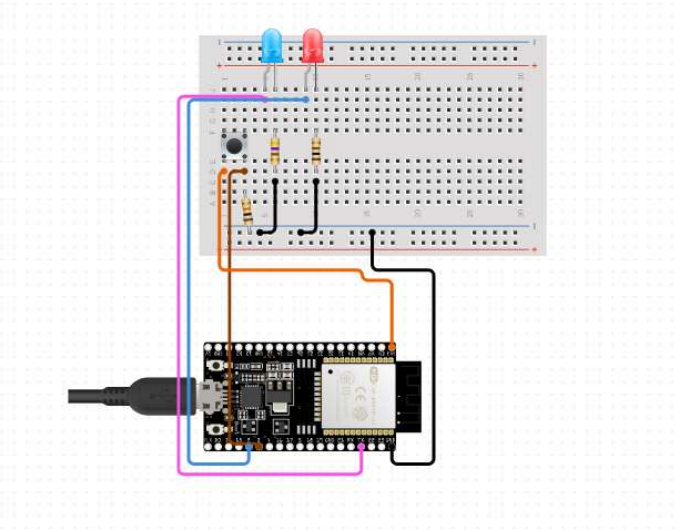
b) Logigramme



3 Schéma de câblage

a) Composants

- 2 LEDs => 1 rouge et 1 bleue
- 1 Bouton-poussoir
- 3 résistances => 1 de 10kΩ et 2 autres de 330Ω
- 1 bred board
- 1 ESP32



4 Code source

a) Code ESP32 :

```
/* bibliothèques */

#include <Arduino.h>


/* Variables */

const int BUTTON = GPIO_NUM_21;    //Pin du bouton

const int LEDred = GPIO_NUM_22;    //Pin de la LED rouge

const int LEDblue = GPIO_NUM_23;     //Pin de la LED bleu


int flag_button;    //flag du bouton

int state_button;    //état du bouton

int count;     //compteur


static QueueHandle_t qh;      //Création de notre file qh


/* Tâche Bouton */

void TaskButton(void *arg)

{

  int bufferSendData;    //variable de la donnée qui sera envoyée
```

```
33
34   for(;;)
35
36   {
37
38     state_button = digitalRead(BUTTON);   //lecture de l'état du bouton
39
40     if (state_button == LOW)     // Système avec flag qui permet d'éviter le
41 rebond du bouton
42     {
43
44         flag_button = 1;
45
46     }
47
48     if (flag_button == 1 && state_button == HIGH)
49
50     {
51
52         count += 1;   //incrémentation du compteur
53
54         flag_button = 0;
55
56     }
57     if (count == 0)
58
59     {
60
61       bufferSendData = 2;   //Valeur envoyée aux autres tâches pour que les
62 2 LED restent éteintes au démarrage
63     }
64     else
65     {
66
67       bufferSendData = count % 2;   //Valeur envoyée aux autres tâches (1 si
68 count est impair et 0 si count est pair)
69     }
70     //(file,  buffer valeur envoyée,  Timeout)
71
72     xQueueSendToBack(qh, &bufferSendData, portMAX_DELAY);   //fonction pour
73 envoyer la valeur dans la file
74   }
75 }
76
77 /* Tâche LED Red */
78
79 void TaskLEDRed(void *arg)
80
81 {
82   int bufferGetDataRed;   //variable de la donnée reçue
83   for(;;)
84   {
85     //(file,  buffer valeur reçue,  Timeout)
86
87     xQueueReceive(qh,&bufferGetDataRed,portMAX_DELAY);  //fonction pour
88 recevoir la valeur de la file
```

```
 89
 90     if (bufferGetDataRed == 1)     //si la valeur est égale à 1, la LED rouge
 91 clignote à une fréquence de 2Hz
 92
 93     {
 94
 95       digitalWrite(LEDred,1);    //LED rouge s'allume
 96
 97       delay(500);                //500ms de délai
 98
 99       digitalWrite(LEDred,0);    //LED rouge s'éteind
100
101       delay(500);                //500ms de délai
102
103     }
104
105     else
106
107     {
108
109       digitalWrite(LEDred,0);    //LED rouge s'éteind
110
111     }
112
113   }
114
115 }
116
117 /* Tâche LED Bleu */
118
119 void TaskLEDBlue(void *arg)
120
121 {
122
123   int bufferGetDataBlue;     //variable de la donnée reçue
124
125   for(;;)
126
127   {
128
129     //(file,  buffer valeur reçue,  Timeout)
130
131     xQueueReceive(qh,&bufferGetDataBlue,portMAX_DELAY);
132 //fonction pour recevoir la valeur de la file
133 //si la valeur est égale à 0, la LED bleu clignote à une fréquence de 4Hz
134     if (bufferGetDataBlue == 0)
135
136     {
137       digitalWrite(LEDblue,1);     //LED bleu s'allume
138
139       delay(250);                  //250ms de délai
140
141       digitalWrite(LEDblue,0);     //LED bleu s'éteind
142
143       delay(250);                  //250ms de délai
144
```

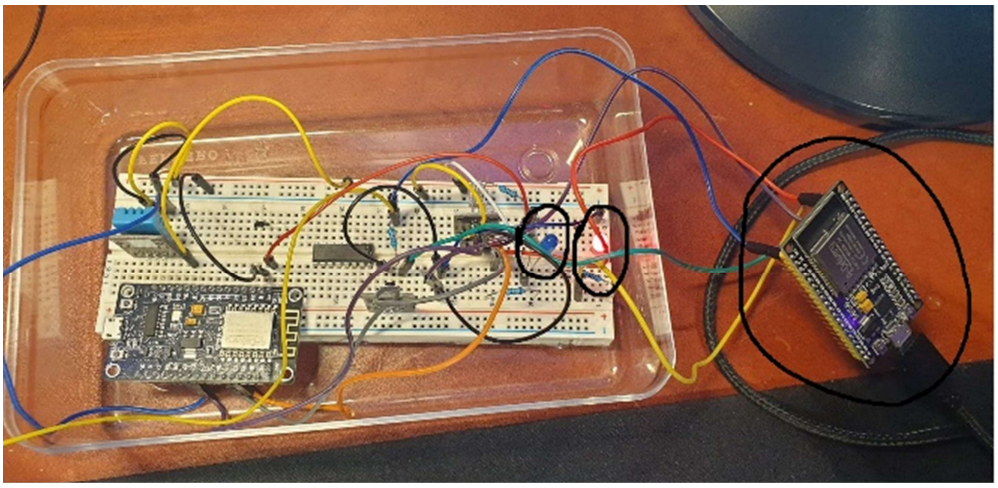
```
145     }
146
147     else
148
149     {
150
151       digitalWrite(LEDblue,0);     //LED bleu s'éteind
152
153     }
154
155   }
156
157 }
158
159 /* fonction setup */
160
161
162 void setup()
163
164 {
165
166   pinMode(LEDred, OUTPUT);      // Initialise la LED rouge en output
167
168   digitalWrite(LEDred, LOW);     //Eteind la LED rouge
169
170   pinMode(LEDblue, OUTPUT);      // Initialise la LED Bleu en output
171
172   digitalWrite(LEDblue, LOW);    //Eteind la LED bleu
173
174   pinMode(BUTTON, INPUT);        // Initialise le bouton en input
175
176
177   //mise à zéro de quelques variables
178
179   flag_button = 0;
180
181   state_button = 0;
182   count = 0;
183
184   qh = xQueueCreate(3,8); //paramètrage/configuration de la file qh
185 //(la) Tâche, le nom de la tâche, taille, paramètre, priorité, handle, CPU
186   xTaskCreatePinnedToCore(TaskButton,"TaskButton",2048,NULL,1,NULL,1);
187
188   xTaskCreatePinnedToCore(TaskLEDRed,"TaskLEDRed",2048,NULL,1,NULL,1);
189
190   xTaskCreatePinnedToCore(TaskLEDBlue,"TaskLEDBlue",2048,NULL,1,NULL,1);
191
192 }
193
194 void loop()
195
196 {
197
198   //Nothing else matters
199
200 }
```

5 Conclusion

En observant la puissance de l'ESP32 avec son dual core et la puissance du multitâche, on s'est rendu compte que le retour en arrière vers l'ESP8266 ne sera sans doute plus possible pour nous. Mais c'est loin d'être une perte. Avec le free RTOS et sa gestion des tâches en envoyant des données dans une file, on arrive aisément à faire tourner plusieurs tâches en même temps.
Ceci nous a permis de nous passer de la fonction « Millis » que nous utilisions dans nos précédents TP pour ne pas être bloqué par des délais.

6 Annexes, bibliographie et illustrations

a) Annexes



b) Bibliographie

- https://www.circuito.io/ : schéma de câblage
- http://draw.io/ : organigramme
- Xmind : Mindmap
- http://www.esp32learning.com/code/esp32-and-freertos-example-create-a-task.php
- https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-reference/system/freertos.html
- file:///C:/Users/olivi/Downloads/Multit%C3%A2che%20en%20pratique%20avec%20lESP32%20(Elektor%202020)%20(1).pdf
- https://cdn.shopify.com/s/files/1/1509/1638/files/ESP_-_32_NodeMCU_Developmentboard_Datenblatt_AZ-Delivery_Vertriebs_GmbH_10f68f6c-a9bb-49c6-a825-07979441739f.pdf?v=1598356497