Programmation FreeRTOS

Installation de l'environnement de développement ESPRESSIF

Vous installerez manuellement :

- ▷ le SDK Espressif pour freeRTOS ;
- ▷ le compilateur, *toolchain*, pour le processeur Xtensa ;

xterm
```
$ cd ~/esp
$ wget
https://dl.espressif.com/dl/xtensa-lx106-elf-gcc8_4_0-esp-2020r3-linux-amd64.tar.gz
$ tar xvfz xtensa-lx106-elf-gcc8_4_0-esp-2020r3-linux-amd64.tar.gz
$ git clone https://github.com/espressif/ESP8266_RTOS_SDK.git
$ echo "export IDF_PATH=~/esp/ESP8266_RTOS_SDK" > environnement
$ echo "export PATH=~/esp/xtensa-lx106-elf/bin:$PATH" >> environnement
```

Vous vérifierez que votre ESP8266 est bien détecté en USB :

xterm
```
$ lsusb
Bus 001 Device 005: ID 10c4:ea60 Silicon Labs CP210x UART Bridge
$ ls /dev/ttyUSB*
/dev/ttyUSB0
```

Ici, non seulement l'ESP8266 est bien reconnu, mais il est connecté sur `/dev/ttyUSB0`

Vous pouvez ensuite vous assurez que l'application est bien configurée pour être flashée sur le bon `/dev` à l'aide du menu de configuration obtenu avec :

xterm
```
$ cd ~/esp
$ source environnement
$ cd ESP8266_RTOS_SDK/examples/get-started/hello_world
$ make menuconfig
```

Ce qui donne le menu suivant :

xterm
```
 Espressif IoT Development Framework Configuration
  Arrow keys navigate the menu.  <Enter> selects submenus ---> (or empty submenus
  ----).  Highlighted letters are hotkeys.  Pressing <Y> includes, <N> excludes,
<M>
  modularizes features.  Press <Esc><Esc> to exit, <?> for Help, </> for Search.
  Legend: [*] built-in  [ ] excluded  <M> module  < > module capable

            SDK tool configuration  --->
            Bootloader config  --->
            Serial flasher config  --->
            Partition Table  --->
            Compiler options  --->
            Component config  --->

                <Select>    < Exit >    < Help >    < Save >    < Load >
```

Puis :

xterm

```
 Serial flasher config
  Arrow keys navigate the menu.  <Enter> selects submenus ---> (or empty submenus
  ----).  Highlighted letters are hotkeys.  Pressing <Y> includes, <N> excludes,
<M>
  modularizes features.  Press <Esc><Esc> to exit, <?> for Help, </> for Search.
  Legend: [*] built-in  [ ] excluded  <M> module  < > module capable

       (/dev/ttyUSB0) Default serial port
           Default baud rate (115200 baud)  --->
       [*] Use compressed upload
           Flash SPI mode (QIO)  --->
           Flash SPI speed (40 MHz)  --->
           Flash size (2 MB)  --->
           Before flashing (Reset to bootloader)  --->
           After flashing (Hard reset after flashing)  --->
           'make monitor' baud rate (74880 bps)  --->


               <Select>    < Exit >    < Help >    < Save >    < Load >
```

Pour compiler l'application :

xterm

```
$ make all
```

Pour flasher le firmware sur l'ESP8266 :

xterm

```
$ make flash
```

Pour se connecter au port série de l'ESP8266 :

xterm

```
$ make monitor
```

Vous taperez `ctrl-]` pour sortir.

Programmation freeRTOS : lancement de tâches et priorité

Vous pourrez récupérer le source complet de l'application sur :

xterm
```
$ git clone
https://git.unilim.fr/pierre-francois.bonnefoi/freeRTOS_clignoter_led.git
```

```
#include <stdio.h>
#include <string.h>
#include "freertos/FreeRTOS.h"
#include "freertos/task.h"
#include "esp_system.h"
#include "esp_spi_flash.h"
#include "freertos/semphr.h"
#include "freertos/event_groups.h"
#include "driver/uart.h"
#include "driver/gpio.h"


// Task: Blink LED at rate set by global variable
void toggleLED(void *parameter) {

  gpio_set_direction(2, GPIO_MODE_OUTPUT);
  while (1) {
    gpio_set_level(2, 1);
        printf("0\n");
    vTaskDelay(700 / portTICK_PERIOD_MS);
    gpio_set_level(2, 0);
        printf("1\n");
    vTaskDelay(700 / portTICK_PERIOD_MS);
  }
}

void app_main()
{
  // Configure serial and wait a second
  vTaskDelay(1000 / portTICK_PERIOD_MS);
  printf("freeRTOS LED Demo\n");

  // Start blink task
  xTaskCreate(
            toggleLED,      // Function to be called
            "Toggle LED",   // Name of task
            1500,           // Stack size (bytes in ESP32, words in FreeRTOS)
            NULL,           // Parameter to pass
            1,              // Task priority
            NULL);           // Task handle
}
```

Comparez à la version Arduino.