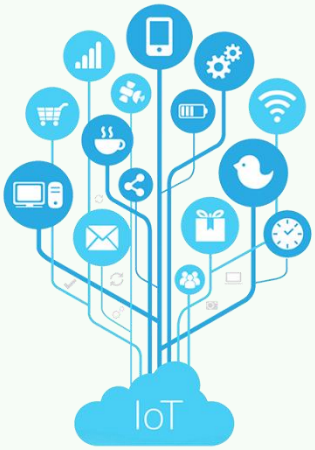
UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE

TP n°0 : Solutions

MASTER RÉSEAUX ET TÉLÉCOM



VERSION : 10 juin 2021

Florent NOLOT
UNIVERSITÉ DE REIMS CHAMPAGNE ARDENNE

Table des matières

I. Solutions

A. Exercice 1 – Clignotement LEDs

1. Question 1

Ecrire un programme qui fait clignoter deux fois la LED rouge puis de deux fois la LED verte en continue. Le délai entre chaque clignotement est de 1 seconde. Attention à ne jamais avoir deux LEDs allumées en même temps.

Dans le fichier « Board.h », on récupère le nom de la LED verte « Board_GPIO_LED1 ».

```
99 #define Board_GPIO_LED0          CC1350_LAUNCHXL_433_GPIO_LED_RED
100 #define Board_GPIO_LED1          CC1350_LAUNCHXL_433_GPIO_LED_GREEN
101 #define Board_GPIO_RLED          CC1350_LAUNCHXL_433_GPIO_LED_RED
102 #define Board_GPIO_GLED          CC1350_LAUNCHXL_433_GPIO_LED_GREEN
103 #define Board_GPIO_LED_ON        CC1350_LAUNCHXL_433_GPIO_LED_ON
104 #define Board_GPIO_LED_OFF       CC1350_LAUNCHXL_433_GPIO_LED_OFF
```

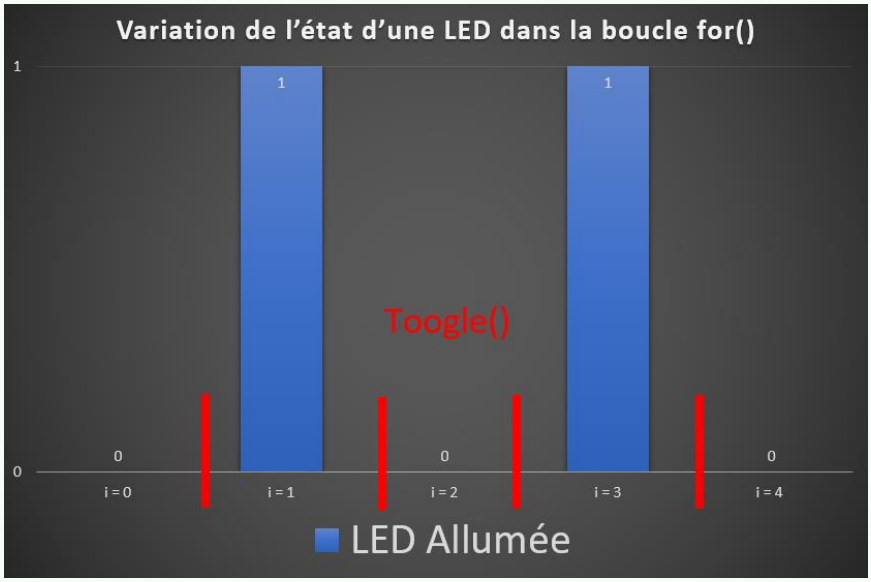
Modification du fichier « empty.c »

```
/*
 *  ======== mainThread ========
 */
void *mainThread(void *arg0)
{
    /* 1 second delay */
    uint32_t time = 1;
    int i;

    /* Call driver init functions */
    GPIO_init();

    /* Configure the LED pin */
    GPIO_setConfig(Board_GPIO_LED0, GPIO_CFG_OUT_STD | GPIO_CFG_OUT_LOW);
    GPIO_setConfig(Board_GPIO_LED1, GPIO_CFG_OUT_STD | GPIO_CFG_OUT_LOW);

    /* Turn off user LED */
    GPIO_write(Board_GPIO_LED0, Board_GPIO_LED_OFF);
    GPIO_write(Board_GPIO_LED1, Board_GPIO_LED_OFF);

    while (1) {

        for( i=0 ; i < 4 ; i++){
            sleep(time);
            GPIO_toggle(Board_GPIO_LED0);
        }
        for( i=0 ; i < 4 ; i++){
            sleep(time);
            GPIO_toggle(Board_GPIO_LED1);
        }

    }
}
```

Positionner de la LED verte en sortie

Eteindre la LED verte

Basculer les LEDs



2. Question 2

Changer le délai entre chaque clignotement pour qu'il soit égale à 500 ms. (Astuce : utiliser la fonction *usleep()*).

Le prototype de la fonction *usleep()* est définie dans le fichier « …/source/ti/posix/ccs/unistd.h »

```
extern unsigned sleep(unsigned seconds);
extern int usleep(useconds_t useconds);
```

Modification du fichier « empty.c »

```
/*
 *  ======== mainThread ========
 */
void *mainThread(void *arg0)
{
    /* 1s   = 1 000 000 microseconds
     * 0.5s =   500 000 microseconds */
    useconds_t time = 500000;
    int i;

    /* Call driver init functions */
    GPIO_init();

    /* Configure the LED pin */
    GPIO_setConfig(Board_GPIO_LED0, GPIO_CFG_OUT_STD | GPIO_CFG_OUT_LOW);
    GPIO_setConfig(Board_GPIO_LED1, GPIO_CFG_OUT_STD | GPIO_CFG_OUT_LOW);

    /* Turn off user LED */
    GPIO_write(Board_GPIO_LED0, Board_GPIO_LED_OFF);
    GPIO_write(Board_GPIO_LED1, Board_GPIO_LED_OFF);

    while (1) {

        for( i=0 ; i < 4 ; i++){
            usleep(time);
            GPIO_toggle(Board_GPIO_LED0);
        }
        for( i=0 ; i < 4 ; i++){
            usleep(time);
            GPIO_toggle(Board_GPIO_LED1);
        }

    }
}
```

B. Exercice 2 – Gestion des interruptions

Modifiez le programme pour que quand aucun bouton n'est pressé, aucune LED s'allume. Si le bouton 1 est pressé la LED rouge clignote une fois, si le bouton 2 est pressé la LED verte clignote une fois, le délai entre chaque clignotement doit être de 500 ms.

Suppression de la boucle infinie dans le fichier « main_nortos.c »

```c
/*
 *  ======== main_nortos.c ========
 */
#include <stdint.h>
#include <stddef.h>

#include <NoRTOS.h>

/* Example/Board Header files */
#include <ti/drivers/Board.h>

extern void *mainThread(void *arg0);

/*
 *  ======== main ========
 */
int main(void)
{
    /* Call driver init functions */
    Board_init();

    /* Start NoRTOS */
    NoRTOS_start();

    /* Call mainThread function */
    mainThread(NULL);

    return 0;
}
```

Modification du fichier « gpiointerrupt.c »

```c
#include <stdint.h>             /* For gpiointerrupt.c */
#include <stddef.h>             /* For gpiointerrupt.c */
#include <unistd.h>             /* For usleep() */
#include <ti/drivers/GPIO.h>    /* For Driver Header files */

#include "Board.h"              /* For Example/Board Header files */


short int num_led=-1;
useconds_t time = 500000;

/*
 *  ======== gpioButtonFxn0 ========
 *  Callback function for the GPIO interrupt on Board_GPIO_BUTTON0.
 */
void gpioButtonFxn0(uint_least8_t index){
    num_led = Board_GPIO_LED0;
}

/*
 *  ======== gpioButtonFxn1 ========
 *  Callback function for the GPIO interrupt on Board_GPIO_BUTTON1.
 *  This may not be used for all boards.
 */
void gpioButtonFxn1(uint_least8_t index){
    num_led = Board_GPIO_LED1;
}
```

Instruction exécutée lors d'une interruption par la pression du bouton 0

Instruction exécutée lors d'une interruption par la pression du bouton 1

```c
/*
 *  ======== mainThread ========
 *  LAUNCHPAD XL CC1350F128 OWNS 2 BUTTONS
 *  Others development boards may have only 1 button
 */
void *mainThread(void *arg0)
{
    /* Call driver init functions */
    GPIO_init();

    /* Configure the LEDs and buttons pins */
    GPIO_setConfig(Board_GPIO_LED0, GPIO_CFG_OUT_STD | GPIO_CFG_OUT_LOW);
    GPIO_setConfig(Board_GPIO_LED1, GPIO_CFG_OUT_STD | GPIO_CFG_OUT_LOW);
    GPIO_setConfig(Board_GPIO_BUTTON0, GPIO_CFG_IN_PU | GPIO_CFG_IN_INT_FALLING);
    GPIO_setConfig(Board_GPIO_BUTTON1, GPIO_CFG_IN_PU | GPIO_CFG_IN_INT_FALLING);

    /* Turn off user LEDs */
    GPIO_write(Board_GPIO_LED0, Board_GPIO_LED_OFF);
    GPIO_write(Board_GPIO_LED1, Board_GPIO_LED_OFF);

    /* install Buttons callback */
    GPIO_setCallback(Board_GPIO_BUTTON0, gpioButtonFxn0);
    GPIO_setCallback(Board_GPIO_BUTTON1, gpioButtonFxn1);

    /* Enable interrupts */
    GPIO_enableInt(Board_GPIO_BUTTON0);
    GPIO_enableInt(Board_GPIO_BUTTON1);

    while(1){

        if(num_led >= 0){
            GPIO_toggle(num_led);
            usleep(time);
            GPIO_toggle(num_led);
            num_led = -1;
        }

    }
}
```

Appels des fonctions ISR si pression sur bouton

Basculer une LED si pression sur bouton

C. Exercice 3 – Communication UART

Modifiez le programme pour qu'à chaque pression d'une touche de clavier la LED verte clignote. Attention on ne doit pas voir le délai pendant le moment de pression de la touche du clavier et celui de l'apparition du caractère sur le terminal de la liaison séries.

Modification du fichier « uartecho.c »

```c
void *mainThread(void *arg0)
{
    char        input;
    const char  echoPrompt[] = "Echoing characters:\r\n";
    UART_Handle uart;
    UART_Params uartParams;
    useconds_t time = 50000;

    /* Call driver init functions */
    GPIO_init();
    UART_init();

    /* Configure the LED pin */
    GPIO_setConfig(Board_GPIO_LED1, GPIO_CFG_OUT_STD | GPIO_CFG_OUT_LOW);

    /* Turn off user LED */
    GPIO_write(Board_GPIO_LED1, Board_GPIO_LED_OFF);

    /* Create a UART with data processing off. */
    UART_Params_init(&uartParams);
    uartParams.writeDataMode = UART_DATA_BINARY;
    uartParams.readDataMode = UART_DATA_BINARY;
    uartParams.readReturnMode = UART_RETURN_FULL;
    uartParams.readEcho = UART_ECHO_OFF;
    uartParams.baudRate = 38400;

    uart = UART_open(Board_UART0, &uartParams);

    if (uart == NULL) {
        /* UART_open() failed */
        while (1);
    }

    UART_write(uart, echoPrompt, sizeof(echoPrompt));

    /* Loop forever echoing */
    while (1) {
        UART_read(uart, &input, 1);
        GPIO_write(Board_GPIO_LED1, Board_GPIO_LED_ON);
        usleep(time);
        GPIO_write(Board_GPIO_LED1, Board_GPIO_LED_OFF);
        UART_write(uart, &input, 1);
    }
}
```

Délai de 50 ms

Affichage de « Echoing Characters : »

Communication UART

Affichage du résultat sur le terminal de la communication série