**Software Timer in FreeRTOS**

Benjamin A. Blouin

*benjamin.blouin@temple.edu*

**Summary**

**Introduction**

**Discussion**

**Conclusions**

**Appendices**

**GitHub**

<https://github.com/3keepmovingforward3/Embedded-System-Design-Sp19/blob/master/software_timer_in_free_rtos/software_timer_in_free_rtos.sdk/lab9/src/main.c>

**Pictures**

*A screenshot of a cell phone

Description automatically generated*

Figure Block Diagram

**Code**

*/\**

*\* main.c*

*\**

*\* Created on: Apr 17, 2019*

*\* Author: bblouin*

*\*/*

*// FreeRTOS includes*

*#include "FreeRTOS.h"*

*#include <task.h>*

*#include <timers.h>*

*#include <stdio.h>*

*// Xilinx includes*

*#include <xgpio.h>*

*#include <xil\_printf.h>*

*// Preprocessor Definitions*

*#define TIMER\_ID 1*

*#define BUTTONS 1*

*#define SWITCHES 2*

*#define FOURSEC 4000UL*

*static TickType\_t four = pdMS\_TO\_TICKS(FOURSEC); // 4 seconds*

*static TaskHandle\_t Task\_BTN\_SW = NULL; // Taskhandle object*

*static XGpio xBTNSWI, xLEDS; // gpio objects*

*static xTimerHandle xtimer1; // timer handle*

*// Function Prototypes*

*static void prvTask\_BTN\_SW(void \*pvParameters); // Task*

*static void vTimer1Callback(TimerHandle\_t pxTimer); // Timer*

*static void gpioInit(); //gpio initialization*

*int main (void){*

*gpioInit();*

*xTaskCreate(prvTask\_BTN\_SW,*

*(const char \*) "TASK\_BTN\_SW",*

*configMINIMAL\_STACK\_SIZE,*

*NULL,*

*tskIDLE\_PRIORITY + 4,*

*&Task\_BTN\_SW);*

*xtimer1 = xTimerCreate((const char \*)"Timer1",*

*four,*

*pdFALSE,*

*(void \*) TIMER\_ID,*

*vTimer1Callback);*

*if(xtimer1!=NULL){*

*xTimerStart(xtimer1,0);*

*vTaskStartScheduler();*

*}*

*while(1){};*

*}*

*// Task*

*static void prvTask\_BTN\_SW(void \*pvParameters){*

*while(1){*

*if(XGpio\_DiscreteRead(&xBTNSWI,BUTTONS)==1&&XGpio\_DiscreteRead(&xBTNSWI,SWITCHES)!=1){*

*xTimerStart(xtimer1,0UL);*

*XGpio\_DiscreteWrite(&xLEDS,1,1);*

*}*

*else if(XGpio\_DiscreteRead(&xBTNSWI,BUTTONS)==2&&XGpio\_DiscreteRead(&xBTNSWI,SWITCHES)!=2){*

*xTimerStart(xtimer1,0UL);*

*XGpio\_DiscreteWrite(&xLEDS,1,2);*

*}*

*else if(XGpio\_DiscreteRead(&xBTNSWI,BUTTONS)==4&&XGpio\_DiscreteRead(&xBTNSWI,SWITCHES)!=4){*

*xTimerStart(xtimer1,0UL);*

*XGpio\_DiscreteWrite(&xLEDS,1,4);*

*}*

*else if(XGpio\_DiscreteRead(&xBTNSWI,BUTTONS)==8&&XGpio\_DiscreteRead(&xBTNSWI,SWITCHES)!=8){*

*xTimerStart(xtimer1,0UL);*

*XGpio\_DiscreteWrite(&xLEDS,1,8);*

*}*

*else if(XGpio\_DiscreteRead(&xBTNSWI,BUTTONS)==1&&XGpio\_DiscreteRead(&xBTNSWI,SWITCHES)==1){*

*xTimerStart(xtimer1,0UL);*

*XGpio\_DiscreteWrite(&xLEDS,1,14);*

*}*

*else if(XGpio\_DiscreteRead(&xBTNSWI,BUTTONS)==2&&XGpio\_DiscreteRead(&xBTNSWI,SWITCHES)==2){*

*xTimerStart(xtimer1,0UL);*

*XGpio\_DiscreteWrite(&xLEDS,1,13);*

*}*

*else if(XGpio\_DiscreteRead(&xBTNSWI,BUTTONS)==4&&XGpio\_DiscreteRead(&xBTNSWI,SWITCHES)==4){*

*xTimerStart(xtimer1,0UL);*

*XGpio\_DiscreteWrite(&xLEDS,1,11);*

*}*

*else if(XGpio\_DiscreteRead(&xBTNSWI,BUTTONS)==8&&XGpio\_DiscreteRead(&xBTNSWI,SWITCHES)==8){*

*xTimerStart(xtimer1,0UL);*

*XGpio\_DiscreteWrite(&xLEDS,1,7);*

*}*

*}*

*}*

*// Timer*

*static void vTimer1Callback(xTimerHandle pxTimer){*

*XGpio\_DiscreteWrite(&xLEDS,1,0);*

*}*

*// Initialize GPIO*

*static void gpioInit(){*

*XGpio\_Initialize(&xBTNSWI,XPAR\_AXI\_GPIO\_0\_DEVICE\_ID);*

*XGpio\_Initialize(&xLEDS,XPAR\_AXI\_GPIO\_1\_DEVICE\_ID);*

*XGpio\_SetDataDirection(&xBTNSWI,1,0xf);*

*XGpio\_SetDataDirection(&xBTNSWI,2,0xf);*

*XGpio\_SetDataDirection(&xLEDS,1,0x0);*

*}*