UCSD Extension

Embedded Controller Programming for Real-Time Systems

Course Number: ECE-40097  
Section ID: 145032

Final Project

Date: 3/19/2020

Chris Isabelle

christopher.j.isabelle@gmail.com

SID: U01136665

Contents

[Project Requirements 3](#_Toc35524459)

[Summary 4](#_Toc35524460)

[Test Report 5](#_Toc35524461)

[Code 7](#_Toc35524462)

[UART 7](#_Toc35524463)

[myDelay1 7](#_Toc35524464)

[myDelay2 8](#_Toc35524465)

[IRQ Handlers & Callback functions 8](#_Toc35524466)

[Global flags 8](#_Toc35524467)

[Functions 8](#_Toc35524468)

[Event Flag processing 9](#_Toc35524469)

[Menu char Processing 10](#_Toc35524470)

# Project Requirements

|  |  |  |
| --- | --- | --- |
| Requirement ID | Description | Compliance |
| **P** | **Final prerequisites** |  |
| P-1 | Use Cube32MX to generate code without UART interrupt. | ✓ |
| P-2 | Expand generated code to use UART in **Interrupt** mode. | ✓ |
| P-3 | Enable GPIO PIN 13 interrupt, used for blue user switch using handler EXTI15\_10\_IRQHandler() | ✓ |
| P-4 | Enable Timer 2 and Timer 3, Timer 6 interrupts | ✓ |
| P-5 | Enable both IWDG and WWDG watchdog timers | ✓ |
| P-6 | Enable RTC | ✓ |
| **R1** | **UART Interrupt, (1 point)** |  |
| R1-1 | Implement logMsg() method to display a character on the terminal using interrupt. | ✓ |
| R1-2 | Implement logGetMsg() method to receive character from terminal using interrupt. | ✓ |
| R1-3 | If ‘g’ is entered on the terminal, toggle the green LED after a 1 second delay. | ✓ |
| R1-4 | If ‘b’ is entered on the terminal, toggle the red LED after a 1 second delay. | ✓ |
| R1-5 | For any other character, print “unknown character received”. | ✓ |
| R1-6 | Comment out MX\_IWDG\_Init(), MX\_WWDG\_Init() & MX\_RTC\_Init() | ✓ |
| **R2** | **Create a software interrupt and use one of the non-used IRQ, (5 points)** |  |
| R2-1 | Use FMC\_IRQn = 48 //FMC Global Interrupt | ✓ |
| R2-2 | Enable FMC\_IRQa in MX\_GPIO\_Init() | ✓ |
| R2-3 | Create a menu item ‘s’ which will generate a software interrupt and print “SW Interrupt detected” | ✓ |
| R2-3a | When ‘s’ is selected enable STIR using FMC\_IRQn | ✓ |
| R2-3b | ISR callback function is called | ✓ |
| R2-3c | Use a global flag to let main menu display the required message | ✓ |
| **R3** | **Create the method myDelay1() using timer2 with input in msec, (5 points)** |  |
| R3-1 | Use myDelay1() as the delay method for 1 second delay requirement in R1-3 | ✓ |
| **R4** | **Create the method myDelay2() using SysTick with input in msec, (5 points)** |  |
| R4-1 | Use myDelay2() as the delay method for 1 second delay requirement in R1-4 | ✓ |
| **R5** | **Use Timer 3 to count events, (6 points)** |  |
| R5-1 | Program the timer in MX\_TIM3\_Init() so that it will expire every 1 second. | ✓ |
| R5-2 | Create a new menu item ‘t’ to start timer3 | ✓ |
| R5-3 | Implement HAL\_TIM\_PeriodElapsedCallback() to set a flag indicating that the 1 sec timer has elapsed. | ✓ |
| R5-4 | In the main loop count 10 events and stop timer3 after 10 iterations | ✓ |
| R5-5 | Print the log message, “Total counted timer3 events = %d\r\n” | ✓ |
| **R6** | **Test IWDG, (6 points)** |  |
| R6-1 | Program prescaler, window and reload value for a timeout of 500 msec. | ✓ |
| R6-2 | Pet the watchdog in main() | ✓ |
| R6-3 | Verify not watchdog event and no reset | ✓ |
| R6-4 | Create a new menu item ‘w’ that will add a 1sec delay and trip the watchdog. | ✓ |
| R6-2 | Verify that the device resets due to the watchdog timeout. | ✓ |
| **R7** | **Test MX\_RTC, (6 points)** |  |
| R7-1 | Set the alarm for hour 0, minute 1 | ✓ |
| R7-2 | Implement a callback to set the flag and detect the alarm. | ✓ |
| R7-3 | Display the message “RTC alarm A detected” | ✓ |
| R7-4 | Verify that message is printed 1 minute following power on or reset. | ✓ |

# Summary

All requirements have been met. Here’s a list of a few clarifications I made in order to get things to work together and for validation.

1. I used STM32CubeMX to initialize the RTC time, data and alarm data structures.
2. myDelay1 & myDelay2 are IWDG safe. HAL\_IWDG\_Refresh(&hiwdg) is called in the msec count down loop.
3. Processed flags associated with TIM3, SW interrupt and RTC in an inner loop while waiting for a char to be input to the screen. This inner loop is also IWDG safe.
4. Requirement R5, timer3 is re-entrant.
5. Provided a teraterm capture file with timestamps to validate event timing. The original teraterm capture log <finalValidationTest.log> is contained in the project .zip file.
6. The test report show timing validation using the teraterm timestamps. The RTC time error of .892 seconds is greater than expected.

# Test Report

[Thu Mar 19 14:56:49.686 2020] **<<- Start of RTC Alarm A test**

[Thu Mar 19 14:56:49.686 2020]

[Thu Mar 19 14:56:49.686 2020]

[Thu Mar 19 14:56:49.686 2020]

[Thu Mar 19 14:56:49.686 2020] Welcome to <Embedded Controller Programming for Real-Time Systems>

[Thu Mar 19 14:56:49.693 2020] ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

[Thu Mar 19 14:56:49.698 2020] ~ Assignment: Final Project ~

[Thu Mar 19 14:56:49.706 2020] ~ Course Number: ECE-40097 ~

[Thu Mar 19 14:56:49.712 2020] ~ Section ID: 145032 ~

[Thu Mar 19 14:56:49.717 2020] ~ Student Name: Chris Isabelle ~

[Thu Mar 19 14:56:49.723 2020] ~ SID: U01136665 ~

[Thu Mar 19 14:56:49.728 2020] ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

[Thu Mar 19 14:56:49.735 2020]

[Thu Mar 19 14:56:49.735 2020]

[Thu Mar 19 14:56:49.735 2020]

[Thu Mar 19 14:56:49.735 2020] Main Menu

[Thu Mar 19 14:56:49.738 2020] ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

[Thu Mar 19 14:56:49.742 2020] Enter g - Toggle the Green LED after 1 one second delay

[Thu Mar 19 14:56:49.747 2020] Enter b - Toggle the Blue LED after 1 one second delay

[Thu Mar 19 14:56:49.751 2020] Enter s - Generate a SW interrupt

[Thu Mar 19 14:56:49.754 2020] Enter t - Start timer3

[Thu Mar 19 14:56:49.758 2020] Enter w - Enable a 1 second delay to trigger the WDT

[Thu Mar 19 14:56:52.085 2020]

[Thu Mar 19 14:56:52.085 2020]

[Thu Mar 19 14:56:52.085 2020] received character = g

[Thu Mar 19 14:56:53.087 2020] Toggle Green LED **<<- myDelay1() 1 second delay validated**

[Thu Mar 19 14:56:53.089 2020] **timestamp diff = 53.087-52.085=1.002 seconds**

[Thu Mar 19 14:56:53.089 2020]

[Thu Mar 19 14:56:53.089 2020]

[Thu Mar 19 14:56:53.089 2020] Main Menu

[Thu Mar 19 14:56:53.093 2020] ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

[Thu Mar 19 14:56:53.096 2020] Enter g - Toggle the Green LED after 1 one second delay

[Thu Mar 19 14:56:53.101 2020] Enter b - Toggle the Blue LED after 1 one second delay

[Thu Mar 19 14:56:53.106 2020] Enter s - Generate a SW interrupt

[Thu Mar 19 14:56:53.110 2020] Enter t - Start timer3

[Thu Mar 19 14:56:53.113 2020] Enter w - Enable a 1 second delay to trigger the WDT

[Thu Mar 19 14:56:54.277 2020]

[Thu Mar 19 14:56:54.277 2020]

[Thu Mar 19 14:56:54.277 2020] received character = b **<<- myDelay2() 2 second delay validated**

[Thu Mar 19 14:56:55.280 2020] Toggle Blue LED **timestamp diff = 55.280-52.277=1.003 seconds**

[Thu Mar 19 14:56:55.282 2020]

[Thu Mar 19 14:56:55.282 2020]

[Thu Mar 19 14:56:55.282 2020]

[Thu Mar 19 14:56:55.282 2020] Main Menu

[Thu Mar 19 14:56:55.284 2020] ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

[Thu Mar 19 14:56:55.289 2020] Enter g - Toggle the Green LED after 1 one second delay

[Thu Mar 19 14:56:55.293 2020] Enter b - Toggle the Blue LED after 1 one second delay

[Thu Mar 19 14:56:55.298 2020] Enter s - Generate a SW interrupt

[Thu Mar 19 14:56:55.302 2020] Enter t - Start timer3

[Thu Mar 19 14:56:55.304 2020] Enter w - Enable a 1 second delay to trigger the WDT

[Thu Mar 19 14:56:56.533 2020]

[Thu Mar 19 14:56:56.533 2020]

[Thu Mar 19 14:56:56.533 2020] received character = s

[Thu Mar 19 14:56:56.536 2020]

[Thu Mar 19 14:56:56.536 2020]

[Thu Mar 19 14:56:56.536 2020]

[Thu Mar 19 14:56:56.536 2020] Main Menu

[Thu Mar 19 14:56:56.540 2020] ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

[Thu Mar 19 14:56:56.543 2020] Enter g - Toggle the Green LED after 1 one second delay

[Thu Mar 19 14:56:56.548 2020] Enter b - Toggle the Blue LED after 1 one second delay

[Thu Mar 19 14:56:56.553 2020] Enter s - Generate a SW interrupt

[Thu Mar 19 14:56:56.556 2020] Enter t - Start timer3

[Thu Mar 19 14:56:56.558 2020] Enter w - Enable a 1 second delay to trigger the WDT

[Thu Mar 19 14:56:56.563 2020]

[Thu Mar 19 14:56:56.563 2020] >>> SW Interrupt Detected <<< **<<- SW Interrupt using FMC\_IRQn validated**

[Thu Mar 19 14:56:56.566 2020]

[Thu Mar 19 14:57:01.581 2020]

[Thu Mar 19 14:57:01.582 2020]

[Thu Mar 19 14:57:01.582 2020] received character = t **<<- Start of Timer 3 test**

[Thu Mar 19 14:57:01.584 2020]

[Thu Mar 19 14:57:01.584 2020] Starting Timer 3 Test

[Thu Mar 19 14:57:01.587 2020]

[Thu Mar 19 14:57:01.587 2020]

[Thu Mar 19 14:57:01.587 2020]

[Thu Mar 19 14:57:01.587 2020] Main Menu

[Thu Mar 19 14:57:01.590 2020] ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

[Thu Mar 19 14:57:01.595 2020] Enter g - Toggle the Green LED after 1 one second delay

[Thu Mar 19 14:57:01.599 2020] Enter b - Toggle the Blue LED after 1 one second delay

[Thu Mar 19 14:57:01.604 2020] Enter s - Generate a SW interrupt

[Thu Mar 19 14:57:01.609 2020] Enter t - Start timer3

[Thu Mar 19 14:57:01.611 2020] Enter w - Enable a 1 second delay to trigger the WDT

[Thu Mar 19 14:57:02.586 2020] Total timer3 events counted = 1 **<<- Timer 3 one second interval validated**

[Thu Mar 19 14:57:03.587 2020] Total timer3 events counted = 2 **timestamp diff = 2.586-1.582=1.004 seconds**

[Thu Mar 19 14:57:04.588 2020] Total timer3 events counted = 3

[Thu Mar 19 14:57:05.587 2020] Total timer3 events counted = 4

[Thu Mar 19 14:57:06.589 2020] Total timer3 events counted = 5

[Thu Mar 19 14:57:07.589 2020] Total timer3 events counted = 6

[Thu Mar 19 14:57:08.589 2020] Total timer3 events counted = 7

[Thu Mar 19 14:57:09.589 2020] Total timer3 events counted = 8

[Thu Mar 19 14:57:10.591 2020] Total timer3 events counted = 9

[Thu Mar 19 14:57:11.591 2020] Total timer3 events counted = 10 **<<- Timer 3 10 second count validated**

[Thu Mar 19 14:57:11.597 2020] **timestamp diff = 11.591-1.582=10.009 seconds**

[Thu Mar 19 14:57:11.597 2020] Timer 3 Test Complete

[Thu Mar 19 14:57:50.578 2020]

[Thu Mar 19 14:57:50.578 2020] >>> RTC alarm A detected <<< **<<- RTC alarm A validated**

[Thu Mar 19 14:57:50.581 2020] **timestamp diff = 57:50.578-56:49.686=**

[Thu Mar 19 14:57:51.806 2020] **60.892 seconds**

[Thu Mar 19 14:57:51.806 2020]

[Thu Mar 19 14:57:51.806 2020] received character = w

[Thu Mar 19 14:57:51.808 2020]

[Thu Mar 19 14:57:51.808 2020] Starting IWDG Test

[Thu Mar 19 14:57:52.376 2020] **<<- IWDG watch dog 500msec timeout validated**

[Thu Mar 19 14:57:52.376 2020] **timestamp diff = 52.376-51.808=0.574 seconds**

[Thu Mar 19 14:57:52.376 2020]

[Thu Mar 19 14:57:52.376 2020]

[Thu Mar 19 14:57:52.376 2020] Welcome to <Embedded Controller Programming for Real-Time Systems>

[Thu Mar 19 14:57:52.382 2020] ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

[Thu Mar 19 14:57:52.388 2020] ~ Assignment: Final Project ~

[Thu Mar 19 14:57:52.394 2020] ~ Course Number: ECE-40097 ~

[Thu Mar 19 14:57:52.400 2020] ~ Section ID: 145032 ~

[Thu Mar 19 14:57:52.409 2020] ~ Student Name: Chris Isabelle ~

[Thu Mar 19 14:57:52.413 2020] ~ SID: U01136665 ~

[Thu Mar 19 14:57:52.418 2020] ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

[Thu Mar 19 14:57:52.424 2020]

[Thu Mar 19 14:57:52.424 2020]

[Thu Mar 19 14:57:52.426 2020]

[Thu Mar 19 14:57:52.426 2020] Main Menu

[Thu Mar 19 14:57:52.428 2020] ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

[Thu Mar 19 14:57:52.436 2020] Enter g - Toggle the Green LED after 1 one second delay

[Thu Mar 19 14:57:52.439 2020] Enter b - Toggle the Blue LED after 1 one second delay

[Thu Mar 19 14:57:52.442 2020] Enter s - Generate a SW interrupt

[Thu Mar 19 14:57:52.445 2020] Enter t - Start timer3

[Thu Mar 19 14:57:52.448 2020] Enter w - Enable a 1 second delay to trigger the WDT

# Code

## UART

// generic USART handler

**void** **USART1\_IRQHandler**(**void**)

{

HAL\_UART\_IRQHandler(&huart1);

}

// logMsg - prints messages to the UART

**void** **logMsg**(UART\_HandleTypeDef \*huart, **char** \_out[])

{

//Use STM32L4xx\_HAL\_Driver for UART TX

uint32\_t numTxChar = **strlen**(\_out);

HAL\_UART\_Transmit\_IT(huart, (uint8\_t \*)\_out, numTxChar);

//need to stall and wait for all characters to be transmitted

**while**(!\_\_HAL\_UART\_GET\_FLAG(huart, UART\_FLAG\_TC));

}

// logGetMsg method

uint8\_t **logGetMsg**(UART\_HandleTypeDef \*huart)

{

uint8\_t rxChar;

HAL\_UART\_Receive\_IT(&huart1, &rxChar, **sizeof**(uint8\_t));

**return** (rxChar);

}

**void** **FMC\_SW\_IRQHandler**(**void**)

{

mySwInterruptFlag = 1;

}

## myDelay1

// myDelay1 method - IWDG safe

**void** **myDelay1**(uint32\_t val)

{

//Confirmed all bits in TIM2 are initialized to zero.

//Only bits required as set are handled in the code below

//TIM2 (CR) Control Register Setup

//set CR[4] DIR = 1 for downcounter

TIM2->CR1 |= 1<<4;

//set CR[0] CEN = 1 for Counter enable

TIM2->CR1 |= 1<<0;

TIM2->PSC = 1; //PSC\_CLK = 80MHz/(PSC+1) = 40MHz

TIM2->ARR = 40000;

TIM2->CCR1 = 1;

**while** (val != 0)

{

//wait for SR[0] UIF = 1 Update Interrupt Pending

**while**(!(TIM2-> SR & 1)) {}

//placing the IWDG refresh here will force a WDT event if the timer is not configured correctly

//pet the IWDG timer

HAL\_IWDG\_Refresh(&hiwdg);

val --;

//clear SR[0] UIF = 0 No update occured

TIM2->SR &= ~1;

}

}

## myDelay2

// myDelay2 method - IWDG safe

**void** **myDelay2**(uint32\_t val)

{

//SysTick\_LOAD = (SysTick\_Interrupt\_Period \* SysTick\_Counter\_Clock\_Frequency) - 1

SysTick->LOAD = (1E-3 \* SystemCoreClock) - 1;

**while** (val != 0)

{

//Added mySysTick\_IRQHandler() in SysTick\_Handler()

//mySysTick\_IRQHandler() sets swSysTickEventFlag each time SysTick\_Handler() is called

//wait until swSysTickEventFlag changes

**while**(!mySysTickEventFlag) {}

//reset swSysTickEventFlag

mySysTickEventFlag = 0;

//pet the IWDG timer

HAL\_IWDG\_Refresh(&hiwdg);

val --;

}

}

## IRQ Handlers & Callback functions

### Global flags

**int** mySwInterruptFlag = 0;

**int** mySysTickEventFlag = 0;

**int** myTim3EventFlag = 0;

**int** myRtcAlarmFlag = 0;

### Functions

**void** **FMC\_SW\_IRQHandler**(**void**)

{

mySwInterruptFlag = 1;

}

// called by SysTick\_Handler()

**void** **mySysTick\_IRQHandler**(**void**)

{

mySysTickEventFlag = 1;

}

**void** **HAL\_TIM\_PeriodElapsedCallback**(TIM\_HandleTypeDef \*htim)

{

**if**(htim == &htim3)

{

myTim3EventFlag = 1;

}

}

**void** **HAL\_RTC\_AlarmAEventCallback**(RTC\_HandleTypeDef \*hrtc)

{

myRtcAlarmFlag = 1;

}

## Event Flag processing

//stall the outerloop until a char is input to the terminal - IWDG safe

//process tim3, rtc & swInterrupt flags

**while**(\_\_HAL\_UART\_GET\_FLAG(&huart1, UART\_FLAG\_RXNE)==0)

{

//pet the IWDG timer

HAL\_IWDG\_Refresh(&hiwdg);

//test to see if the TIM3 interrupt flag has been triggered

**if**(myTim3EventFlag)

{

//clear the flag

myTim3EventFlag=0;

//throw away for first event

//enabling the interrupt causes an event that is not a timed event

//see test log for timing analysis.

**if**(tim3Cntr==0)

{

**sprintf**(tempStr, "Total timer3 enabled\r\n", tim3Cntr);

tim3Cntr++;

**continue**;

}

**sprintf**(tempStr, "Total timer3 events counted = %d\r\n", tim3Cntr);

//increment counter

tim3Cntr++;

logMsg(&huart1, tempStr);

**if**(tim3Cntr>10)

{

//Stop timer 3

HAL\_TIM\_Base\_Stop\_IT(&htim3);

//reset tim3Cntr

tim3Cntr = 0;

logMsg(&huart1, "\r\nTimer 3 Test Complete\r\n");

}

}

//test to see if the SW interrupt flag has been triggered

**if**(mySwInterruptFlag)

{

//print message

logMsg(&huart1, "\r\n>>> SW Interrupt Detected <<<\r\n\r\n");

//clear the flag

mySwInterruptFlag=0;

}

//test to see if the RTC interrupt flag has been triggered

**if**(myRtcAlarmFlag)

{

//print message

logMsg(&huart1, "\r\n>>> RTC alarm A detected <<<\r\n\r\n");

//clear the flag

myRtcAlarmFlag=0;

}

}

## Menu char Processing

//get the character that was input to the terminal

tempChar[0] = logGetMsg(&huart1);

logMsg(&huart1, "\n\r\nreceived character = ");

logMsg(&huart1, tempChar);

logMsg(&huart1, " \r\n");

//process character

**switch**(tempChar[0])

{

**case**('g'): //Toggle Green LED

myDelay1(1000);

HAL\_GPIO\_TogglePin(GRN\_LED\_GPIO\_Port, GRN\_LED\_Pin);

logMsg(&huart1, "Toggle Green LED \r\n");

**break**;

**case**('b'): //Toggle blue LED

myDelay2(1000);

HAL\_GPIO\_TogglePin(BLUE\_LED\_GPIO\_Port, BLUE\_LED\_Pin);

logMsg(&huart1, "Toggle Blue LED \r\n");

**break**;

**case**('s'): //Generate a SW interrupt

NVIC->STIR = *FMC\_IRQn*;

**break**;

**case**('t'): //Start timer 3

logMsg(&huart1, "\r\nStarting Timer 3 Test\r\n");

HAL\_TIM\_Base\_Start\_IT(&htim3);

**break**;

**case**('w'): //Enable 1 second delay to trigger the IWDG

logMsg(&huart1, "\r\nStarting IWDG Test\r\n");

HAL\_Delay(1000);

**break**;

**default**: logMsg(&huart1, "\n\r\nunknown character received \r\n");

}