**ECE - 4097 Embedded Controller programing for Real Time Systems**

**Final Exam - Total points 35**

**Name:**

**Here are some hints and pointers. I hope you will find it useful.**

1. Create a software interrupt and use one of the non-used IRQ – **5 points**
   1. You could any one but let’s use 48 for this assignment.

FMC\_IRQn = 48, /\*!< FMC global Interrupt

* 1. Enable the FMC\_IRQn in **MX\_GPIO\_Init()**
  2. Create another menu with character ‘s’, which will generate the software interrupt and you should print the message “ SW Interrupt detected”. Here are steps involved.
     1. When the key is pressed, you should enable STIR using FMC\_IRQn
     2. This will trigger the interrupt and callback method will be called
     3. Set a flag or do something else to print the message “ SW Interrupt detected”.

**Hints**:

* Like any interrupt, we need to first enable it  in main.c

// Enable FMC interrupt  
HAL\_NVIC\_SetPriority(FMC\_IRQn, 0, 0);  
HAL\_NVIC\_EnableIRQ(FMC\_IRQn);

* Now use STIR to generate the interrupt when 's' is pressed

// Trigger Forced SW interrupt  
void enableSWIntr(uint32\_t IRQn) {

//Use STR register to trigger the   
 }

-  When interrupt is generated, we need to go to ISR and hence create ISR handler in file stm32l4xx\_it.c file where others ISR's are.

/\* USER CODE BEGIN 1 \*/  
void FMC\_IRQHandler(void)  
{  
           FMC\_SW\_IRQHandler();  
}

- Add callback in  FMC\_SW\_IRQHandler()

- In callback, you log the desired message

1. Create a method myDelay1() using timer2 and should take input in MilliSec – **5 Points** 
   1. Replace the existing HAL\_Delay() with myDelay1() for Blue LED

// Here is the method signature and feel free to implement the way you want to

void myDelay1(uint32\_t val) {

}

**Hints:**

* One could do it in different ways and here is one way to do it
* Since the ask is to take input in milliseconds, I would recommend to have timer programmed for 1 millisecond delay. This makes user life easier.
* You could then run a loop with input value to create a desire delay.

void myDelay1(uint32\_t val) {

// If the val is 10 then this will delay for 10 ms.

while (val != 0) {

           // Program the TIM2 register for 1 millisec delay

// Wait for SR   
while (!(TIM2->SR & 1)) {  
  
}  
val--;

}  
TIM2->SR &= ~1;

}

1. Create a method myDelay2() using SysTick and should take input in Millisecond – **6 Points**
   1. Replace the existing HAL\_Delay() with myDelay2() for Green LED

// Here is the method signature and feel free to implement the way you want to

void myDelay2(uint32\_t val) {

}

**Hints:** Same as above but to program SysTick register

1. Use Timer 3 to count events – **6 points**
   1. Program the timer in **MX\_TIM3\_Init()** with values , which will expire every second.
   2. Create a menu with character ‘t’ to start the timer3
   3. You will need to implement HAL\_TIM\_PeriodElapsedCallback() method and set a flag here to know when 1 sec has lapsed.
   4. Count for 10 (equivalent to 10 sec) in main loop and stop the timer3 when count reaches 10.
   5. Print the log message - "Total counted timer3 event = %d\r\n"

**Hints:**

* + You could use HAL\_TIM\_Base\_Start\_IT() when ‘t’ is pressed.
  + Callback is HAL\_TIM\_PeriodElapsedCallback()

1. Uncomment MX\_IWDG\_Init() code to test the watchdog – **6 Points**
   1. Program the prescaler, window and reload value for timeout of 0.5 sec.
   2. Pet the watchdog in main () code – to avoid board reset
      1. Make sure you don’t have any delay in main() code.
   3. There should be no reset and software should be working properly.
   4. Now simulate a failure by introducing a delay more than 0.5 sec to miss the watchdog pet.
   5. Create a menu with character ‘w’ and delay of 1 sec. Since 1 sec is more that 0.5 sec, SW will miss the pet and unit will reset.

**Hints:**

- There are few ways to do this. I have added example to calculate min and max delay values.

- You could calculate to get 0.5 sec delay.

- You could use HAL\_IWDG\_Refresh() to referesh watchdog.

\* USER CODE BEGIN IWDG\_Init 0 \*/

/\*

Watchdog freq. is 32 kHz

Prescaler: Min\_Value = 4 and Max\_Value = 256

Reload: Min\_Data = 0 and Max\_Data = 0x0FFF

TimeOut in seconds = (Reload \* Prescaler) / Freq.

MinTimeOut = (4 \* 1) / 32000 = 0.000125 seconds (125 microseconds)

MaxTimeOut = (256 \* 4096) / 32000 = 32.768 seconds

1. Uncomment MX\_RTC\_Init() to test RTC Alarm – **6 Points**
   1. Set the alarm for hour 0, minute 1
   2. Implement callback and set the flag to detect the alarm
   3. 1 minute after start of the board , alarm should be set.
   4. Display a message “ RTC alarm A detected”

**Hints:**

* Callback is HAL\_RTC\_AlarmAEventCallback()