Dylan Dang
026158052

CECS 346 Lab 4 – SysTick Timer

By Dr. Min He

Preparation: You will need a LaunchPad, two push buttons or switches, two 10kΩ resistors, four color LEDs:red, yellow, green, and white, and four resistors for the LEDs (between 330Ω to 1kΩ).

Book Reading: Textbook Sections 4.4

Starter project: A working CECS346Lab 3

Reference Project: SysTick

Purpose:
The purpose of this lab is to learn how to use SysTick timer to generate accurate timing for an embedded system, and learn how to use logic analyzer in Keil uVision simulator.

System Requirements:
In this lab you will implement the same features as Lab 3. Instead of using software loop to generate 0.5s delay, you will use SysTick timer busy waiting approach to generate the same time delay.

Hardware requirements:

1) Port E will be used to control 4 LEDs: white(PE3), red (PE2), yellow (PE1), green (PE0).
2) Port A will be used for the two switches: sw1 (PA2), sw2 (PA3).

Software requirements:

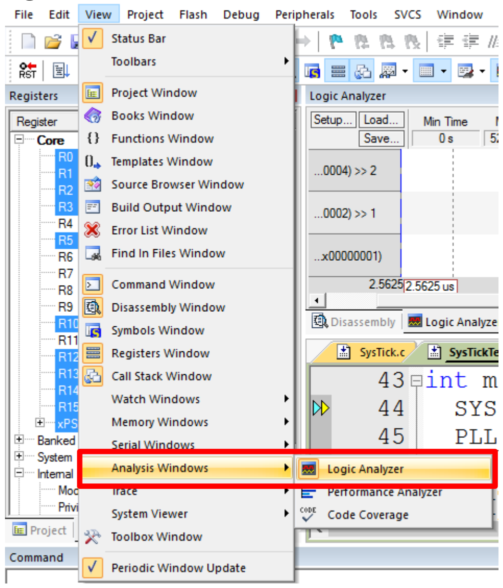
1. Make a copy of your Lab 3 project and rename it to CECS346Lab4, save CECS346Lab3.c as CECS346Lab4.c, Remove CECS346Lab3.c. Add Lab3.c to Source folder. Clean unused code. Please refer to NewProjectBasedOnOld.mp4 for a live demo.
2. Study the example project SysTick: understand how to initialize SysTick timer and how to generate a specified time delay using busy waiting approach. #include <stdint.h> for data type alias. Write two functions to your CECS346Lab4.c with specified function prototypes shown below:
   ```
   // initialize the systick timer with maximum reload value,
   // enable SysTick timer with system clock.
   void SysTick_init(void);

   // Use busy waiting approach to generate n*0.5s delay,
   // n is specified by the parameter of this function.
   void Wait_HalfSecond(uint8_t char n);
   ```
3. Remove Delay() function and function calls to Delay(). Replace Delay() function call with a call to Wait_HalfSecond(n).

Simulation and Demonstration:

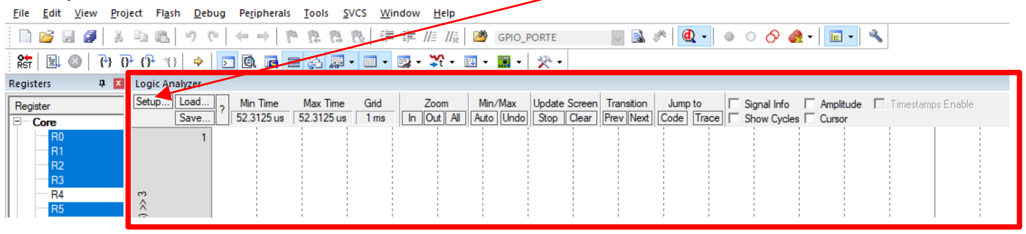
Test cases: please follow the instructions given below to test your embedded system in keil uVision simulator and on board:

1) Start with green light on for a minimum of 2s
2) Press sw2 to observe the following light changes: green(2s)->yellow(1s)->red/white(blue).
3) Press sw1 to observe the following light changes: white(blue) off 0.5s, then on 0.5s, then off and green on.
4) Start with green light on, press both sw1 and sw2 to observe the following light sequence for at least once: green(2s)->yellow(1s)->red/white(blue)(2s)->white(blue) off 0.5s, then 0.5 on, then off and green on.

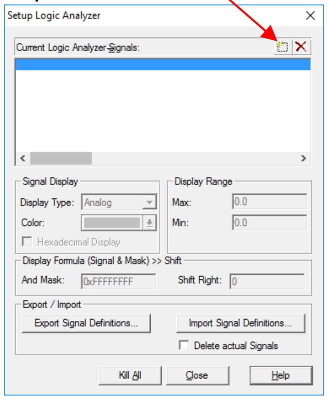
1. **Simulation**: Compile and simulate it with edXLab10 DLL(-dedXLab10 at the debug tab on options window) and Logic Analyzer. Screenshot all four cases listed above.
   <u>How to setup the logic analyzer:</u>
   First, set up the logic analyzer to observe bits PE0, PE1, PE2, and PE3. After starting debug session, click view → Analysis Windows → Logic Analyzer to open the Logic Analyzer. See figure below.



   On the left top of the Logic Analyzer window Click **Setup** button to bring up “Setup Logic Analyzer” window.

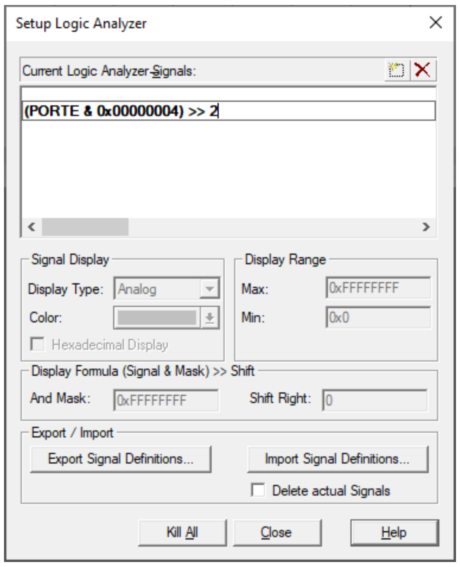


Then click **New(insert)** button on the "Setup Logic Analyzer" window to add signals to be captured.

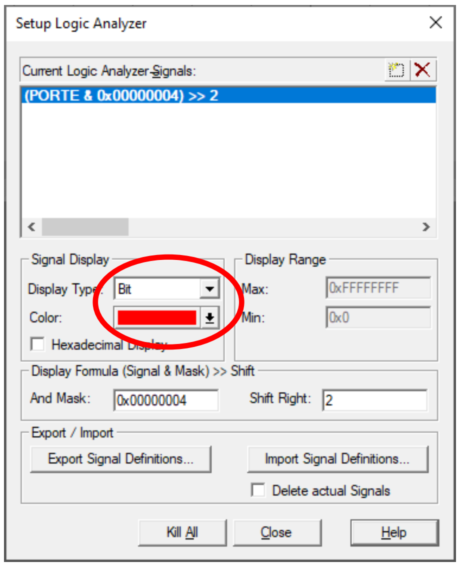

Setup Logic Analyzer
Current Logic Analyzer Signals:
Signal Display
Display Range
Hexadecimal Display
Display Formula (Signal & Mask) >> Shift
Export / Import
Export Signal Definitions...
Import Signal Definitions...
Delete actual Signals
Kill All
Close
Help

Display Type: Analog
Color:
Max: 0.0
Min: 0.0
And Mask: 0xFFFFFFFF
Shift Right: 0

We need to capture four signals: PE3, PE2, PE1, and PE0. To capture PE2(Red LED) output, enter **(PORTE & 0x00000004) >> 2** in the "setup Logic Analyzer" window and hit **Enter** when done. Then select the setting just entered and make sure Display Type is "Bit" and pick a color the matches the LED color. Close the window to finish current signal setup. Repeat the process to setup for PE1(Yellow) and PE0(Green). You can use Blue for the PE3.


Setup Logic Analyzer
Current Logic Analyzer Signals:
Signal Display
Display Range
Hexadecimal Display
Display Formula (Signal & Mask) >> Shift
Export / Import
Export Signal Definitions...
Import Signal Definitions...
Delete actual Signals
Kill All
Close
Help

(PORTE & 0x00000004) >> 2
Display Type: Analog
Color:
Max: 0xFFFFFFFF
Min: 0x0
And Mask: 0xFFFFFFFF
Shift Right: 0


Setup Logic Analyzer
Current Logic Analyzer Signals:
Signal Display
Display Range
Hexadecimal Display
Display Formula (Signal & Mask) >> Shift
Export / Import
Export Signal Definitions...
Import Signal Definitions...
Delete actual Signals
Kill All
Close
Help

(PORTE & 0x00000004) >> 2
Display Type: Bit
Color: 🟥
Max: 0xFFFFFFFF
Min: 0x0
And Mask: 0x00000004
Shift Right: 2

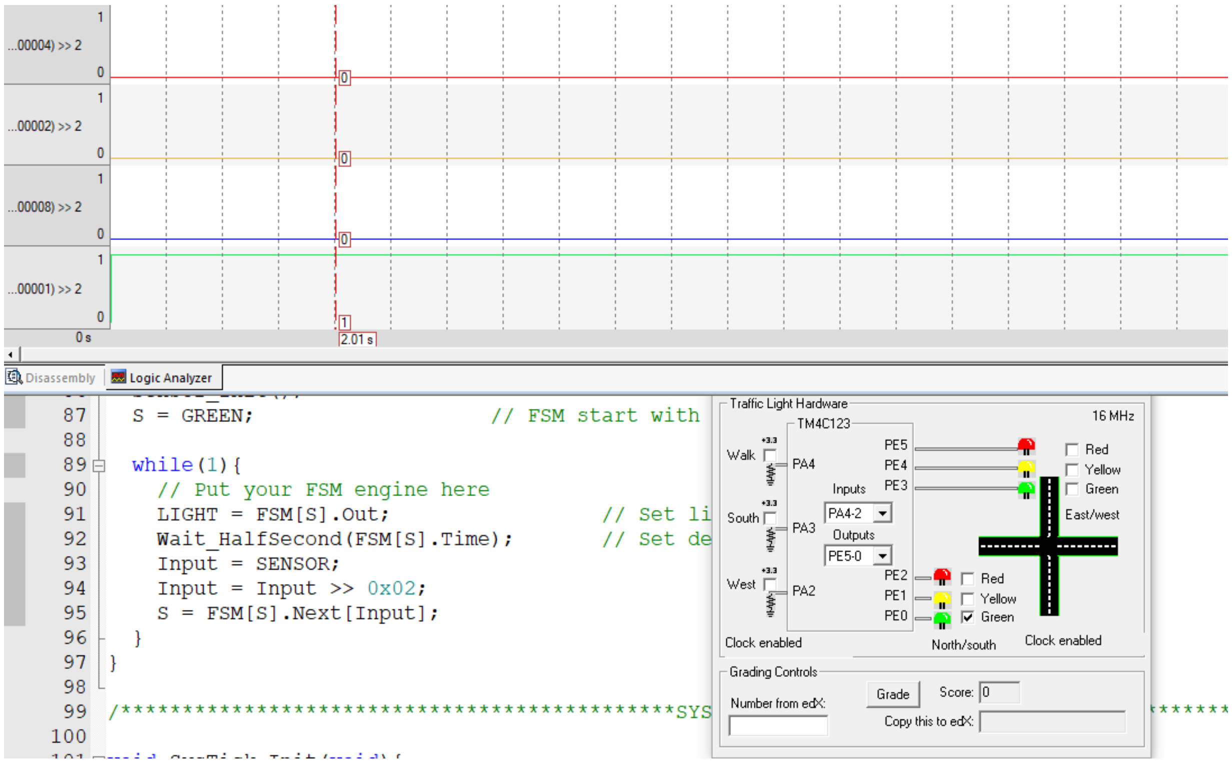
2. Download your program to Launchpad, test and demonstrate all four cases on board.

Deliverable:

Submit the following two files:

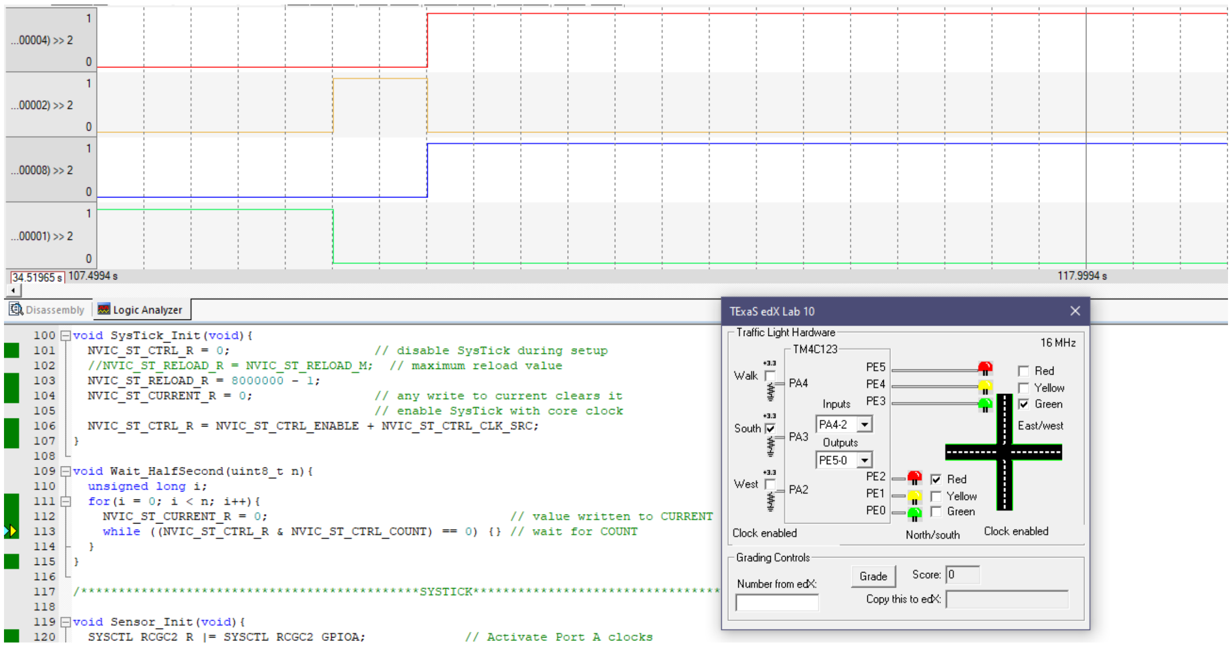
1. A lab report in PDF or Word format: attach required screen shots of logic analyzer outputs to end of this document.
2. Source code: CECS346Lab4.c.

start GREEN on (2s) :



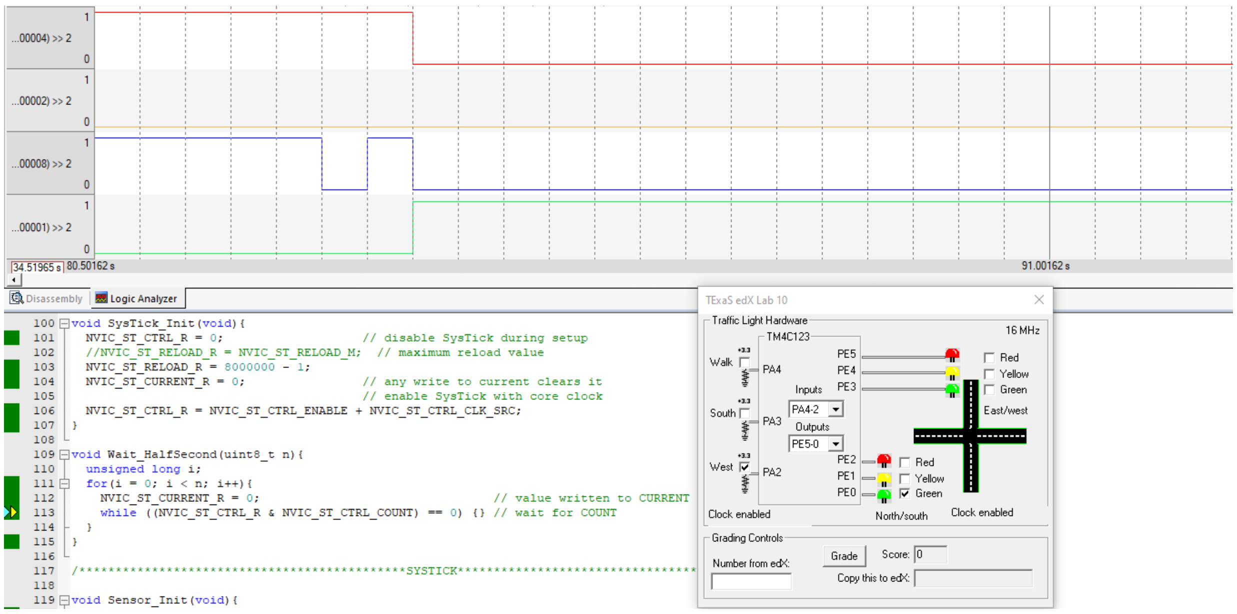
```
  S = GREEN;                        // FSM start with

  while(1){
    // Put your FSM engine here
    LIGHT = FSM[S].Out;                    // Set li
    Wait_HalfSecond(FSM[S].Time);          // Set de
    Input = SENSOR;
    Input = Input >> 0x02;
    S = FSM[S].Next[Input];
  }
}

/********************************************SYS
```

SW2(PA3) pressed: GREEN (2s) -> YELLOW (1s) -> RED/WHITE(BLUE):



```
void SysTick_Init(void){
  NVIC_ST_CTRL_R = 0;                    // disable SysTick during setup
  //NVIC_ST_RELOAD_R = NVIC_ST_RELOAD_M;  // maximum reload value
  NVIC_ST_RELOAD_R = 8000000 - 1;
  NVIC_ST_CURRENT_R = 0;                 // any write to current clears it
                                         // enable SysTick with core clock
  NVIC_ST_CTRL_R = NVIC_ST_CTRL_ENABLE + NVIC_ST_CTRL_CLK_SRC;
}

void Wait_HalfSecond(uint8_t n){
  unsigned long i;
  for(i = 0; i < n; i++){
    NVIC_ST_CURRENT_R = 0;                             // value written to CURRENT
    while ((NVIC_ST_CTRL_R & NVIC_ST_CTRL_COUNT) == 0) {} // wait for COUNT
  }
}

/*********************************************SYSTICK*********************************

void Sensor_Init(void){
  SYSCTL_RCGC2_R |= SYSCTL_RCGC2_GPIOA;           // Activate Port A clocks
```

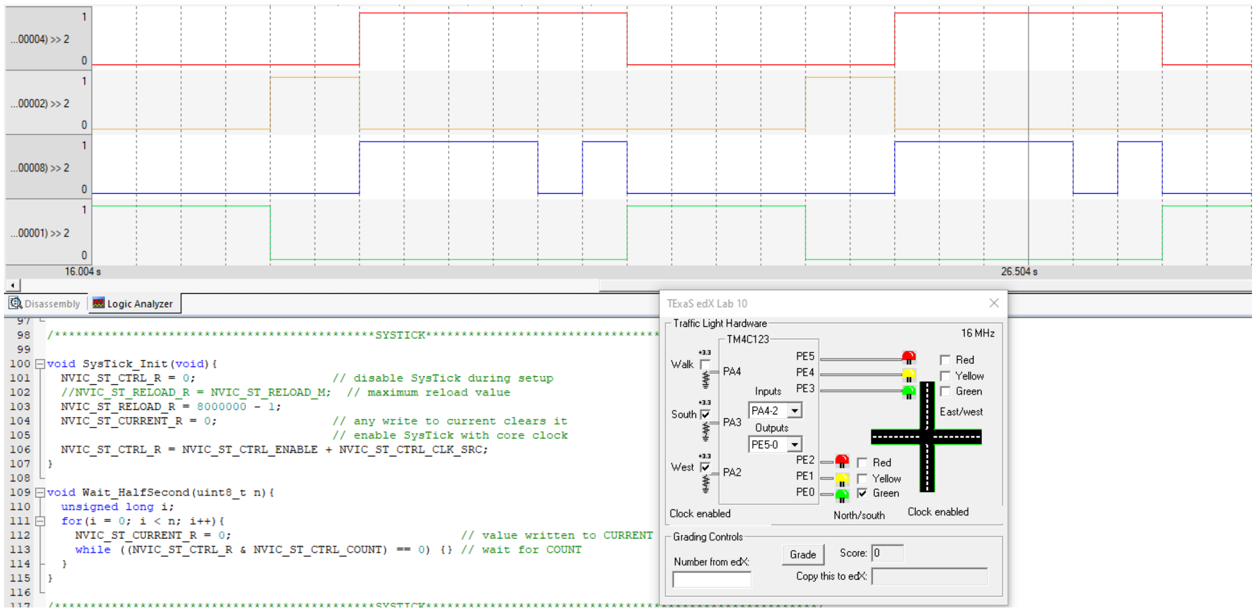
SW1(PA2) pressed: WHITE(BLUE) off (0.5s) -> WHITE(BLUE) on (0.5s) -> WHITE(BLUE) off -> GREEN on:



```
void SysTick_Init(void){
  NVIC_ST_CTRL_R = 0;                    // disable SysTick during setup
  //NVIC_ST_RELOAD_R = NVIC_ST_RELOAD_M;  // maximum reload value
  NVIC_ST_RELOAD_R = 8000000 - 1;
  NVIC_ST_CURRENT_R = 0;                 // any write to current clears it
                                         // enable SysTick with core clock
  NVIC_ST_CTRL_R = NVIC_ST_CTRL_ENABLE + NVIC_ST_CTRL_CLK_SRC;
}

void Wait_HalfSecond(uint8_t n){
  unsigned long i;
  for(i = 0; i < n; i++){
    NVIC_ST_CURRENT_R = 0;                                  // value written to CURRENT
    while ((NVIC_ST_CTRL_R & NVIC_ST_CTRL_COUNT) == 0) {} // wait for COUNT
  }
}

/*****************************************SYSTICK*******************************

void Sensor_Init(void){
```

Walk [ ] South [ ] West [✓]
Inputs: PA4-2
Outputs: PE5-0
East/west: Red [ ] Yellow [ ] Green [ ]
North/south: Red [ ] Yellow [ ] Green [✓]
Score: 0

start GREEN on, SW1(PA2) pressed & SW2(PA3) pressed: GREEN(2s) -> YELLOW(1s) -> RED/WHITE(BLUE)(2s) -> WHITE(BLUE) off (0.5s) -> WHITE(BLUE) on (0.5s), WHITE(BLUE) off -> GREEN on:



```
/*****************************************SYSTICK*******************************

void SysTick_Init(void){
  NVIC_ST_CTRL_R = 0;                    // disable SysTick during setup
  //NVIC_ST_RELOAD_R = NVIC_ST_RELOAD_M;  // maximum reload value
  NVIC_ST_RELOAD_R = 8000000 - 1;
  NVIC_ST_CURRENT_R = 0;                 // any write to current clears it
                                         // enable SysTick with core clock
  NVIC_ST_CTRL_R = NVIC_ST_CTRL_ENABLE + NVIC_ST_CTRL_CLK_SRC;
}

void Wait_HalfSecond(uint8_t n){
  unsigned long i;
  for(i = 0; i < n; i++){
    NVIC_ST_CURRENT_R = 0;                                  // value written to CURRENT
    while ((NVIC_ST_CTRL_R & NVIC_ST_CTRL_COUNT) == 0) {} // wait for COUNT
  }
}

/*****************************************SYSTICK*******************************
```

Walk [ ] South [✓] West [✓]
Inputs: PA4-2
Outputs: PE5-0
East/west: Red [ ] Yellow [ ] Green [ ]
North/south: Red [ ] Yellow [ ] Green [✓]
Score: 0