LAB: EXTI & SysTick

Date: 2023-09-26

Author: NohYunKi

Demo Video:

- **EXTI:** https://youtube.com/shorts/5mQDUwYdhdg?feature=share
- **SysTick:**https://youtube.com/shorts/lAM8qyBFHq4?feature=share

Introduction

In this lab, each numbers(0~9) will be counted on the 7-Segment display by using EXTI(External Interrupt) and SysTick(System Tick) . In the previous lab, counting number was performed in the infinite loop(polling method). However in this lab, this work will be performed when specific external interrupt arise.(interrupt method). To achieve this goal, EXTI and SysTick will be used.

Requirement

Hardware

- MCU
  - NUCLEO-F411RE
- Actuator/Sensor/Others:
  - 7-segment display(5101ASR)
  - Array resistor (330 ohm)
  - decoder chip(74LS47)
  - breadboard

Software

- Keil uVision, CMSIS, EC_HAL library

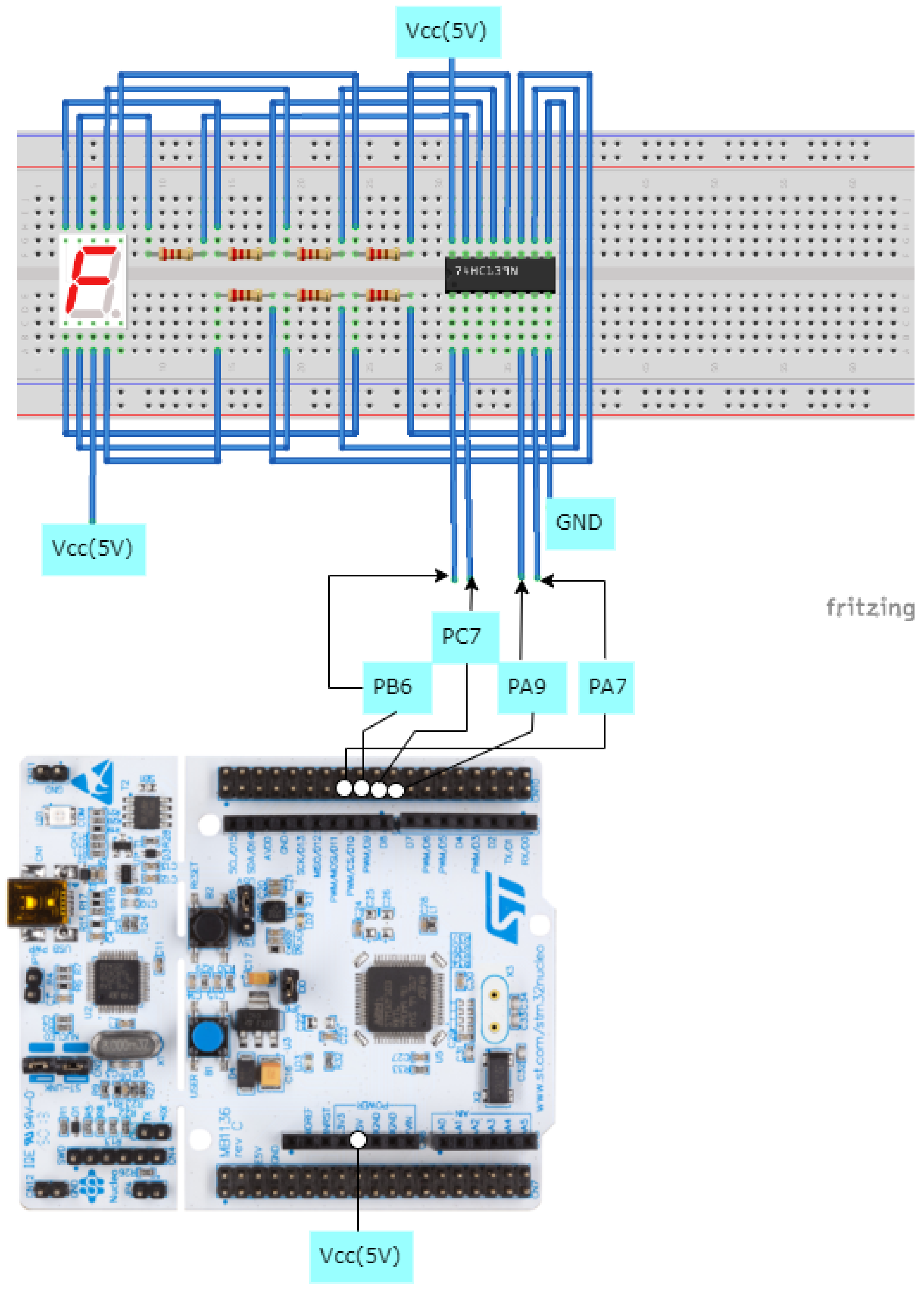
Problem 1: Counting numbers on 7-Segment using EXTI Button

Configuration

Digital In for Button (B1)	Digital Out for 7-Segment decoder
Digital In	Digital Out
PC13	PA7, PB6, PC7, PA9

Digital In for Button (B1)	Digital Out for 7-Segment decoder
PULL-UP	Push-Pull, No Pull-up-Pull-down, Medium Speed

Circuit Diagram



Discussion

1. We can use two different methods to detect an external signal: polling and interrupt. What are the advantages and disadvantages of each approach?

- Answer: While using polling to get external signal, computer constantly focuses on whether specific external interrupt arise or not. Therefore, even though when any external interrupt is not occurred, computer uses some parts of it's memory to check whether external interrupt is occured or not. However, if computer uses external interrupt function, computer can detect external interrupt's occurrence right away. Because computer don't focus on to check the occurrence of external interrupt constantly , computer can save it's memory.

2. What would happen if the EXTI interrupt handler does not clear the interrupt pending flag? Check with your code

- Answer: 'Pending' signal means permission sign of designated function's performance. In this reason, when interrupt pending flag is not cleared after finishing it's performance, this execution will be performed repeatedly. In reality, when not clearing pending state in this lab's code, numbers are counted constantly after pressing once time.

code

- main

```
/**
******************************************************************************
* @author  NohYunKi 21800226
* @content   Embedded Controller_EXTI LAB
*
******************************************************************************
*/

#include "stm32f4xx.h"
#include "ecGPIO.h"
#include "ecRCC.h"
#include "ecEXTI.h"




#define LED_PIN    5
#define BUTTON_PIN 13

void setup(void);
void EXTI15_10_IRQHandler(void);


unsigned int cnt = 0;


int main(void) {

   // System CLOCK, GPIO Initialiization ----------------------------------------
   setup();


   // EXTI Initialiization ------------------------------------------------------

   EXTI_init(GPIOC, 13, FALL, 0);



   while (1);
}


void EXTI15_10_IRQHandler(void) {
   if (is_pending_EXTI(BUTTON_PIN)) {
      cnt++;
          if (cnt > 9) cnt = 0;
        sevensegment_display(cnt % 10);
          for(volatile int i = 0; i < 250000;i++){}

      clear_pending_EXTI(BUTTON_PIN); // cleared by writing '1'
   }
}

// Initialiization
void setup(void)
{
   RCC_PLL_init();                         // System Clock = 84MHz
   // Initialize GPIOA_5 for Output
   GPIO_init(GPIOA, LED_PIN, OUTPUT);    // calls RCC_GPIOA_enable()
   // Initialize GPIOC_13 for Input Button
   GPIO_init(GPIOC, BUTTON_PIN, INPUT);  // calls RCC_GPIOC_enable()
     // Initialize sevensegment
     sevensegment_display_init();
}
```

- EXTI_init

```
void EXTI_init(GPIO_TypeDef *Port, int Pin, int trig_type,int priority){

   // SYSCFG peripheral clock enable
   RCC->APB2ENR |= RCC_APB2ENR_SYSCFGEN;

   // Connect External Line to the GPIO
   int EXTICR_port;
   if         (Port == GPIOA) EXTICR_port = 0;
   else if   (Port == GPIOB) EXTICR_port = 1;
   else if   (Port == GPIOC) EXTICR_port = 2;
   else if   (Port == GPIOD) EXTICR_port = 3;
   else                               EXTICR_port = 4;

   SYSCFG->EXTICR[Pin/4] &= ~(15UL<<4*(Pin%4));         // clear 4 bits
   SYSCFG->EXTICR[Pin/4] |= EXTICR_port<<4*(Pin%4);         // set connect with
proposed Port

   // Configure Trigger edge
   if (trig_type == FALL) EXTI->FTSR |= 1UL<<Pin;   // Falling trigger enable
   else if   (trig_type == RISE) EXTI->RTSR |= 1UL<<Pin;   // Rising trigger
enable
```

```
    else if   (trig_type == BOTH) {          // Both falling/rising trigger enable
       EXTI->RTSR |= 1UL<<Pin;
       EXTI->FTSR |= 1UL<<Pin;
    }

    // Configure Interrupt Mask (Interrupt enabled)
    EXTI_enable(Pin);     // not masked


    // NVIC(IRQ) Setting
    int EXTI_IRQn = 0;

    if (Pin < 5)     EXTI_IRQn = Pin+6;
    else if   (Pin < 10)     EXTI_IRQn = 23;
    else           EXTI_IRQn = 40;

    NVIC_SetPriority(EXTI_IRQn, priority);   // EXTI priority
    NVIC_EnableIRQ(EXTI_IRQn);    // EXTI IRQ enable
}
```

Results

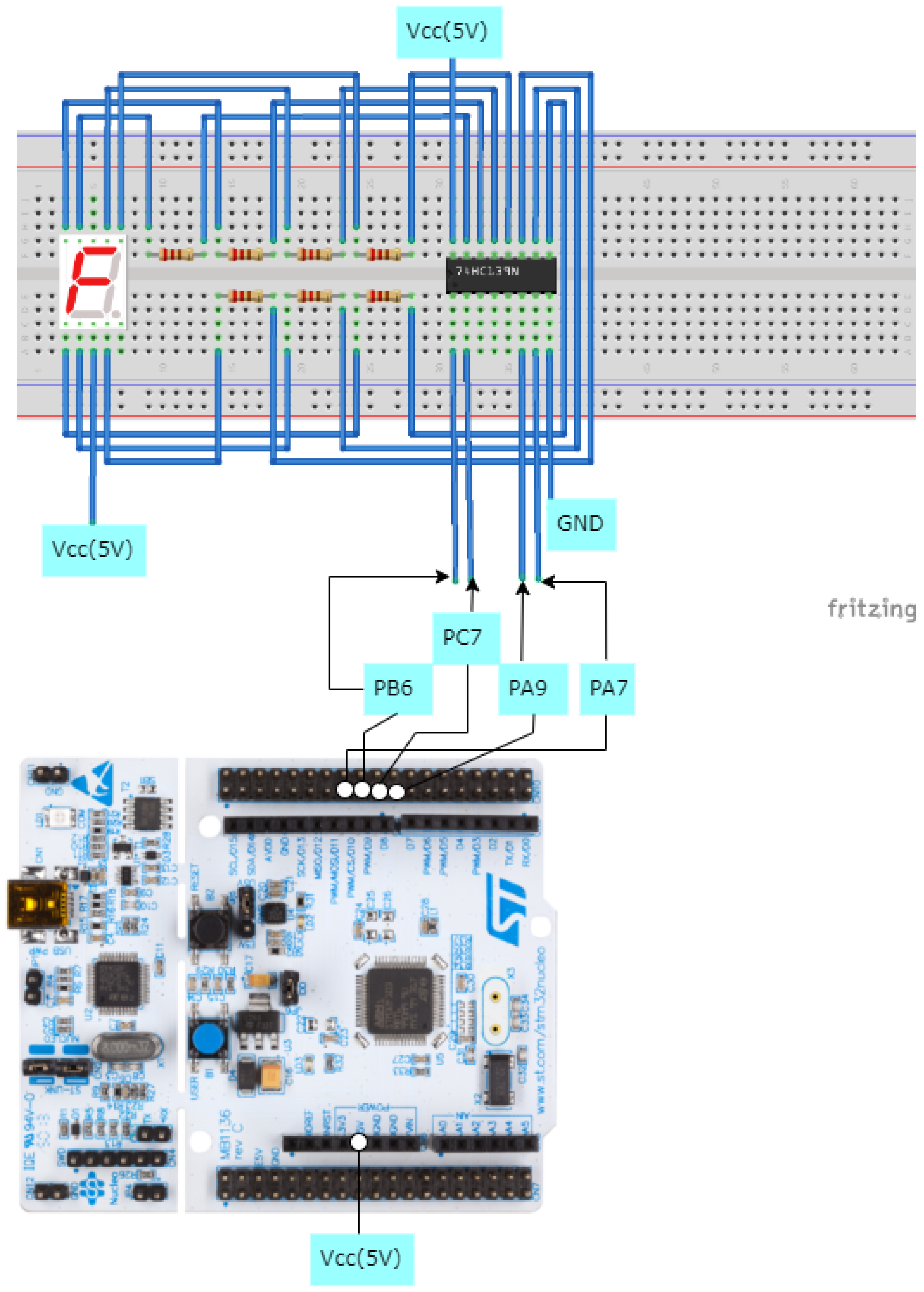
Link: https://youtube.com/shorts/5mQDUwYdhdg?feature=share

Problem 2: Counting numbers on 7-Segment using SysTick

Configuration

Digital In for Button (B1)	Digital Out for 7-Segment decoder
Digital In	Digital Out
PC13	PA7, PB6, PC7, PA9
PULL-UP	Push-Pull, No Pull-up-Pull-down, Medium Speed

Circuit Diagram



Code

- main

```
/**
******************************************************************************
* @author  SSSLAB
* @Mod       2021-8-30 by YKKIM
* @brief   Embedded Controller:  LAB Systick&EXTI with API
*                     - 7 segment
*
******************************************************************************
```

```
*/

#include "stm32f411xe.h"
#include "ecGPIO.h"
#include "ecRCC.h"
#include "ecSysTick.h"

int count = 0;
// Initialiization
void setup(void)
{
    RCC_PLL_init();
    SysTick_init();
    sevensegment_init();
}

int main(void) {
    // Initialiization --------------------------------------------------------
        setup();

    // Inifinite Loop ----------------------------------------------------------
    while(1){
        sevensegment_display(count);
        delay_ms(1000);
        count++;
        if (count >9) count =0;
        SysTick_reset();
    }
}
```

- SysTick.c

```
/*----------------------------------------------------------------\
@ Embedded Controller by Young-Keun Kim - Handong Global University
name: NohYunKi
/----------------------------------------------------------------*/




#include "ecSysTick.h"
#define MCU_CLK_PLL 84000000
#define MCU_CLK_HSI 16000000

volatile uint32_t msTicks=0;

//EC_SYSTEM_CLK

void SysTick_init(void){
    //  SysTick Control and Status Register
    SysTick->CTRL = 0;                                          // Disable
SysTick IRQ and SysTick Counter

    // Select processor clock
    // 1 = processor clock;  0 = external clock
    SysTick->CTRL |= SysTick_CTRL_CLKSOURCE_Msk;
    // uint32_t MCU_CLK=EC_SYSTEM_CLK
    // SysTick Reload Value Register
    SysTick->LOAD = MCU_CLK_PLL / 1000 - 1;                     // 1ms, for HSI
PLL = 84MHz.

    // SysTick Current Value Register
    SysTick->VAL = 0;

    // Enables SysTick exception request
    // 1 = counting down to zero asserts the SysTick exception request
    SysTick->CTRL |= SysTick_CTRL_TICKINT_Msk;

    // Enable SysTick IRQ and SysTick Timer
    SysTick->CTRL |= SysTick_CTRL_ENABLE_Msk;

    NVIC_SetPriority(SysTick_IRQn, 16);     // Set Priority to 1
    NVIC_EnableIRQ(SysTick_IRQn);           // Enable interrupt in NVIC
}



void SysTick_Handler(void){
    SysTick_counter();
}

void SysTick_counter(){
    msTicks++;
}


void delay_ms (uint32_t mesc){
  uint32_t curTicks;

  curTicks = msTicks;
  while ((msTicks - curTicks) < mesc);

    msTicks = 0;
}

//void delay_ms(uint32_t msec){
//  uint32_t now=SysTick_val();
//  if (msec>5000) msec=5000;
//  if (msec<1) msec=1;
//  while ((now - SysTick_val()) < msec);
//}


void SysTick_reset(void)
{
    // SysTick Current Value Register
    SysTick->VAL = 0;
}

uint32_t SysTick_val(void) {
    return SysTick->VAL;
```

```
}

//void SysTick_counter(){
//  msTicks++;
//  if(msTicks%1000 == 0) count++;
//}
```

Results

Link: https://youtube.com/shorts/lAM8qyBFHq4?feature=share