Re: CENG 255 -Lab 4 notes

Philip Baback Alipour

Wed 2016-11-23 6:34 PM

Inbox

To: Philip Baback Alipour <philipbaback_orbsix@msn.com>;

Cc: alimst@uvic.ca <alimst@uvic.ca>; maryam.bhr1989@hotmail.com <maryam.bhr1989@hotmail.com>; sjokhio@uvic.ca <sjokhio@uvic.ca>;

Importance: High

Dear groups G04, G05 and G06 CENG 255 members,

On behalf of my co-TAs of these sections I am also sharing the following notes among their students as BCC'ed.

1- You are basically programming the I/O i.e. buttons BTN1 and BTN2; then observing the LEDs toggle ON/OFF.

2- In order to make sure that the debugger switches are setup properly as well as any prior syntax errors in your program, compile your basic template code first then add and edit before your next code compilation.

I will introduce to you the interrupt signal process (like a keystroke on your keyboard) which is sent to the micro-controller output port from the lower board.
From there after processing the signal (per 8-bit register) you'll instead of displaying characters on screen, in fact lit up an LED.

3- However, I will give extra/bonus marks, if you purposely write a code (separately) where the signal bounces and not all of it (information/data) is returned back to where the LEDs are and the light dims away incrementally (as you toggle a few times) (an engineering issue where an exception can handle it... this in the real world detecting defected boards will become extremely handy). So in return a **debouncing algorithm** must be written as the main code (see below):

In general, these are expected during the lab:
4- Be prepared to see lots of issues with the "**switch bounce**" - that is, the buttons on the board rather than registering 1 rising edge per button press;
will instead bounce the signal more than once (multiple rising edges for each button press :o) therefore making the lab that much more difficult....

Capable students for such performance points need to include a denouncing algorithm (which is what I have pointed out above), otherwise the button pressing will work 75% of the time on some boards. So you are encouraged to keep working on it and test.

5- Also, extra points will be given if you turn on the lights in a series and random like Christmas lights.

Bear in mind, the evaluation will not take place until the week after tomorrow.

The following is some complementary notes from Brent:

“A couple of notes regarding lab 4 interrupt project.

For the interrupts to work with the second button you need to insure the following conditions have been met.

1) Push button two is connect to pin PA5. You can configure it just line PA4 for push button 1. You can even OR the pins to simplify the process as shown below.

Original code

```
/* Configure PA4 pin as input floating */
  GPIO_InitStructure.GPIO_Pin = GPIO_Pin_4;
  GPIO_InitStructure.GPIO_Mode = GPIO_Mode_IN;
  GPIO_InitStructure.GPIO_OType = GPIO_OType_PP;
  GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
  GPIO_InitStructure.GPIO_PuPd = GPIO_PuPd_DOWN;
  GPIO_Init(GPIOA, &GPIO_InitStructure);
```

Modified code

```
  /* Configure PA4 and PA5 pins as input floating */
  GPIO_InitStructure.GPIO_Pin = GPIO_Pin_4 | GPIO_Pin_5;
  GPIO_InitStructure.GPIO_Mode = GPIO_Mode_IN;
  GPIO_InitStructure.GPIO_OType = GPIO_OType_PP;
  GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
  GPIO_InitStructure.GPIO_PuPd = GPIO_PuPd_DOWN;
  GPIO_Init(GPIOA, &GPIO_InitStructure);
```

2) You need to enable the external interrupt for PA5 (Push Button 2). The procedure of OR the pins can also be applied to this code.

Original code

```
/* Configure EXTI4 line */
  EXTI_InitStructure.EXTI_Line = EXTI_Line4;
  EXTI_InitStructure.EXTI_Mode = EXTI_Mode_Interrupt;
  EXTI_InitStructure.EXTI_Trigger = EXTI_Trigger_Falling;
  EXTI_InitStructure.EXTI_LineCmd = ENABLE;
  EXTI_Init(&EXTI_InitStructure);
```

Modified code

```
 /* Configure EXTI4 and EXTI5 line */
  EXTI_InitStructure.EXTI_Line = EXTI_Line4 | EXTI_Line5;
  EXTI_InitStructure.EXTI_Mode = EXTI_Mode_Interrupt;
  EXTI_InitStructure.EXTI_Trigger = EXTI_Trigger_Falling;
  EXTI_InitStructure.EXTI_LineCmd = ENABLE;
  EXTI_Init(&EXTI_InitStructure);
```

3) In the interrupt service routine to verify which button is pushed the interrupt pending register will need to be examined. It will also need to be cleared before the interrupt is exited.

Original code

```
void EXTI4_15_IRQHandler(void) {
	if (EXTI->IMR & EXTI_IMR_MR4)
	{
		EXTI->PR |= EXTI_PR_PR4;
		EXTI4Counter = ( EXTI4Counter + 1 ) & 15;
	}
}
```

Modified code.

```
void EXTI4_15_IRQHandler(void) {
	if (EXTI->IMR & EXTI_IMR_MR4)
	{
	  if ( EXTI->PR & EXTI_PR_PR4 )    // Test for push button 1
	  {
	    EXTI->PR |= EXTI_PR_PR4;     // Clear pending interrupt for push button 1
	    EXTI4Counter = ( EXTI4Counter + 1 ) & 15;
	  }
	}
	if (EXTI->IMR & EXTI_IMR_MR5)
	{
	  if ( EXTI->PR & EXTI_PR_PR5 )    // Test for push button 2
	  {
	    EXTI->PR |= EXTI_PR_PR5;    // Clear pending interrupt for push button 2
	    EXTI4Counter = ( EXTI4Counter - 1 ) & 15;
	  }
	}
}
```

You can read more about the pending register in the cortex reference manual on page 227. Link below

http://www.st.com/resource/en/reference_manual/dm00031936.pdf

---

RM0091 Reference manual - STMicroelectronics

www.st.com

July 2015 DocID018940 Rev 8 1/1008 1 RM0091 Reference manual STM32F0x1/STM32F0x2/STM32F0x8 advanced ARM®-based 32-bit MCUs Introduction This reference manual targets ...

Have a good lab."

I will also post these notes on my webpage under the teaching section.

See you tomorrow,
Philip

PS- To the students of my section G04: please bring your lab#3 reports in the lab for submission.

==========

Philip B. Alipour,
Ph.D. Candidate in Electrical, Computer Engineering and Quantum Physics,
Dept. of Electrical and Computer Engineering, University of Victoria, V8W 3P6, Canada,
Office: ELW Room # A358,
Emails: phibal12@uvic.ca or philipbaback_orbsix@msn.com
Homepage: http://web.uvic.ca/~phibal12/