CE2107 Lab3 Assignment Sheet (to be submitted to NTULearn before next lab)

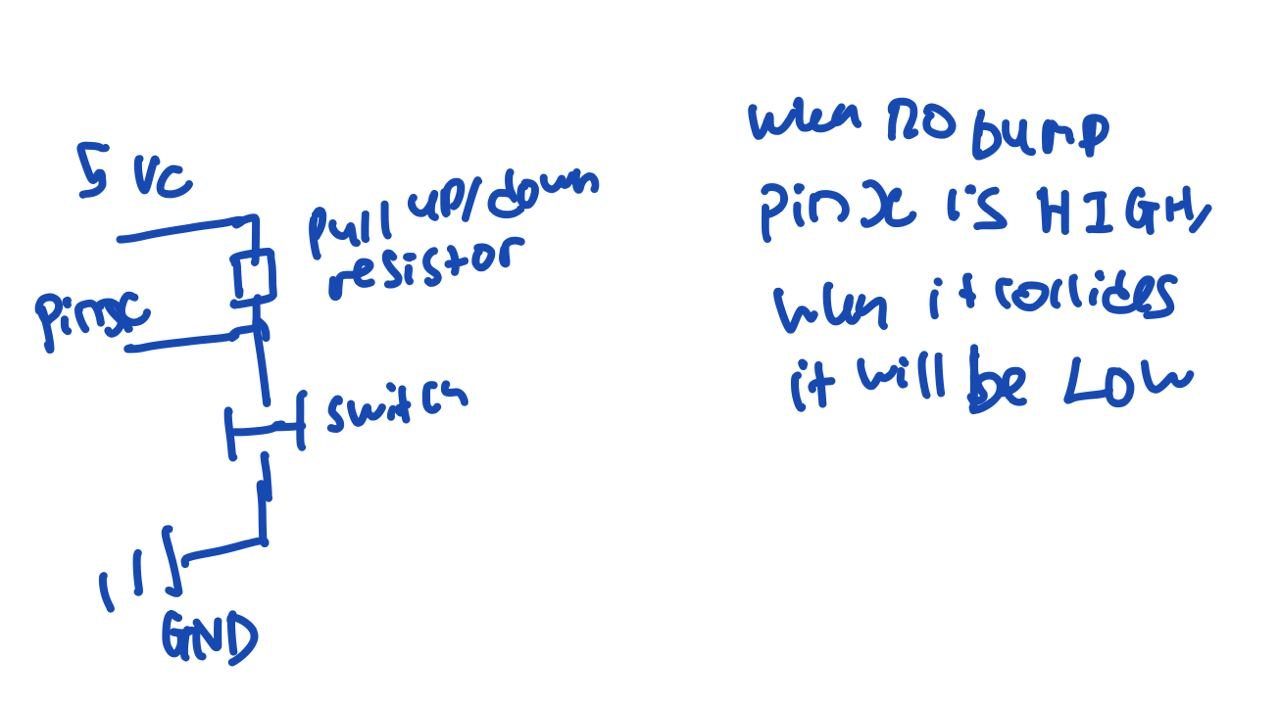
Name:Mervyn Chiong Jia Rong Lab Group: SEP4 Date:

1. Section 6. Other than the procedures outlined in the introduction of Exception Handling, what other registers need to be noted when using the Exception Handling System in ARM Cortex M4F processor? Think globally…

**PRIMASK and BASEPRI**

**PRIMASK regardless of priority level is to enable/disable all interrupts and exceptions aside from NMI and a hard fault.**

**BASEPRI is an 8bit value that will support 255 priorities and will disable interrupts less than or equal to the set priority level**

1. Section 6.2. The bump switch used in the lab is shown below. Pin 1 and 3 of the bump switch are connected to the MSP432. Draw the internal circuit of the bump switch and describe how the MSP432 GPIO can be used to detect that the switch is closed?  
    
2. Section 6.3. Write down the GPIO configuration used for pins connected to pin 3 of the Bump switch.

P4->SEL0 &= ~0xED;

P4->SEL1 &= ~0xED;

P4->DIR &= ~0xED;

P4->REN |= 0xED;

P4->OUT |= 0xED;

uint8\_t **Bump\_Read**(**void**){

uint8\_t result=0x00;

uint8\_t result1,result2,result3;

result1=(P4->IN)&0x01;

result2=(P4->IN)&0x0C;

result2=result2>>1;

result3=(P4->IN)&0xE0;

result3=result3>>2;

result=result1+result2+result3;

**return** (result);

}

1. Section 6.3. Illustrate with detail working and APIs used how systick timer is configure to interrupt the system at 1000Hz frequency.

**SysTick\_Init(48000,2);**

**Clock is set to 48Mhz bit systic is 48000 cycles so to find frequency is 48Mhz/48000cycles=1000hz/cycle**

**Systic will count down from the specified before 1 interrupt.**

1. Section 6.4. In the Simple motor project, the api used to move the motor forward is Motor\_ForwardSimple(uint16\_t duty, uint32\_t time), where time is number of 10ms units, i.e. if time=2, motot will run for 2ms. Show and explain the code in the function that enable this 10ms unit timing.
2. **void** Motor\_ForwardSimple(uint16\_t duty, uint32\_t time){  
   *// Drives both motors forward at duty (100 to 9900)  
   // Runs for time duration (units=10ms), and then stops  
   // Stop the motors and return if any bumper switch is active  
   // Returns after time\*10ms or if a bumper switch is hit*uint32\_t counter;  
   *//uint8\_t temp;* counter = time;  
    *// Forward direction* P1->OUT &= ~0xC0;  
    *//temp=Bump\_Read();* **while**(counter>0 && Bump\_Read()==0x3F){  
    *//while(counter>0){  
    // Enable L and R motor* P3->OUT |= 0xC0;  
    *// PWM HIGH* P2->OUT |= 0xC0;  
    SysTick\_Wait1us(duty);  
    *//PWM LOW* P2->OUT &= ~0xC0;  
    SysTick\_Wait1us(10000-duty);  
    counter--;  
    }  
    Motor\_StopSimple();  
   }

**In the main loop, both motors will be enabled and it will move forward for “duty” microseconds. period is decided by SysTick\_wait1us(duty) Which will then be turned off after 10ms.**

**Counter then decreased, and the same enablers and disablers are restarted at intervals of 10ms**

1. Section 6.5. Reference to PWM\_Init1() in PWM.c, what is the base clock used to increment the counters in Timer\_A0? Show the details of how this base clock of Timer\_A0 is derived.

**Base clock = SMCLK \*div ratio IDEX \* div ratio ID**

**void** PWM\_Init34(uint16\_t period, uint16\_t duty3, uint16\_t duty4){  
  
 *// write this as part of Lab 3* **if**(duty3 >= period) **return**; *// bad input* **if**(duty4 >= period) **return**; *// bad input* P2->DIR |= 0xC0; *// P2.6, P2.7 output* P2->SEL0 |= 0xC0; *// P2.6, P2.7 Timer0A functions* P2->SEL1 &= ~0xC0; *// P2.6, P2.7 Timer0A functions* TIMER\_A0->CCTL[0] = 0x0080; *// CCI0 toggle* TIMER\_A0->CCR[0] = period; *// Period is 2\*period\*8\*83.33ns is 1.333\*period* TIMER\_A0->EX0 = 0x0000; *// divide by 1* TIMER\_A0->CCTL[3] = 0x0040; *// CCR1 toggle/reset* TIMER\_A0->CCR[3] = duty3; *// CCR1 duty cycle is duty3/period* TIMER\_A0->CCTL[4] = 0x0040; *// CCR2 toggle/reset* TIMER\_A0->CCR[4] = duty4; *// CCR2 duty cycle is duty4/period* TIMER\_A0->CTL = 0x02F0; *// SMCLK=12MHz, divide by 8, up-down mode  
 // bit mode  
 // 9-8 10 TASSEL, SMCLK=12MHz  
 // 7-6 11 ID, divide by 8  
 // 5-4 11 MC, up-down mode  
 // 2 0 TACLR, no clear  
 // 1 0 TAIE, no interrupt  
 // 0 TAIFG*}

*// SMCLK = 48MHz/4 = 12 MHz, 83.33ns*

**^ this is from the above code**

**SMCLK=12Mhz because the**

**ID div ratio=1 due to”** TIMER\_A0->CTL = 0x02F0;**”**

**And IDEX div ratio is 1 because “**TIMER\_A0->EX0 = 0x0000;**”**

1. Section 6.5. What is the PWM frequency generated to the motor? illustrate with detail working.

TIMER\_A0->CCR[0] = period;

**^this code defines the number period or in other words how far It goes starting from 0 all the way to the max\_count. So it will be in 1 period, 2\*max\_count\*clock period for a PWM cycle to repeat.**

1. Section 6.5. Is interrupt mechanism used in the PWM generation via Timers?

**No**

1. Section 6.5. What is the IRQ number corresponding to the interrupt used by Timer\_A1 in Lab3\_TimerCompare\_Motor project use? What is the corresponding Exception number?

**Taken from the NVIC table:**

**Timer\_A1 is input interrupt 10**

**Therefore IRQ 10 is used and config via ISER0 reg, leading to 16(external interrupts start at offset 16)+10 to be 26**