Priority Registers on the NVIC

NVIC_PRIx_R

- Each priority register contains an 8-bit priority field for four devices.
- Only the top three bits of the 8-bit field are used. 0-7

Address	31 – 29	23 – 21	15 – 13	7 – 5	Name
0xE000E400	GPIO Port D	GPIO Port C	GPIO Port B	GPIO Port A	NVIC_PRI0_R
0xE000E404	SSI0, Rx Tx	UART1, Rx Tx	UART0, Rx Tx	GPIO Port E	NVIC_PRI1_R
0xE000E408	PWM Gen 1	PWM Gen 0	PWM Fault	I2C0	NVIC_PRI2_R
0xE000E40C	ADC Seq 1	ADC Seq 0	Quad Encoder	PWM Gen 2	NVIC_PRI3_R
0xE000E410	Timer 0A	Watchdog	ADC Seq 3	ADC Seq 2	NVIC_PRI4_R
0xE000E414	Timer 2A	Timer 1B	Timer 1A	Timer 0B	NVIC_PRI5_R
0xE000E418	Comp 2	Comp 1	Comp 0	Timer 2B	NVIC_PRI6_R
0xE000E41C	GPIO Port G	GPIO Port F	Flash Control	System Control	NVIC_PRI7_R
0xE000E420	Timer 3A	SSI1, Rx Tx	UART2, Rx Tx	GPIO Port H	NVIC_PRI8_R
0xE000E424	CAN0	Quad Encoder 1	I2C1	Timer 3B	NVIC_PRI9_R
0xE000E428	Hibernate	Ethernet	CAN2	CAN1	NVIC_PRI10_R
0xE000E42C	uDMA Error	uDMA Soft Tfr	PWM Gen 3	USB0	NVIC_PRI11_R
0xE000ED20	SysTick	PendSV	--	Debug	NVIC_SYS_PRI3_R

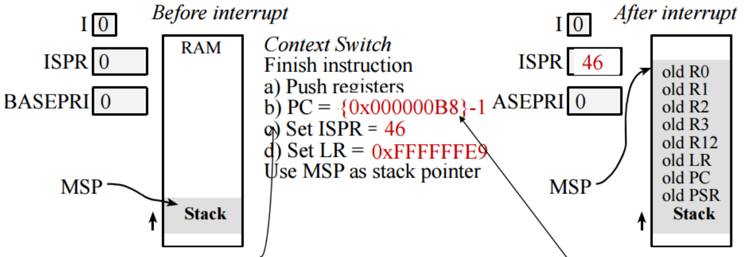
Shared Interrupt Vector

- If multiple pins on one GPIO port are armed, the shared ISR must poll to determine which one(s) requested service.

Example (TM4C123):

```
volatile unsigned long count = 0;   //global variables

void GPIOPortF_Handler(void){
  if(GPIO_PORTF_RIS_R&0x10){     // poll PF4
    GPIO_PORTF_ICR_R |= 0x10;    // acknowledge flag4
    count++;                     // signal SW1 occurred
  }
  if(GPIO_PORTF_RIS_R&0x01){     // poll PF0
    GPIO_PORTF_ICR_R |= 0x01;    // acknowledge flag0
    count++;                     // signal SW2 occurred
  }
}
```

Interrupt Context Switch



Context Switch
Finish instruction
a) Push registers
b) PC = {0x000000B8}-1
c) Set ISPR = 46
d) Set LR = 0xFFFFFFE9
Use MSP as stack pointer

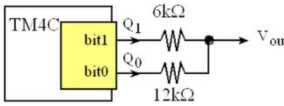
Vector address for GPIO Port F

PC={0x000000B8}-1=0x000003BB-1
= 0x000003BA

Vector Number **46** corresponds to GPIO Port F

DAC Example

Design a 2-bit **binary-weighted** DAC with a range of 0 to 3.3v using resistors

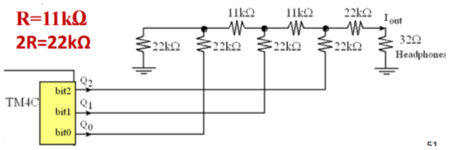


N	Q_1	Q_0	V_{out} (V)
0	0	0	0.0
1	0	3.3	1.1
2	3.3	0	2.2
3	3.3	3.3	3.3

Design a 3-bit **R-2R** DAC

N	Q_2	Q_1	Q_0	I_{out} (μA)
0	0	0	0	0.0
1	0	0	3.3	12.5
2	0	3.3	0	25.0
3	0	3.3	3.3	37.5
4	3.3	0	0	50.0
5	3.3	0	3.3	62.5
6	3.3	3.3	0	75.0
7	3.3	3.3	3.3	87.5

R=11kΩ
2R=22kΩ



Edge-triggered Modes

- **IS** (Interrupt Sense) bit in **GPIO_PORTx_IS_R**
  - If IS=0, edge triggering; If IS=1, level triggering
- **IBE** (Interrupt Both Edges) in **GPIO_PORTx_IBE_R**
- **IEV** (Interrupt Event) in **GPIO_PORTx_IEV_R**
  - 1: rising edge triggering; 0: falling edge triggering
- **IME** (Interrupt Mask Enable) in **GPIO_PORTx_IM_R**
  - 1: arm interrupt; 0: disarm interrupt

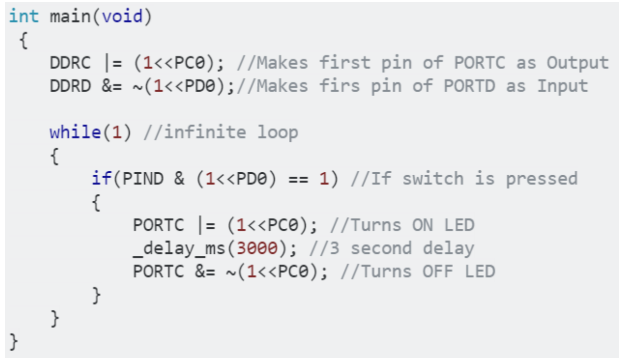
IS	IBE	IEV	IME	Port mode
0	0	0	0	Input, falling edge trigger, busy wait
0	0	1	0	Input, rising edge trigger, busy wait
0	1	-	0	Input, both edges trigger, busy wait
0	0	0	1	Input, falling edge trigger, interrupt
0	0	1	1	Input, rising edge trigger, interrupt
0	1	-	1	Input, both edges trigger, interrupt

Interrupt Processing (Context Switch)

1. **The execution of the main program is suspended**
   1. the current instruction is finished,
   2. suspend execution and push 8 registers (R0-R3, R12, LR, PC, PSR) on the stack
   3. LR set to 0xFFFFFFE9 (indicates interrupt return)
   4. IPSR set to vector number (ISR_NUMBER)
   5. sets PC to ISR address
2. **The interrupt service routine (ISR) is executed**
   – clears the flag that requested the interrupt
   – performs necessary operations
   – communicates using global variables
3. **The main program is resumed when ISR executes BX LR**
   – pulls the 8 registers from the stack

Exercise 3: Assume a switch is connected to PB5 using **positive logic**, an LED is connected to PD1 using **negative logic**. Write a C program to toggle the LED when the switch is pressed, and turn off the LED when the switch is not pressed.

```
int main() {
        port_init();
        while(1){
                ...
}
```



```
int main(void)
 {
    DDRC |= (1<<PC0); //Makes first pin of PORTC as Output
    DDRD &= ~(1<<PD0);//Makes firs pin of PORTD as Input

    while(1) //infinite loop
    {
        if(PIND & (1<<PD0) == 1) //If switch is pressed
        {
            PORTC |= (1<<PC0); //Turns ON LED
            _delay_ms(3000); //3 second delay
            PORTC &= ~(1<<PC0); //Turns OFF LED
        }
    }
}
```

Exercise

1. The system has two external input switches *SW1* and *SW2* and two external output LEDs *LED1* and *LED2*.

2. The counter is controlled by edge-triggered interrupt – the counter is incremented by 1 when *SW1* is pressed and decremented by 1 when *SW2* is pressed.

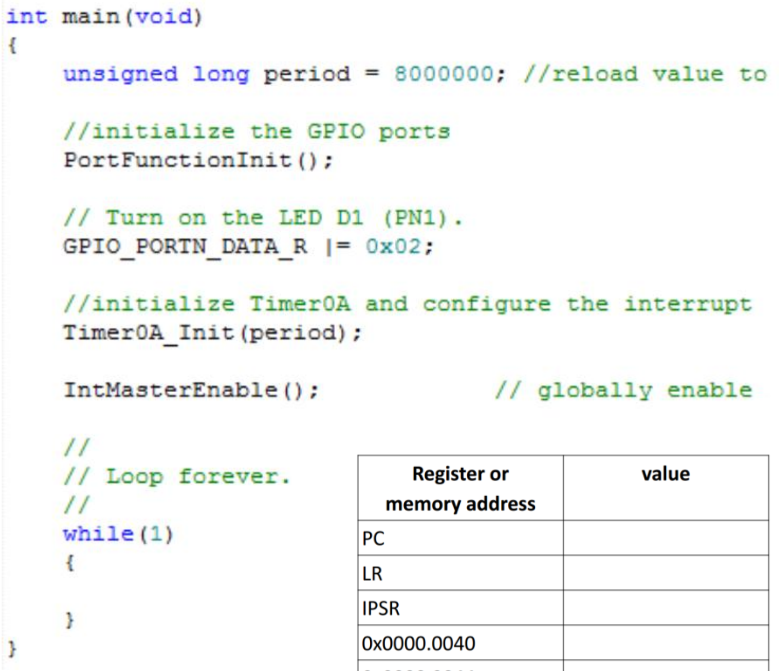
3. The LEDs are used to display the counter. *LED1* is used to display bit 0 of the counter and *LED2* is used to display bit 1. *LED1* and *LED2* are connected with pins **PD1** and **PD2** on the microcontroller using positive logic, respectively.

4. *SW1* and *SW2* are wired with pins **PA0** and **PB0** using negative logic respectively.

Exercise

Beginning address in the mem	Function name
0x0000.0368	PortFunctionInit()
0x0000.03E8	GPIOInterrupt_Init()
0x0000.047E	GPIOPortA_Handler()
0x0000.049C	GPIOPortB_Handler()
0x0000.0524	main()
0x0000.0284	IntDefaultHandler

Toggle_timer_interrupt_TivaWare



```
int main(void)
{
    unsigned long period = 8000000; //reload value to

    //initialize the GPIO ports
    PortFunctionInit();

    // Turn on the LED D1 (PN1).
    GPIO_PORTN_DATA_R |= 0x02;

    //initialize Timer0A and configure the interrupt
    Timer0A_Init(period);

    IntMasterEnable();              // globally enable

    //
    // Loop forever.
    //
    while(1)
    {

    }
}
```

Register or memory address	value
PC	
LR	
IPSR	
0x0000.0040	
0x0000.0044	
0x0000.004C	

E

What will be the values stored in **PC**, **LR**, **IPSR**, memory addresses **0x0000. 0040, 0x0000. 0044,** and **0x0000. 004C** after *SW1 (PA0)* is pressed?

Vector Number	Interrupt Number	Vector Address	Description
16	0	0x0000.0040	GPIO Port A
17	1	0x0000.0044	GPIO Port B
19	3	0x0000.004C	GPIO Port D

Exercise 2: The following program uses **bit specific addressing** to access Port B. Complete the C program. If Port B has an initial value of 0x53, what will be the value of "data" and the value on PortB after the code segment is executed?

```
#define PB24   (*((volatile unsigned long *)
0x40005050))

data = PB24;      // data = ?
PB24 = 0x24;
```

Exercise 4: Given the code below, configure the reload value of the timer "period" to generate 100 ms periodic delay.

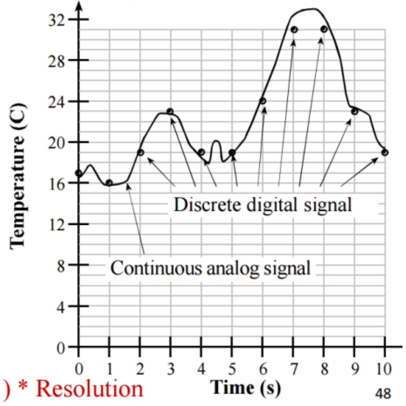
```
unsigned long period =  ______________________;

SysCtlClockSet(SYSCTL_SYSDIV_10|SYSCTL_USE_PLL|
SYSCTL_XTAL_16MHZ|SYSCTL_OSC_MAIN);

Timer0A_Init(period);
```

Digital Representation of Analog Signals

Digitization: Amplitude and time quantization

- Range
  - 0 to 31 °C
- Resolution
  - 31 °C /31 = 1 °C
- Precision
  - 5 bits
  - 32 alternatives

Sampling Rate
- 1Hz



Range = (Precision-1) * Resolution