Texas Instruments Robotics System Learning Kit
Module 13
Activity: Timers
Activity: Timers

**Question 1**
There is a 32-bit timer on the MSP432. If this timer is clocked at 12 MHz, what is the slowest period at which you could create a periodic interrupt?

**Question 2**
Read the MSP432 data sheet on the 32-bit timer. Write software that uses this timer to execute a periodic task, Task(), once a second.

**Question 3**
Use the Timer A2 example without changing the Timer A2 initialization, from the lecture slides, to run Task1 once a second, Task2 twice a second, and Task3 three times a second. To reduce latency, set it up so no two tasks run during the same ISR. Assume SMCLK is 12 MHz.

**Question 4**
Write software to generate four PWM cycles with period 1ms, but have independent duty cycles.

**Question 5**
List all the MSP432 pins that could be used to generate PWM outputs.

**Question 6**
Read the MSP432 data sheet for the Timer_A module. Describe the behavior of this software system, assuming the main program calls TimerA0_Init and then enables interrupts.

```c
#define N1 1000
#define N2 1500
#define N3 2000

void TimerA0_Init(void){
    TA0CTL &= ~0x0030;       // 0) halt Timer A0
    TA0CTL = 0x0240;         // 1) SMCLK, divide by 2
    TA0EX0 = 0x0005;         // divide by 6
    TA0CCTL1 = 0x0010;       // 2) compare mode, arm CCIFG
    TA0CCTL2 = 0x0010;       // compare mode, arm CCIFG
    TA0CCTL3 = 0x0010;       // compare mode, arm CCIFG
    TA0CCR1 = N1/2;          // 3) time of first interrupt
    TA0CCR2 = N2/2;          //
    TA0CCR3 = N3/2;          //
    NVIC_IPR2 = (NVIC_IPR2&0xFFFF00FF)|0x00004000;
    NVIC_ISER0 = 0x00000200; // 5) enable interrupt 9
    TA0CTL |= 0x0024;        // 6) reset and start
}

void TA0_N_IRQHandler(void){
    if(TA0CCTL1&0x0001){
        TA0CCTL1 &= ~0x0001;   // acknowledge interrupt 1
        TA0CCR1 = TA0CCR1+N1;  // set up for next time
        Task1(); // execute user task
    }
    if(TA0CCTL2&0x0001){
        TA0CCTL2 &= ~0x0001;   // acknowledge interrupt 2
        TA0CCR2 = TA0CCR2+N2;  // set up for next time
        Task2(); // execute user task
    }
    if(TA0CCTL3&0x0001){
        TA0CCTL3 &= ~0x0001;   // acknowledge interrupt 3
        TA0CCR3 = TA0CCR3+N3;  // set up for next time
        Task3(); // execute user task
    }
}
```
ti.com/rslk