GpioSetup ()

- Configure registers LATA, ANSELA, TRISA
- TRISA = 0x00 (make all PORTA pins an output)
- LATA = 0x80 (sets RA7 = 1 and the counter bits RA0-RA5 = 0)

UserAppInitialization ()

- LATA = 0x80 (sets RA7 = 1 and the counter bits RA0-RA5 = 0)
- T0CON0 = 0xF0; When I looked at the data sheet, bits 5 and 6 were unspecified so I assumed they were don't cares, meaning I also could have set T0CON0 = 0x90 and nothing would have changed
- T0CON1 = 0x44

UserAppRun()

- Need a static u16 counter variable
- Need an array of type u8 and initialize it to store the following pattern that I want to implement, 0x01, 0x02, 0x04, 0x08, 0x10, 0x20
- I need a variable to represent the index of the array so I can pass each index into LATA later in this function. I am going to use an INT instead of a u8 variable as I only need to index 5 elements and would prefer to use INT.
- Increment the counter up to 500 as per the instruction in the activity handout
- Use index variable to pass array element into LATA so the patterns can be displayed on the LEDs, and increment index variable as well
- Need an if statement for when index variable is equal to the length of the array + 1 and reset the variable so we don't get an "OutofBound" Error

TimeXus()

- I want to create a separate source file to contain this time delay function and have it called from main(). This also means I will have to create another header file which contains a description of the necessary "requires" and "promises"
- First step is to disable the timer. This can be done by setting the enable bit of the T0CON0 register (bit 7) equal to zero. Since T0CON0 was already initialized a value back in UserAppInitialization(), I will use a bit-wise AND operation. Therefore, T0CON0 &= 0x70.
- Next to preload the TMR0H and TMR0L registers, the given INPUT_PARAMETER must be subtracted from the max value of 0xFFFF then add 1 due to roll over. This will time out the requested period.
- The 16-bit result must be split into the 8 high bits for TMR0H and 8 low bits for TMR0L. The high bits can be isolated by left shifting the result by 8, and the low bits can simply be attained by zeroing out the high bits from the result
- Clear the flag bit by setting TMR0IF = 0, from the PIR3 register
- Enable the timer again by setting the enable bit on T0CON0 back to 1

Main()

- Call the TimeXus() and pass on an INPUT_PARAMETER of 1000 since the instructions specified to time out a 1ms delay
- Instructions also specified to set the flag bit equal to zero and make the function wait in that state until roll over occurs where TMR0IF = 1. This can be done through an empty while loop.