

EECE.3170: Microprocessor Systems Design I

Fall 2016

Lecture 29: Key Questions

November 28, 2016

1. Describe the following assembly program, which blinks a single LED:

```
cblock 0x70      ;shared memory accessible from all banks
Delay1          ;Two registers for delay loop in shared mem
Delay2
    endc

    ORG 0
Start:
    banksel      OSCCON          ;bank1
    movlw        b'00111000'     ;set cpu speed of 500KHz
    movwf        OSCCON          ;OSCCON configures
                                ; internal clock
    bcf          TRISC,0         ;Pin C0 = output for DS1
    banksel      LATC            ;bank2
    clrf         LATC            ;Turn off all of the LEDs
MainLoop:
    bsf          LATC, 0         ;turn on DS1

OndelayLoop:
    decfsz       Delay1,f        ;Waste time.
    bra         OndelayLoop      ;Inner loop takes 3 inst
                                ; per loop * 256 loops =
                                ; 768 instructions
    decfsz       Delay2,f        ;The outer loop takes an
                                ; additional 3
                                ; instructions per loop
                                ; * 256 loops
    bra         OndelayLoop      ; (768+3) * 256 = 197376
                                ; instructions /
                                ; 125K instructions per
                                ; second = 1.579 sec
    bcf          LATC,0          ;Turn off LED C0

OffDelayLoop:
    decfsz       Delay1,f        ;same delay as above
    bra         OffDelayLoop
    decfsz       Delay2,f
    bra         OffDelayLoop
    bra         MainLoop         ;Do it again...
    end
```

1. Extra space to describe first program.

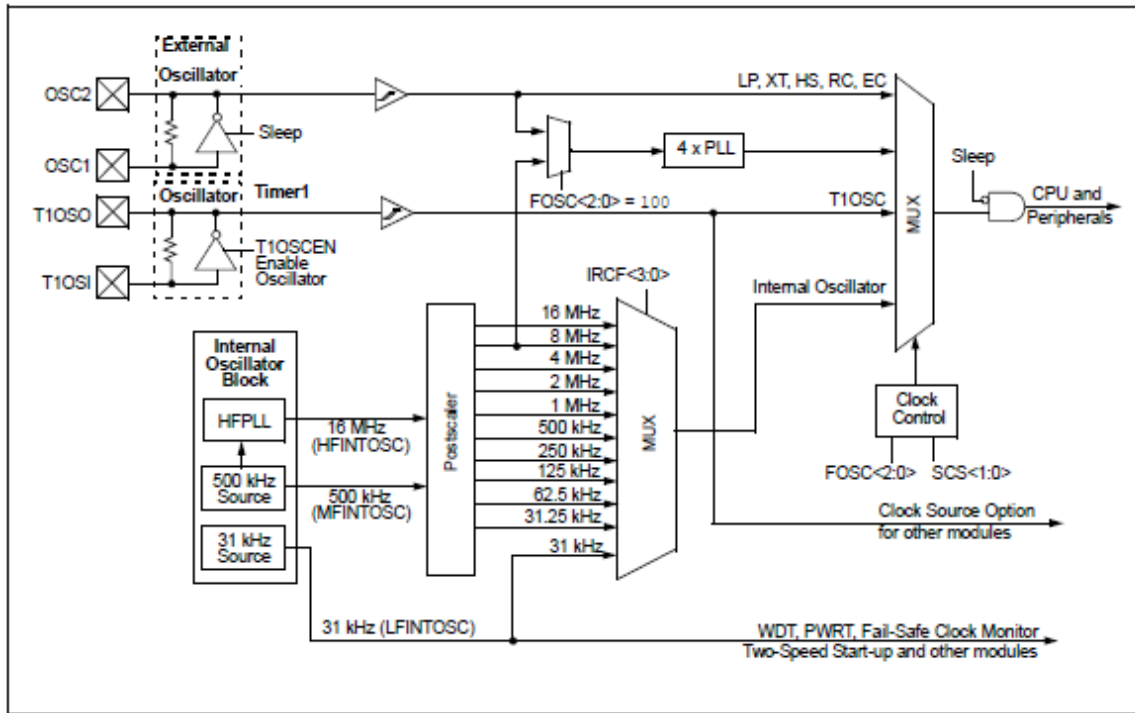
2. Describe the equivalent program in C, shown below:

```
void main(void) {
    unsigned int delay; // 16 bit variable

    OSCCON = 0b00111000; //500KHz clock speed
    TRISCbits.TRISC0 = 0; //using pin as output
    delay = 11250;
    while (1) {
        //each instruction is 8us (1/(500KHz/4))
        while(delay-- != 0)continue;

        LATCbits.LATC0 ^= 1; //toggle LED
        delay = 11250; //reset delay counter
    }
}
```

3. Describe the basic functionality of the PIC16F1829 clock generation module below:



4. Explain the operation of the programs used to rotate the LEDs using an instruction count-based delay loop (rotate.asm and rotate.c).

5. Explain the features of a typical microcontroller timer module.

6. Explain the operation of the programs used to rotate the LEDs using a timer-based delay loop (timer0.asm and timer0.c).

```

; *****
; Lesson 3 - "Rotate"
;
; This lesson will introduce shifting instructions as well as bit-oriented skip operations to
; move the LED display.
;
; LEDs rotate from right to left at a rate of 1.5s
;
;
; PIC: 16F1829
; Assembler: MPASM v5.43
; IDE: MPLABX v1.10
;
; Board: PICkit 3 Low Pin Count Demo Board
; Date: 6.1.2012
;
; *****
; * See Low Pin Count Demo Board User's Guide for Lesson Information*
; *****

#include <p16F1829.inc>
    __CONFIG _CONFIG1, (_FOSC_INTOSC & _WDTE_OFF & _PWRTE_OFF & _MCLRE_OFF & _CP_OFF & _CPD_OFF &
    _BOREN_ON & _CLKOUTEN_OFF & _IESO_OFF & _FCMEN_OFF);
    __CONFIG _CONFIG2, (_WRT_OFF & _PLLEN_OFF & _STVREN_OFF & _LVP_OFF);

    errorlevel -302                ;supress the 'not in bank0' warning
    cblock 0x70                    ;shared memory location that is accessible from all banks
Delay1
Delay2
    endc

; -----LATC-----
; Bit#:  -7---6---5---4---3---2---1---0---
; LED:   -----|DS4|DS3|DS2|DS1|-
; -----

    ORG 0                ;start of code
Start:
    banksel              ;bank1
    movlw                b'00111000' ;set cpu clock speed of 500KHz
    movwf                OSCCON
    clrf                 TRISC        ;make all of PORTC an output
    banksel              ;select the bank where LATC is (bank2)
    movlw                b'00001000' ;start the rotation by setting DS4 ON
    movwf                LATC        ;write contents of the working register to the latch
MainLoop:
OndelayLoop:
    decfsz              Delay1,f     ;Waste time.
    goto                OndelayLoop ;The Inner loop takes 3 instructions per loop * 256 loopss = 768
    instructions
    decfsz              Delay2,f     ;The outer loop takes an additional 3 instructions per lap * 256
    loops
    goto                OndelayLoop ;(768+3) * 256 = 197376 instructions / 125K instructions per second
    = 1.579 sec.

Rotate:
    lsrwf               LATC,F       ;shift the LEDs and turn on the next LED to the right
    btfsc               STATUS,C     ;did the bit rotate into the carry (i.e. was DS1 just lit?)
    bsf                 LATC, 3      ;yes, it did and now start the sequence over again by turning on
    DS4
    goto                MainLoop    ;repeat this program forever

    end                  ;end code section

```

```

/**
*****
* Lesson 3 - "Rotate"
*
* This lesson will introduce shifting instructions as well as bit-oriented skip operations to
* move the LED display.
*
* LEDs rotate from right to left at a rate of 1.5s
*
* PIC: 16F1829
* Compiler: XC8 v1.00
* IDE: MPLABX v1.10
*
* Board: PICkit 3 Low Pin Count Demo Board
* Date: 6.1.2012
*
* *****
* See Low Pin Count Demo Board User's Guide for Lesson Information*
* *****
*/

#include <htc.h> //PIC hardware mapping
#define _XTAL_FREQ 500000 //Used by the XC8 delay_ms(x) macro

//config bits that are part-specific for the PIC16F1829
__CONFIG(FOSC_INTOSC & WDTE_OFF & PWRTE_OFF & MCLRE_OFF & CP_OFF & CPD_OFF & BOREN_ON & CLKOUTEN_OFF &
IESO_OFF & FCMEN_OFF);
__CONFIG(WRT_OFF & PLLEN_OFF & STVREN_OFF & LVP_OFF);

/* -----LATC-----
* Bit#: -7---6---5---4---3---2---1---0---
* LED: -----|DS4|DS3|DS2|DS1|-
* -----
*/

void main(void) {
    TRISC = 0; //all pins are outputs
    OSCCON = 0b00111000; //500KHz clock speed
    LATC = 0b0001000; //start the rotation by setting DS4 ON - rotate
    from the right to left

    while (1) {
        __delay_ms(500); //delay 500ms
        LATC >> = 1; //shift to the right by 1
        if (STATUSbits.C) //when the last LED is lit, restart the pattern
            LATCbits.LATC3 = 1;
    }
}

```



```
banksel    LATC           ;bank2
lshf      LATC, f
btfsc     STATUS,C       ;did the bit rotate into the carry?
bsf       LATC,3         ;yes, put light DS4 back up

bra       MainLoop       ;continue forever

end
```

```

/**
*****
* Lesson 9 - "Timer0"
*
* Timer0 is a counter implemented in the processor. It may be used to count instruction
* cycles or external events, that occur at or below the instruction cycle rate.
* In the PIC18, Timer0 can be used as either an 8-bit or 16-bit counter, or timer. The
* enhanced mid-range core implements only an 8-bit counter.
* This lesson configures Timer0 to count instruction cycles and to set a flag when it rolls
* over. This frees up the processor to do meaningful work rather than wasting instruction
* cycles in a timing loop.
* Using a counter provides a convenient method of measuring time or delay loops as it
* allows the processor to work on other tasks rather than counting instruction cycles.
*
*
* LEDs rotate from right to left, similar to Lesson 3, at a rate of ~.5 seconds.
*
* PIC: 16F1829
* Compiler: XC8 v1.00
* IDE: MPLABX v1.10
*
* Board: PICkit 3 Low Pin Count Demo Board
* Date: 6.1.2012
*
*****
* See Low Pin Count Demo Board User's Guide for Lesson Information*
*****
*/

#include <htc.h> //PIC hardware mapping
#define _XTAL_FREQ 500000 //Used by the XC8 delay_ms(x) macro

//config bits that are part-specific for the PIC16F1829
__CONFIG(FOSC_INTOSC & WDTE_OFF & PWRTE_OFF & MCLRE_OFF & CP_OFF & CPD_OFF & BOREN_ON & CLKOUTEN_OFF &
IESO_OFF & FCMEN_OFF);
__CONFIG(WRT_OFF & PLEN_OFF & STVREN_OFF & LVP_OFF);

/* -----LATC-----
* Bit#: -7---6---5---4---3---2---1---0---
* LED: -----|DS4|DS3|DS2|DS1|-
*-----
*/

void main(void) {
    OSCCON = 0b00111000; //500KHz clock speed
    TRISC = 0; //all LED pins are outputs
    LATC = 0;
    OPTION_REG = 0b00000111; //1:256 prescaler for a delay of: (instruction-cycle *
256-counts)*prescaler = ((8uS * 256)*256) =~ 524mS
    LATCbits.LATC4 = 1; //start with DS4 lit

    while (1) {
        while (!INTCONbits.TMR0IF) continue; //you can let the PIC do work here, but for now we will
        wait for the flag
        INTCONbits.T0IF = 0; //flag MUST be cleared in software
        LATC >> = 1; //rotate the LEDs
        if (STATUSbits.C) //when the last LED is lit, restart the pattern
            LATCbits.LATC3 = 1;
    }
}
}

```