ECED 3204 Microprocessor

Midterm Reference Solution

Date: October 26 2017 **Room: B225, B227, B229**
Time: 7:00pm-9:00pm

Student name	
ID	

1) Problem one has following two sub problems:
   a. Write an instruction to add the two 8 bit numbers stored in program memory at 0x100A and 0x100F together and leave the sum in the data memory at 0x2000
   [5points]

```
ldi     ZL,0x0A         ; use Z to point to program memory location 0x100A
ldi     ZH,0x10         ;    "
ldi     YL,0x00         ; use Y to point to data memory location 0x2000
ldi     YL,0x20         ;    "
lpm     r0,Z            ; fetch program memory location 0x100A
lpm     r1,Z+5          ; fetch program memory location 0x100F
add     r1,r0
st      Y,r1
```

   b. Write a sequence of instructions to subtract 100 from data memory locations at 0x200A.
   [5points]

```
ldi     YL,0x0A         ; use Y to point to data memory location 0x200A
ldi     YH,0x20         ;    "
ldi     r20,100         ; place 100 in r20
ld      r0,Y            ; fetch data memory location 0x200A
sub     r0,r20          ; subtract 100 from data memory location 0x200A
st      Y,r0            ; save the difference back to data memory location 0x200A
```

2) Problem two has following two sub problems:
   a) Write a macro to find the square of the value from register R16, and store the results in R17 (MSB Byte) and R18 (LSB byte)
   [5points]

```
.macro      squarer16
Mov @1, @0
mul @0, @1
mov @1, r0
mov @2, r1
.endmacro
```

   b) Invoke the micro defined in a to calculate square of 5
   [5points]

```
ldi r16,5
Squarer16 R16,R17,R18
```

3) Read following program and determine the value of r16, r17 and r18 when the program is done. [10points]

```
        .include    <m2560def.inc>
        .def    count   = r16
        .def    lpcnt   = r17
        .equ    NN      = 40
        .cseg
        .org    0x00
        rjmp    start
        .org    0xF6
start:  ldi     ZL,low(array<<1)
        ldi     ZH,high(array<<1)           "
        ldi     count,0;
        ldi     lpcnt,NN
loop:   cpi     lpcnt,0
        breq    done
        lpm     r18,z+ ;
        andi    r18,0x07        ;
        brne    next    ;
        inc     count   ;
next:   dec     lpcnt   ;
        rjmp    loop
done:   rjmp    done
array:  .db     1,2,3,4,5,6,7,8,9,10
        .db     11,12,13,14,15,16,17,18,19,20
        .db     21,22,23,24,25,26,27,28,29,30
        .db     31,32,33,34,35,36,37,38,39,40
// end of program
```

	r16	r17	r18
binary value	0x05	0x00	0x00

4) Write a C function to test whether an integer between 100 and 1000 is a prime number. [10 points]

```
#include <avr\io.h>
unsigned int prim_num[6];          // array to hold the prime numbers
#define     yes       1
#define     no        0
char isprime(unsigned int x);
unsigned char sqroot(unsigned int y);
void main (void)
{   unsigned  count;
    unsigned int testnum;              // number to be tested
    testnum = 999;           // first number to be tested for prime under 1000
    count = 450;    // total test
    while (count){
            if(isprime(testnum)){ // if the number is prime, then save it
                    prim_num[i] = testnum;
            }
            testnum -= 2;
            count--;
    }
    while(1);
}//-----------------------------------------------------------------------------------------
// This function tests whether the unsigned integer k is a prime number.
char isprime (unsigned int k)
{
    int i;
    char limit;
    if (k < 2) return 0;
    else if(k < 4)
            return 1;
    limit = sqroot(k);        // use the square root as prime test limit
    for (i = 2; i <= limit; i++)
            if ((k % i) == 0) return 0; // divisible?
    return 1;
}
//-----------------------------------------------------------------------------
// This function finds the closest integral square root of the unsigned integer y.
unsigned char sqroot (unsigned int y)
{
    unsigned int i, sar, mask, test;
    sar = 0;
    mask = 0x80;
    for (i = 0; i < 8; i++) {
            test = sar | mask;
            if ((test * test) <= y)
                    sar = test;
            mask = mask >> 1;
    }
    if ((y - sar * sar) < ((sar+1) * (sar+1) - y))
            return sar;
    else return (sar+1);
}
```

5) Execute each following instruction and write out the new Rd value. Assume carry has the value of 1.

[15points]

Rd=11001100		
	New Rd value	New C value
lsl Rd	10011000	1
lsr Rd	01100110	0
rol Rd	10011001	1
ror Rd	11100110	0
asr Rd	11100110	1
swap Rd	11001100	1
inc Rd	11001101	1
push Rd	11001100	1
ser Rd	11111111	1
dec Rd	11001011	1

6) What are the contents of PORTA after the following AVR program is assembled:

[5points]

```
ldi     r16, 1<<1
andi    r16, 0x0F
out     PORTA, r16
```

PortA=0x02

7) Write an ASM subroutine to calculate S where $S = \sum_{k=1}^{100} k$

[10points]

```
.include "m644Adef.inc"
            .def   lpcnt  = r17
            .def   SUMH   = r25
            .def   SUML   = r24

            .equ   NN     = 100
            .cseg
            .org   0x00
            rjmp   start
            .org   0xF6
start:      ldi    lpcnt,NN
            ldi     r24,0
            ldi    r25,0  ;
loop:       cpi    lpcnt,0
            breq   done

            add    suml,lpcnt    ;
            ldi    r16,0
            adc    sumh,r16
            dec    lpcnt  ;
            rjmp   loop
done:       rjmp   done
```

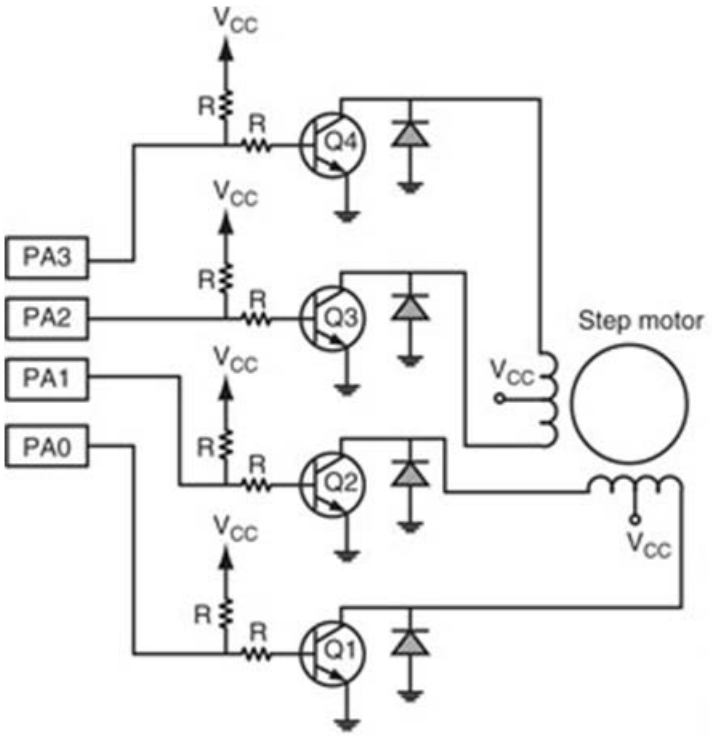
8) Write an ASM subroutine to find numbers of elements in a program array that are odd integers.

[10points]

```
.include    <m2560def.inc>
    .def    count   = r16
    .def    lpcnt   = r17
    .equ    NN      = 40
    .cseg
    .org    0x00
    rjmp    start
    .org    0xF6
start:      ldi     ZL,low(array<<1)
    ldi     ZH,high(array<<1)           "
    ldi     count,0;
    ldi     lpcnt,NN
loop:       cpi     lpcnt,0
    breq    done
    lpm     r18,z+  ;
    andi    r18,0x01        ;
    brne    next    ;
    inc     count   ;
next:       dec     lpcnt   ;
    rjmp    loop
done:       rjmp    done
array:      .db     1,2,3,4,5,6,7,8,9,10
            .db     11,12,13,14,15,16,17,18,19,20
            .db     21,22,23,24,25,26,27,28,29,30
            .db     31,32,33,34,35,36,37,38,39,40
// end of program
```

9) Write a subroutine that rotates the stepper motor shown in figure counterclockwise in full steps. Assume the subroutine for delay is available **delayby10ms**

[10points]



The subroutine to rotate the stepper motor counterclockwise full-step is as follows:

```
        .include <m644Adef.inc>

        .equ    FSTEP1 = 0x0C   ; values to rotate a full step
        .equ    FSTEP2 = 0x06   ;    "
        .equ    FSTEP3 = 0x03   ;    "
        .equ    FSTEP4 = 0x09   ;    "

FSccw:  ldi     r20,FSTEP4
        out     PORTA,r20
        rcall   wait10ms
        ldi     r20,FSTEP3
        out     PORTA,r20
        rcall   wait10ms
        ldi     r20,FSTEP2
        out     PORTA,r20
        rcall   wait10ms
        ldi     r20,FSTEP1
        out     PORTA,r20
        rcall   wait10ms
        ret
```

10) Configure even interrupt pins (INTn) to interrupt on the rising edge and the odd interrupt pins to interrupt on the falling edge.

[10points]

```
ldi     r16,0xBB      ; configure odd interrupt pins to interrupt on the rising edge
sts     EICRA,r16     ; and even interrupt to interrupt on the falling edge
sts     EICRB,r16     ;   “
ldi     r16,0xFF      ; enable all INTn pin to interrupt locally
sts     EIMSK,r16     ;   “
sei                   ; enable global interrupt
```

Bonus question:

1) Which of following timer is 8 bit timer? [1 point]

a) timer 1 **b) timer 2** c) timer 3

2) Which of following timer has input capture function? [1 point]

a) timer 0 **b) timer 1** c) timer 2

3) How many interrupt sources does timer 1 have? [1 point]

a) 3 b) 4 **c) 5**

4) How many operation modes does timer 1 have? [1 point]

b) 8 b) 16 **c) 5**

5) What can CTC mode do? [1 point]

a) measure duty cycle b) measure signal period **c) create delay**