CS273 HW 5 - Ziad Arafat

1. Without a status register we can't check if the last operation was a zero or negative or positive or overflow etc. So operations like Branching that rely on the specific results of the previous math operation or comparison won't work making most logic near impossible.

2. CZVSN

    1. S:1, C:1, Z:0, V:0, H:1, N:1
    2. S:0, C:1, Z:0, V:0, H:1, N:0

3. R1, R0

    1. R1: 0xfa, R0: 0x05
    2. R1: 0x00, R0: 0x80
    3. R1: 0x8b, R0: 0xBA

4. CODE:

```
; POS/POS: 3 1
; NEG/POS:-3 -1
; POS/NEG:-3 1
; NEG/NEG: 3 -1
; R16 numerator
; R17 denomonator
; R18 Case: 0, 1, 2, 3
; R22 Quotient
; R20 Remainder

.text

divide:
    lds R16, A // load numerator
    lds R17, B // load denomonator
    clr R18    // set case to 0

isnumneg: // is the numerator negative?
    cpi R16, 0
    brlt numneg // it is!
    jmp isdenomneg

numneg: // The numberator is negative
    neg R16
    ldi R18, 1 // set case to 1
    jmp isdenomneg // is the denom negative?

isdenomneg: // is denom negative?
    cpi R17, 0
    brlt denomneg // IT IS
```

```
    brit denomneg // if is
    cpi R18, 1    // check if we were in case 1
    breq posneg   // if we were then it's posneg
    jmp pospos    // otherwise it was pospos

denomneg:
    neg R17
    cpi R18, 0    // are we in case 0?
    breq negpos   // If so then it's negpos
    jmp negneg    // else it's negneg


// here we make sure our case is set then jump to division.
pospos:
    ldi R18, 0
    jmp division
negpos:
    ldi R18, 1
    jmp division
posneg:
    ldi R18, 2
    jmp division
negneg:
    ldi R18, 3
    jmp division

division:
    lds R20, R16
    lds R21, R17
    clr R22
L1:
    inc R22
    sub R20, R21
    brcc L1

    dec R22
    add R20, R21

// check the case and negate the quotient/remainder accordingly.
chkcase:
    cpi R18, 0
    breq case0
    cpi R18, 1
    breq case1
    cpi R18, 2
    breq case2
    cpi R18, 3
    breq case3

; POS/POS: 3 1
; NEG/POS:-3 -1
; POS/NEG:-3 1
; NEG/NEG: 3 -1
case0:
    jmp done
```

```
    jmp done
case1:
    neg R22
    neg R20
    jmp done
case2:
    neg R22
    jmp done
case3:
    neg R20

done:
    ret
```

5. CODE:

```
; R18: x
; R19: y
; R20: 2
; R21: 5
; R22: 7

conditionals:
    ldi R18, 5
    ldi R19, -5

    cp R18, R19
    breq elseif
    cp R18, R19
    breq


    cp R19, R18
    brlt if
    dec R19
    dec R19
    dec R19
    cp R19, R18
    brlt elseif
    jmp else

if:
    ldi R20, 2
    add R18, R20
    lds R19, R18
    jmp done
elseif:
    ldi R21, 5
    add R18, R21
    lds R19, R18
    jmp done
else:
```

```
        ldi R22, 7
        add R18, R22
        lds R19, R18

    done:
        ret
```