Jose Franco Baquera

April 13, 2018

CS – 273

Homework #9

1. **Decode the following AVR binary code located in the program memory. What are the registers involved? What are their values before and after the program finishes? You will need to pay attention to the endianess of the coding. (The dashes are not part of the code but are added to increase readability)**

Starting program memory address is 0x0100

	LOW BYTE | HIGH BYTE
0x0100:	0010-0001-1110-0000
0x0101:	0011-0001-1110-0000
0x0102:	0100-1000-1110-0000
0x0103:	0011-0010-0000-1111
0x0104:	0010-0011-0010-1111
0x0105:	0100-1010-1001-0101
0x0106:	1110-0001-1111-0111

Answer:

```
# Here is the decoded AVR program:
LDI R18, 1
LDI R19, 1
LDI R20, 8
LOOP: ADD R19, R18
MOV R18, R19
DEC R20
BRNE LOOP
```

The registers involved are R18, R19, and R20. For this AVR assembly code, R20 acts as a counter for the BRNE loop. Before the program finishes, R18 = 1, R19 = 1, and R20 = 8. After the program finishes, R18 = 0, R19 = 0, and R20 = 0.

Before the program finishes:

R18 = 1

R19 = 1

R20 = 8

After the program finishes:

R18 = 0

R19 = 0

R20 = 0

*(NOTE: R18 and R19 contain 0 because after the loop ends, we add up to 256, which means that **overflowed** occurred since registers can only hold a maximum value of 255 unsigned. The number 256 is represented as 0 by the registers.)*

2. **Indicate the value loaded into R30, R31, and R20 in the following program:**

```
 .ORG 0x0
        LDI R30, LO8(OUR_DATA )
        LDI R31, HI8(OUR_DATA)
        LPM  R20, Z
.ORG 0x0524
OUR_DATA: .byte 20, '$', '5'
```

Answer:

```
R30 = 0x24

R31 = 0x05

R20 = 0x14 (or Decimal: 20).
```

3. **Write a program to read the following message from program ROM and place it in data RAM starting at 0x200:**

.ORG 0x0500

MYDATA: .asciz "Will artificial intelligence rule human?"

Answer:

```
.data

.org 0x0200
```

```
final_string: .byte 40 ;Dummy value
.text
.global readmessage
readmessage:
ldi r30, lo8(MYDATA≪1)
ldi r31, hi8(MYDATA≪1)
ldi r26, lo8(final_string) ;Could have also done LDI r26, lo8(0x200)
ldi r27, hi8(final_string) ;Could have also done LDI r27, hi8(0x200)
loop: lpm r18, Z+
st X+, r18
cpi r18, 0
brne loop
ret
.org 0x0500
MYDATA: .asciz “Will artificial intelligence rule human?”
```

****NOTE: The program above will also copy the ending zero of the string (i.e. it will also copy the null-terminated symbol). If we didn’t want this zero, we would have to modify the loop slightly.*

4. **Write a program that calculates the checksum of the values at location 0x00D5 to 0x0300 of EEPROM.**

```
.data
# No data segment for this program.
.text
.global programstart
programstart:
clr r27 ;Clear register for checksum
ldi r24, 0xD5 ;Load the low and high bytes of the starting address
```

```
ldi r25, 0x00
loop:
sbic EECR, EEWE ;Check EWEE to see if last write is finished
rjmp loop
out EEARL, r24 ; Load low byte of EEPROM address
out EEARH, r25 ; Load high byte of EEPROM address
sbi EECR, EERE ;Set read enable to 1
in r26, EEDR ;Load EEPROM data register into r26
add r27, r26 ;Add the value read into the checksum register
addiw r25:r24, 1 ;Add 1 to EEPROM address
cpi r24, 0x00 ;Compare low byte to target address low byte
brne loop
cpi r25, 0x03 ;Compare high byte to target address high byte
brne loop
end:
neg r27 ;Do the negation of the checksum
ret
```

5. **In each of the following cases perform checksum calculation to see if data is corrupted or not.**

   a. **Data=$62, $F3, and $15; checksum=$72**

      0x62 + 0xF3 + 0x15 + 0x72 = **0x1DC** (NOTE: We ignore drop the carry.)

      Since 0xDC does not equal to 0x00, **data IS corrupted.**

   b. **Data=$50, $88, $3C, and $8E; checksum=$6D**

      0x50 + 0x88 + 0x3C + 0x8E + 0x6D = **0x20F** (NOTE: We drop the carry.)

      Since 0x0F does not equal to 0x00, **data IS corrupted.**

c. Data=0, 0, 0, 0, 0, 0; checksum=0

0 + 0 + 0 + 0 + 0 + 0 + 0 = **0x00**

Since 0x00 equals to 0x00, **data is NOT corrupted.**

d. Data=1,-1,1,-1,1,-1; checksum=1

1 – 1 + 1 – 1 + 1 – 1 + 1 = **0x01**

Since 0x01 does not equal to 0x00, **data IS corrupted.**