(5) Question 1. Using the basis elements, %10110010 = -128+32+16+2 = -128+50 = -78

(5) Question 2. ROM, EEPROM and flash EEPROM are nonvolatile.

(5) Question 3. What will be the value of the carry (C) bit?
Step 1: make sure both numbers are in unsigned 8-bit format, they are in format already
Step 2: perform addition, 110+140=250
Step 3: check to see if the result fits into 8-bit unsigned format, it fits so C=0

(5) Question 4. What will be the value of the overflow (V) bit?
Step 1: make sure both numbers are in signed 8-bit format, they are in format already
Step 2: perform subtraction, -90- -100 = -90+100 = 10
Step 3: check to see if the result fits into 8-bit signed format, it fits so V=0

(5) Question 5. $A5 = %10100101.
Part a) Shift right maintaining the sign bit. Register A=%11010010 = $D2
Part b) The C bit contains the least significant bit that is shifted out, C=1

(5) Question 6. The reset vector is a 16-bit address at $FFFE.

```
    org  $FFFE
    fdb  $6000
```

(10) Question 7. The PC will be $4122 after the opcode and operand are fetched. $F0 will be sign-extended to $FFF0. This means -16. $4122+$FFF0 = $4112

```
$4120 07F0          bsr MyFunction
```

R/W	Addr	Data	Changes to A,B,X,Y,S,PC,IR,EAR
1	$4120	$07	IR=$07, PC=$4121
1	$4121	$F0	PC=$4122 (return address)
0	$3FF3	$22	Push return address, SP=$3FF3
0	$3FF2	$41	SP=$3FF2, PC=$4112

(20) Question 8.
Part a) (friendly)

```
    bset  DDRT,#$10   ; Set bit 4 DDRT so PT4 is an output
    bclr  DDRT,#$02   ; Clear bit 1 DDRT so PT1 is an input
```

or (friendly)

```
    ldaa  DDRT
    oraa  #$10   ; Set bit 4 DDRT so PT4 is an output
    anda  #$FD   ; Clear bit 1 DDRT so PT1 is an input
    staa  DDRT
```

or (not friendly)

```
    ldaa  #$10   ; Set bit 4 DDRT so PT4 is an output
    staa  DDRT   ; Clear bit 1 DDRT so PT1 is an input
```

Part b) (friendly)

```
     brset PTT,#$02,next  ; test bit 1, skip over if PT1==1
     bset  PTT,#$10       ; set bit 4 of PTT so PT4=1
next
```

or (friendly)

```
     ldaa PTT   ; read PTT
     anda #$02  ; test bit 1, skip over if PT1==1
     bne  next  ; not equal to zero, if PT1==1
     ldaa #$10
     oraa PTT   ; friendly
     staa PTT   ; set bit 4 of PTT so PT4=1
next
```

or (not friendly)

```
     ldaa PTT   ; read PTT
     anda #$02  ; test bit 1, skip over if PT1==1
     bne  next  ; not equal to zero, if PT1==1
     ldaa #$10
     staa PTT   ; set bit 4 of PTT so PT4=1
next
```

(20) Question 9.

```
;*****Add1 subroutine*********************
;Inputs:  RegA is the first number
;         RegB is the second number
;Outputs: RegA is the sum of the first+second numbers
;         RegA is returned as 255 if an unsigned overflow occurs
 Add1  aba          ;RegA=RegA+RegB
       bcc  done
       ldaa #$255   ;unsigned overflow
done   rts
```

(20) Question 10. This is an if-then-else structure.

```
       ldd  Input       ;bring in first number
       cpd  #-100       ;subtract second number
       blt  less        ;skip if Input < -100
       movw #0,Output   ;Input >= -100
       bra  done
less   movw #200,Output ;Input < -100
done
```