(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 $FF0. This means -16. $4122 + $FF0 = $4112

```
$4120  07F0   bsr MyFunction

<table>
<thead>
<tr>
<th>R/W</th>
<th>Addr</th>
<th>Data</th>
<th>Changes to A, B, X, Y, S, PC, IR, EAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$4120</td>
<td>$07</td>
<td>IR = $07, PC = $4121</td>
</tr>
<tr>
<td>1</td>
<td>$4121</td>
<td>$F0</td>
<td>PC = $4122 (return address)</td>
</tr>
<tr>
<td>0</td>
<td>$3FF3</td>
<td>$22</td>
<td>Push return address, SP = $3FF3</td>
</tr>
<tr>
<td>0</td>
<td>$3FF2</td>
<td>$41</td>
<td>SP = $3FF2, PC = $4112</td>
</tr>
</tbody>
</table>
```

(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
ora #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)

\[
\begin{align*}
\text{brset PTT,}\#\$02,\text{next} &; \text{test bit 1, skip over if PT1==1} \\
\text{bset PTT,}\#\$10 &; \text{set bit 4 of PTT so PT4=1} \\
\text{next}
\end{align*}
\]

or (friendly)

\[
\begin{align*}
\text{ldaa PTT} &; \text{read PTT} \\
\text{anda}\#\$02 &; \text{test bit 1, skip over if PT1==1} \\
\text{bne next} &; \text{not equal to zero, if PT1==1} \\
\text{ldaa}\#\$10 & \\
\text{ora PTT} &; \text{friendly} \\
\text{staa PTT} &; \text{set bit 4 of PTT so PT4=1} \\
\text{next}
\end{align*}
\]

or (not friendly)

\[
\begin{align*}
\text{ldaa PTT} &; \text{read PTT} \\
\text{anda}\#\$02 &; \text{test bit 1, skip over if PT1==1} \\
\text{bne next} &; \text{not equal to zero, if PT1==1} \\
\text{ldaa}\#\$10 & \\
\text{staa PTT} &; \text{set bit 4 of PTT so PT4=1} \\
\text{next}
\end{align*}
\]

(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.

\[
\begin{align*}
\text{ldd Input} &; \text{bring in first number} \\
\text{cpd}\#-100 &; \text{subtract second number} \\
\text{blt less} &; \text{skip if Input} < \text{-100} \\
\text{movw} #0,\text{Output} &; \text{Input} \geq \text{-100} \\
\text{bra done} & \\
\text{less} \ movw \#200,\text{Output} &; \text{Input} < \text{-100} \\
\text{done}
\end{align*}
\]