This is a closed book exam. You must put your answers on this piece of paper only. You have 50 minutes, so allocate your time accordingly. *Please read the entire quiz before starting.*

(5) **Question 1.** Specify A, B, C, D, or E ….

(5) **Question 2.** Answer the decimal value….

(4) **Part 3a.** Specify `RegA`………………..

(2) **Part 3b.** Specify 0 or 1 ………………..

(2) **Part 3c.** Specify 0 or 1 ………………..

(2) **Part 3d.** Specify 0 or 1 ………………..

(2) **Part 3e.** Specify 0 or 1 ………………..

(10) **Question 4.** Specify the range………..

(10) **Question 5.** Show the machine code….

(5) **Question 6.** What is the EA?………..

(15) **Question 7.** Simplified memory cycles (you may or may not need all 5 entries)

<table>
<thead>
<tr>
<th>R/W</th>
<th>Addr</th>
<th>Data</th>
<th>Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
(40) Question 8. Write the two assembly language subroutines.

DDRM equ $0252  ; Port M Direction
DDRT equ $0242  ; Port T Direction
PTM  equ $0250  ; Port M I/O Register
PTT  equ $0240  ; Port T I/O Register
org $3800  ; RAM

org $4000  ; ROM
aba 8-bit add RegA+RegB
abx unsigned add RegX+RegB
aby unsigned add RegY+RegB
adca 8-bit add with carry to RegA
adcb 8-bit add with carry to RegB
adda 8-bit add to RegA
addb 8-bit add to RegB
add 16-bit add to RegD
anda 8-bit logical and to RegA
andb 8-bit logical and to RegB
andcc 8-bit logical and to RegCC
asl/lsl 8-bit left shift Memory
asla/lsla 8-bit left shift RegA
aslb/lslb 8-bit arith left shift RegB
asld/lsld 16-bit left shift RegD
asr 8-bit arith right shift Memory
asra 8-bit arith right shift
asrb 8-bit arith right shift to RegB
bcc branch if carry clear
bclr clear bits in memory
bcs branch if carry set
beq branch if result is zero (Z=1)
bge branch if signed ≥
bgnd enter background debug mode
bgt branch if signed >
bhi branch if unsigned >
bhs branch if unsigned =
bita 8-bit and with RegA, sets CCR
bitb 8-bit and with RegB, sets CCR
ble branch if signed ≤
blt branch if unsigned <
bls branch if unsigned =
bit branch if signed <
bmi branch if result is negative (N=1)
bne branch if result is nonzero (Z=0)
bpl branch if result is positive (N=0)
bra branch always
brclr branch if bits are clear,
brn branch never
brset branch if bits are set
bset set bits in memory
bsr branch to subroutine
bvc branch if overflow clear
bvs branch if overflow set
call subroutine in expanded memory
cba 8-bit compare RegA with RegB
clc clear carry bit, C=0
cli clear I=0, enable interrupts
clr 8-bit Memory clear
cira RegA clear
cirb RegB clear
clv clear overflow bit, V=0
cmpa 8-bit compare RegA with memory
cmpb 8-bit compare RegB with memory
com 8-bit logical complement to Memory
com 8-bit logical complement to RegA
comb 8-bit logical complement to RegB
cpd 16-bit compare RegD with memory
cpx 16-bit compare RegX with memory
cpy 16-bit compare RegY with memory
da 8-bit decimal adjust accumulator
dba 8-bit add RegA+RegB
dbx unsigned add RegX+RegB
dby unsigned add RegY+RegB
deac 8-bit decrement RegA
dec 8-bit decrement memory
decb 8-bit decrement RegB
des 16-bit decrement RegSP
dex 16-bit decrement RegX
dey 16-bit decrement RegY
deq 8-bit logical exclusive or to RegA
deqb 8-bit logical exclusive or to RegB
lsr 8-bit logical right shift memory
lsra 8-bit logical right shift RegA
lsrb 8-bit logical right shift RegB
lsrd 16-bit logical right shift RegD
maxa 8-bit unsigned maximum in RegA
maxm 8-bit unsigned maximum in memory
mina 8-bit unsigned minimum in RegA
minm 8-bit unsigned minimum in memory
movb 8-bit move memory to memory
movw 16-bit move memory to memory
mul RegD=RegA*RegB
neg 8-bit 2’s complement negate memory
nega 8-bit 2’s complement negate RegA
negb 8-bit 2’s complement negate RegB
oraa 8-bit logical or to RegA
orab 8-bit logical or to RegB
orcc 8-bit logical or to RegCC
psha push 8-bit RegA onto stack
pshb push 8-bit RegB onto stack
pshc push 8-bit RegCC onto stack
pshd push 16-bit RegD onto stack
pshx push 16-bit RegX onto stack
pshy push 16-bit RegY onto stack
pula pop 8 bits off stack into RegA
pulb pop 8 bits off stack into RegB
pulc pop 8 bits off stack into RegCC
puld pop 16 bits off stack into RegD
pulx pop 16 bits off stack into RegX
puly pop 16 bits off stack into RegY
rev Fuzzy logic rule evaluation
revw weighted Fuzzy rule evaluation
rol 8-bit roll shift left Memory
rola 8-bit roll shift left RegA
rolb 8-bit roll shift left RegB
ror 8-bit roll shift right Memory
rorh 8-bit roll shift right RegA
rorb 8-bit roll shift right RegB
rtc return sub in expanded memory
rti return from interrupt

**ANDB**

**Logical AND B**

<table>
<thead>
<tr>
<th>Source Form</th>
<th>Address Mode</th>
<th>Object Code</th>
<th>Cycles</th>
<th>Access Detail</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANDB RopXli</td>
<td>IMM</td>
<td>C4 il</td>
<td>1</td>
<td>P</td>
</tr>
<tr>
<td>ANDB opr8a</td>
<td>DIR</td>
<td>D4 dd</td>
<td>3</td>
<td>rFP</td>
</tr>
<tr>
<td>ANDB opr16a</td>
<td>EXT</td>
<td>F4 hh 1l</td>
<td>3</td>
<td>rOP</td>
</tr>
<tr>
<td>ANDB opr0_XYSp</td>
<td>IDX</td>
<td>E4 xb</td>
<td>3</td>
<td>rFP</td>
</tr>
<tr>
<td>ANDB opr9,XYSp</td>
<td>IDX1</td>
<td>E4 xb ff</td>
<td>3</td>
<td>rPO</td>
</tr>
<tr>
<td>ANDB opr16,XYSp</td>
<td>[IDX2]</td>
<td>E4 xb ee ff</td>
<td>4</td>
<td>frFP</td>
</tr>
<tr>
<td>ANDB [D,XYSp]</td>
<td>[D,IDX]</td>
<td>E4 xb</td>
<td>6</td>
<td>fIPfP</td>
</tr>
<tr>
<td>ANDB [oprx16,XYSp]</td>
<td>[IDX2]</td>
<td>E4 xb ee ff</td>
<td>6</td>
<td>fIDPrFP</td>
</tr>
</tbody>
</table>
(5) **Question 1.** An integer divide sometimes causes a/an ______ error because information is lost in the least significant bits. Fill in the blank

A) overflow  B) underflow  C) drop out  D) floor  E) ceiling

(5) **Question 2.** What is signed integer value (in decimal) of the 8-bit binary number %11010011?

**Question 3.** Consider the result of executing the following two 6812 assembly instructions.

```assembly
ldaa #101
adda #210
```

(4) **Part a)** What is the value in Register A after these two instructions are executed? Give your answer as an unsigned decimal.

(2) **Part b)** What will be the value of the carry (C) bit?

(2) **Part c)** What will be the value of the overflow (V) bit?

(2) **Part d)** What will be the value of the zero (Z) bit?

(2) **Part e)** What will be the value of the negative (N) bit?

(10) **Question 4.** What range of values can be represented an 8-bit signed binary fixed-point format with a resolution, \(\Delta\), of 1/8? Give the smallest and the largest values.

(10) **Question 5.** Show the machine code generated by the instruction

```assembly
andb -2, y
```

(5) **Question 6.** Assume Reg X is initially $3A00, Reg Y is initially $3900, what is the effective address of this instruction

```assembly
leax 2, y-
```

(15) **Question 7.** Show the simplified bus cycles that occur when the bsr sub instruction is executed. In the “changes” column, specify which registers get modified during that cycle, and the corresponding new values. Just show the one instruction.

```assembly
$F120 CF4000 [ 2]( 0){OP }main lds #$4000
$F123 87 [ 1]( 2){O } clra
$F124 0766 [ 4]( 3){PPPS }loop bsr sub
$F126 20FC [ 3]( 7){PPP } bra loop
```

***************some stuff in here ***************

; Purpose: increment a number  
; Input: RegA, range 0 to 255  
; Output: RegA=Input+1  
; Errors: overflow if 255

```assembly
$F18C 42 [ 1]( 10){O }sub inca
$F18D 3D [ 5]( 11){UfPPP} rts
$FFFE org $fffe
$FFFE F120 fdb main
```

(40) **Question 8.**

Part a) Write an assembly language subroutine that sets all of Port T to outputs and all of Port M to inputs. Basically, this function sets the appropriate direction registers.

Part b) Write a second assembly language subroutine that sets Port T to $55 if the value of Port M is less than $20 (treated as an unsigned number). Conversely, if the value of Port M is greater than or equal to $20, then Port T is unchanged. Do this once then return from the subroutine.