Spring 2017
Final Quiz

UF ID#_____

Open book/open notes, **90-minute** exam. ***No electronic devices permitted.***

Point System: Page 1 => 4 points Veronica | Page 2 => 10 points Daniel #2 / Cody #3

Page 3 => 6 points Kevin #4 / Alan #5 | Page 4 => 10 points Lucas

Page 5 => 15 points 6-9 Aaron / #10 Casey | Page 6 => 5 points Caleb

TOTAL => _____ (50 pt.)

Grade Review Information: (*NOTE: deadline of request for grade review is the day the exam is returned.*)

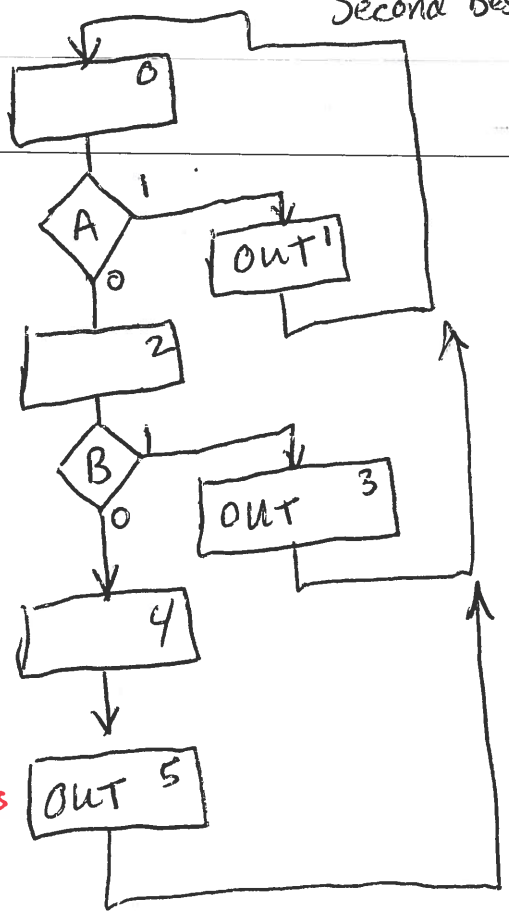
Total Pts. → Canvas Kyle, Wyatt

You are given a **1 KHz** clock for the following problem. Design a state machine that creates an output (**OUT.H**) based on three input variables: **A.L**, **B.L** and **C.L** ***Assume ONE and ONLY one of the three INPUT variables will be TRUE at any given time.***

If **A = T**, output a **500 Hz 50% Duty Cycle** square wave. Else If **B = T**, output a **333.3 Hz 33.3% Duty Cycle** square wave. Else if **C = T** output a **250 Hz 25% Duty Cycle** square wave. Recall: **Frequency = 1/Period** and **Duty Cycle = 100% * (Time High/Period).** Best Design = Most pts.

1. Draw the **Flow Chart** below for the State Machine. Don't forget to ***number your states!*** (4 pt.).

Second Best!

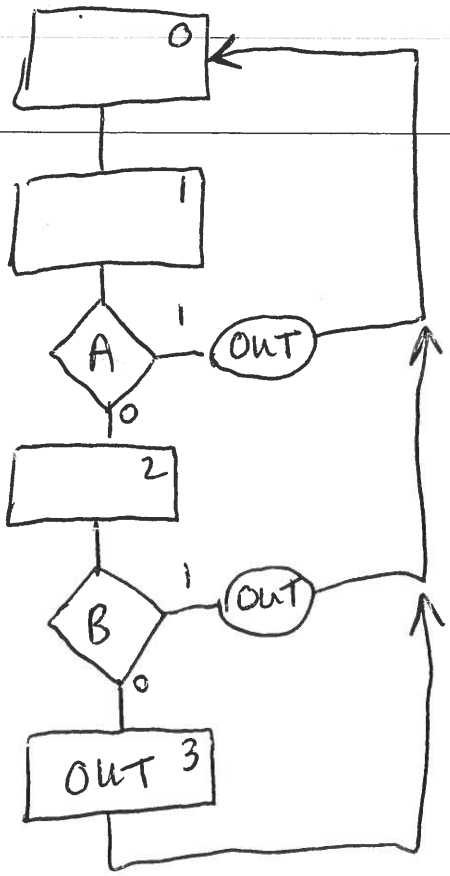


inverted duty cyle -0.5

Crazy outputs -0.5

Extra State -1
No Out.H -0.5 in state
no states between -1

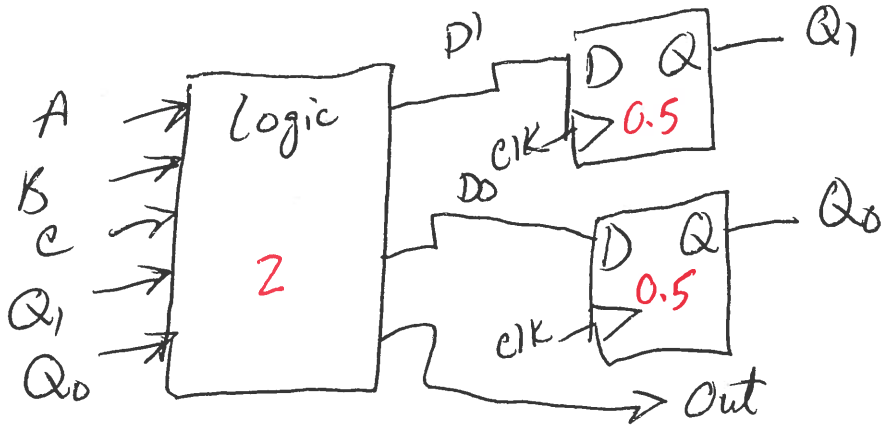
Best!



Pg. Score =

Key

2. Draw the **Functional Block Diagram** for the system assuming that **logic gates** & **D flip-flops** will be used in the implementation. ***Label all signals***. (3 pt.)



3. Fill in the **Next State Table** below and use only the Present and Next States required in your design (7 pt.)

Note: You may assume that we have created a **reset circuit** to always **start** in the first state '**State 0**'.

	A.L	B.L	C.L	Q3.H	Q2.H	Q1.H	Q0.H	D4.H	D3.H	D2.H	D1.H	D0.H	OUT.H
1	X	X	X			0	0				0	1	0
1	1	X	X			0	1				0	0	1
1	0	X	X			0	1				1	0	0
1	X	1	X			1	0				0	0	1
1	X	0	X			1	0				1	1	0
1	X	X	1			1	1				0	0	1
1	X	X	0			1	1				0	0	0

1 also ok

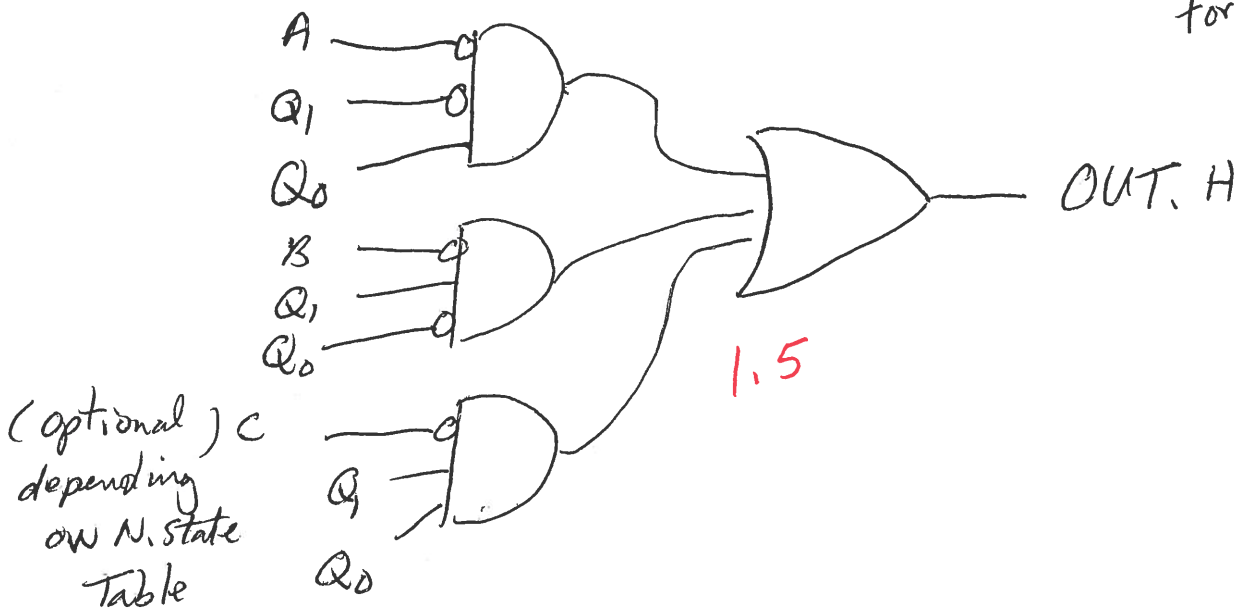
Page Score =

4. Create a **Logic Equation (SOP is OK!)** & **circuit** for **OUT.H** using inputs **A.L, B.L, C.L** and **Qx:Q0.** (3 pt.)

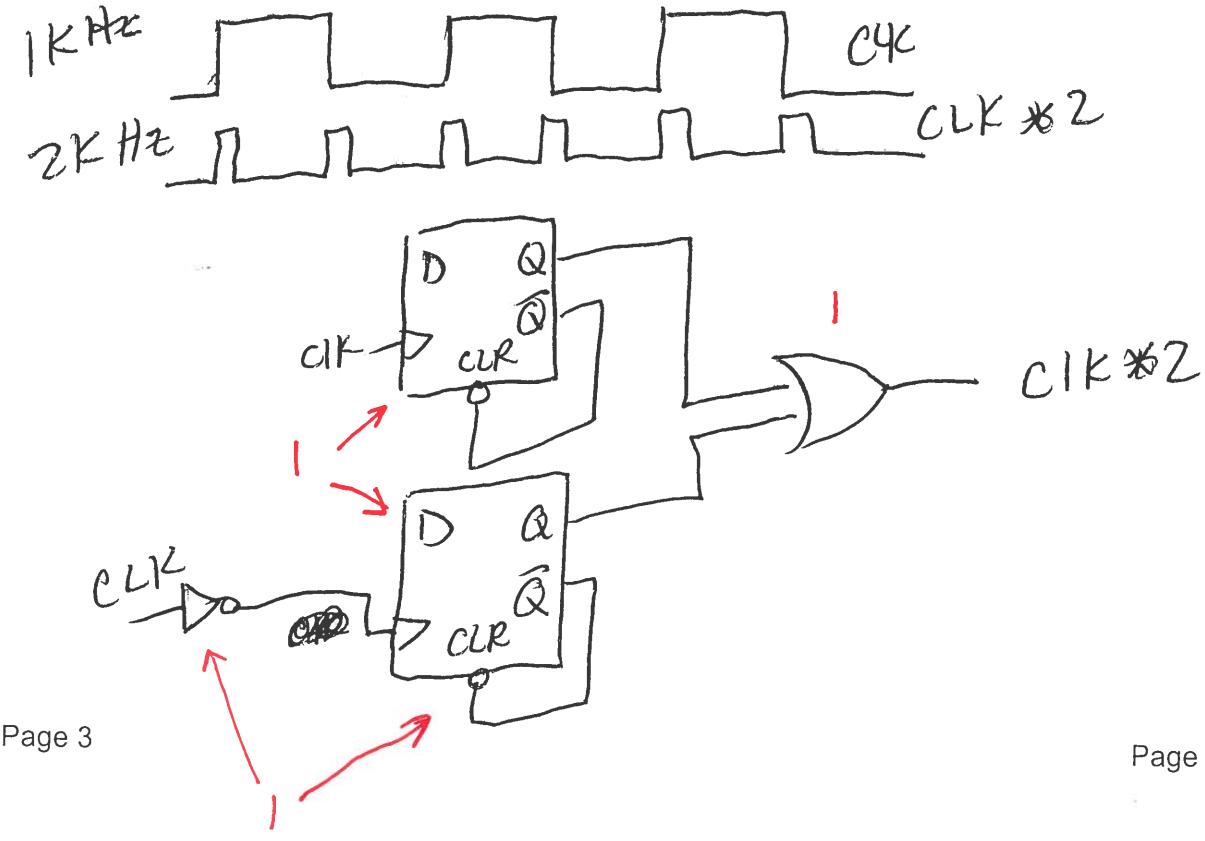
$$Out = A\bar{Q}_1Q_0 + BQ_1\bar{Q}_0 + CQ_1Q_0$$

1.5

{ goes away CQ_1Q_0 for inputs 011



5. Suppose we desired OUT.H to run at twice it's current frequency, to do this we would need to create a clock that is twice the original 1KHz clock. Show how to ***create a 2 KHz clock from the original 1 KHz clock***. Hint: Use the asynchronous clear and/or set mechanism in a D flip-flop. (3 pt.)



Page Score =

Key

5. A vector is stored in 4Kx8 RAM at address **$1000.** It consists of **37_{10} signed numbers that are each two bytes in length**. Write a GCPU program to count the number of positive numbers in the vector. Write your ***final sum to the address immediately after the vector*** stored in RAM. Use **X as a pointer to the data** and **use address $1FFF for your loop counter.** (10 pt.)

37_{10} = 37 words = 74 bytes

74 = 4A Hex 0-73 = 74 byt

37_{10} = 25 Hex ↑ 49 Hex

+1 Ldaa #37 ($25) ; Loop Counter

staa $1FFF

Ldx #$1000 ; data ptr

+1 Top: Ldaa 1,X ; get data

+1 BN SKIP ; check if positive

Ldaa $4A } ; inc Positive counter

Ldab #1 } 2

SUMBA

STAA $4A

$4A wrong -0.5

+1 Ldab #0 ; Zero Pos. counter

staa $4A

Skip: INX } ; inc ptr to next word

INX } +1

Ldaa $1FFF ; Loop Counter = loop counter - 1

Ldab #$FF } +2

SUMBA

STAA $1FFF

BNE TOP +1

DONE

Page Score =

Key

For the next set of questions, consult the G-CPU code in **Appendix A.**

6. What is the effective address for the LDY #$0 instruction? 0006 **Hex** (1 pt.) 6, 7 ok

7. What is the effective address for the STAA 0,X instruction the 1st time it is executed?

1000 **Hex** (1 pt.)

8. How many RAM memory locations are modified by this program? 13 **Decimal** (2 pt.) 0xD

9. What are the values in RAM when the program has completed? (2 pt.)

$1000 = 8B Hex $1001 = FB Hex

88 or 03
1000 1000
0000 0011
8 B

83
1111 0011
1000 1000
F B

10. In Appendix A, we are now using a ***SLOW ROM*** to fetch code. This ROM takes ***2 cycles per memory fetch***. i.e. ***2 clock periods instead of 1 to fetch a byte out of memory.*** Fill in the cycle table below for the ***1st loop pass*** for the lines of code shown below assuming that ***2 cycles instead of one will be required to fetch a memory byte out of ROM***. RAM still has 1 cycle R/W to/from memory. Label simply "**NEW**" for new states in the controller ASM below. (9 pt.)

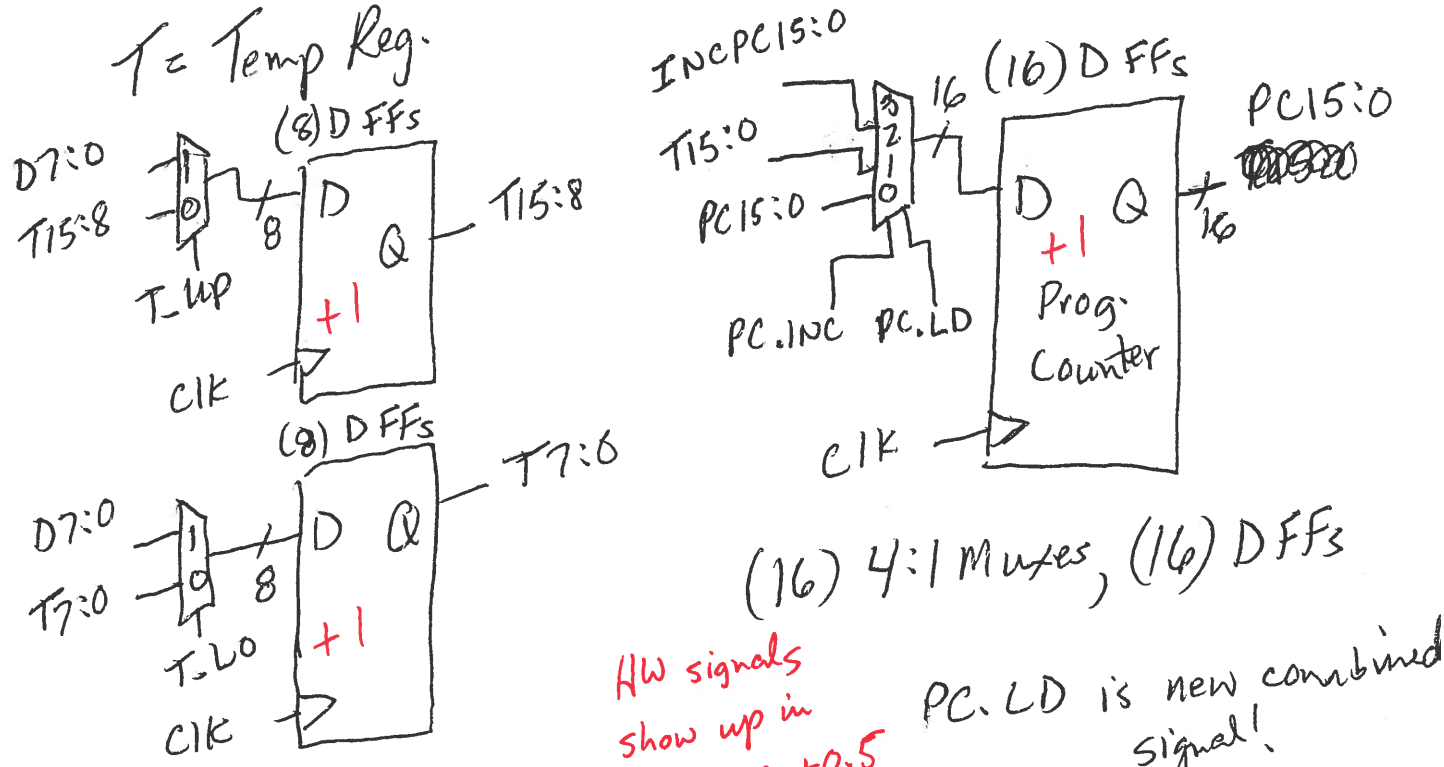
```
OR_BA
STAA    0,X
```
;From Appendix A, show execution of the 1st pass of this code

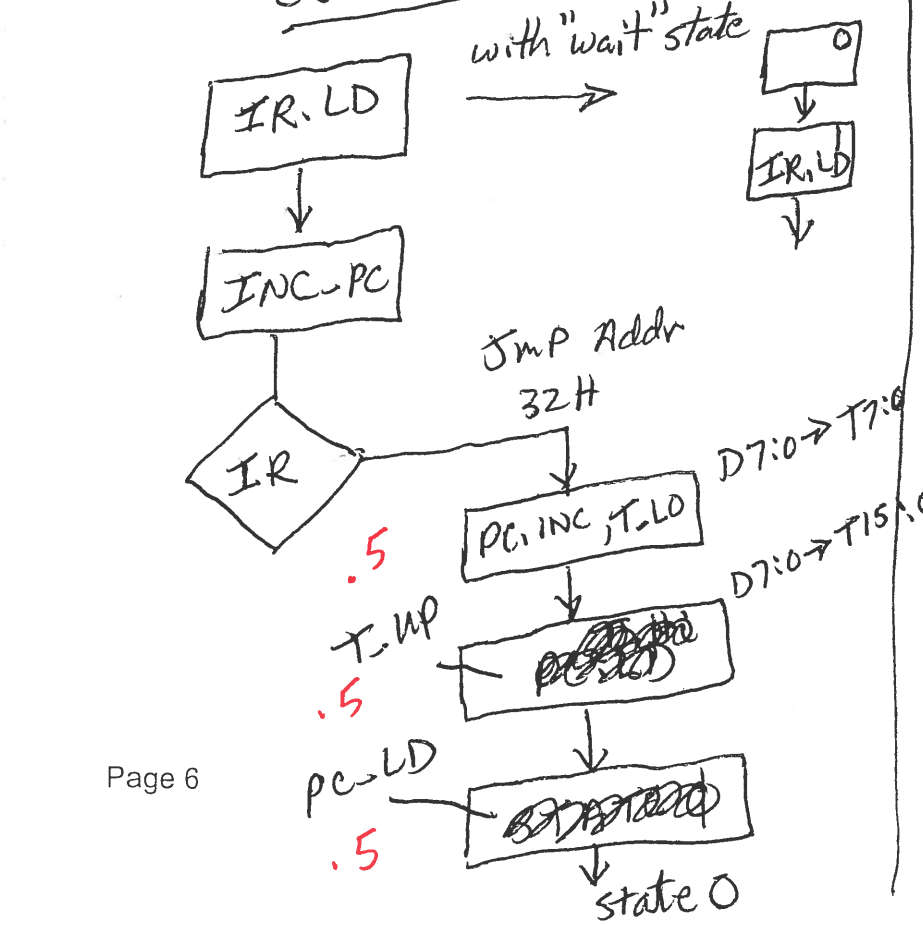
Cycle	Controller State	Addr Bus	Data Bus	IR	R/-W	A Reg	Address Mux Sel1:0	Dev Outputting To Data Bus
1	0	000F	XX	0C	1	03	0 PC	Rom
2	New	000F	18	DC	1	03	0 PC	Rom
3	1	000F	18	18	1	03	0 PC	Rom
4	0	0010	XX	18	1	8B	0 PC	Rom
5	New	0010	10	18	1	8B	0 PC	Rom
6	1	0010	10	10	1	8B	0 PC	Rom
7	24	0011	XX	10	1	8B	PC	Rom
8	New	0011	00	10	1	8B	PC	Rom
9	25	1000	8B	10	0	8B	2 MAR	CPU

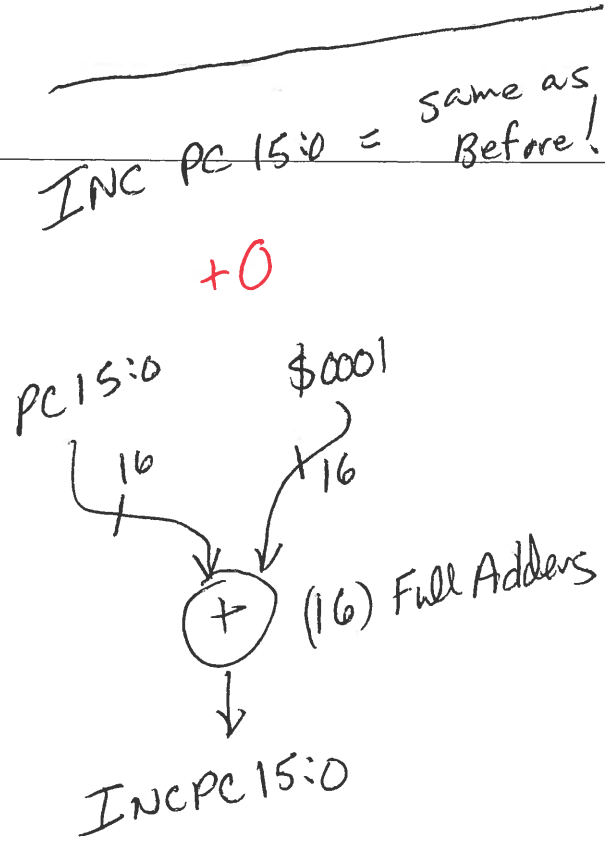
Page Score =

Key

11. Using only decoders, flip-flops, muxes, full adders and other elementary digital components, ***show the new hardware and modifications to the controller*** required to create a JMP ADDR instruction where ADDR is a 16 bit operand that corresponds to the jump address. Use **BUS labeling** when possible. (5 pt.)







Page Score =

Appendix A. G-CPU Code for **Problems 6 - 10:**

Note1: **8K ROM starting address $0** and **8K RAM starting at address $1000**.

Note2: **ROM** is a **slow device** that requires **TWO cycles** to **read** in a **value** from it.

```
        ORG     $0
        LDAB    #$F3
        STAB    $2FFF
        LDY     #$0
        LDX     #$1000
T1:     LDAA    0,Y
        LDAB    #$88
        OR_BA
        STAA    0,X
        INX
        INY
        LDAA    $2FFF
        LDAB    #1
        SUM_BA
        BNE     T1
END1    BEQ     END1
```

B=F3
2FFF[F3]
Y=0
X=1000
A=03
B=88
1000 1011 (8B)=A
1000[8B]
X=1001 y=1
A=F3
B=1
A=F4

STAA $2FFF ; added in Exam!

2nd Pass

A=F3 1111 0011
B=88 1000 1000
A=FB
1001[FB]
y=2 X=1002

F3

Note: Use the space below to hand assemble instructions as needed in 6 - 10:

Addr	Data	
0	03	Ldaa #
1	F3	
2	07	
3	FF	staa addr
4	2F	
5	09	Ldy #
6	00	
7	00	
8	08	
9	00	LDX #
A	10	
B	0C	LdX #
C	00	
D	03	Ld aa 0,y
E	88	
F	18	OR_BA
10	10	STAA 0,X
11	00	
12		

Page Score =