

Jonathan W. Valvano
 May 14, 2005, 9am - 12 noon

First: _____ Last: _____

(4) Question 1.
 Give max number of bytes on the stack

(4) Question 2.
 Yes/no.
 If yes, state where

21
Yes, in LFP between ldd y and stx y
E (partial credit B)
D
C
A
D
(950,1010)
A C E in any order
B
20mV
CPOL=0 CPHA=0
B) asyn serial
N) full duplex

Choose A-Z, or AA-JJ

(2) Question 15.
 Choose A-Z, or AA-JJ

(2) Question 16.
 Choose A-Z, or AA-JJ

(2) Question 17.
 Choose A-Z, or AA-JJ

(2) Question 18.
 Choose A-Z, or AA-JJ

(2) Question 19.
 Choose A-Z, or AA-JJ

(2) Question 20.
 Choose A-Z, or AA-JJ

(2) Question 21.
 Choose A-Z, or AA-JJ

(2) Question 22.
 Choose A-Z, or AA-JJ

(4) Question 23.
 Choose A-F

(4) Question 24.
 Choose yes/no, if no then give an example

(4) Question 25.
 Choose A-F

(2) Question 26.
 Choose A-F

(2) Question 27. If you deliver software with bugs, then you should implement a plan allowing customers to receive software patches to fix the errors.

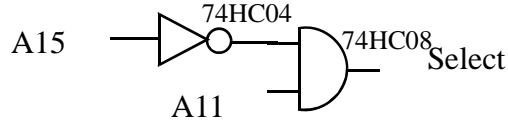
A) accuracy
J) desk check
D) bandwidth
P) latency
U) polled interrupt
W) private
R) nonintrusive
H) buffered I/O
F
No $(10*1)/5=2$ $10*(1/5)=0$
B
C

(4) Question 28. Minimal-cost positive-logic address decoder

RAM \$6000-\$67FF 0110,0XXX,XXXX,XXXX
 YourDevice \$6800-\$6FFF 0110,1XXX,XXXX,XXXX
 ROM \$E000-\$FFFF 111X,XXXX,XXXX,XXXX

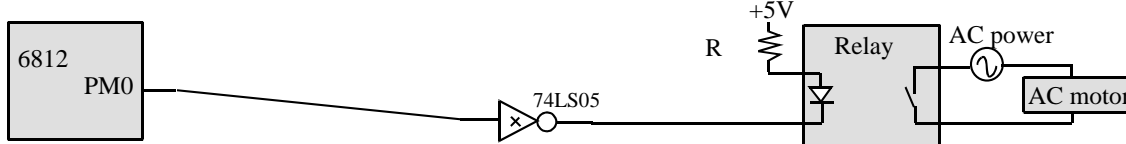
Need addresses A15, A11

Select = not(A15)*A11



(6) Question 29. Interface a solid-state relay

$$R = (5 - V_d - V_{OL}) / I_d = (5 - 2.6 - 0.4) / 2\text{mA} = 1000\Omega$$



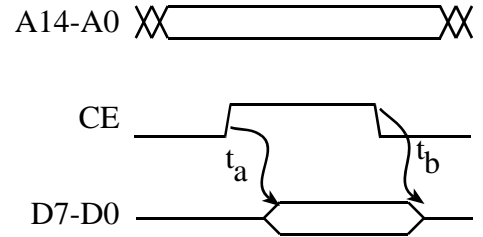
(4) Question 30. Write the C code to implement the following equation using fixed-point math

```
// *****Multiply fixed point Z=X*Y, resolution = 1/16
// Inputs X,Y range from 0 to 4095.9375
// Outputs Z range from 0 to 4095.9375 (return 4095.9375 if overflow)
void FixMult(void){unsigned long product;
    product = (IX*IY)/16;    // promote to higher precision
    if(product<65535){
        IZ = (unsigned short) product;    // demote result
    } else{
        IZ = 65535;    // overflow
    }
}
```

(8) Question 31. Design the interface between a 32k by 8 bit PROM and a 6811

EE345L Spring 2005 Final

(8) Question 31. Design the interface between a 32k by 8-bit PROM and a 6811 running at 2 MHz. Implement full address decoding for addresses \$8000-\$FFFF. Assume $t_a = 100\text{ ns}$, and $t_b = 20\text{ ns}$. Assume a 10ns gate delay through any digital logic and the 74HC573 address latch. The PROM has one control signal, CE, which when high reads data at the 15-bit address. Show just the circuit including chip numbers, but not pin numbers. The timing analysis is not required for this question.



(8) Question 31. Design the interface between a 32k by 8 bit PROM and a 6811

Step 1. Design the address decoder

\$8000-\$FFFF is 1xxx,xxxx,xxxx,xxxx

For fully decoded, specify all 0s and 1s. Select = A15 in positive logic

Step 2. Create a status table. The status table always starts like this

Select	R/W	Control Signals	Explanation
0	0		write cycle to another device
0	1		read cycle from another device
1	0		write cycle to our device
1	1		read cycle from our device

In this situation there is one control signal called CE

Select	R/W	CE	Explanation
0	0		write cycle to another device
0	1		read cycle from another device
1	0		write cycle to our device
1	1		read cycle from our device

Basically in the status table you make the memory do the necessary functions. In this case we want to turn off the device if the cycle is accesses another device. We will also turn it off if a write cycle occurs.

Select	R/W	CE	Explanation
0	0	0	write cycle to another device
0	1	0	read cycle from another device
1	0	0	write cycle to our device
1	1		read cycle from our device

Finally, we will activate it if there is a read cycle to our device

Select	R/W	CE	Explanation
0	0	0	write cycle to another device
0	1	0	read cycle from another device
1	0	0	write cycle to our device
1	1	1	read cycle from our device

Step 3. During the timing analysis we have to do two things. First, guarantee RDA overlaps RDR. And, second we have to make sure the memory doesn't drive the data bus during the first half of the cycle (because the 6811 is driving the low address during the first half of the cycle. To prevent the memory data from colliding we will only drive data out of the memory when E=1. This is called synchronizing the read operation to E. To create a combined table, we expand the status table, adding the E as an input. The data from the status table is entered in the positions where E=1.

E	Select	R/W	CE	Explanation
0	0	0		
1	0	0	0	write cycle to another device
0	0	1		
1	0	1	0	read cycle from another device
0	0	0		
1	1	0	0	write cycle to our device
0	1	1		
1	1	1	1	read cycle from our device

To make the control signal low throughout the cycles when we wish to disable the memory, we place additional zeros

E	Select	R/W	CE	Explanation
0	0	0	0	
1	0	0	0	write cycle to another device
0	0	1	0	

1	0	1	0	read cycle from another device
0	0	0	0	
1	1	0	0	write cycle to our device
0	1	1		
1	1	1	1	read cycle from our device

To make the control signal synchronized positive logic, we place a 0,1 in those entries for the cycle we wish to activate. See the book Figure 9.28 and Table 9.9 for the choices that can be entered into the combined table.

E	Select	R/W	CE	Explanation
0	0	0	0	
1	0	0	0	write cycle to another device
0	0	1	0	
1	0	1	0	read cycle from another device
0	0	0	0	
1	1	0	0	write cycle to our device
0	1	1	0	
1	1	1	1	read cycle from our device

The question did not ask to verify timing, but if it did, we would write equations for RDA and RDR. The gate delay in this question is 10ns.

$$RDA = (\text{rise of CE} + t_a, \text{fall of CE} + t_b)$$

Because CE is synchronized to the E clock, the rise of CE will be 250+gate delay = 260. Similarly, the fall of CE will be 500+gate delay = 510. $t_a = 100 \text{ ns}$, and $t_b = 20 \text{ ns}$

$$RDA = (260+100, 510+20) = (360, 530)$$

RDR comes from the 6811. At 2 MHz it is

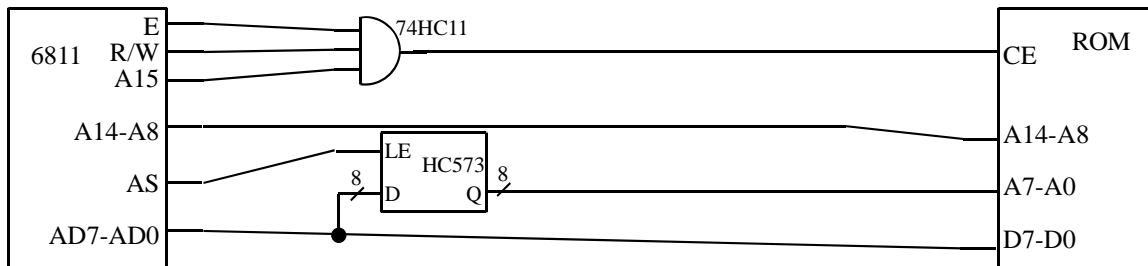
$$RDR = (450, 510)$$

Notice that RDA overlaps RDR

Step 4. Develop the logic equation for the control signal and build it with real gates

$$CE = A_{15} \cdot R/W \cdot E$$

Synchronized positive logic, activate PROM when R/W=1 and A15=1



(6) **Question 32.** Write C code that arms and enables an input capture 7 interrupt

```
// -----IC7_Init-----
// enable input capture 7 interrupts on falling edge
// inputs: none
// outputs: none
```

```

void IC7_Init(void){
    TSCR1 = 0x80;           // enable TCNT
    TIOS &= ~0x80;         // channel 7 is IC
    TCTL3 = (TCTL3&0x3F)|0x40; // rising edge IC7
    TIE |= 0x80;           // Arm IC7
asm cli                     // enable interrupts
}

```

```

Main      ; <=start execution here      2
4126 3b          PSHD                    4
4127 c680        LDAB #128               4
4129 5b00        STAB _CRGINT            4
412b 8633        LDAA #51                4
412d 5a00        STAA _RTICTL            4
412f c7          CLRB                    4
4130 87          CLRA                     4
4131 7c3800      STD Num                 4
4134 10ef        CLI                     4
4136 fc3800      LDD Num                 4
4139 07b3        BSR LowPassFilter 6
LowPassFilter
4100 3b          PSHD                    8
4101 fc3802      LDD y                   8
; interrupt pushes CCR,A,B,X,Y,PC      17
handler
4112 c680        LDAB #128               17
4114 5b00        STAB _CRGFLG            17
4116 fe3800      LDX Num                 17
4119 08          INX                     17
411a 7e3800      STX Num                 17
411d fc3800      LDD Num                 17
4120 07de        BSR LowPassFilter19
LowPassFilter
4100 3b          PSHD                    21
4101 fc3802      LDD y                   21

```