Ryerson Polytechnic University
Department of Electrical and Computer Engineering
ELE538 – Microprocessor Systems

Final Examination **December 4, 1997**

Name:______________________________ **Section:**____________

Time: 3 hours Examiner: A.Lee
Closed-book except for 68HC11 Technical Data
Answer all questions on this paper
Use the same symbolic name for any internal registers as presented in the Data book.
Assume that these symbolic names have been declared properly already

---

1. (a) **Translate the following C code into a 6811 assembly code segment**:

```
char i;
for (i=N;i!=0;i--);
```

Credit is based on the efficiency of your solution.

Solution: **(4 marks)**

(b) **Write a subroutine in 6811 assembly code** to convert a single hex digit (already passed in Accumulator A) into its ascii equivalent back into Accumulator A.
[Reminder: '0' = 0x30, 'A' = 0x41]

Solution: **(4 marks)**

2. (a) **Write a program segment in C (or pseudo) code** to wait for the process of conversion to be completed at the on-chip A/D converter system, then read the result from the A/D Result Register #2 and convert this binary value into a two-digit ascii characters (ch_hi, ch_lo), with a final range of 00 to 30.

   Solution: **(4 marks)**

   (b) **Devise an algorithm in C (or pseudo) code** to convert an 8-bit binary value *fuel_byte* to the corresponding 8-element display *fuel_display* driving 8 LEDs.

   The input *fuel_byte* is an 8-bit binary value that represents the fuel level.

   The output *fuel_display* is to display the fuel level using 8 LED's such that each activated LED (1 = ON) represents 1/8 of a tank of fuel. For example:

All LED's ON:	nearest to a full tank
First 4 LED's ON:	about 1/2 tank
First LED ON:	about 1/8 tank
No LEDs ON:	nearest to an empty tank

   Credit is based on the efficiency of your solution.

   Solution: **(4 marks)**

3. Software delay can be implemented via the following C code:

```
void delay(void) {
     unsigned int i;
     for (i=0;i<N;i++);
     }
```

The output of the gcc6811 compiler indicates that the usage of *delay()* (where N is replaced by 100) is compiled into:

```
0010 BD 00 37        jsr _delay  ; CALL: R:*ZD0 = _delay (0 bytes) 6~
                  _delay:
                  ;;;----------------------------------------
                  ;;;  PROLOGUE for delay
                  ;;;----------------------------------------
0037 18 3C           pshy    ; Save stack frame                      5~
0039 18 30           tsy ; Set current stack frame                   4~
003B DE 00           ldx *ZD5                                        4~
003D 3C              pshx    ; pushed register *ZD5                  4~
                  ;;;END PROLOGUE
003E CC 00 00        ldd #0                                          3~
0041 DD 00           std *ZD5    ; movhi: #0 -> *ZD5                 4~
0043              L24:
0043 DE 00           ldx *ZD5                                        4~
0045 08              inx                                             3~
0046 DF 00           stx *ZD5    ; addhi3: *ZD5 by #1 -> *ZD5        4~
0048 DC 00           ldd *ZD5                                        4~
004A 1A 83 00 63     cpd #99 ; cmphi *ZD5 with #99                   5~
004E 22 03           bhi .+5                                         3~
0050 7E 00 43        jmp L24 ; (bleu) long branch                    3~
                  ;;;EPILOGUE
0053 38              pulx    ; Pulling register *ZD5                 5~
0054 DF 00           stx *ZD5                                        4~
0056 18 38           puly    ; Restore stack frame                   6~
0058 39              rts ; return from function                      5~
                  ;;;----------------------------------------
                  ;;; END EPILOGUE for delay
                  ;;;----------------------------------------
```

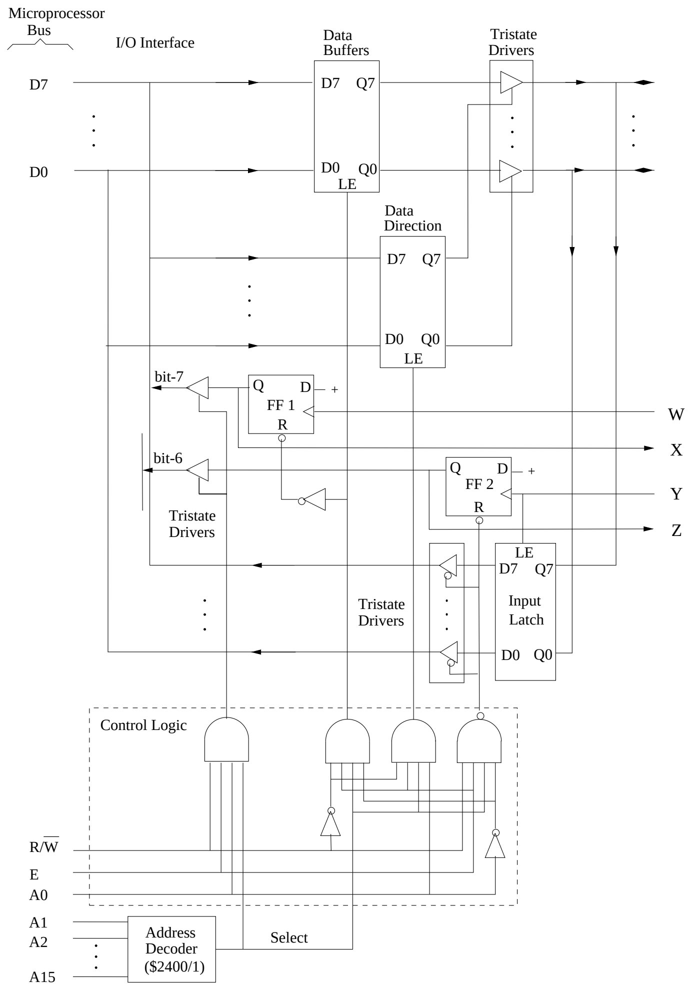
A certain delay (in cycles) will be generated whenever *delay()* is called:

(a) **Estimate this delay value**. Ignore any overhead. (Show your calculation) **(3 marks)**

(b) **Calculate the exact total delay value** by including all overhead. **(3 marks)**

(c) **What is the new value of N** that would yield a delay time closest to 10,000 cycles? **(2 mark)**

4. Given the following basic I/O interface circuits:



(a) **Label each of the points (W, X, Y, Z)** with an appropriate lebel from one of the following more descriptive term (DAV, $\overline{DAV}$, DAC, $\overline{DAC}$) and what direction of data transfer (input, output) they are associated with. **(2 marks)**

(b) **Indicate all circuit functions** that would be accomplished after executing each of the following steps: **(3 marks)**

i. write to 0x2400

ii. read from 0x2400

iii. write to 0x2401

iv. read from 0x2401

(c) **Give the necessary 6811 assembly code segment** to read the incoming data byte whenever a valid input data has been latched. (**Do not use any symbolic name here**). **(3 marks)**

5. (a) **Write a program module** *delay(int N)* **in C (or pseudo) code** to generate a delay of N system clock cycles using the on-chip 6811 timer facility. Your solution should **not** involve the use of interrupt.

(Due to the unavoidable time overhead before readjustment, the actual result of delay may be slightly larger than N cycles. You may ignore this fact for now.)

Solution: **(4 marks)**

(b) **Write a program segment in C (or pseudo) code** to ouput on bit_3 and bit_2 of PORTD the following waveforms:

```
     PD3          ---------------------------
         ________|                           |____
         0   1   2   3   4   5   6   7   8   9   10ms
         . . . . . . . . . . . . . . . . . . . . .
     PD2 ________             --------------------
         ^       |___________|                   ^
         |                                       |
program start                                program end
```

Use the delay module *delay(int N)* in 5(a) to generate all necessary time delays. Assume a 1 MHz system clock (E). Your solution must not alter the operations of any other bits on PORTD.

(Do not assume any existing status of these two lines prior your program segment.)

Solution: **(4 marks)**

6. A 68HC11 controller based system is required to communicate with another similar system using the on-board SCI facility. The required configuration is:

   - full duplex serial communication
   - 8N1 (i.e. 8-bit data, No parity, 1 stop bit) data format at 4800 baud, with system clock (E) running at 1 MHz.
   - a single interrupt service routine to handle both transmit and receive operations (i.e. whenever one or both of TDRE and RDRF flags are set).

   Use the labels *SCI_soft_vector* (assume defined) and *ISR_SCI* to complete the needs of your solution.

   (a) **Write a program segment in C (or pseudo) code** to configure all relevant SCI registers and to implement all necessary steps to initialize the controller for proper interrupt capability.

   Solution: **(4 marks)**

   (b) **Write in C (or pseudo) code** all necessary steps that must be carried out inside the ISR to handle both transmit and receive operations. Be sure to include the checking of any erroneous conditions.

   Note: A couple of character pointers, *send_ptr* and *receive_ptr*, should be used to handle the data transfer. The first pointer specifies the source of the character to be sent, and the second pointer specifies the destination of the received character. Each pointer should be incremented after successful data transfer.

   Solution: **(4 marks)**