“There is no reason anyone would want a computer in their home.”
Ken Olson, president, chairman and founder of Digital Equipment Corporation, 1977

Quiz 1 review
1) Definitions (matching or multiple choice)
volatile, nonvolatile
RAM, ROM, port
static efficiency, dynamic efficiency
structured program, call graph, data flow graph
basis, nibble, precision, decimal digits (see table below)
fixed-point,
overflow, ceiling and floor, drop out,
bus, address bus, data bus,
memory-mapped, I/O mapped
bus cycle, read cycle, write cycle,
IR, EAR, BIU, CU, ALU, registers,
device driver,
friendly,
mask, toggle,
reset vector
Table 2.2. Standard definition of decimal digits.

<table>
<thead>
<tr>
<th>decimal digits</th>
<th>exact range</th>
<th>exact alternatives</th>
<th>approximate bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>0 to 999</td>
<td>1,000</td>
<td>~ 10</td>
</tr>
<tr>
<td>3½</td>
<td>0 to 1999</td>
<td>2,000</td>
<td>~ 11</td>
</tr>
<tr>
<td>3¾</td>
<td>0 to 3999</td>
<td>4,000</td>
<td>~ 12</td>
</tr>
<tr>
<td>4</td>
<td>0 to 9999</td>
<td>10,000</td>
<td>~ 13</td>
</tr>
<tr>
<td>4½</td>
<td>0 to 19,999</td>
<td>20,000</td>
<td>~ 14</td>
</tr>
<tr>
<td>4¾</td>
<td>0 to 39,999</td>
<td>40,000</td>
<td>~ 15</td>
</tr>
<tr>
<td>5</td>
<td>0 to 99,999</td>
<td>100,000</td>
<td>~ 17</td>
</tr>
<tr>
<td>5½</td>
<td>0 to 199,999</td>
<td>200,000</td>
<td>~ 18</td>
</tr>
<tr>
<td>5¾</td>
<td>0 to 399,999</td>
<td>400,000</td>
<td>~ 19</td>
</tr>
<tr>
<td>6</td>
<td>0 to 999,999</td>
<td>1,000,000</td>
<td>~ 20</td>
</tr>
<tr>
<td>6½</td>
<td>0 to 199,999</td>
<td>2,000,000</td>
<td>~ 21</td>
</tr>
<tr>
<td>6¾</td>
<td>0 to 3,999,999</td>
<td>4,000,000</td>
<td>~ 22</td>
</tr>
<tr>
<td>N</td>
<td>0 to 10^N-1</td>
<td>10^N</td>
<td>~ N*\log_2(10)</td>
</tr>
<tr>
<td>N½</td>
<td>0 to 2*10^N-1</td>
<td>2*10^N</td>
<td>~ N*\log_2(10)+1</td>
</tr>
<tr>
<td>N¾</td>
<td>0 to 4*10^N-1</td>
<td>4*10^N</td>
<td>~ N*\log_2(10)+2</td>
</tr>
</tbody>
</table>

2) Number conversions, 8-bit (fill in the blank)
convert one format to another without a calculator

- signed decimal e.g., $-56$
- unsigned decimal e.g., $200$
- binary e.g., $\%11001000$
- hexadecimal e.g., $\$C8$

I won’t ask you to convert signed binary or signed hex:
- signed binary e.g., $-\%00101111$
- signed hexadecimal e.g., $-\$2F$
**fixed-point representations**

given resolution convert between **value** and **integer**
given precision and range choose the fixed-point format

3) **Details of executing single instructions**

8-bit addition, subtraction yielding result, N, Z, V, C
(like Homework)
simplified cycle by cycle execution
assembly listing to execution cycles (Lab 3.2)
machine code to execution cycles (Lab 3.4)
for indexed mode addresses, for example

```
ldaa 4,x
ldaa 40,x
ldaa -4,x
ldaa -40,x
ldaa $400,x
ldaa 4,+x
ldaa 4,-x
ldaa 4,x+
ldaa 4,x-
```

calculate effective address
go from assembly to machine code **xb**
go from machine code **xb** to assembly
simple multiply and divide (**mul** **idiv** **fdiv**) 
stack functions for **bsr** and **rts**

4) **Simple programs** (look at assembly code in Chap 1,2,3)
initialize stack
create global variables
set reset vector
specify an I/O pin is an input
specify an I/O pin is an output
clear an I/O output pin to zero
set an I/O output pin to one
toggle an I/O output pin
check if an I/O input pin is high or low
e.g., if PA4 is low then make PB2 high

8-bit operations
    add, sub, shift left, shift right, and, or, eor
simple if-then like examples in Chapter 3
no if-then-else
no if((uG1>5)&&(uG2<100)){}  

simple while-loop like examples in Chapter 3
simple subroutines, parameters passed in registers
four lines of comments for client
* purpose
* inputs: registers, format, units
* outputs: registers, format, units
* error possibilities
called with bsr, returns using rts

Look at last year’s quiz to see the types of information given to you. Notice also the format of the quiz and the expected answers.
4. Assembly Language Programming

Chapter 4 objectives are to:

- Present an introduction to product development,
- Discuss the TExaS editor and assembler,
- Basic approach to assembly language programming,
- Define the pseudo-ops supported by TExaS.

4.1.1. Assembly language development

- editor
- source code
- assembler
- crossassembler
- object code
- loader
- debugger

Source code

```
PORTA equ $0000
DDRA equ $0002
org $0800
cnt rmb 2
org $F000
main lds #$C000
movb #$80,DDRA
off bclr PORTA,#$80
look ldd #$4444
std cnt
loop ldaa PORTA
   anda #$7F
   cmpa key
   bne off
   ldx cnt
   dex
   stx cnt
   bne loop
   bset PORTA,#$80
   bra look
key fcb %00100011
org $FFFE
fdb main
```
The first pass
check for syntax errors
size of each line of object code
create the symbol table

The second pass
check for syntax errors, errors reported in listing file
create object code, S19 file if needed for real 6812
recreate the symbol table, phasing error if different
put object code, details and source into listing file
add symbol table to the end of the listing file

A symbol table is a mapping between symbolic names
PORTC equals $0003

The actual object code is in bold face.

Errors that occur during the assembly process
1) Label previously defined error: label occurs multiple times
2) Undefined opcode error: operation does not exist
3) Operand error: syntax error within the operand
   expression error
   undefined symbol in expression
   addressing mode error
   size of allocated storage too big, e.g., \texttt{ds 100*1000}
4) Phasing error: value of symbol changes from pass1 to pass2
5) Can't program address error
6) Branch too far error: destination address is too far away
7) Illegal string termination: e.g., "Hello World!"