EE443 - Embedded Systems

Exercise - 2

Assembly Process

1. What are the assembler operations performed during:
a) Parsing
b) Making the machine code

2. Describe how the symbol table is utilized by the assembler during the assembly process.

3. The following assembly code is written for an 8-bit microprocessor with 16-bit addressing.
a) Fill in the "**Memory Address**" column according to the required machine instruction lengths.
b) Fill in the symbol table for the assembly code.
c) Complete the data and address fields for the machine instructions.

```
Sub1: ORG  0x2C00
      Load    A, Number;    load one byte direct
      Sub     A, Limit;     subtract one byte direct
      JumpIZ  Done;         jump-if-zero to address
      Load    A, #0x01;     load immediate data
      Store   A, Dready;    store one byte direct
      Jump    Cont;         jump to address
Done: Load    A, #0x36;     load immediate data
      Store   A, Number;    store one byte direct
Cont: Return;
      VAR     Number;       reserve one byte memory
      VAR     Limit;        reserve one byte memory
      VAR     Dready;       reserve one byte memory
```

SYMBOL TABLE	
Symbol	**Address**
Sub1	2C00

Memory Address	**Opcode**	**Data / Address**	
2C00	5A		
	62		
	A3		
	36		
	15		
	A1		
	36		
	15		
	B1		
	Reserved for **Number**		
	Reserved for **Limit**		
	Reserved for **Dready**		

Instruction:

```
Load   A, Number;
Sub    A, Limit;
JumpIZ Done;
Load   A, #0x01;
Store  A, Dready;
Jump   Cont;
Load   A, #0x36;
Store  A, Number;
Return;
```

4. What are the differences between subprogram calls and macro calls? Describe the advantages and disadvantages of macro calls compared to subprogram calls.

5. Write the assembly language instructions corresponding to the following statements in C language on a typical 8-bit microprocessor with 16-bit memory address.

a)
```
MyChar ++;  // increment one-byte variable
```
b)
```
MyInt ++;   // increment short integer (two-byte) variable
```
c)
```
if (MyChar == 0)  // check if one-byte variable is zero
  <calculate A>
else
  <calculate B>
```
d)
```
if (MyInt == 0)  // check if two-byte variable is zero
  <calculate A>
else
  <calculate B>
```