Ch6

Assembly Language Programming

CHAPTER ASSIGNMENT · 37 MARKS

Name _____ Class _____ No. _____ Date _____

Answer every question in the space given. Show your working. You may use the chapter page to help you.

Questions from *Computer Science for Cambridge International AS & A Level Coursebook*, 2nd edition, Sylvia Langfield and Dave Duddell (Cambridge University Press), for classroom use.

Tasks and questions from the chapter

DP **Discussion Point:** Although writing a program in assembly language is much easier than using machine code, many would argue that its use is no longer justified. Can you investigate the arguments for and against?

T6.01 **Task 6.01:** Check the ASCII coding table to see why the subtraction in Table 6.01 works.

T6.02 **Task 6.02:** Use register transfer notation to describe the execution of an LDI instruction.

EQ6.01 **Extension Question 6.01:** Carry out a comparable calculation for the addition in binary of –66 to +68. What do you think the processor should do with the carry bit?

Q6.01 **Question 6.01:** Can you follow through the changes in the trace table for Worked Example 6.03? Could it be possible for the program to change the content in one of the memory locations 100–107 during execution?

T6.03 **Task 6.03:** Without looking at the explanations provided, trace the assembly language program shown in Table 6.01. Use a value of 3 for MAX and then 7, 8 and 9 as input values.

RP **Reflection Point:** There are several references in this chapter to the content in earlier chapters. Have you checked that you understand how the topics are related by revising the content in the earlier chapters?

Exam-style questions

1 Three instructions for a processor with an accumulator as the single general purpose register are:

`LDD <address>` for direct addressing

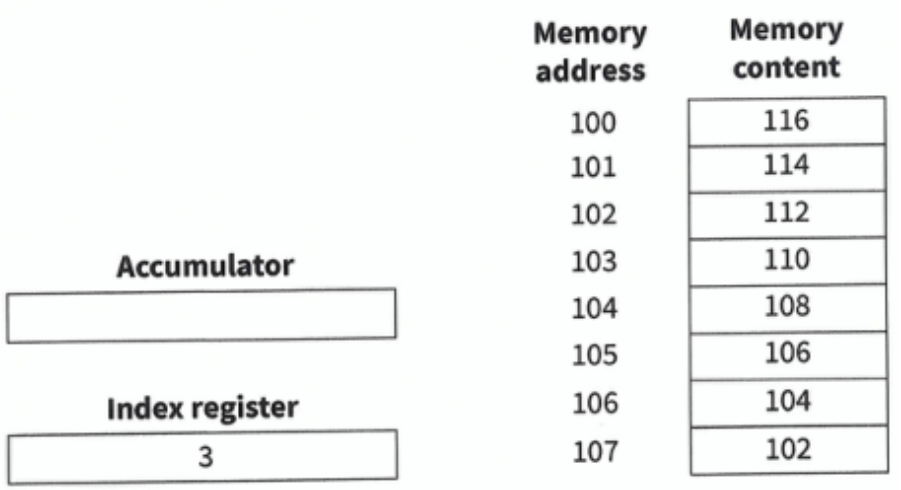
`LDI <address>` for indirect addressing

`LDX <address>` for indexed addressing

In the diagrams below, the instruction operands, the register content, memory addresses and the memory contents are all shown as denary values.

(a) Consider the instruction `LDD 103`.

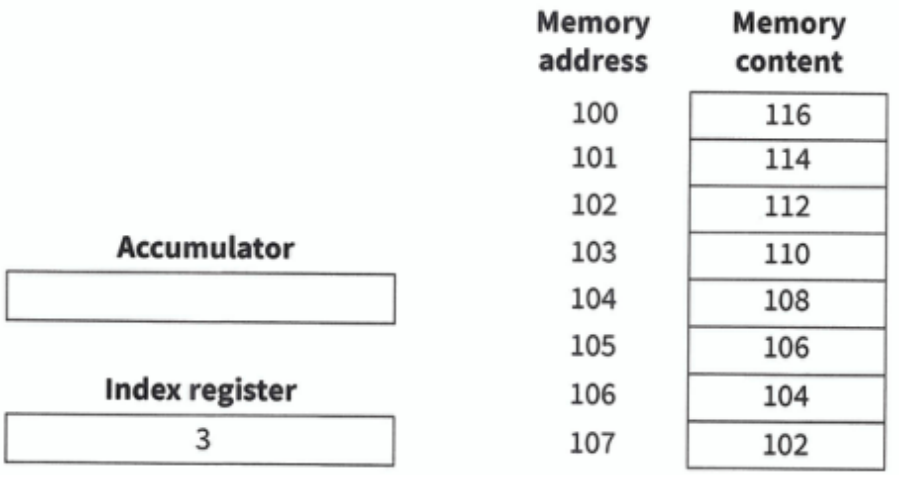
(i) Draw arrows on a copy of the diagram below to explain execution of the instruction. [2]



(ii) Give the contents of the accumulator as a denary value after execution of the instruction. [1]

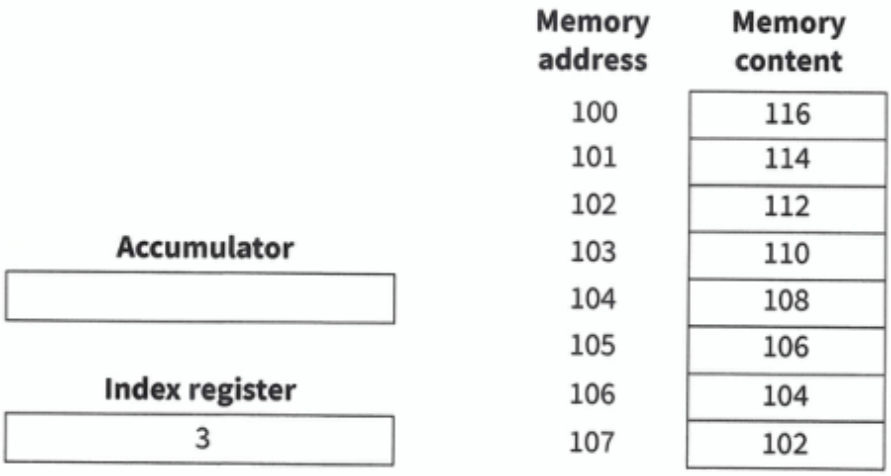
(b) Consider the instruction `LDI 107`.

(i) Draw arrows on a copy of the diagram below to explain execution of the instruction. [3]



(ii) Give the contents of the accumulator as a denary value after execution of the instruction. [1]

(i) Draw arrows on a copy of the diagram below to explain the execution of the instruction `LDX 103`. [3]



(ii) Give the contents of the accumulator as a denary value after the execution. [1]

---

2 Every machine code instruction has an equivalent in assembly language. An assembly language program will contain assembly language instructions. An assembly language program also contains components not directly transformed into machine code instructions when the program is assembled.

(a) Describe the use of three types of component of an assembly language program that are not intended to be directly transformed into machine code by the assembler. [6]

(b) Complete the trace table for the following assembly language program. Note that the LDI instruction uses indirect addressing. [6]

Assembly language program

Memory address	Memory content
100	LDD 201
101	INC ACC
102	STO 202
103	LDI 203
104	DEC ACC
105	STO 201
105	ADD 204
107	STO 201
108	END
201	10
202	0
203	204
204	5

Accumulator
0

Memory addresses			
201	202	203	204
10	0	204	5

3 Consider the following assembly language program:

```
            IN
            STO CHARACTER
            IN
            SUB #48
START:      CMP #0
            JPN OUTPUT
            END
OUTPUT:     OUT
            DEC ACC
            JMP START
CHARACTER:
```

(a) Explain what the program takes as input. [4]

(b) Explain what the program outputs. [3]

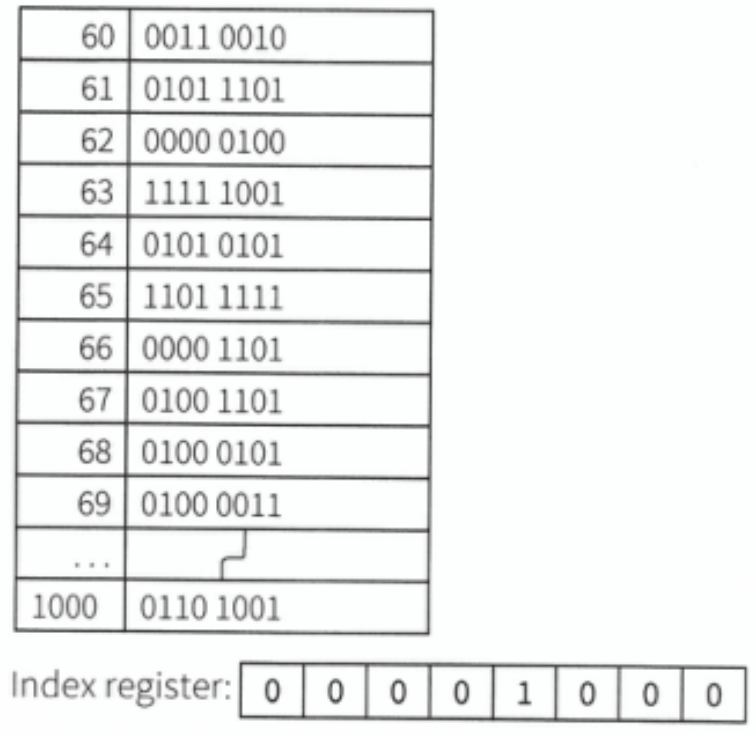
(c) Complete the symbol table shown below which would be obtained from the first pass of a two-pass assembler. You can use denary numbers for addresses and you can assume that the first instruction is stored in address 0. [4]

Label	Address

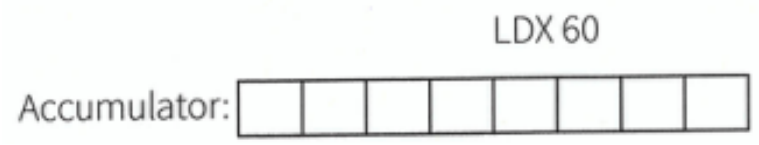
4 The table shows assembly language instructions for a processor which has one general purpose register, the Accumulator (ACC) and an index register (IX).

Instruction Op code	Instruction Operand	Explanation
`LDD`	`<address>`	Direct addressing. Load the contents of the given address to ACC.
`LDX`	`<address>`	Indexed addressing. Form the address from `<address>` + the contents of the index register. Copy the contents of this calculated address to ACC.
`STO`	`<address>`	Store contents of ACC at the given address.
`ADD`	`<address>`	Add the contents of the given address to ACC.
`INC`	`<register>`	Add 1 to the contents of the register (ACC or IX).
`DEC`	`<register>`	Subtract 1 from the contents of the register (ACC or IX).
`CMP`	`<address>`	Compare contents of ACC with contents of `<address>`.
`JPE`	`<address>`	Following a compare instruction, jump to `<address>` if the compare was True.
`JPN`	`<address>`	Following a compare instruction, jump to `<address>` if the compare was False.
`JMP`	`<address>`	Jump to the given address.
`OUT`		Output to screen the character whose ASCII value is stored in ACC.
`END`		Return control to the operating system.

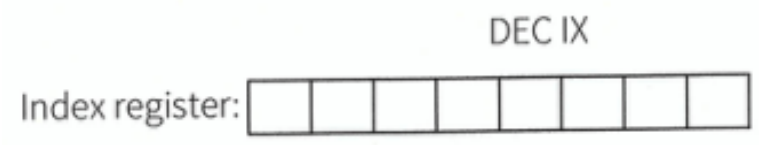
(a) The diagram shows the current contents of a section of main memory and the index register:



(i) Show the contents of the Accumulator after the execution of the instruction: `LDX 60`. Show how you obtained your answer. [2]



(ii) Show the contents of the index register after the execution of the instruction: `DEC IX` [1]



Cambridge International AS & A Level Computer Science 9608 paper 11 Q9a June 2016

End of assignment · Check your answers against the chapter at gurujazzy.com/subjects/computer-science/as-a-level/assembly-language/