A-level
Computer Science

Assembly languages

Lesson Objectives

Students will learn about:

- Assembly language
- Machine codes
- Instructions used in assembly language
- Writing simple programs in assembly language
- Different types of addressing modes

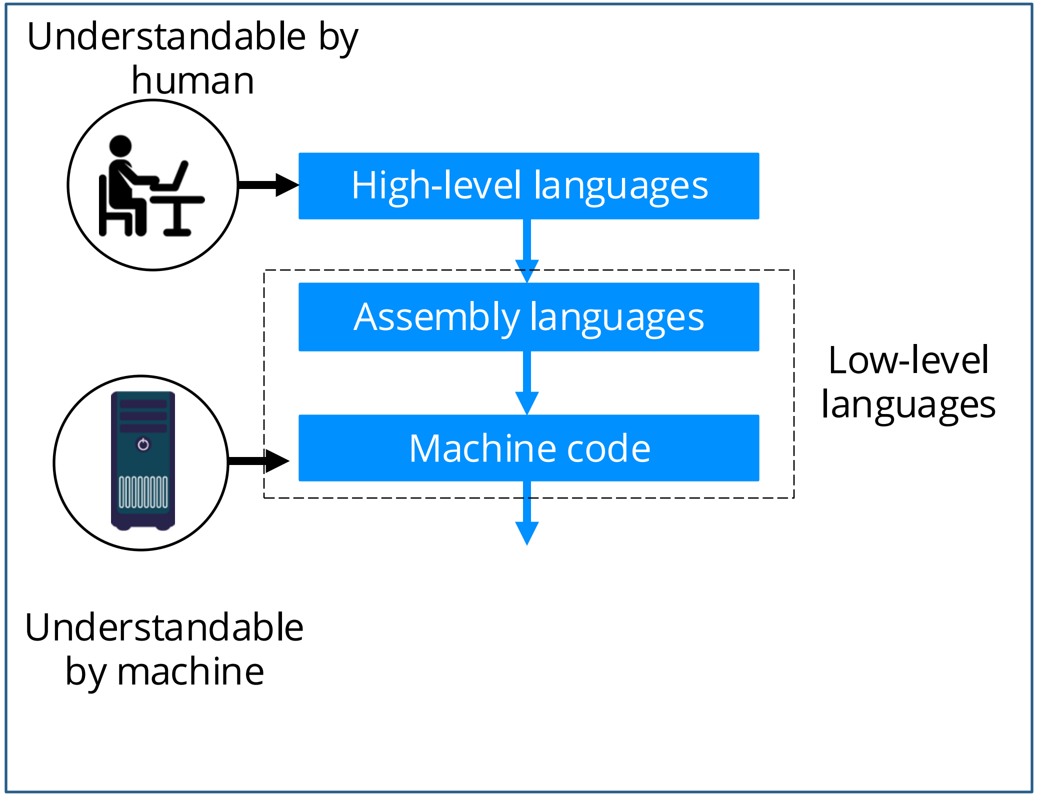
1.

Content

Low-level languages

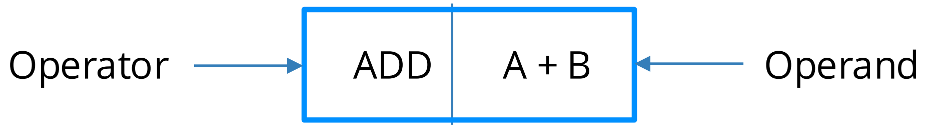
- Low-level languages are specific to hardware and include assembly language and machine code.
- Each hardware has a different instruction set.
- Instruction set is the list of instructions such as ADD, SUB, INC, DEC, etc.

Low-level languages: Assembly language & Machine code



Instruction

- An instruction consists of two parts: Operator and Operand.
- Operator is a part of an instruction that tells the CPU about the operation to be performed.
- Operand is that part of instruction that represents the data or memory location where the operation is required to be carried out.



Low-level languages

- Low-level languages are related to the hardware architecture and its instruction set.
- There are two types of low-level languages:
  - Machine code, binary instructions that are understandable by the computer.
  - Assembly language that needs to be converted to machine code.

Assembly language

- Assembly language is used by programmers to make use of special hardware.
- Instructions used are dependent upon the type of machine.
- The code does not take up much space of primary memory and performs its task quickly.

Assembly language

- Code to add two numbers:

Instruction	Operand	Function
MOV	AX, 22h	Move the hex number 22 to the register AX
ADD	AX, 15h	Add the number in register AX with hex number 15
HLT		Stop

- In instruction, ADD AX, [15]. ADD is an example of instruction.
- The numbers in registers AX and 15 are examples of operand.

What are registers?

- Registers are high-speed data storage areas in the computer.
- Register AX represents the accumulator, sometimes denoted ACC
- Accumulator: Register where the processor performs the calculations and stores the result.
- Program counter: Register that contains the memory location of the address of the next instruction.

Operators in assembly language

- Operators are represented in mnemonics.
- Mnemonics is a set of characters.
- Use of mnemonics makes assembly instructions easier to remember.
- Variable is a named memory location to store data.

Instructions

INP and OUT refers to the accumulator and hence, the operands need not be specified.

Instruction	Function
INP	Receive input from user and copy to accumulator
OUT	Output to user the value stored in accumulator
ADD	Add the value in accumulator with the value given in instruction
SUB	Subtract the value given in instruction from the value in accumulator
HLT	Stop
mem1 DAT	Label the current memory location as mem1
STA mem1	Store variable in the memory location mem1

Instructions

INP and OUT refers to the accumulator and hence, the operands need not be specified.

Instruction	Function
LDA mem1	Load the accumulator with the contents of memory location mem1
BRA	Branch instruction. The next instruction is the address provided in this instruction
BRP	Branch if positive. Branch to the address if the contents of accumulator is not zero
BRZ	Branch if zero. Branch to the address if the contents of accumulator is zero.

Machine code

- Machine code is written in hexadecimal or binary form.
- It is hard to understand and complicated to manage the storage and manipulation of data.

Machine code

- Code to add two numbers:

Opcode	Operand	Function
A1	22	Move the hex number 22 to the register AX
05	15	Add the number in register AX with hex number 15
F4		Stop

- It can be noticed that the code is not understandable.
- Opcode is a part of an instruction that tells the CPU about the operation to be performed.

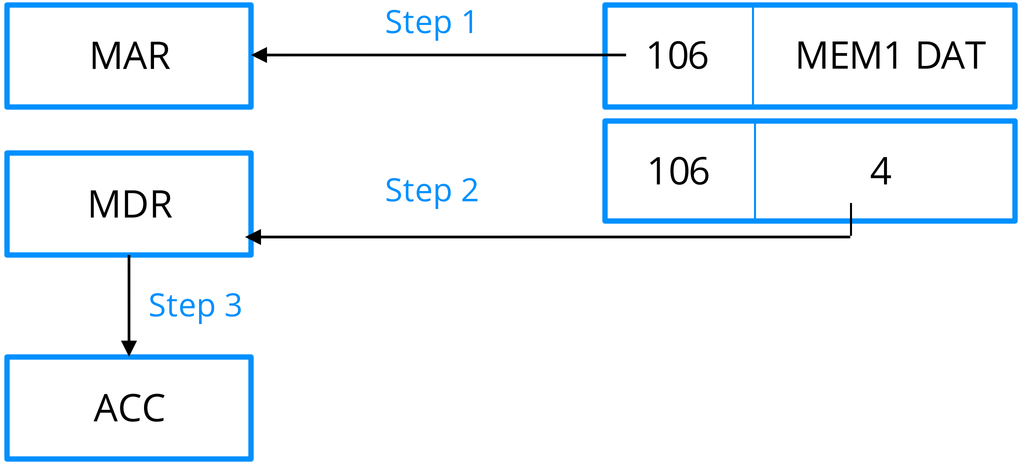
Program to add two numbers

- A simple program to add two numbers is given.
- First number: stored at 106
- Second number: stored in ACC

Instruction	Operand	Function
INP		Input a number and store in accumulator
STA	MEM1	Store the number in memory location MEM1
INP		Input another number and store in accumulator
ADD	MEM1	Add the number in accumulator with the number in 'MEM1'
OUT		Output the result
HLT		Stop the program
MEM1 DAT		Location where the first number is stored

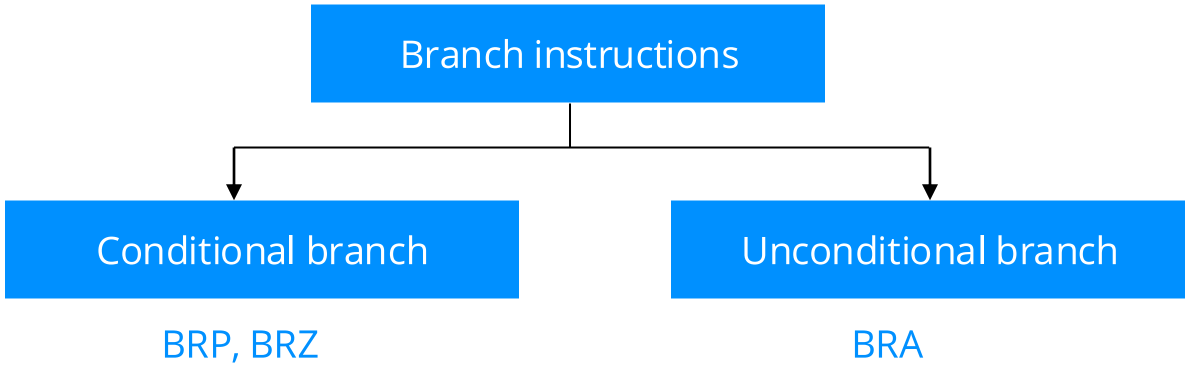
Instruction: MEM1 DAT

- MAR: Memory Address Register
- MDR: Memory Data Register
- MAR stores the address of the variable.
- MDR stores the value of the variable and this value is copied to ACC.



Branch instructions

- Branch instructions alter the control flow of the program.



Branch instructions

Instruction	Function
BRA	Branch instruction. The next instruction is the address provided in this instruction.
BRP	Branch if positive. Branch to the address if the contents of accumulator is zero or positive.
BRZ	Branch if zero. Branch to the address if the contents of accumulator is zero.

Program to find the smaller number out of two numbers

- Compare two numbers: using BRP statement
- If answer is positive, first number is the smaller number

2nd num	-	1st num	=	Answer
MEM2 & ACC		MEM1		ACC

	Inst	Operand	Function
	INP		Input a number and store in accumulator
	STA	MEM1	Store the number in memory location MEM1
	INP		Input another number and store in accumulator
	STA	MEM2	Store the number in memory location MEM2
	SUB	MEM1	Subtract the number in MEM1 from the number in accumulator
	BRP	label1	Branch if positive to location label1
	LDA	MEM2	Load the accumulator with the second number
	OUT		Output the result
	HLT		Stop the program
label 1	LDA	MEM1	Load the accumulator with the first number
	OUT		Output the result
	HLT		Stop the program

Division in 8085 microprocessor

- A and B with numbers in memory location 1040 and 1041 respectively.
- Repeated subtraction.
- Stores the quotient in 2041 and remainder in 2040.
- This is the program specific for 8085 microprocessor.
- Instruction set is slightly different.

	Instruction	Function
	LXI H, 1040	Loads the HL pair register with the address 1040 of memory location
	MOV B, M	Copies the content of memory into register B.
	MVI C, 00	Register C is assigned with value 00h
	INX H	Increments the HL pair register to point to 1041
	MOV A, M	Copies the content of memory into register A.
	CMP B	Compares contents of reg A and B
Label 1	JC Label2	Jumps to Label 2 if carry flag is set
	SUB B	$A \rightarrow A-B$
	INR C	Increments the value of C register
Label 2	JMP Label1	Jumps to Label 1
	STA 2040	Stores remainder in 2040
	MOV A, C	Copies the contents of reg C to A
	STA 2041	Stores the quotient in 2041
	HLT	Terminates the program

Addressing modes

In opcodes, the last 2 bits are allocated for addressing modes. There are four types of addressing modes:

- Immediate addressing: The actual value of the operand is specified in immediate addressing.
- Direct addressing: The memory address of the operand to be operated on is specified.

Addressing modes

In opcodes, the last 2 bits are allocated for addressing modes. There are four types of addressing modes:

- Indirect addressing: The memory address of the location (a register) that holds the address of the data to be operated on is the operand.
- Indexed addressing: The memory address of the operand is calculated by adding the value of index register with a constant value. This type of addressing is suited to access arrays where the elements are stored in successive memory locations.

Addressing modes

- Assume the contents of: Accumulator is 15, Index register is 7 & Register B is 0.
- LDA immediate 5: Accumulator is loaded with value 5.
- LDA direct 5: Accumulator is loaded with contents of memory location 5, that is, 10.

Memory location	Contents
1	
2	
3	
4	
5	10
6	
7	32
8	29
9	
10	37
11	

Addressing modes

- Assume the contents of: Accumulator is 15, Index register is 7 & Register B is 0.
- LDA indirect 5: Accumulator is loaded with the contents of memory location 10 (value present in location 5). ACC is loaded with value 37.

Memory location	Contents
1	
2	
3	
4	
5	10
6	
7	32
8	29
9	
10	37
11	

Addressing modes

- Assume the contents of: Accumulator is 15, Index register is 7 & Register B is 0.
- LDA indexed B: ACC is loaded with 32 (contents of (loc. 7 + reg. B) = loc. 7)
- If Register B is incremented, next memory location is pointed. Contents of Reg. B is 1.
- LDA indexed B: ACC is loaded with 29 (contents of (loc. 7 + reg. B) = loc. 8)

Memory location	Contents
1	
2	
3	
4	
5	10
6	
7	32
8	29
9	
10	37
11	

Let’s review some concepts

Low-level languages

Low-level languages are related to the hardware architecture and its instruction set. Two types: Assembly language and machine code.

Assembly language

A type of low-level language, that is understandable to humans. Operators are written in mnemonics.

Machine code

Written in hexadecimal or binary form. It is hard to understand and complicated to manage the storage and manipulation of data.

List of instructions learnt
INP
OUT
LDA, STA
DAT
ADD, SUB
HLT
Branch instructions: BRA, BRP & BRZ

List of registers learnt
Accumulator
Program counter
Memory Address Register
Memory Data Register

Addressing modes
Immediate
Direct
Indirect
Indexed

2.

Activity

Activity-1

Duration: 15 minutes

1. Write down an assembly program to multiply two numbers in 8085 microprocessor. The numbers are stored in memory location 1000 and 1001. Your program stores the result in memory location 2000 and 2001.

3.

End of topic questions

End of topic questions

1. What are the two different types of codes in low-level programming? How are they different from each other?
2. What is the function of the following instructions?
   i. INP
   ii. STA
   iii. DAT
   iv. OUT
   v. HLT

End of topic questions

3. What happens during the instruction: FIRST DAT? Explain using a diagram. Assume that this instruction is at memory location 94.
4. What are branch instructions? State the different branch instructions with its function.
5. Write an assembly language program to compare two numbers and print the largest number.
6. What are the different addressing modes? Explain each of them with an example along with values of accumulator, index register and register B.