CSE-3108

Spring, 2016

Introduction to 8086 and MDA-8086

Contents Of Presentation

1. Reference Books
2. Evaluation Process in Laboratory
3. Basic Idea on MDA-8086 Trainer KIT
4. Basic Idea on 8086 Microprocessor
5. How to load Machine Instruction in MDA-8086

Reference Books

- **8086 Microprocessor Laboratory Experiments**

  --------Golam Mostofa

- **Microprocessor and Microprocessor Based System**

  ----------M. Rafiquzzaman

Evaluation Process

A student will be evaluated by the processes:

Process	Credit
Attendance	**20%**
Class Performance	**5%**
Assignment	**5%**
Midterm	**20%**
Final Exam	**50%**
Total	**100%**

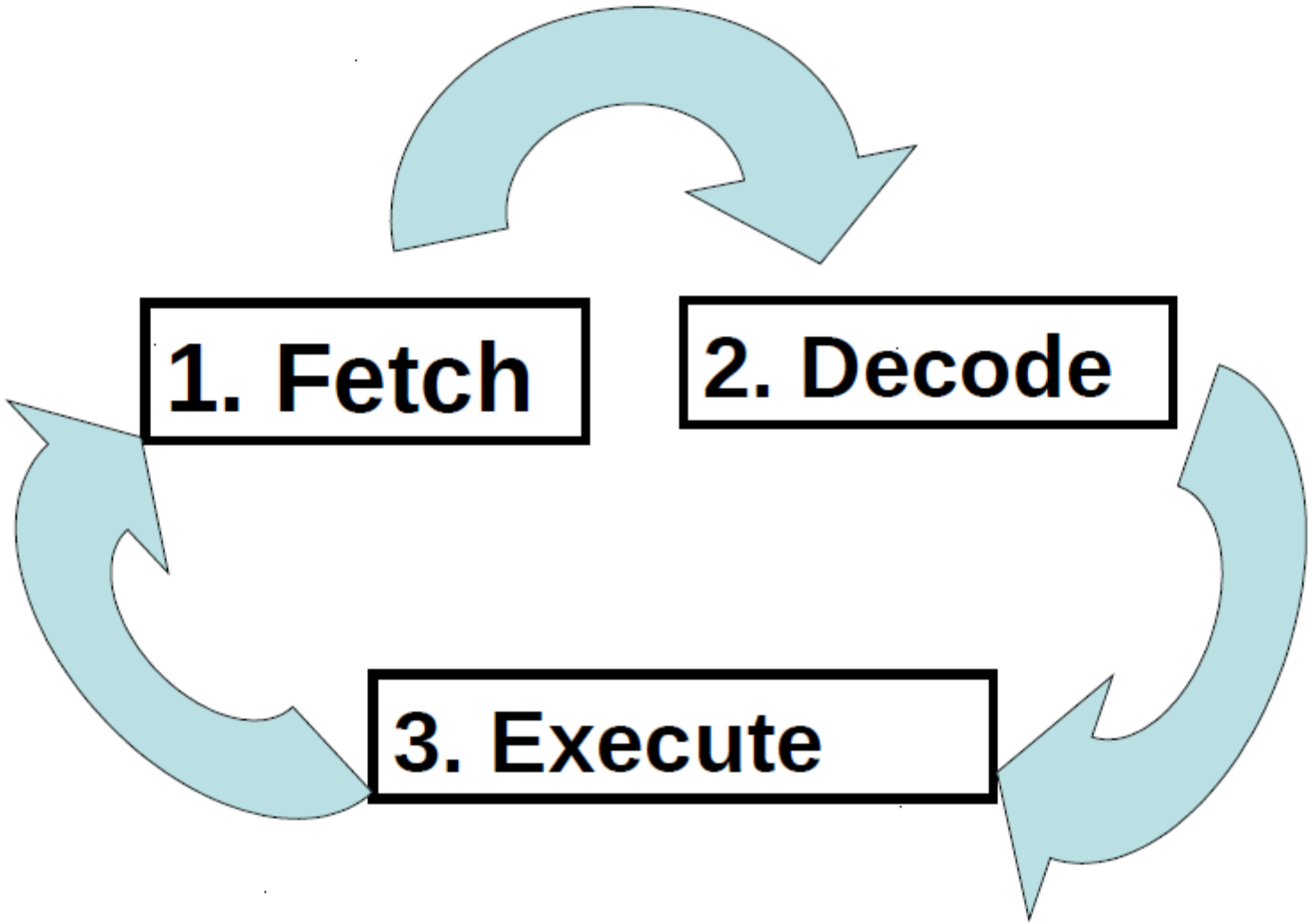
Why Intel 8086?

- Intel 8085 was a rather simple microprocessor, which did not provide many functionalities, or extendibility.
- Intel 8086 was introduced in 1978 as a fully 16-bit extension of Intel's 8-bit based 8080 microprocessor
- Intel 8086 was a birth of a standard for many modern microprocessors. All of the current microprocessors from Intel, like the Core™ i7, Core™ i5, Pentium Processors have instruction set, which are supersets of instruction set of 8086
- One of the first microprocessors to use support pipelining and memory segmentation

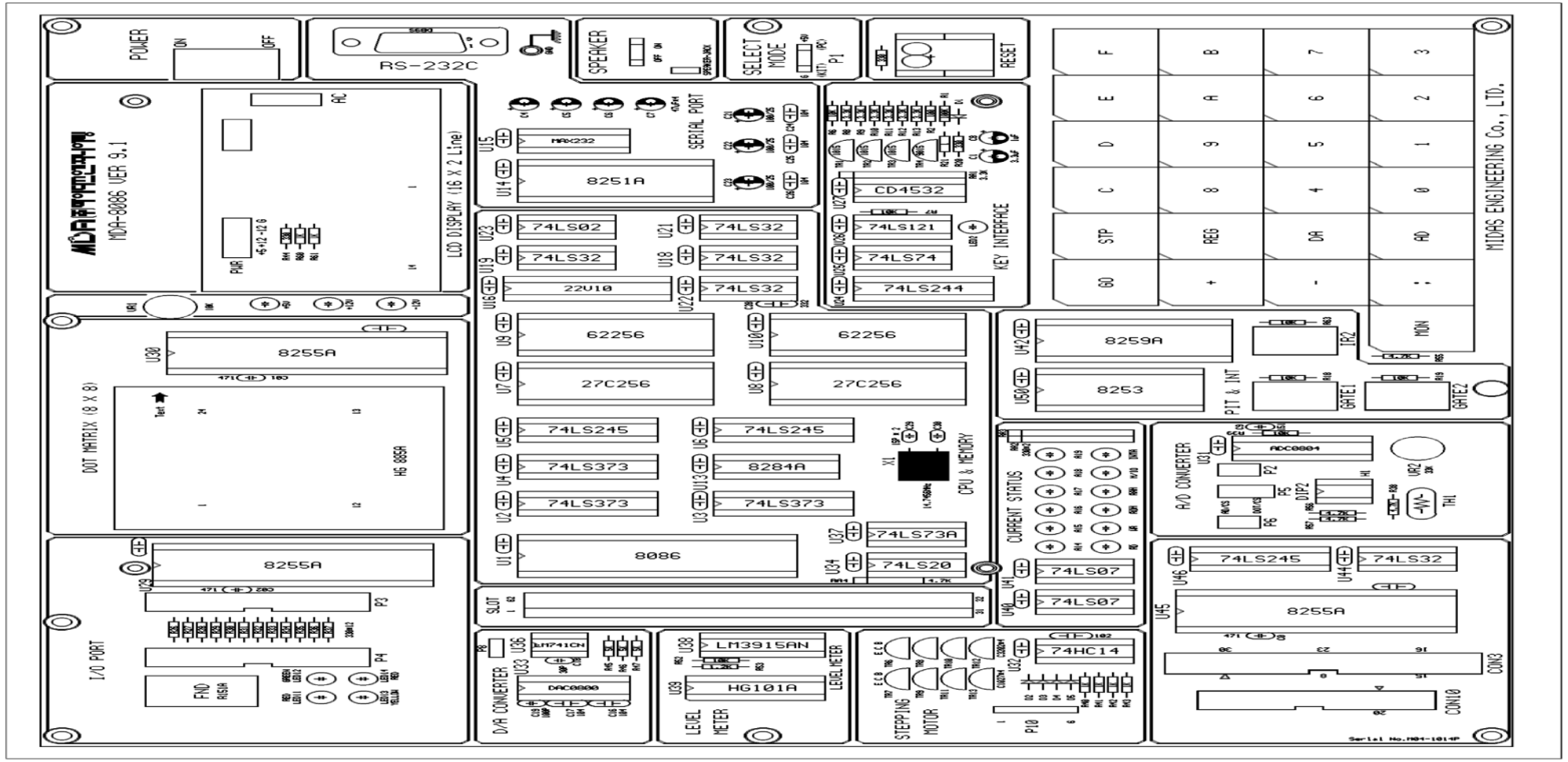
Operation of 8086

The 8086 microprocessor continually performs three steps in a cyclic manner-

1. **Fetch an instruction from memory(RAM).**
   - Specifically from the address ‘CS:IP’
2. **Decode the instruction**
   - In a manner so that the microprocessor control unit will be able to understand it. Operands necessary for this instruction are also fetched at this step.
3. **Execute the instruction**
   - This is the part of the cycle when data processing actually takes place. This step perform necessary calculation with ALU. Then write the appropriate value to the register or memory address.


1. Fetch
2. Decode
3. Execute

Basic Idea on MDA-8086 Trainer KIT



Keys of MDA-8086:

Function of Keys of MDA-8086:

Type 1 - CPU Control Keys

RES: If pressed and then released, it resets 8086 and starts from a cold state. After reset, the PC looks for a valid instruction at CS=FFFF, IP=0000.

NMI: *N*on *M*askable *I*nterrupt key. CPU interrupted immediately if pressed.

Type 2 - Command Keys

AD: Set memory *AD*dress key. Allows user to set 20-bit address of a memory location in the RAM, in the format of [Segment:Offset]. By pressing this key we enter into the 'address input' mode.

":" key allows switching from editing segment to editing offset during the set memory address operation.

Function of Keys of MDA-8086:

DA: This key brings cursor to the data field. User can use the hexadecimal keyboard for entering desired data into selected address.

"+": Move to the next memory location.

"-": Move to the previous memory location.

GO: Key to start the execution of a program. Pressing this button makes the system go to the beginning point of the program to be executed.

REG: To examine and change contents of the 8086 internal *REG*isters.

STP: Allows executing one instruction at a time.

Type 3 - Data Keys

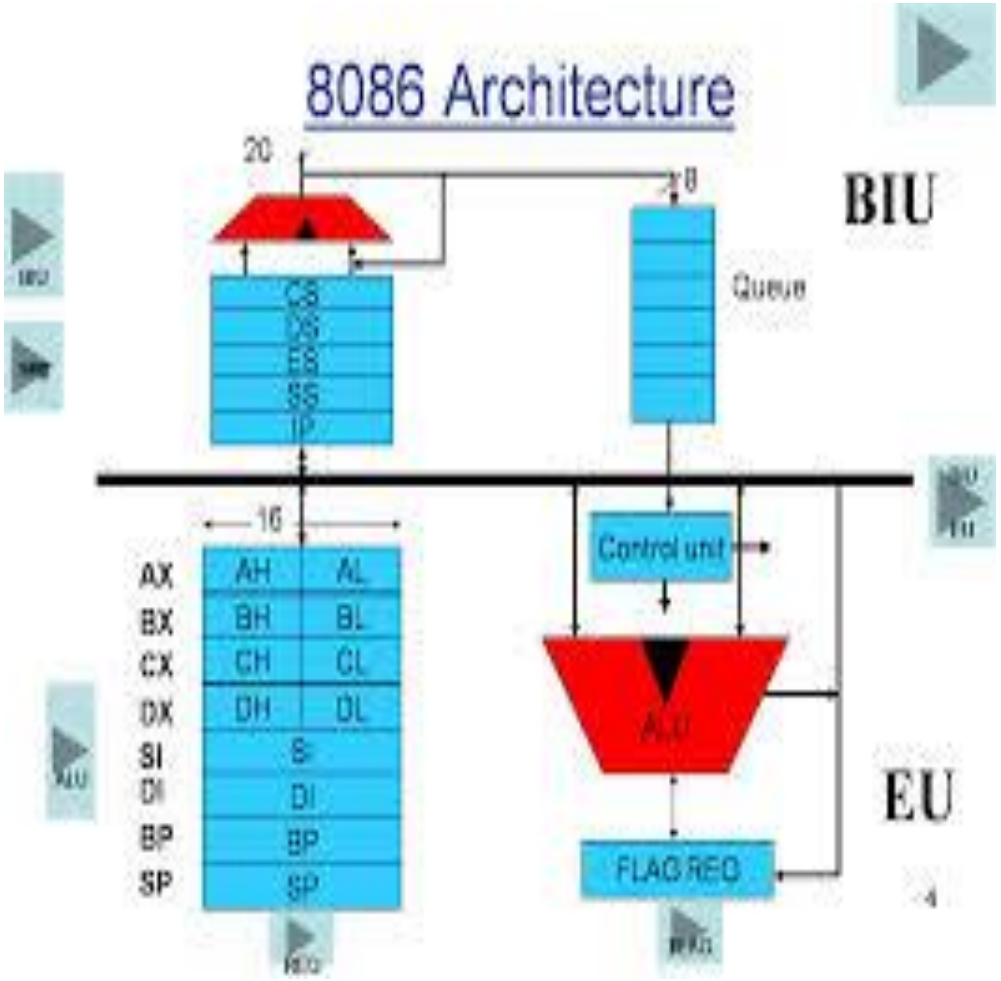
Hexademical keys with labels 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F.

To select the Machine Code and Serial Monitor with P1 Switch

Machine code

Serial Monitor

8086 Architecture

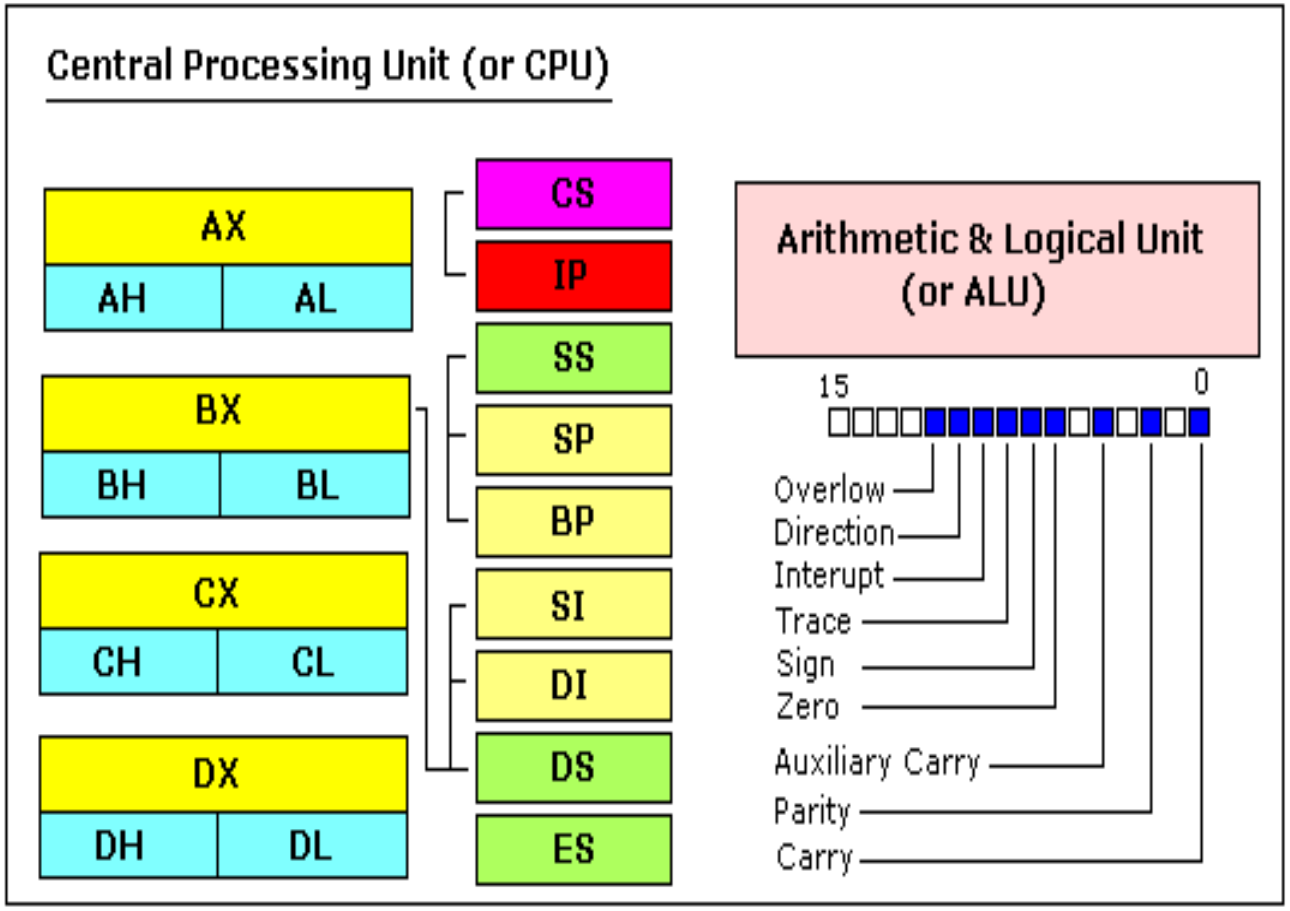


Information inside the microprocessor is stored in the Register.

Register are two types general purpose and segment register.

General purpose register stores data or *memory offsets* whereas segment register generate memory segment address.

8086 Architecture



8086 Architecture

GENERAL PURPOSE REGISTERS

8086 CPU has 8 general purpose registers, each register has its own name:

- AX **- the accumulator register (divided into** AH / AL**).**
- BX **- the base address register (divided into** BH / BL**)**
- CX **- the count register (divided into** CH / CL**)**
- DX **- the data register (divided into** DH / DL**)**
- SI **- source index register**
- DI **- destination index register**
- BP **- base pointer**
- SP **- stack pointer.**

8086 Architecture

SEGMENT REGISTERS:

- CS **- points at the segment containing the current program**
- DS **- generally points at segment where variables are defined**
- ES **- extra segment register, it's up to a coder to define its usage**
- SS **- points at the segment containing the stack**

CPU makes a calculation of physical address(an address into the RAM) by multiplying the segment register by 10h and adding general purpose register to it Example:

Segment = *1230H*
Offset = *0045H*

So, Physical Address
= *(Segment * 10H) + Offset*
= *(1230H * 10H) + 0045H*
= *12300H + 0045H*
= *12345H*

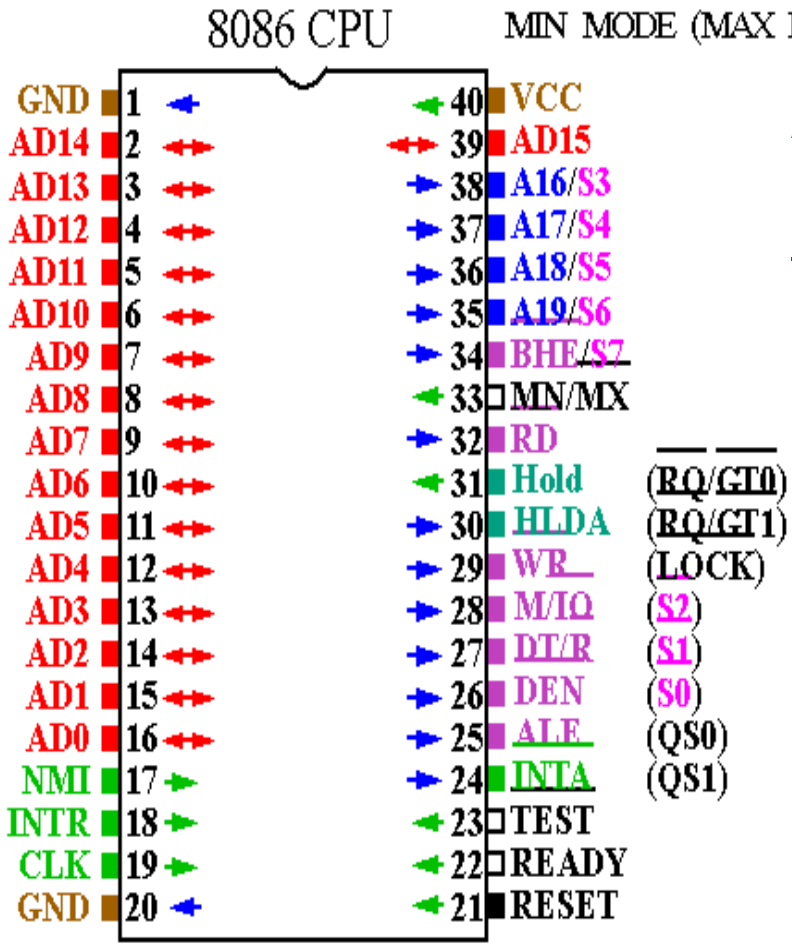
8086 Architecture

SPECIAL PURPOSE REGISTERS

- **IP** - The instruction pointer. IP register always works together with CS segment register and it points to currently executing instruction.

- **Flags** - Determines the current state of the processor. Flags Register is modified automatically by CPU after mathematical operations, this allows to determine the type of the result, and to determine conditions to transfer control to other parts of the program. Generally you cannot access these registers directly.

Basics of 8086 Architecture



A **memory segment** is a block of 64k consecutive memory bytes. Each segment is defined by a segment number starting with 0s. A memory location may be specified by providing a segment number and an offset.

KEY

AD

F

0

0

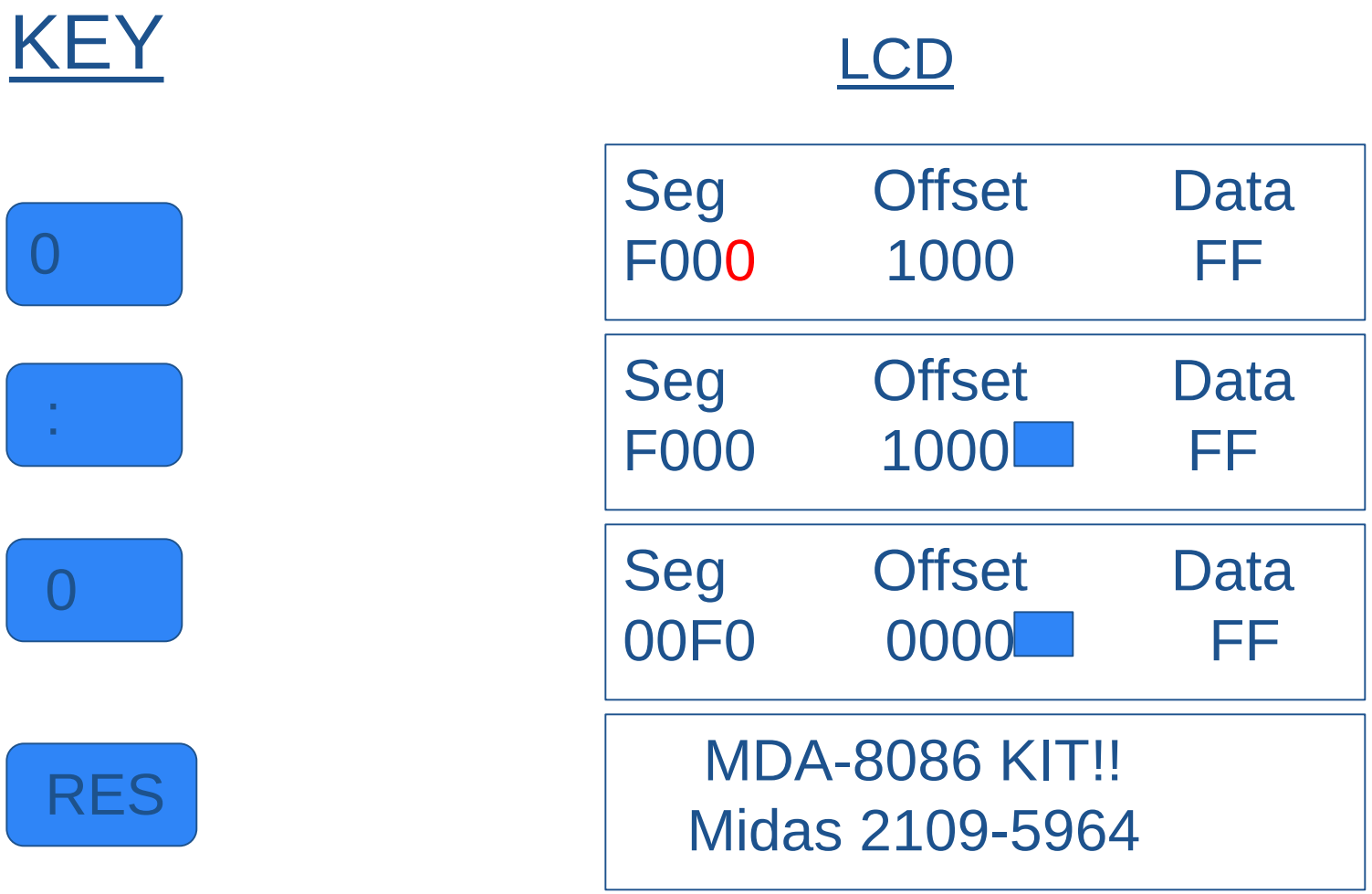
LCD

Seg	Offset	Data
0000	1000	FF

Seg	Offset	Data
000F	1000	FF

Seg	Offset	Data
00F0	1000	FF

Seg	Offset	Data
0F00	1000	FF


KEY
0
:
0
RES
LCD
Seg
Offset
Data
F000
1000
FF
Seg
Offset
Data
F000
1000
FF
Seg
Offset
Data
00F0
0000
FF
MDA-8086 KIT!!
Midas 2109-5964

KEY

AD

+

+

-

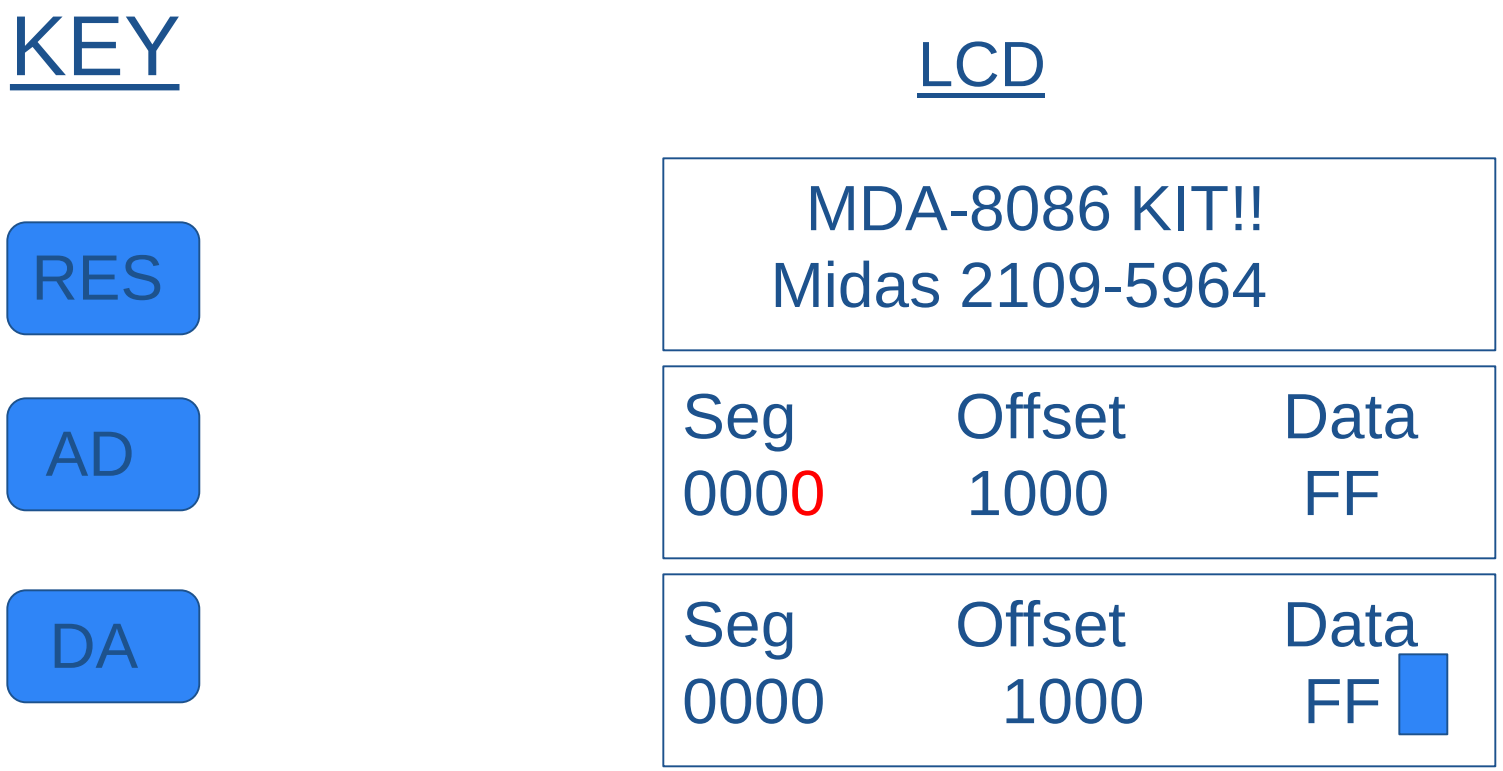
LCD

Seg	Offset	Data
0000	1000	FF

Seg	Offset	Data
0001	1000	FF

Seg	Offset	Data
0002	1000	FF

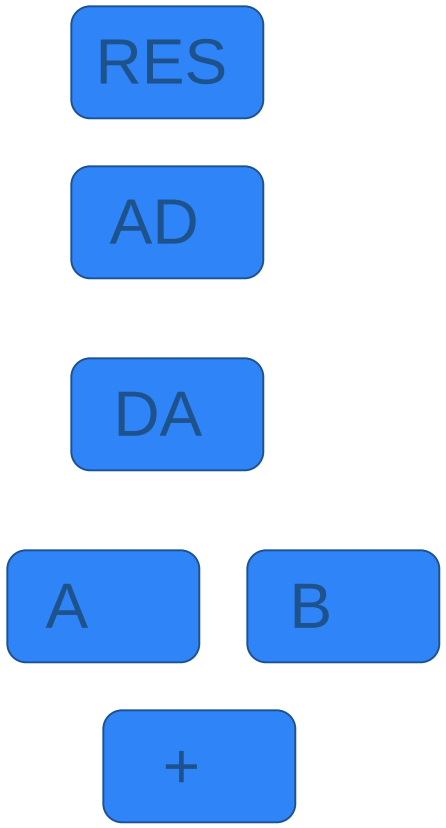
Seg	Offset	Data
0001	1000	FF



Let's Store the following like in your MDA-8086!

<Address>	<DATA>
01000	**AB**
01001	**CD**
01002	**EF**
01003	**34**

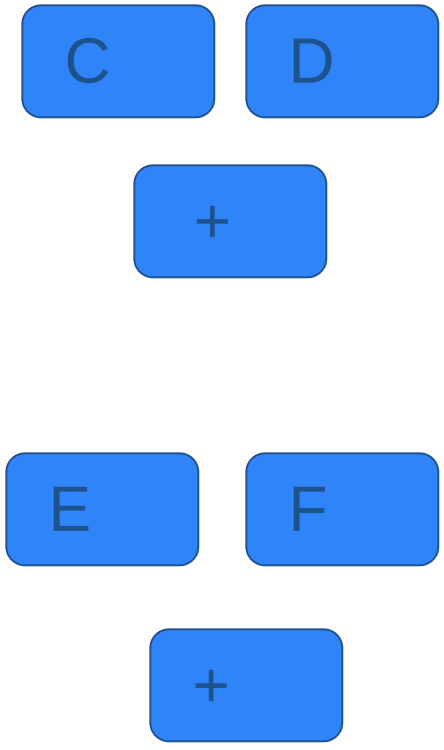
KEY



LCD

MDA-8086 KIT!! Midas 2109-5964		
Seg 0000	Offset 1000	Data FF
Seg 0000	Offset 1000	Data FF
Seg 0000	Offset 1000	Data AB
Seg 0000	Offset 1001	Data FF

KEY



LCD

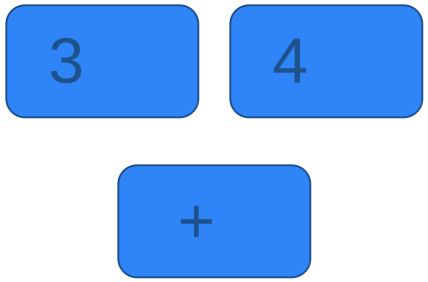
Seg	Offset	Data
0000	1001	CD

Seg	Offset	Data
0000	1002	FF

Seg	Offset	Data
0000	1002	EF

Seg	Offset	Data
0000	1003	FF

KEY



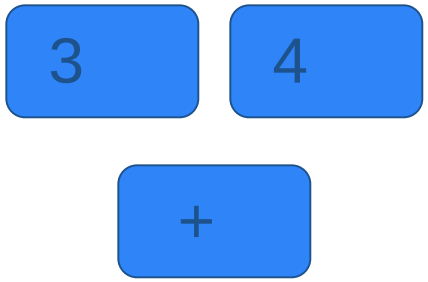
LCD

Seg	Offset	Data
0000	1003	34
Seg	Offset	Data
0000	1004	FF

Do it yourself!!

<Address>	<DATA>
01000	**FF**
01001	**CE**
01002	**DF**
01003	**43**

KEY



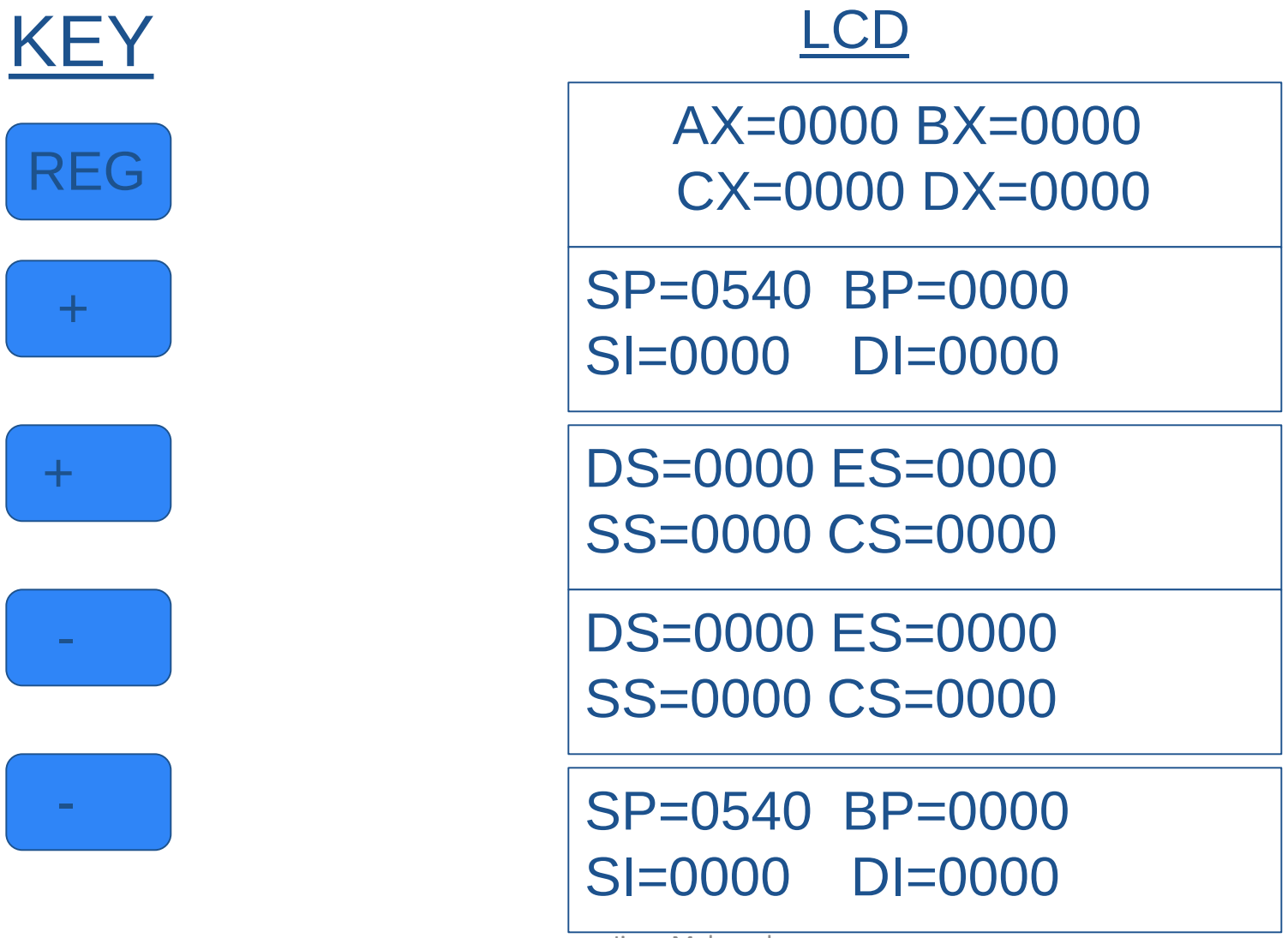
LCD

Seg	Offset	Data
0000	1003	34
Seg	Offset	Data
0000	1004	FF

Do it yourself!!

<Address>	<DATA>
01000	**FF**
01001	**CE**
01002	**DF**
01003	**43**

Experiment 1: Familiarization with the organization and the operating procedure of the MDA-8086 Learning system



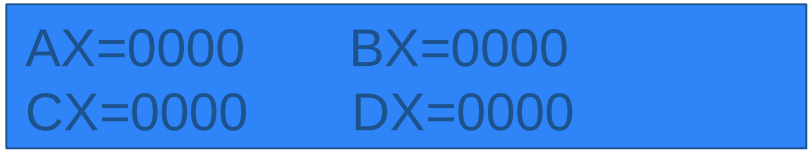
How to load Machine Instruction in MDA-8086

MOV destination, source
Operation: Copies the source operand to the destination

MOV AX,0000 Machine Code: B8 0000

Steps to load it in MDA-8086:

1.RES
2.AD
3.DA
4.B8
5.+
6.00
7.+
8.00
9.STP
10.+



Experiment 2: How to load Machine Instruction in MDA-8086

Do it yourself!!! I will check!

MOV AX,00F0H Machine Code:B8 F000

MOV BL,10H
Machine Code:B3 10

MOV DX,-1
Machine Code:BA FFFF

MOV AX,-1
Machine Code:B8 FFFF

MOV BX,1
Machine Code:BB 0100

Experiment 2: How to load Machine Instruction in MDA-8086

Do it yourself!!! I will check!

MOV AL,FFH Machine Code:B0 FF

MOV AL,F0H
Machine Code:B0 F0

MOV BX,1234H
Machine Code:BB 3412

MOV BL,11H
Machine Code:B3 11

MOV AX,F000H
Machine Code:B8 00F0

Experiment 2: How to load Machine Instruction in MDA-8086

Do it yourself!!! I will check!

ADD AX,4789
Machine Code:05 8947

ADD AL,88H
Machine Code:04 88

ADC AX,6488H
Machine Code:15 8864

MOV BL,11H
Machine Code:B3 11

MOV AX,F000H
Machine Code:B8 00F0

Experiment 3: How to store an assembly language programming in MDA-8086 kit

Program 1: MOV AX,05
ADD AX,03

Machine Code: B8 05
05 03

Steps:

- RES
- AD
- DA
- B8
- +
- 05
- +
- 00
- +
- 05
- +
- 03
- +
- 00
- STP +

How to load Machine Instruction in MDA-8086

Subtract: Sub destination, source
Operation: Subtracts source from destination. The result is placed in the destination.

SUB AX,3567H Machine Code:2D 6735
SBB AX,8000H Machine Code: 1D 0080

Multiplication: MUL source
Operation: The multiplier is the source operand which is either memory or register. For Byte multiplication the multiplicand is AL and for word multiplication the multiplicand is AX. The product is returned to AX.

MUL BL
Machine Code: F6 E3

ADD AX,4789H
Machine Code:05 8947

How to load Machine Instruction in MDA-8086

IMUL BX
Machine Code:F7 EB

DIVISION: DIV source
Operation: The divisor is the source operand which is either memory or register. For Byte division the dividend is AX for word division the dividend is DX. For word division the dividend is DX:AX. The quotient is returned to AH.

DIV BL
Machine Code: F6 F3

How to load Machine Instruction in MDA-8086

IDIV Source
Machine Code:F7 FB

ADC: Add with Carry
Operation: The carry Flag is added to the sum of the source and destination.

ADC AX,6488H
Machine Code:15 8864

How to load Machine Instruction in MDA-8086

DEC: Decrement
Format: DEC destination
Operation: decrements the destination operand by 1.

DEC AL
Machine Code: FE C8

INC: Increment
Format: Increment the destination operand by 1

INC AL
Machine Code: FE C0