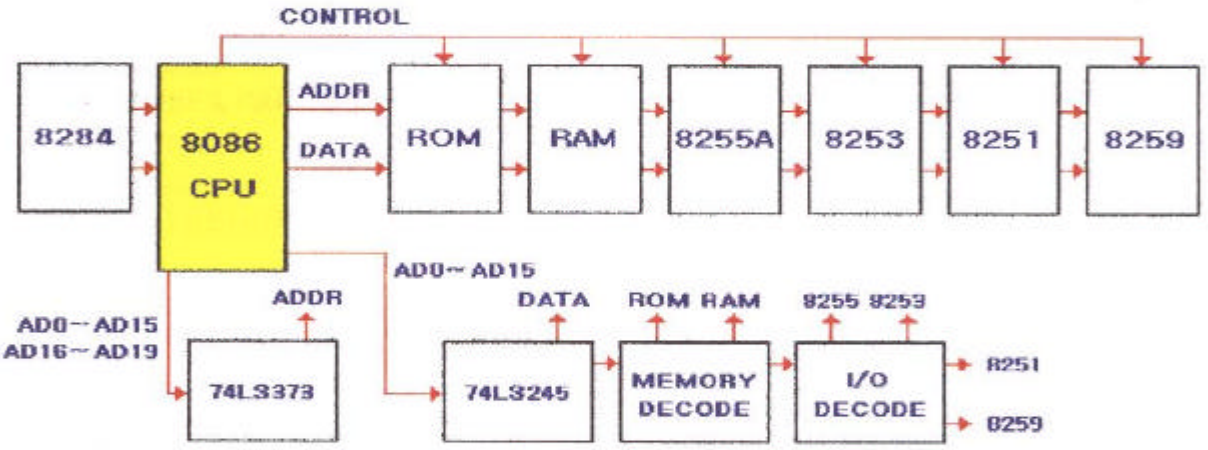
CHAPTER 1 COMPOSITION OF HARDWARE

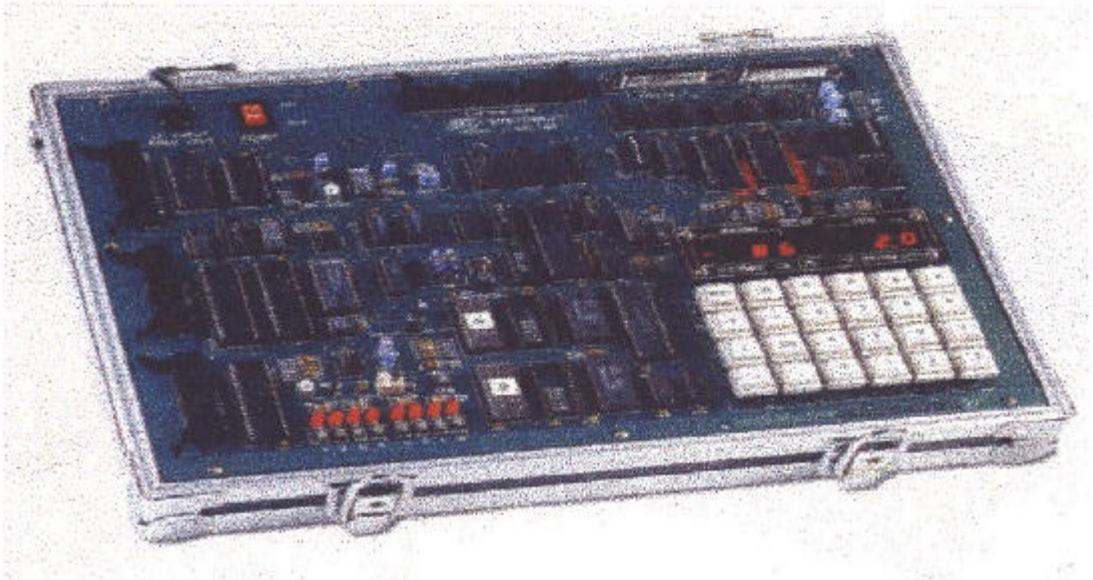
1. SYSTEM INTRODUCTION

The SKC-86 is composed of minimum mode used 8086 CPU of INTEL an the following are some of the main parts of SKC-86 :

8284 – Clock Generator
8255 – Parellel I/O
8251 – Seial Interface
8279 – Keypad/Display
27256 – ROM
62256 – RAM
8259 – Interrupt Control
8253 – Counter/Timer
ADC0809 – A/D (Analog to Digital) Converter
DAC0808 – D/A (Digital To Analog) Converter
LM380 – Amp etc.



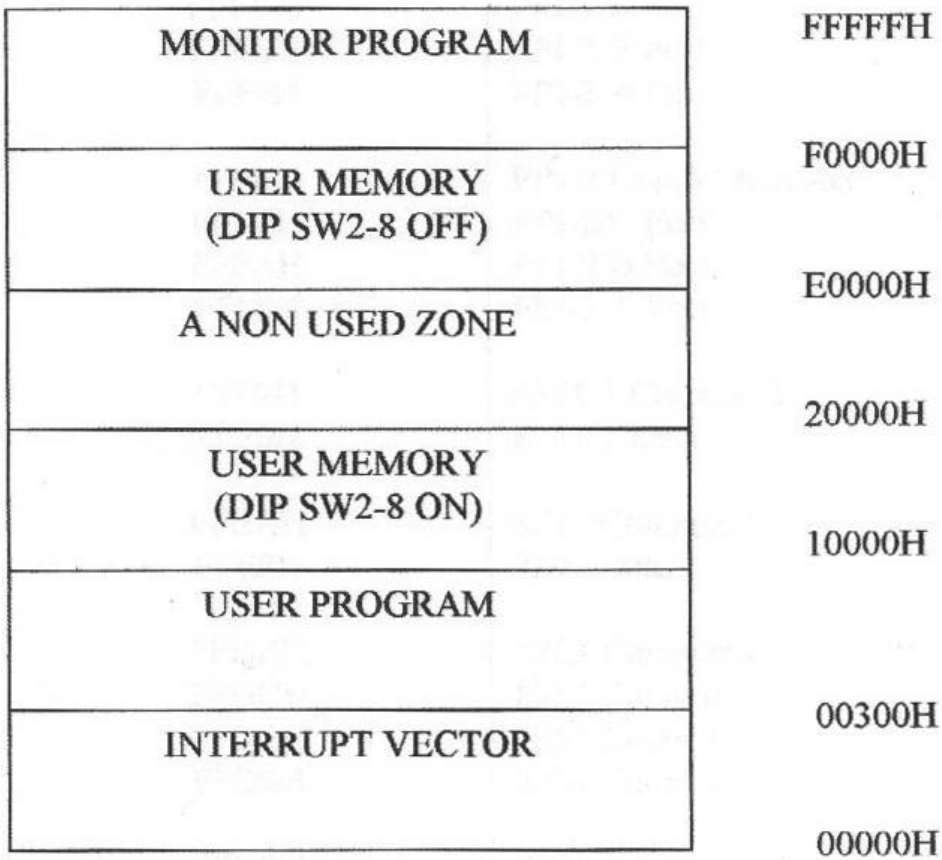
Above picture is the original 8086 CPU and its environment.



And also above you can see a picture of SKC86.

2. MEMORY MAP

You know that to do the operations on a system we need a memory to store, read and write information, to control the system. In SKC86 The memory map is composed of 6 individual parts and the adressation is as follows :



- User memory consists of double constuction in order to easily use ROM, RAM to be sequence of address
- A program which is downloaded on PC is saved at 1000h address
- In case of input of keyboard it can be used from 300h address
- In case of an experiment for interrupts is used setting interrupt vector

3 . I/O MAP

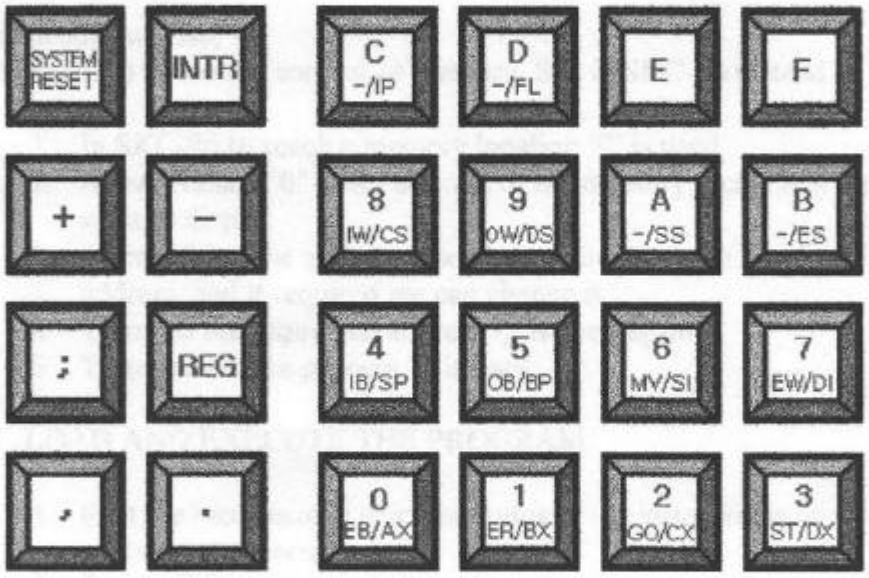
I/O Address	I/O PORT	Remarks
FFFFH	PPI-1 Control Register	
FFFDH	PPI-1 C Port	
FFFBH	PPI-1 B Port	
FFF9H	PPI-1 A Port	
FFFEH	PPI-2 Control Register	
FFFCH	PPI-2 C Port	
FFFAH	PPI-2 B Port	
FFF8H	PPI-2 A Port	RS-232 Port
FFF2H	8251-1 Command	
FFF0H	8251-1 Data	Keypad Control & Display
FFEAH	8297 Command	
FFE8H	8297 Data	Counter $Timer
FFDEH	8253 Command	
FFDCH	8253 Count 0	
FFDAH	8253 Count 1	
FFD8H	8253 Count 2	RS-232 Port
FFD2H	8251-2 Command	
FFD0H	8251-2 Data	Interrupt Controller
FFCAH	8259 Data(A0=High)	
FFC8H	825 Command (A0=Low)	
3FD8H	DA Converter	For Internal Test
3FD6H	PPI-3 Control Register	Output Test (8-bit) Input Test (8-bit)
3FD4H	PPI-3 C Port	
3FD2H	PPI-3 B Port	
3FD0H	PPI-3 A Port	
3FCEH	AD Converter IN 3 or 7	Change IN0-3 IN4-7 Into Short Pin
3FCCH	AD Converter IN 2 or 6	
3FCAH	AD Converter IN 1 or 5	
3FC8H	AD Converter IN 6 or 4	

CHAPTER 2 HOW TO USE SKC-86

1. HOW TO USE KEYPAD

Basically, the functions of some keys on the keypad of SKC-86 is :

',' to continue data coding or to continue observing the coded data
'.' to stop data coding or observing
'0' to see the memory content
'1' to see the register content
'2' to execute the code between the required addresses(starting & final addresses)
'3' to trace the codes



When you look at the keypad you will see some aliases. These aliases are instruction/register names. As an example :
GO/CX : GO is execution & CX register

2. REGISTER CONTENTS AND CHANGING

In 'debug' we use,
-r (register name) to see the content of the register. But in SKC-86 instead of 'r' we use '1' and instead of register name we use required key for that register.
For example: to see the CS content, first of all we press 1 and after it we press 8 which is for register CS.

1. To reach the register '1' is used
2. After pressing '1' press the key of the register you want to display the content
3. 0 for AX register
   1 for BX register

2 for CX register
3 for DX register
4 for SP register
5 for BP register
6 for SI register
7 for SI register
8 for CS register
9 for DS register
A for SS register
B for ES register
C for IP register
D for FL register

4. While displaying the content of the register you can enter a new value using the keypad
5. To terminate the process press '.'

3. MEMORY CONTENTS AND CHANGING

In 'debug' we use,
-e (address) to see the content of memory. But in SKC-86 instead of 'e' we use '0'.

1. In SKC-86 to reach a memory location '0' is used
2. After pressing '0' offset address of the memory location is written which we want to display
3. After writing the address when we press ',' we can see the1 byte data in that address, and if required we can change it.
4. To reach the sequential address ',' is used again
5. To terminate the process '.' is used

4 . LOAD AND EXECUTE THE PROGRAM

1. Find the hexadecimal machine codes of the instructions of the program that you want to execute
2. Set the CS register with the value where you want to restore the codes as described above (2)
3. As described in (3) go to the offset address and enter byte by byte the code
4. The starting address and the address after the last byte entered is the address that the program will operate
5. To terminate entering codes (.) is pressed
6. IP register is set to the starting offset of our code and (.) is pressed to terminate the process
7. After this it`s time to execute the program. After (2) (go) key is pressed the instruction code in CS:IP address is displayed. This is the starting address of our instruction and the code of the instruction.
8. To enter the address that out program will stop execution is entered by pressing (,) and the address which is the address after the last byte of our instruction codes is entered.
9. To start execution (.) is pressed
10. After the execution if (E) sysmbol is displayed this means that our program is in an infinite loop or there may be an error in the code. To get rid of this error

(RESET) key can be pressed and after this the system is restarts; so the codes must be reviewed.

11. If the execution finishes normally (br) sysmbol is displayed. After this register values are reviewed to see weather the program is correct or not.

5 . HOW TO TRACE A PROGRAM

1. The things that we mantioned in the above section (4) are repeated until 7th step.
2. To trace the code (3) (ST) key is pressed.You will see the first code of the first instruction in CS:IP.
3. After this step (,) is pressed to execute the code step by step.
4. If you want to see the register values while you are tracing the program, by pressing (.) you can pause tracing and control the addresses you want.
5. To continue tracing repeat step 2 and 3 . If you won`t change the value of IP during the trace operation it will continue from where you left.