Practical-1

AIM: To study basic terminology and architecture of a Microprocessor.

Terminology:

- **Instruction Set –**
- **Bus –**
- **Instruction Per Cycle –**
- **Bandwidth –**
- **Clock Speed –**
- **Word Length –**
- **Data Type –**

Architecture of a Microprocessor:

Definition: A microprocessor is nothing but the Central Processing Unit of a computer that has been constructed on a single chip.

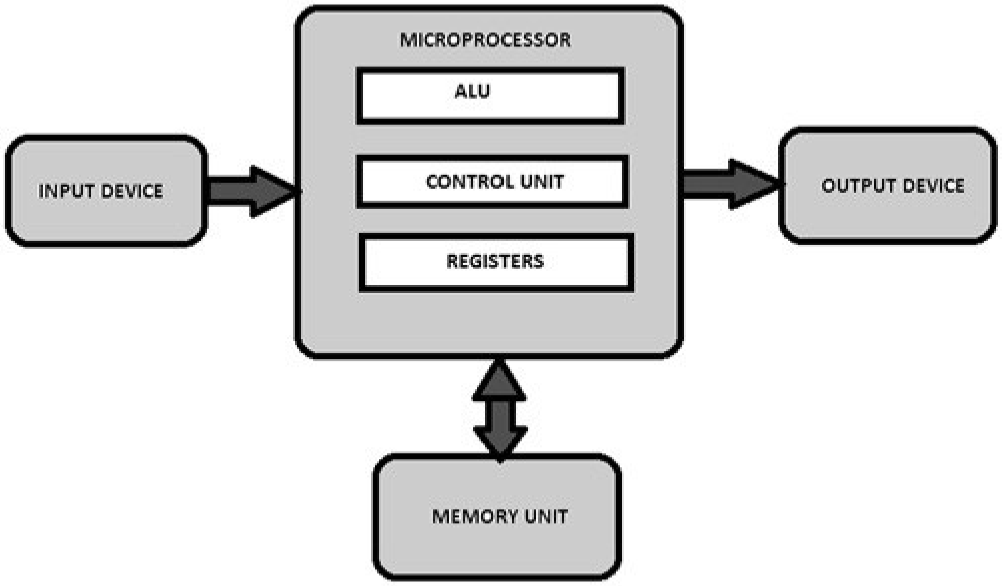
It is an integrated circuit and is able to implement all the important functions of the CPU. It is built on a silicon chip and is a clock-driven. The device is register-based.

It accepts binary data and produces the necessary output after processing the data on the basis of the instructions which are stored in the memory.

A Microprocessor takes a bunch of instructions in machine language and executes them, telling the processor what it has to do. Microprocessor performs three basic things while executing the instruction:

1. It performs some basic operations like addition, subtraction, multiplication, division and some logical operations using its Arithmetic and Logical Unit (ALU). New Microprocessors also perform operations on floating point numbers also.
2. Data in Microprocessor can move from one location to another.
3. It has a Program Counter (PC) register that stores the address of next instruction based on the value of PC, Microprocessor jumps from one location to another and takes decision.

A typical Microprocessor structure looks like this:



Each component of the architecture is discussed as below.

- **The Arithmetic and Logic Unit (ALU):**

  The arithmetic and logic Unit (ALU) is that part of the computer that actually performs arithmetic and logical operations on data. All other elements of the computer system - control unit, register, memory, I/O - are there mainly to bring data into the ALU to process and then to take the results back out.

  An arithmetic and logic unit and, indeed, all electronic components in the computer are based on the use of simple digital logic devices that can store binary digits and perform simple Boolean logic operations. Data are presented to the ALU in registers. These registers are temporary storage locations within the CPU that are connected by signal paths of the ALU.

  The ALU is the area of the computer in which arithmetic and logic operations are performed on data. The type of operation that is to be performed is determined by signals from the control unit. The data that are to be operated on by the ALU can come from either the memory unit or the input unit. Results of operations performed in the ALU can be transferred to either the memory unit for storage or the output unit.

- **Control Unit:**

  Control unit is like the conductor of an orchestra, who is responsible for keeping each of the orchestra members in proper synchronization. This unit contains logic and timing circuits that generate the proper signals necessary to execute each instruction in a program.

  The control unit fetches an instruction from memory by sending an address and read command. The instruction word stored at the memory location is then transferred to the control unit. This instruction word, which is in some form of binary code, is then decoded by logic circuitry in the control unit to determine which instruction is being called for. The control unit uses this information to send the proper signals to the rest of the units in order to execute the specified operation.

  This sequence of fetching an instruction code and then executing the indicated operation is repeated over and over by the control unit.

- **Memory Unit:**

  The memory section usually consists of a mixture of RAM and ROM. It may also have magnetic floppy disks, magnetic hard disk or laser optical disks, magnetic hard disk or laser optical disks. Memory has two purposes. The first purpose is to store the instructions (Program) that the computer is to perform. The second purpose of the memory is to store the data that are to be operated on by the program. The memory stores these instruction and data as groups of binary digits (words).

- **System Bus:**

  Two different-size arrows are used; the larger arrows represent data or information that actually consists of a relatively large number of parallel lines, and the smaller arrows represent control signals that are normally only one or a few lines. The various arrows are also numbered to allow
  easy reference to them in the following descriptions. These components are common communication path called a bus.

  - **Address Bus:** This is a unidirectional bus, because information flows over it in only one direction, from the CPU to the memory or I/O elements. The CPU alone can place logic levels on the lines of the address bus, thereby generating 216 = 65,536 different possible addresses. Each of these addresses corresponds to one memory location or one I/O element. When the CPU wants to communicate (read or write) with a certain memory location or I/O device, it places the appropriate 16-bit address code on its 16 address pin outputs, A0 through A15, and onto the address bus. These address bits are then decoded to select the desired memory location or I/O device.

  - **Data Bus:** This is a bi-directional bus, because data can flow to or from the CPU. The CPU's eight data pins, D0 through D7,can be either inputs or outputs, depending on whether the CPU is performing a read or a write operation. During data bus by the memory or I/O element. During a write operation the CPU's data pins act as outputs and place data on the data bus, which are then sent to the selected memory or I/O element. in all cases, the transmitted data words are 8bits wide because the CPU handles 8-bit data words, making this an 8-bit µC.

  - **Control Bus:** This is the set of signals that is used to synchronize the activities of the separate µC elements. Some of these control signals, such as RD and WR are sent by the CPU to the other elements to tell them what type of operation is currently in progress. The I/O elements can send control signals to the CPU. An example is the rest input (RES) of the CPU which, when driven LOW, causes the CPU to reset to a particular starting state.

- **Input Unit:**

  The input unit consists of all of the devices used to take information and data that are external to the computer and put them into the memory unit or the ALU. The control unit determines where the input information is sent. The input unit is used to enter the program and data into the
  memory unit or into the ALU from an external device during the execution of a program. Some of the common input devices are keyboards, Joysticks, mouse, OCR, OMR etc.

- **Output Unit:**

  The output unit consists of the devices used to transfer data and information from the computer to the "outside world." The output devices are directed by the control unit and can receive data from memory or the ALU. Examples of common output devices are printers, disk or tape units, video monitors, and digital-to-analog converters (DACs).

<u>Practical-2</u>

AIM: To study features and architecture of 8085 Microprocessor with its pin configuration.

Features:

Architecture of 8085:

- **Memory**

Program, data and stack memories occupy the same memory space. The total addressable memory size is 64 KB.
Program memory - program can be located anywhere in memory. Jump, branch and call instructions use 16-bit addresses, i.e. they can be used to jump/branch anywhere within 64 KB. All jump/branch instructions use absolute addressing.
Data memory - the data can be placed anywhere as the 8085 processor always uses 16-bit addresses.
Stack memory is limited only by the size of memory. Stack grows downward.
First 64 bytes in a zero-memory page should be reserved for vectors used by RST instructions.

- **Interrupts**

The 8085 microprocessor has 5 interrupts. They are presented below in the order of their priority (from lowest to highest):
INTR is maskable 8080A compatible interrupt. When the interrupt occurs the processor fetches from the bus one instruction, usually one of these instructions:
One of the 8 RST instructions **(RST0 - RST7).** The processor saves current program counter into stack and branches to memory location N * 8 (where N is a 3-bit number from 0 to 7 supplied with the RST instruction).
CALL instruction (3 byte instruction). The processor calls the subroutine, address of which is specified in the second and third bytes of the instruction.
RST5.5 is a maskable interrupt. When this interrupt is received the processor saves the contents of the PC register into stack and branches to 2Ch (hexadecimal) address.
RST6.5 is a maskable interrupt. When this interrupt is received the processor saves the contents of the PC register into stack and branches to 34h (hexadecimal) address.
RST7.5 is a maskable interrupt. When this interrupt is received the processor saves the contents of the PC register into stack and branches to 3Ch (hexadecimal) address.
Trap is a non-maskable interrupt. When this interrupt is received the processor saves the contents of the PC register into stack and branches to 24h (hexadecimal) address.
All maskable interrupts can be enabled or disabled using EI and DI instructions. RST 5.5, RST6.5 and RST7.5 interrupts can be enabled or disabled individually using SIM instruction.

- **I/O ports**

256 Input ports
256 Output ports

- **Registers**

Accumulator or A register is an 8-bit register used for arithmetic, logic, I/O and load/store operations.
Flag is an 8-bit register containing 5 1-bit flags:

Sign - set if the most significant bit of the result is set.

Zero - set if the result is zero.

Auxiliary carry - set if there was a carry out from bit 3 to bit 4 of the result.

Parity - set if the parity (the number of set bits in the result) is even.

Carry - set if there was a carry during addition, or borrow during subtraction/comparison.

General registers:

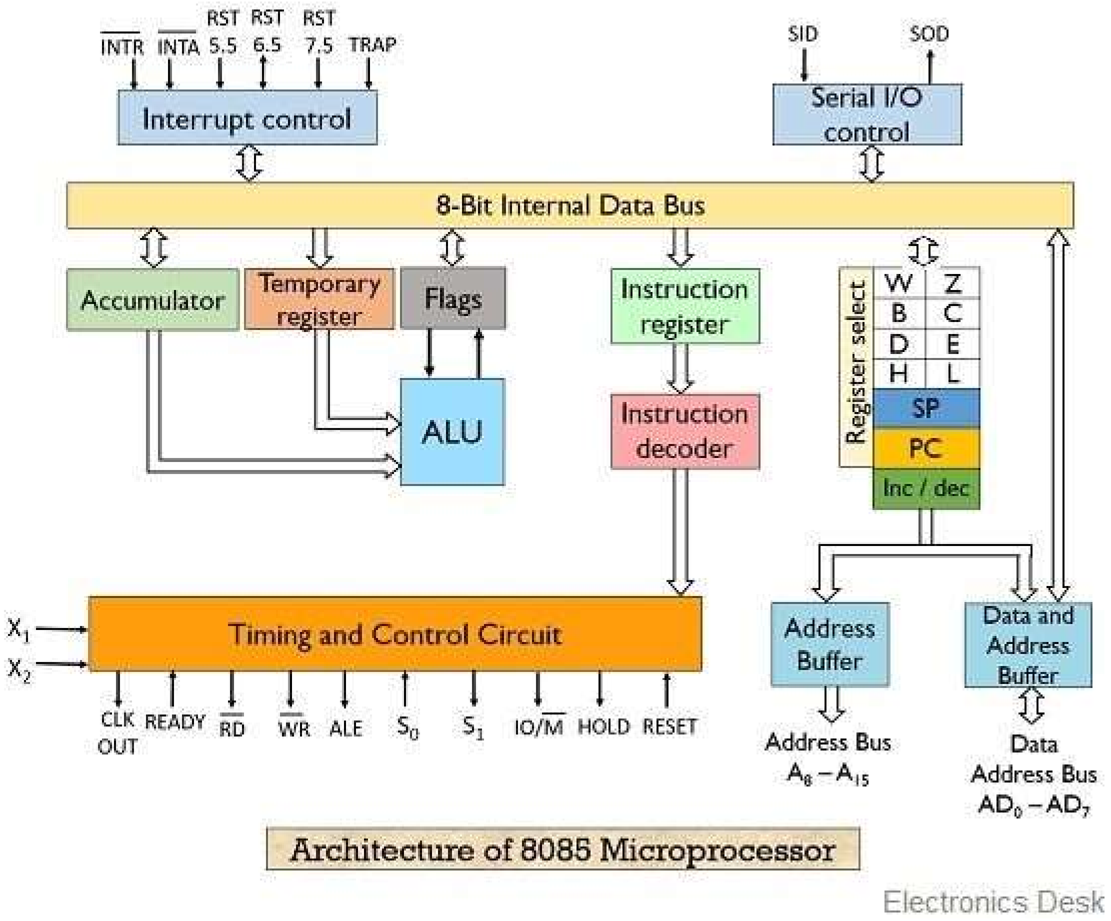
8-bit B and 8-bit C registers can be used as one 16-bit **BC** register pair. When used as a pair the C register contains low-order byte. Some instructions may use BC register as a data pointer.

8-bit D and 8-bit E registers can be used as one 16-bit **DE** register pair. When used as a pair the E register contains low-order byte. Some instructions may use DE register as a data pointer.

8-bit H and 8-bit L registers can be used as one 16-bit **HL** register pair. When used as a pair the L register contains low-order byte. HL register usually contains a data pointer used to reference memory addresses.

Stack pointer is a 16 bit register. This register is always incremented/decremented by 2.

Program counter is a 16-bit register.

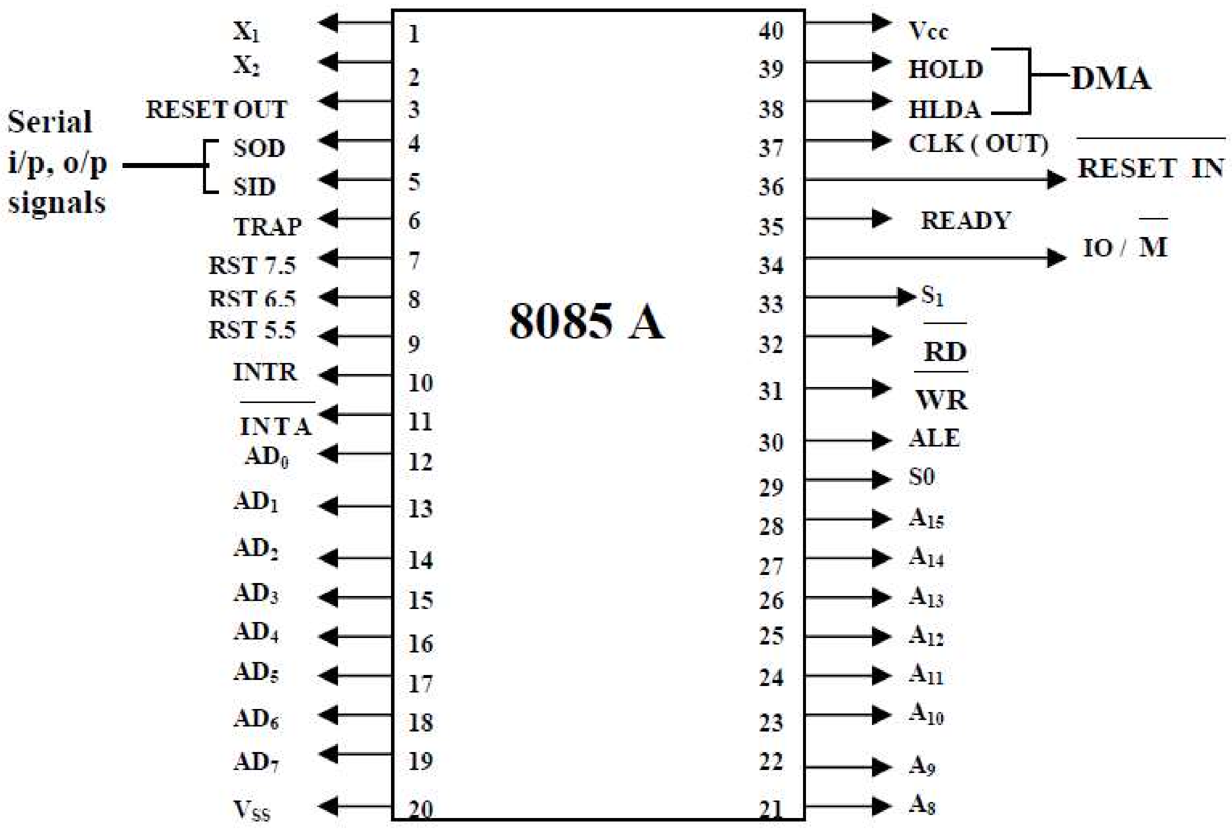


The descriptions of various pins are as follows:

Address Bus and Data Bus

A8 ? A15 (Output): These are address bus and are used for the most significant bits of the memory address or 8-bits of I/O address.

AD0 ? AD7 (Input/output): These are time multiplexed address/data bus i.e. they serve dual purpose. They are used for the least significant 8 bits of the memory address or I/O address during the first cycle. Again, they are used for data during 2nd and 3rd clock cycles.



Pin Diagram of 8085

Control and Status Signals

ALE (Output): ALE stands for Address Latch Enable signal. ALE goes high during first clock cycle of a machine cycle and enables the lower 8-bits of the address to be latched either into the memory or external latch.

IO/M (Output): It is a status signal which distinguishes whether the address is for memory or I/O device.

S0, S1 (Output): These are status signals sent by the microprocessors to distinguish the various types of operation.

RD (Output): RD is a signal to control READ operation. When it goes low, the selected I/O device or memory is read.

WR (Output): WR is a signal to control WRITE operation. When it goes low, the data bus' data is written into the selected memory or I/O location.

READY (Input): It is used by the microprocessor to sense whether a peripheral is ready to transfer a data or not. If READY is high, the peripheral is ready. If it is low the microprocessor waits till it goes high.

Interrupts and Externally Initiated Signals

HOLD (INPUT): HOLD indicates that another device is requesting for the use of the address and data bus.

HLDA (OUTPUT): HLDA is a signal for HOLD acknowledgement which indicates that the HOLD request has been received. After the removal of this request the HLDA goes low.

INTR (Input): INTR is an Interrupt Request Signal. Among interrupts it has the lowest priority. The INTR is enabled or disabled by software.

INTA (Output): INTA is an interrupt acknowledgement sent by the microprocessor after INTR is received.

RST 5.5, 6.5, 7.5 and TRAP (Inputs): These all are interrupts. When any interrupt is recognized the next instruction is executed from a fixed location in the memory.

Reset Signals

RESET IN (Input): It resets the program counter (PC) to 0. It also resets interrupt enable and HLDA flip-flops. The CPU is held in reset condition till RESET is not applied.

RESET OUT (Output): RESET OUT indicates that the CPU is being reset.

Clock Signals

X1, X2 (Input): X1 and X2 are terminals to be connected to an external crystal oscillator which drives an internal circuitry of the microprocessor. It is used to produce a suitable clock for the operation of microprocessor.

CLK (Output): CLK is a clock output for user, which can be used for other digital ICs. Its frequency is same at which processor operates.

Serial I/O Signals

SID (Input): SID is data line for serial input. The data on this line is loaded into the seventh bit of the accumulator when RIM instruction is executed.

SOD (Output): SOD is a data line for serial output. The seventh bit of the accumulator is output on SOD line when SIM instruction is executed.

Power Supply

Vcc : +5 Vlots supply

Vss : ground reference

Practical-3

AIM: To study and execute basic instructions of 8085.

1. **Store the data byte 32H into memory location 4000H.**

2. **Subtract the contents of memory location 4001H from the memory location 2000H and place the result in memory location 4002H.**

3. **Add the 16-bit number in memory locations 4000H and 4001H to the 16-bit number in memory locations 4002H and 4003H. The most significant eight bits of the two numbers to be added are in memory locations 4001H and 4003H. Store the result in memory locations 4004H and 4005H with the most significant byte in memory location 4005H.**

4. **Find the 2's complement of the number stored at memory location 4200H and store the complemented number at memory location 4300H.**

5. **Pack the two unpacked BCD numbers stored in memory locations 4200H and 4201H and store result in memory location 4300H. Assume the least significant digit is stored at 4200H.**

6. **Write a set of instructions to alter the contents of flag register in 8085.**

7. **Calculate the sum of series of numbers. The length of the series is in memory location 4200H and the series begins from memory location 4201H.**
    a. **Consider the sum to be 8 bit number. So, ignore carries. Store the sum at memory location 4300H.**
    b. **Consider the sum to be 16 bit number. Store the sum at memory locations 4300H and 4301H**

a.

b.

8. Multiply two 8-bit numbers stored in memory locations 2200H and 2201H by repetitive addition and store the result in memory locations 2300H and 2301H.

9. **Divide 16 bit number stored in memory locations 2200H and 2201H by the 8 bit number stored at memory location 2202H. Store the quotient in memory locations 2300H and 2301H and remainder in me Sample problem**

10. **Find the number of negative elements (most significant bit 1) in a block of data. The length of the block is in memory location 2200H and the block itself begins in memory location 2201H. Store the number of negative elements in memory location 2300H.**

11. **Divide the 16-bit unsigned number in memory locations 2200H and 2201H (most significant bits in 2201H) by the B-bit unsigned number in memory location 2300H store the quotient in memory location 2400H and remainder in 2401H.**

Assumption: The most significant bits of both the divisor and dividend are zero.

12. DAA instruction is not present. Write a sub routine which will perform the same task as DAA.

Practical-4

AIM: To study interfacing memory as well as I/O elements with a Microprocessor.

Memory Interfacing:

An 8085 microprocessor has a 16-bit address bus. Each bit can take the value of either 0 or 1. So, the total number of addresses that can be generated on a 16-bit address bus will be 256. And each unique address refers to a memory block containing 8 bits or 1 byte of space.

$$2^{16} \text{ bytes} = 1024 * 2^{6} \text{ bytes} = 2^{6} \text{ kiloBytes} = 64 \text{ kB}$$

Thus, we can say that 8085 can support a memory chip of size up to 64 kB. We can interface a memory chip of size less than that too. Also, we can interface several memory chips to a single 8085 microprocessor, until and unless their combined size does not exceed 64 kB. Let us learn how to achieve all that.

Memory chips of different types and sizes

Memory chips come in a variety of types and with different storage capacities. A broad classification of memory chips based on their read and write capability is:

- **RAM (Random Access Memory):** We can read as well as write data on this type of memory. The chip of this type has pins for both memory read and memory write signals.
- **ROM (Read Only Memory):** As the name suggests, we can only write data on this type of memory chip. The chip of this type has a pin only for memory read signal.

Now, you must be wondering what does a microprocessor read from a ROM if data cannot be written on it.

Data cannot be written on it by a microprocessor when it is connected in the circuit. But data can be written on it using some special techniques. This kind of memory is used to store programs, while RAM is used to store the data.

Example -1: Interface a 1kB EPROM and a 2 kB RAM with microprocessor 8085. The address allotted to 1 kB EPROM should be 2000H to 22FFH. You can assign the address range of your choice to the 2 kB RAM.

starting and ending address of the 1kB EPROM:

3000H = 0011 0000 0000 0000

33FFH = 0011 0011 1111 1111

Address bit number	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
Starting address	**0**	**0**	**1**	**1**	**0**	**0**	0	0	0	0	0	0	0	0	0	0
Ending Address	**0**	**0**	**1**	**1**	**0**	**0**	1	1	1	1	1	1	1	1	1	1

From the above table, we can observe that ten bits from A0 to A9 are changing. These Meanwhile, bits A11 to A15 do not change and don't have any effect on the addressing process inside the memory chip. So, we can conclude that the values of bits A15-A11 (0011 00) given in the above table are in a unique, unchanging configuration for this memory chip. If even one of these bits changes, the address won't belong to this memory chip. **So, we can use these values of A15-A11 to uniquely identify this memory chip, which is exactly what the CS signal is supposed to do.**

We can say that when A15 = A14 = A11 = A10 = 0 and A13 = A12 = 1, then our memory chip should be selected. Now, we need to design the logic to generate the CS signal. The resulting Boolean equation of CS will be:

CS = Complement of (A15* . A14* . A13 . A12 . A11* . A10*)

Now, we have to generate a chip select signal for the second memory chip, which is 2kB RAM. The process is quite similar and differs from the previous one in two ways:

The size of the memory is different. So, there are 11 address lines instead of 10.

We are not given an address range here. We are given the liberty to decide on our own.

Similar to the previous case, we connect the first 11 address lines of the 8085 microprocessor to the 11 address lines of the 2kB RAM. These bits will take values of 0 and 1 and will generate 2 * 1024 different addresses. The address bits A10-A0 will vary from 000 0000 0000 to 111 1111 1111.

What about the remaining address bits? Well, they don't have any role in the addressing of the memory in this 2kB RAM. So, we can fix them to a certain value without affecting anything. Let's fix them to 0000

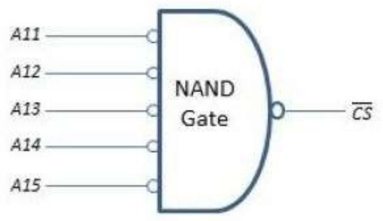
0. Thus, the address range for this chip becomes 0000 0000 0000 0000 to 0000 0111 1111 1111. In hexadecimal, the address range will be from 0000H to 07FFH.

Address bit number	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
Starting address	**0**	**0**	**0**	**0**	**0**	0	0	0	0	0	0	0	0	0	0	0
Ending Address	**0**	**0**	**0**	**0**	**0**	1	1	1	1	1	1	1	1	1	1	1

We use a similar technique here. We use the remaining bits A15-A11 to uniquely identify this chip i.e., to generate chip select signal. So, the boolean equation will be

CS = Complement of (A15* . A14* . A13* . A12* . A11*)

The implementation of this equation using NAND Gate to generate the CS signal is shown in the following image.



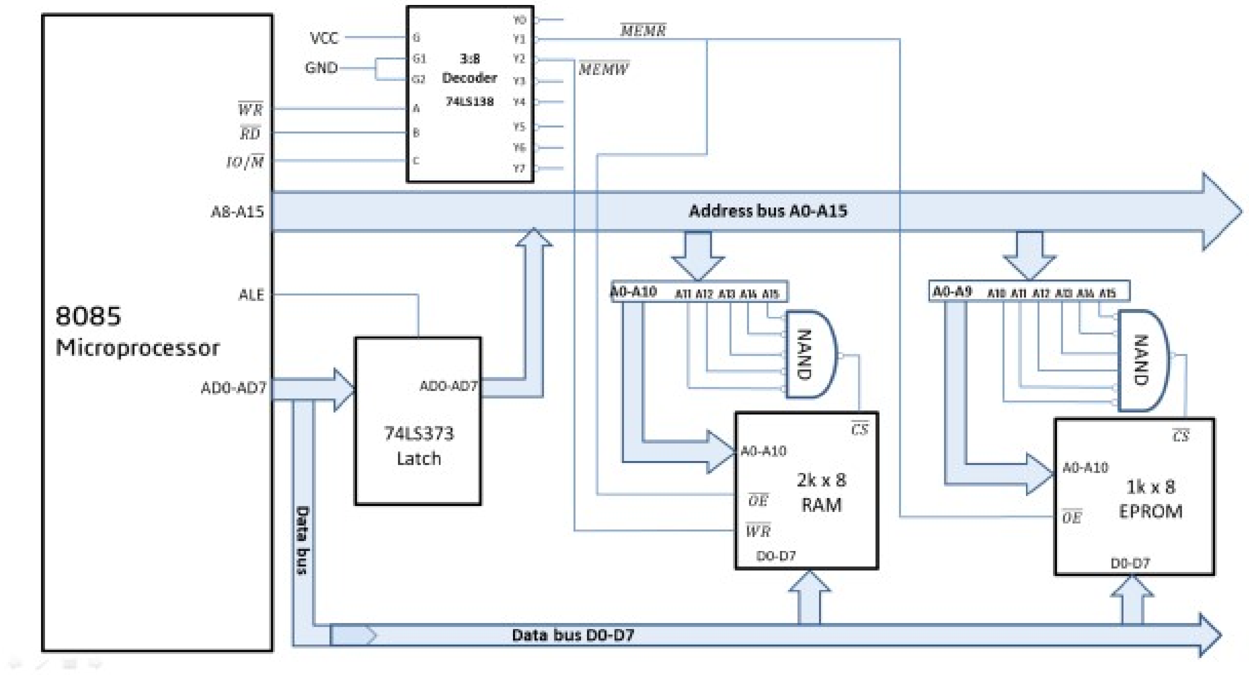
Chip select logic for 2k x 8 RAM

The final circuit:

Since we now have the chip select logic and have decided all the connections, it's time to finalize the circuit. The entire external memory interfacing circuit can be broken up into five different parts:

- 8085 microprocessor
- Demultiplexing of address/data bus
- Generation of control signals
- Generation of chip select signals
- Memory chips

The images below show the final circuit with all the five parts listed above integrated into a single circuit. Just the connections are shown in the first diagram. In the diagram following it, different subsections of the circuit are labeled.



Exercise:

1. Interface a 4kB EPROM and a 2 kB RAM with microprocessor 8085. EPROM and RAM chips are available in size of 2kB. (Draw below.)

2. Interface a 2kB EPROM and a 4kB RAM with microprocessor 8085. Starting address of RAM should 1000 H and starting address of RAM can be anything. (Draw below.)

3. Write a brief note on I/O interfacing in 8085. (Draw below.)

<u>**Practical-5**</u>

AIM: To study features of advanced processors.

Features of 8086:

Features of 80286:

Features of 80386: