Assembler vs Compiler.

Factor	Compiler	Assembler
Operation	Compilers translate high level programming language code to machine level code	Assemblers convert the assembly level language to machine level code.
Input	High level programming language.	Assembly level code.
Conversion type	Compilers check and convert the complete code at one time.	Assemblers generally do not convert complete code at one time.
Output	Mnemonic version of machine code.	Binary version of machine code.
Examples	C, C++ , Java compilers.	GAS, GNU assemblers.

$\overline{TEST}$ vs. TEST in 8086.

$\overline{TEST}$	TEST
It is a signal	It is a logical instruction
If low, execution continue, else, processor remains in an idle state	Performs bit by bit logical AND operation on the two operands.
Input is examined by WAIT instruction	Operand maybe register, memory or immediate data.
No example	e.g. TEST AX,BX

Maskable vs. Non Maskable interrupts.

Features	Maskable Interrupts	Non-Maskable Interrupts
Definition	Maskable interrupts are those that may be accepted or ignored by the CPU based on their priority.	The CPU must accept the non-maskable interrupts.
Priority	It aids in the management of low-priority jobs.	It aids in the management of high-priority jobs.
Response Time	It has a higher response time.	It has a very low response time.
Examples	INTR	NMI

Tightly coupled vs. loosely coupled configuration.

Loosely Coupled	Tightly Coupled
Distributed memory is present	Shared memory is present.
Low data rate.	High data rate.
Low cost	More cost
Memory conflicts absent	Memory conflicts present
Low degree of interaction between tasks.	High degree of interaction between tasks.
Used in distributed computing systems.	Used in parallel processing systems.

Synchronous vs. Asynchronous mode of 8251.

Synchronous mode	Asynchronous mode
It uses a clock signal to synchronize the data transfer between 8251 and other device.	Data transfer is synchronized by the start and stop bits that are prepended to each data byte.
High data throughput	Low data throughput
Applications: Networking, high-speed data transfer	Applications: Serial communication, low-speed data transfer

Pipelining vs. memory segmentation.

Pipelining	Memory Segmentation
It is the feature of fetching the next instruction while executing the current instruction.	It is the process in which the main memory of the computer is logically divided into different segments and each segment has its own base address.
It allows the CPU to execute multiple instructions at the same time.	It divides the memory space into multiple segments.
It improves the performance of the CPU	It allows the CPU to access different parts of the memory at the same time.
It can be complex to implement and debug	It can add overhead to the memory access time

Mode 0 vs. Mode 1 of IC-8253.

Feature	Mode 0	Mode 1
Purpose	To generate an interrupt after a certain interval	To generate a one-shot pulse
Output	Low when the count reaches zero	Low for a specified period of time
Applications	Periodic tasks, polling sensors, updating displays	Timing functions, triggering cameras

What is the use of 6-byte instruction queue in 8086?

To speed up program execution, the BIU fetches six instruction bytes ahead of time from the memory. These prefetched instruction bytes are held for the execution unit in a group of registers called Queue.

The process of fetching the next instruction when the present instruction is being executed is called as pipelining.

Advantages: Increased performance, Increased throughput and Reduced latency.

Short notes:- (a) Microcontroller Architecture (IC-8051c) (b) Pentium Processors. (c) 8085 Addressing modes.

Microcontroller Architecture (IC-8051c)

Accumulator (ACC) The accumulator register (ACC or A) acts as an operand register, in case of some instructions.

B Register This register is used to store one of the operands for multiply and divide instructions.

Program Status Word (PSW) This set of flags contains the status information

Stack Pointer (SP) This 8-bit wide register is incremented before the data is stored onto the stack

Data Pointer (DTPR) This 16-bit register contains a higher byte (DPH) and the lower byte (DPL) of a 16-bit external data RAM address.

Port 0 to 3 Latches and Drivers These four latches and driver pairs are allotted to each of the four on-chip I/O ports.

Serial Data Buffer The serial data buffer internally contains two independent registers.

Timer Registers These two 16-bit registers can be accessed as their lower and upper bytes.

Control Registers The special function registers contain control and status information for interrupts, timers/counters and serial port.

Timing and Control Unit This unit derives all the necessary timing and control signals required for the internal operation of the circuit.

Oscillator This circuit generates the basic timing clock signal for the operation of the circuit using crystal oscillator.

Instruction Register This register decodes the opcode of an instruction to be executed

EPROM and Program Address Register These blocks provide an on-chip EPROM and a mechanism to internally address it.

RAM and RAM Address Register These blocks provide internal 128 bytes of RAM and a mechanism to address it internally.

ALU The arithmetic and logic unit performs 8-bit arithmetic and logical operations

SFR Register Bank This is a set of special function registers, which can be addressed using their respective addresses which lie in the range 80H to FFH.

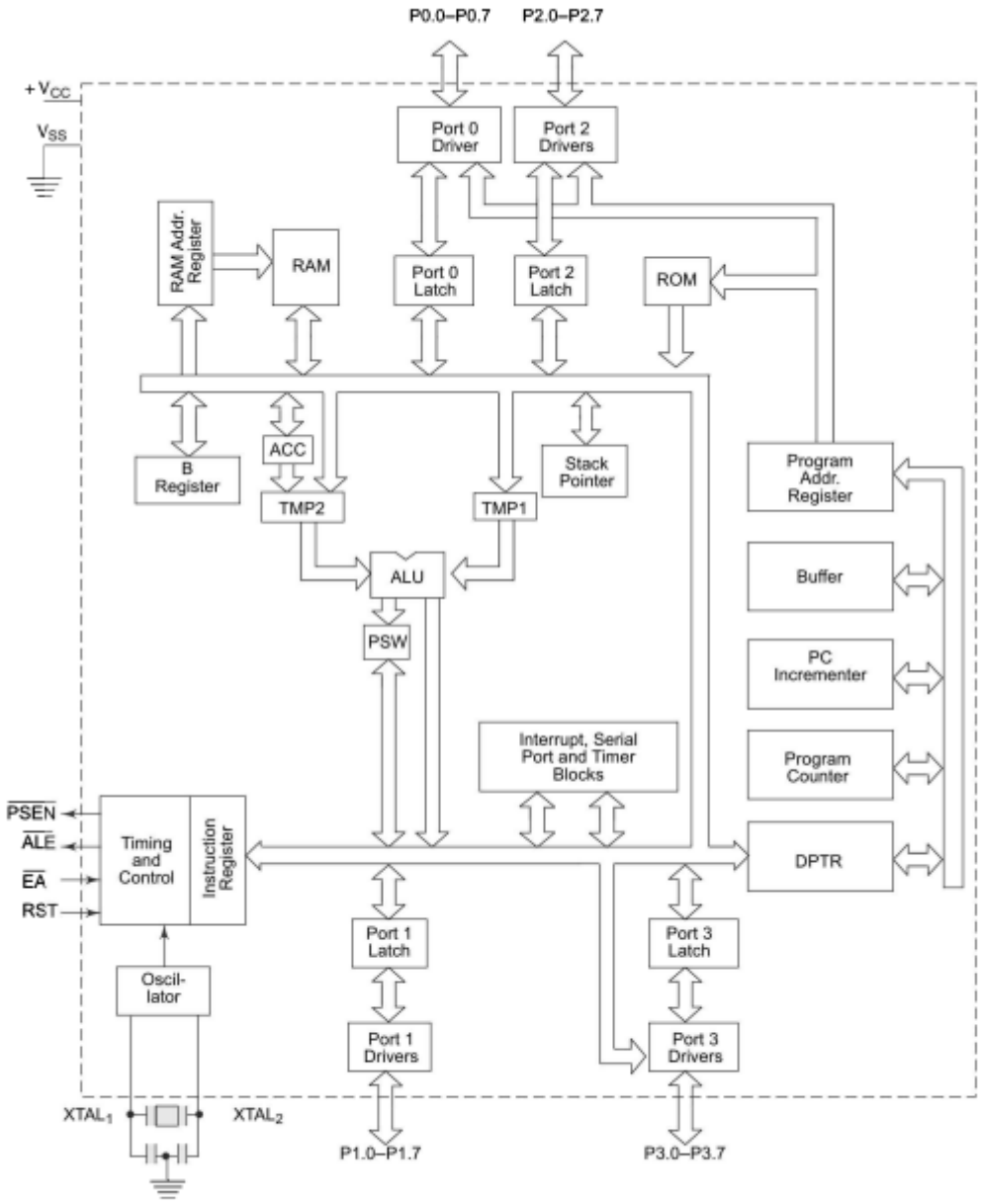

Fig. 17.2 *8051 Block Diagram (Intel Corp.)*

8085 Addressing Modes

Immediate addressing mode

In this mode, the data is specified in the instruction itself as one of its operand. **For example:** MVI K, 20F: means 20F is copied into register K.

Register addressing mode

In this mode, the data is copied from one register to another. **For example:** MOV K, B: means data in register B is copied to register K.

Direct addressing mode

In this mode, the data is directly copied from the given address to the register. **For example:** LDB 5000K: means the data at address 5000K is copied to register B.

Indirect addressing mode

In this mode, the data is transferred from one register to another by using the address pointed by the register. **For example:** MOV K, B: means data is transferred from the memory address pointed by the register to the register K.

Implied addressing mode

This mode doesn't require any operand; the data is specified by the opcode itself. **For example:** CMP.

Pentium Processor

The Pentium is a 32-bit processor. It was introduced by Intel in 1993.

Some of the key features of the Pentium processor are:

- **Branch prediction:** The Pentium uses branch prediction to predict which branch will be taken next. This can improve performance by reducing the number of times the processor needs to fetch instructions from main memory.
- **Pipelined architecture:** The Pentium uses a pipelined architecture, which means that it can fetch, decode, and execute instructions simultaneously. This can significantly improve performance, especially for programs that contain a lot of sequential instructions.
- **Larger instruction set:** The Pentium has a larger instruction set than earlier x86 processors. This allows it to perform more complex operations, which can also improve performance.
- **Faster clock speed:** The Pentium has a faster clock speed than earlier x86 processors. This means that it can execute instructions more quickly, which can also improve performance.

Draw and explain the General bus (read and write) cycle timing diagrams of 8086.

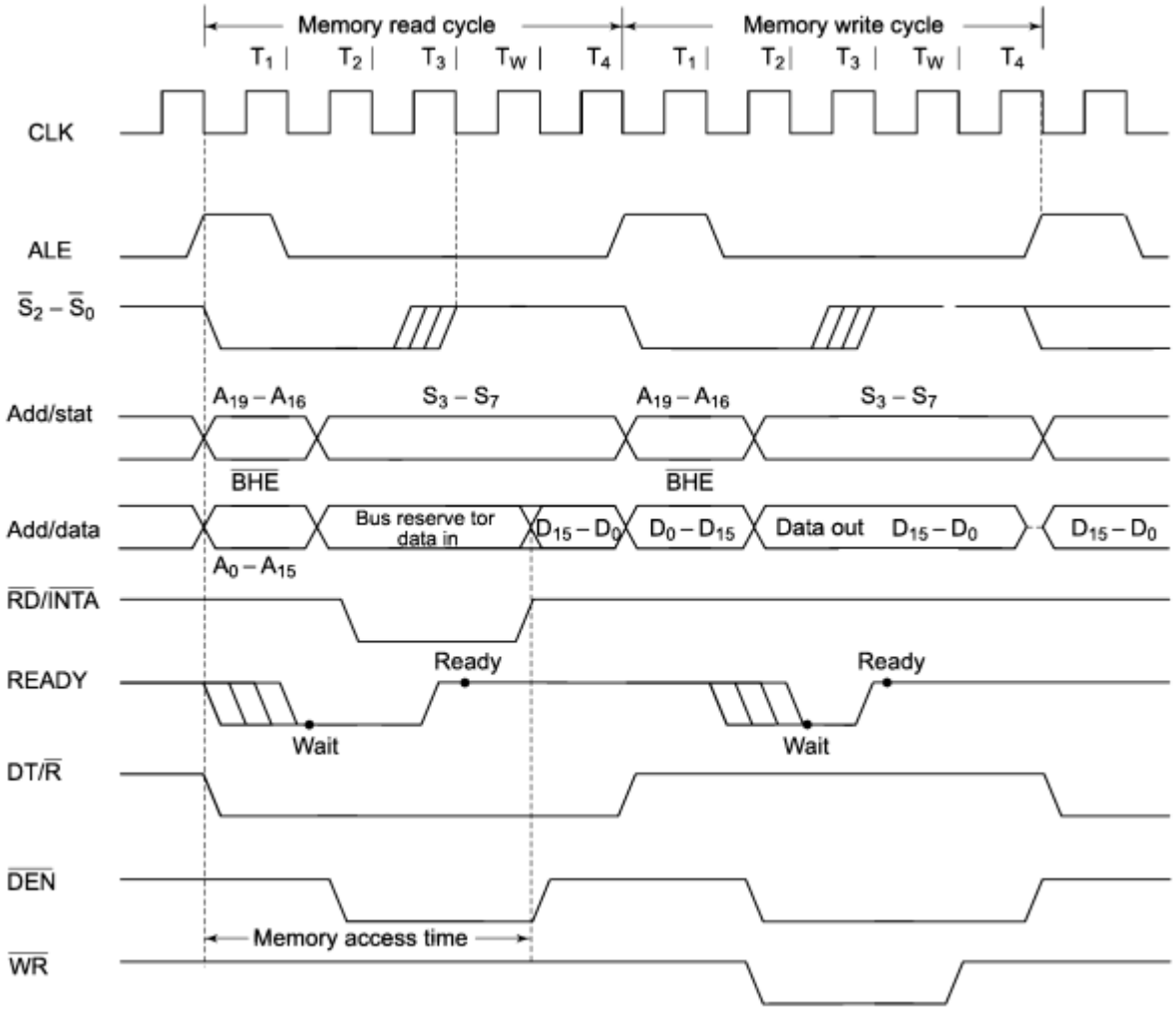

Fig. 1.8 *General Bus Operation Cycle of 8086*

The address is transmitted by the processor during T1. It is present on the bus only for one cycle. During T2, i.e. the next cycle, the bus is tristated for changing the direction of bus for the following data read cycle. The data transfer takes place during T3 and T4.

In case, an addressed device is slow and shows 'NOT READY' status the wait states Tw are inserted between T3 and T4.

The Address Latch Enable (ALE) signal is emitted during T1 by the processor (minimum mode) or the bus controller (maximum mode) depending upon the status of the MN/MX input. The negative edge of this ALE pulse is used to separate the address and the data or status information.

Status bits S3 to S7 are multiplexed with higher order address bits and the BHE signal. Address is valid during T1 while the status bits S3 to S7 are valid during T2 through T4.

Explain the flag register of 8088.

Same as that of 8086

What is stack, stack segment and stack pointer in 8086?

Stack is a portion of memory that is used to temporarily store the contents of registers when a procedure is called. The stack is accessed using the stack pointer register (SP).

Stack segment is a memory segment that is used to store the call stack. The stack segment is identified by the SS register. SS register contains the base address of the stack segment in the memory.

Stack pointer (SP) is a 16-bit register that contains offset of the address that lies in stack segment.

Explain stack structure and stack top address calculations.

The **stack structure** in 8086 is a LIFO (Last In First Out) data structure. This means that the last item pushed onto the stack is the first item popped off the stack. The stack is implemented as a region of memory that is reserved for this purpose. The stack pointer (SP) register is used to keep track of the top of the stack. The stack pointer is incremented whenever data is pushed onto the stack and decremented whenever data is popped off the stack.

SS	⇒ 5000 H					
SP	⇒ 2050 H					
SS	⇒	0101	0000	0000	0000	
10H * SS	⇒	0101	0000	0000	0000	0000
	+					
SP	⇒		0010	0000	0101	0000
Stack-top address		0101 5	0010 2	0000 0	0101 5	0000 0

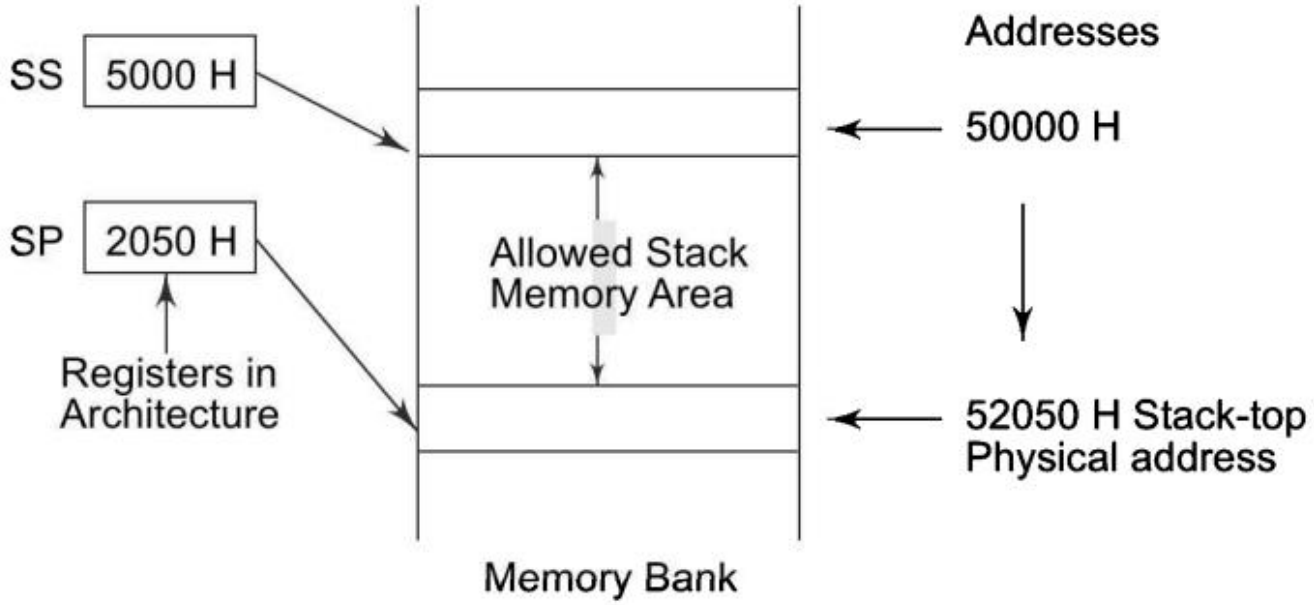

Fig. 4.1 *Stack-top Address Calculation*

Give an example program of using PUSH and POP instructions.

Program 4.2

Write a program to change a sequence of sixteen 2-byte numbers from ascending to descending order. The numbers are stored in the data segment. Store the new series at addresses starting from 6000 H. Use the LIFO property of the stack.

```
ASSUME      CS : CODE, DS : DATA, SS : DATA
DATA          SEGMENT
LIST        DW 10H
STACKDATA   DB FFH DUP (?)
              ORG6000H
              RESULT DW 10H
DATA        ENDSCOUNT EQU 10H
CODE        SEGMENT
START: MOV AX, DATA                      ; Initialize data segment and
       MOV DS, AX                        ; stack segment
       MOV SS, AX
       MOV SP, OFFSET LIST               ; Initialize stack pointer
       MOV CL, COUNT                     ; Initialize counter for word
                                              number
       MOV BX, OFFSET RESULT + COUNT     ; Initialize BX at last
                                           address
```

```
                                 ; (stack) for destination series
NEXT: POP AX                     ; Get the first word from series
    MOV DX, SP                   ; Save source stack pointer
    MOV SP, BX                   ; Get destination stack pointer
    PUSH AX                      ; Save AX to stack
    MOV BX, SP                   ; Save destination stack pointer
    MOV SP, DX                   ; Get source stack pointer for
                                 ; the next number
    DCR CL                       ; Decrement count
    JNZ NEXT                     ; If count is not zero, go to the next num
    MOV AH, 4CH                  ; Else, return to DOS
    INT 21H                      ; prompt
CODE ENDS
    END   START
```

What is IC-8255? Draw a block diagram and discuss the architecture of it.

8255 is Programmable Input-Output Port. It has 24 input/output lines which may be individually programmed in two groups of twelve lines each, or three groups of eight lines. The two groups of I/O pins are named as Group A and Group B. Each of these two groups contain a subgroup of eight I/O lines called as 8-bit port and another subgroup of four I/O lines or a 4-bit port.

The port A lines are identified by symbols **PA0-PA7** while the port C lines are identified as **PC4-PC7**. Similarly, Group B contains an 8-bit port B, containing lines **PB0- PB7** and a 4-bit port C with lower bits **PC0-PC3**.

All of these ports can function independently either as input or as output ports using **Control Word Register (CWR).**

The 8-bit **data bus buffer** is controlled by the **read/write control logic**. The read/write control logic manages all internal and external transfers of both data and control words. **RD, WR, A0, A1 and RESET** are the inputs, provided by the microprocessor to the READ/WRITE control logic.

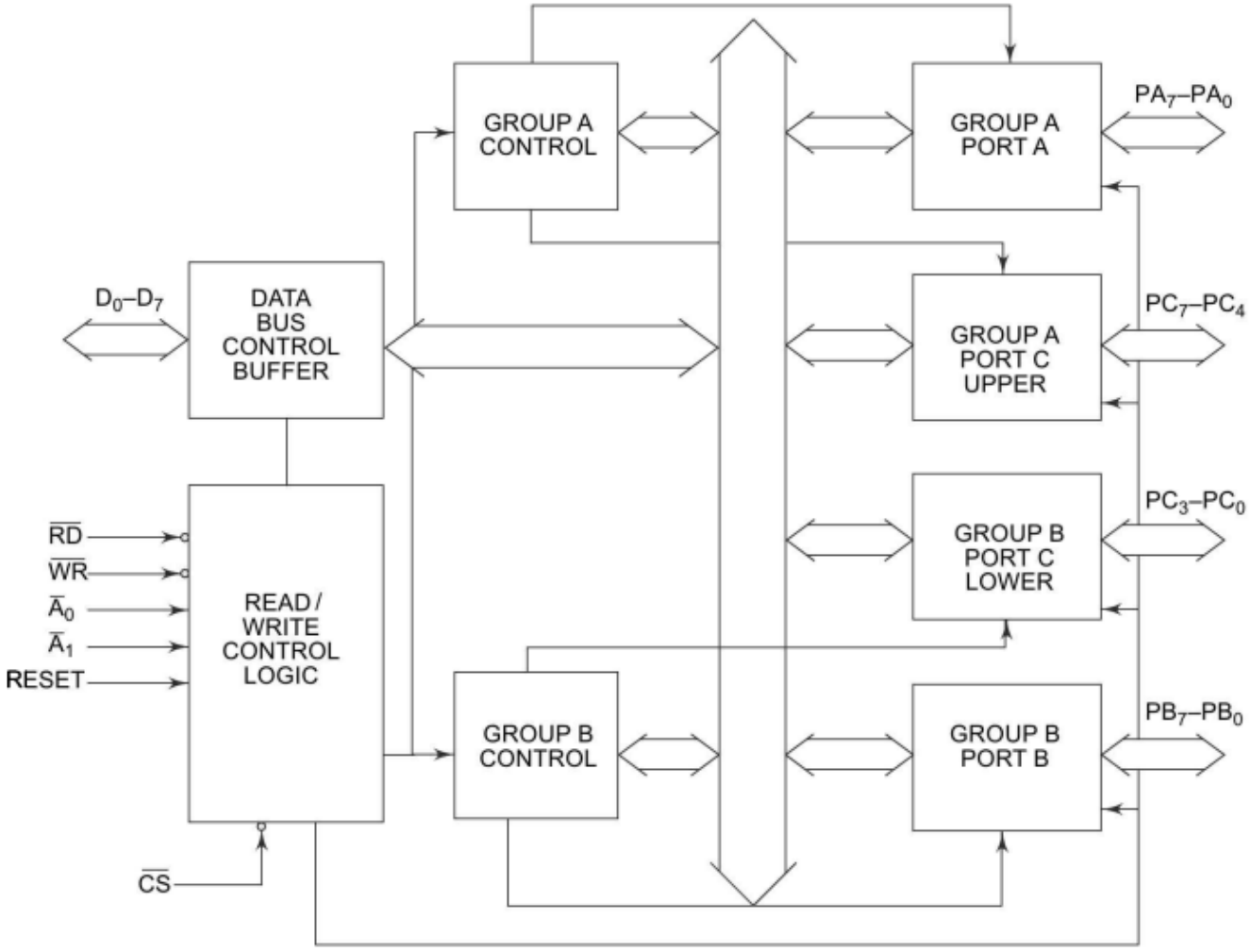

Fig. 5.17(a) *8255 Internal Architecture*

What is IC-8279? Draw a block diagram and discuss its various modes of operation.

8279 is a general purpose **keyboard display controller** that simultaneously drives the display of a system and interfaces a keyboard with the CPU, leaving it free for its routine task.

Architecture of 8279

I/O Control and Data Buffers: The I/O control section controls the flow of data to/from the 8279.

Control and Timing Register and Timing Control: These registers store the keyboard and display modes and other operating conditions programmed by CPU.

Scan Counter: The scan counter has two modes to scan the key matrix and refresh the display.

Return Buffers and Keyboard Debounce and Control: This section scans for a key closure row- wise. If it is detected, the keyboard debounce unit debounces the key entry.

FIFO/Sensor RAM and Status Logic: In keyboard or strobed input mode, this block acts as 8-byte first-in-first-out (FIFO) RAM.

Display Address Registers and Display RAM: The display address registers hold the address of the word currently being written or read by the CPU to or from the display RAM.

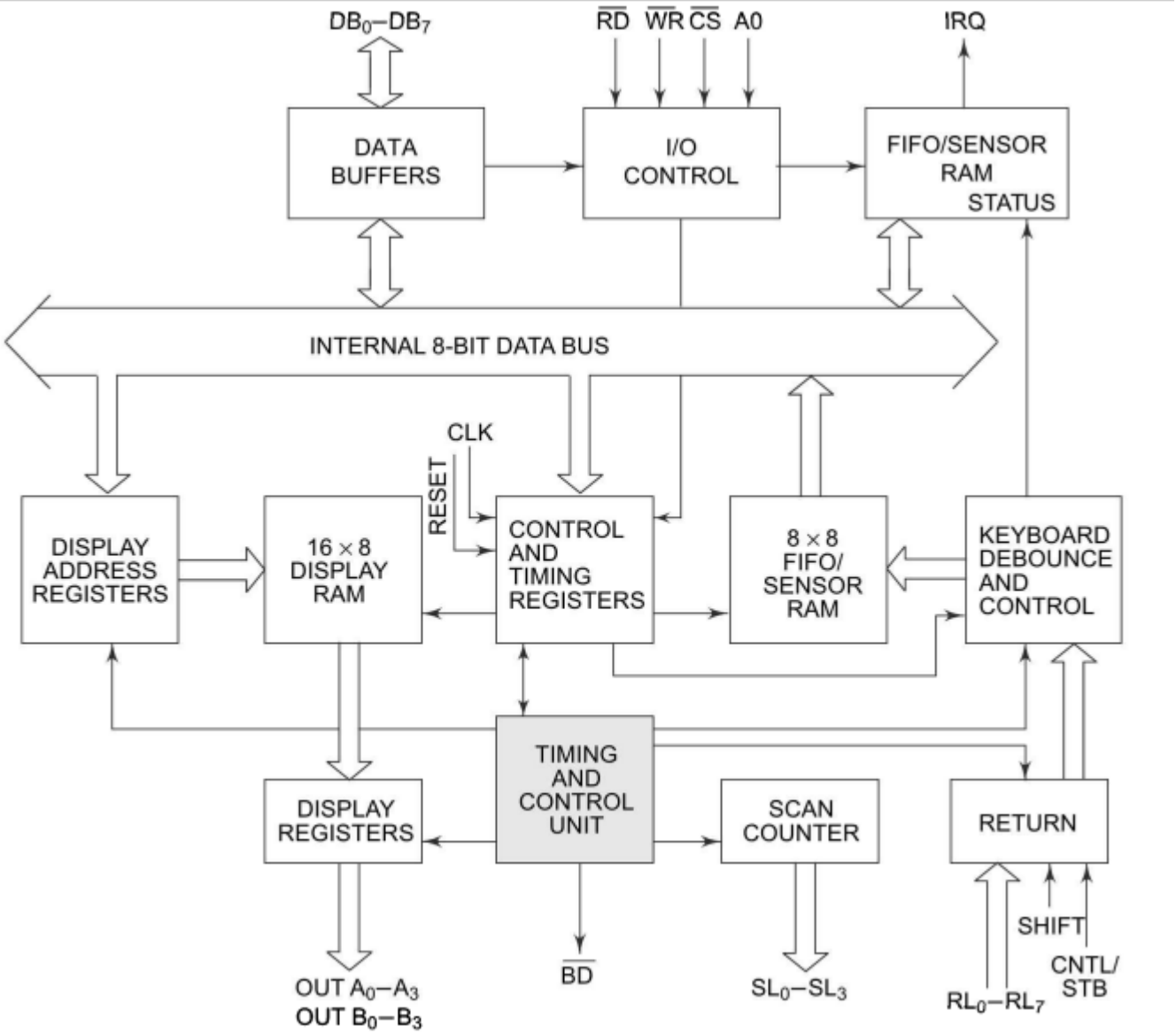

Fig. 6.23 *8279 Internal Architecture*

Modes of operation of 8279

Input (Keyboard) Modes

1. **Scanned Keyboard Mode:** This mode allows a key matrix to be interfaced using either encoded or de- coded scans. In the encoded scan, an 8 x 8 keyboard or in decoded scan, a 4x8 keyboard can be interfaced.

2. **Scanned Sensor Matrix:** In this mode, a sensor array can be interfaced with 8279 using either encoded or decoded scans. With encoded scan 8 x 8 sensor matrix or with decoded scan 4 x 8 sensor matrix can be interfaced.

3. **Strobed input:** In this mode, if the control line goes low, the data on return lines, is stored in the FIFO byte by byte.

Output (Display) Modes

1. **Display Scan:** In this mode, 8279 provides 8 or 16 character multiplexed displays those can be organized as dual 4-bit or single 8-bit display units.

2. **Display Entry:** 8279 allows options for data entry on the displays. The display data is entered for display either from the right side or from the left side.