# *The 8086 Microprocessor Hardware Specifications*

## Pin Diagram of 8086 and Pin description of 8086

Figure (1) shows the Pin diagram of 8086. The **8086** can be configured to work in either of **two modes**:

- The **minimum mode** is selected by applying **logic 1** to the $\mathbf{MN/\overline{MX}}$ input. It is typically used for smaller **single microprocessor** systems.
- The **maximum mode** is selected by applying **logic 0** to the $\mathbf{MN/\overline{MX}}$ input. It is typically used for larger **multiplemicroprocessor** systems.

Depending on the **mode** of operation selected, the 8086 signals can be categorized in three groups.

- The first are the signal having common functions in minimum as well as maximum mode.
- The second are the signals which have special functions for minimum mode.
- The third are the signals having special functions for maximum mode.

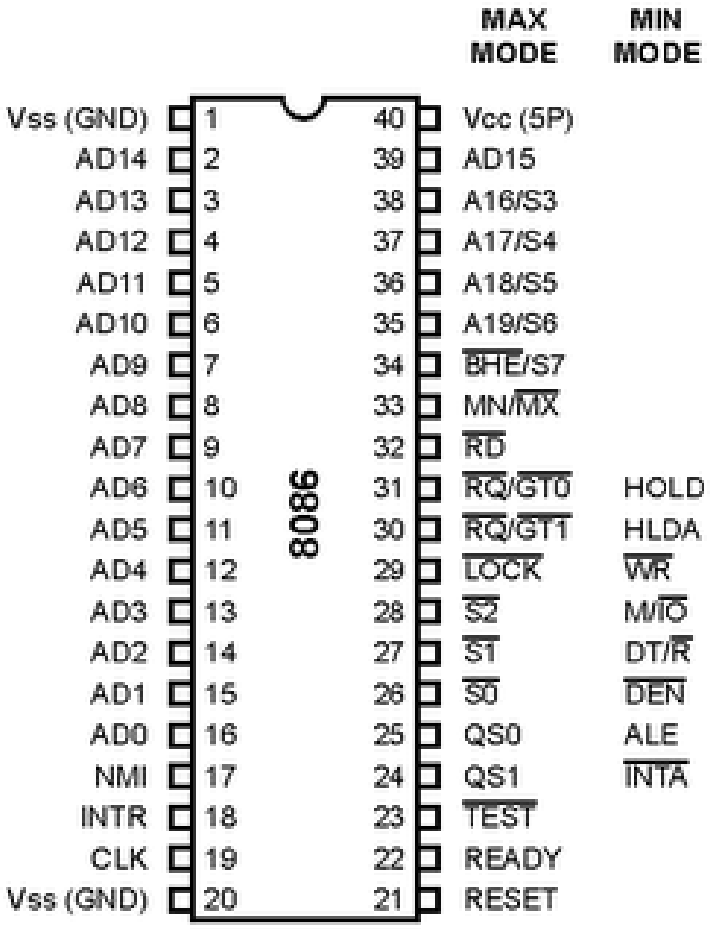


**Fig. 1: Pin Diagram of 8086**

## Minimum mode Operation

- Figure (2) show block diagram of minimum mode.
- Minimum mode operation is the least expensive way to operate the 8086.
- In minimum mode, the 8086 itself provides all the control signals needed to implement the memory and I/O interfaces.
- These control signals are identical to those of the Intel 8085A an earlier 8-bit microprocessor.
- This mode allows the 8085A peripherals to be used with the 8086 without any special consideration.

## Maximum mode Operation

- Figure (3) show block diagram of maximum mode.
- This mode supports existence of more than one processor in a system i.e. multiprocessor system.
- In a multiprocessor system environment more than one processor exists in the system, and each processor is executing its own program.
- In maximum mode the 8086 provides facilities by generating some of the control signals externally.
- In maximum-mode, a separate chip (the 8288 Bus Controller) is used to help in sending control signals over the shared bus.

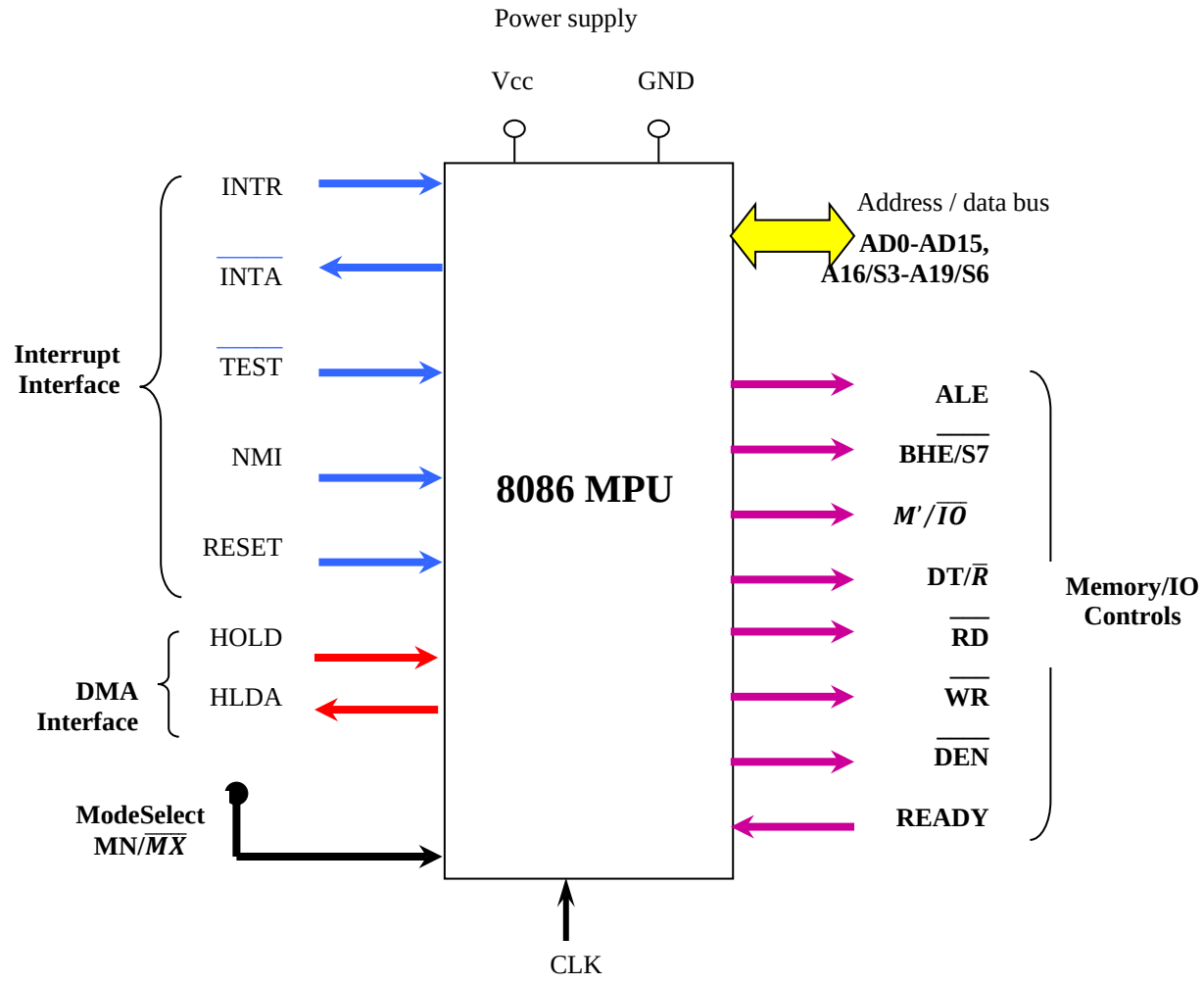


**Fig. 2: Minimum mode block diagram of 8086**

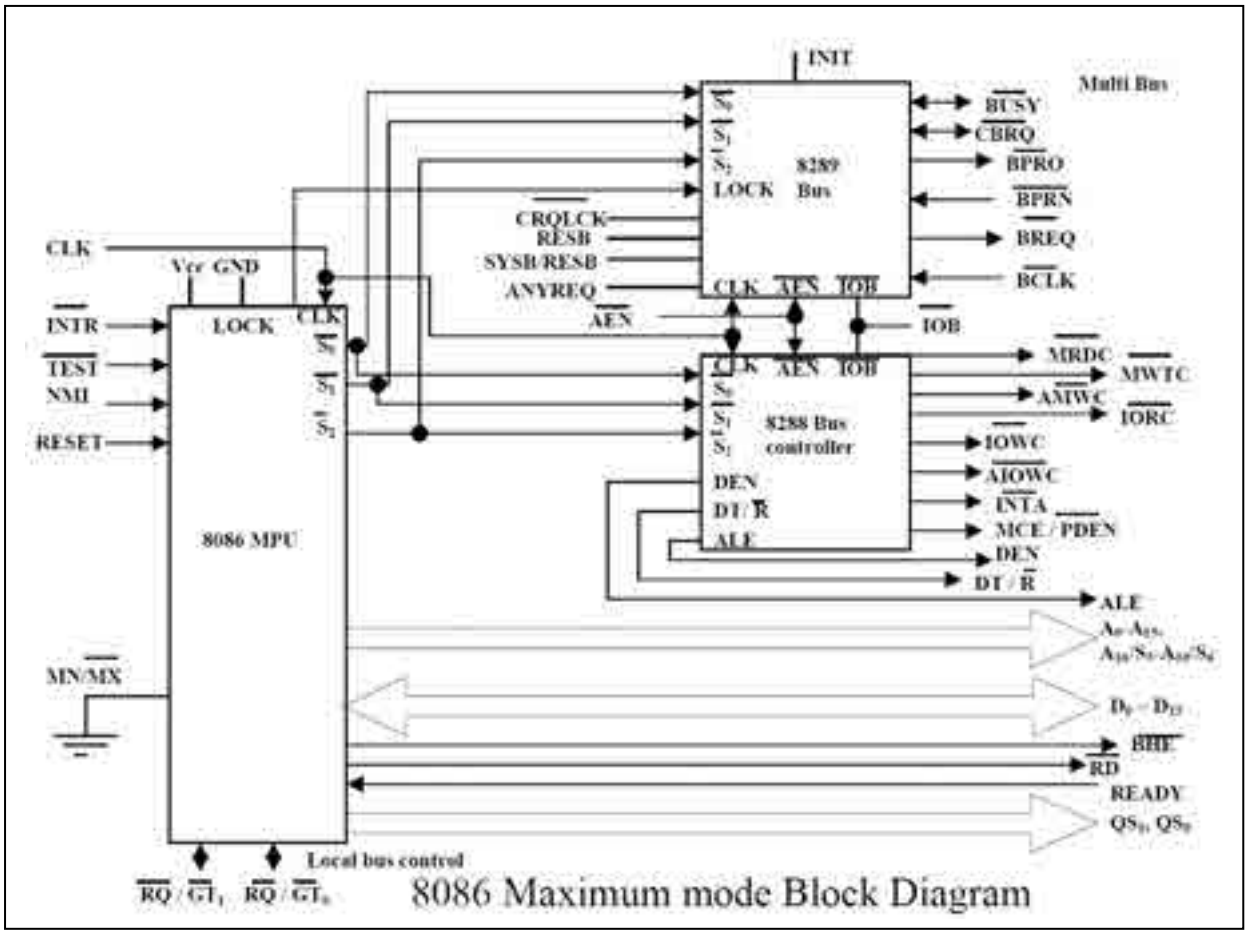


**Fig. 3: Maximum mode block diagram of 8086**

## The signals common for both minimum & maximum modes:

| Common Signals | | |
|---|---|---|
| **Name** | **Function** | **Type** |
| **AD15-AD0** | Address/Data Bus | Bidirectional , 3-state |
| **A19/S6 - A16/S3** | Address/Status | Output , 3-state |
| **MN / $\overline{\text{MX}}$** | Minimum/Maximum mode control | Input |
| **$\overline{\text{RD}}$** | Read Control | Output , 3-state |
| **$\overline{\text{TEST}}$** | Wait on test control | Input |
| **$\overline{\text{READY}}$** | Wait state control | Input |
| **RESET** | System reset | Input |
| **CLK** | System clock | Input |
| **Vcc** | +5V | Input |
| **GND** | Ground | Input |
| **$\overline{\text{BHE}}$ /S7** | Bus High Enable/Status | Input |
| **INTR** | Interrupt Request | Input |
| **NIM** | non maskable interrupt Request | Input |

**Table (1): The common signals for both minimum & maximum modes**

- **Address/Data Bus** (**AD15-AD0):** These line contain the address bus which is 20 bits long [A0 (the LSB) to $A_{19}$ (the MSB)]whenever ALE is logic 1, and contain the data bus which is 16 bits long [D0 (the LSB) to D15 (the MSB)] whenever ALE is logic 0.

- **Address/Status signals (A19/S6,A18/S5,A17/S4,A16/S3) :** These are the time multiplexed to provide address signals (A19-A16) and status lines (S6-S3). Bits S6 always remains logic 0, bit S5 indicates the condition of interrupt flag (IF) bits. S4 and S3 together form a 2-bit binary code that identifies which of the internal segment registers was used to generate the physical address that was output on the address bus during the current bus cycle (See Table 2)

| S4 | S3 | Indication |
|---|---|---|
| 0 | 0 | Extra Data Segment |
| 0 | 1 | Stack Segment |
| 1 | 0 | Code or No Segment |
| 1 | 1 | Data Segment |

**Table (2)**

- **MN / $\overline{\text{MX}}$ :** is an input pin used to select one of this mode .when MN/MX is high the 8086 operates in minimum mode .In this mode the 8086 is configured to support small single processor system using a few devices that the system bus .when MN/MX is low 8086 is configured to support multiprocessor system.
- **Read ( $\overline{\text{RD}}$ ):**is logic 0 (low) when the data is read from memory or I/O location.
- $\overline{\textbf{TEST}}$ **:** is an input pin and is only used by the wait instruction. If the $\overline{\textbf{TEST}}$ pin goes LOW (logic 0), execution will continue (WAIT instruction functions as a NOP), else if $\overline{\textbf{TEST}}$ pin goes HIGH (logic 1) the processor remains in an idle state.
- $\overline{\textbf{READY}}$ **:** If the $\overline{\textbf{READY}}$ pin goes LOW (logic 0) the processor enters into wait state and remains in an idle state. If the $\overline{\textbf{READY}}$ pin goes HIGH (logic 1) it has no effect on the operation of the processor.
- **RESET:** is the system set reset input signal. If thispin held HIGH for a minimum of four clocking periods causes to processor to reset itself and start execution from FFFF0H i.e reinitialize the system.
- **Clock Input (CLK):** The clock input provides the basic timing for processor operation and bus control activity. Its an asymmetric square wave with 33% duty cycle (HIGH for one-third of the clocking period and LOW for two-third).
- **Vcc:**+5V power supply for the operation of the internalcircuit.
- **GND:**Ground for the internal circuit. The 8086 microprocessor have two pins labeled GND both must be connected to ground for proper operation.

- **Bus High Enable/Status($\overline{BHE}$/S7):**The bus high enable signal goes low to indicate thetransfer of data over the higher order (D15-D8). The state S7 is always logic 1.
- **Interrupt Request (INTR) :**is a maskable interrupt input. This is a triggered input. This is sampled during the last clock cycles of each instruction to determine the availability of the request. If any interrupt request is pending (IF=1), the processor enters the interrupt acknowledge cycle ($\overline{INTA}$ becomes active) after the currentinstruction has complete execution.
- **NIM:**is the non maskable interrupt input.The NMI is not maskable internally by software. Atransition from low to high initiates theinterrupt response at the end of the currentinstruction.

## Minimum mode interface signals

| Minimum mode Signals ($MN/\overline{MX}$ =Vcc) | | |
|---|---|---|
| **Name** | **Function** | **Type** |
| **(M/$\overline{IO}$ )** | Memory/IO Control | Output , 3-state |
| **$\overline{WR}$** | WRITE Control | Output , 3-state |
| **ALE** | Address Latch Enable | Output |
| **$DT/\overline{R}$** | Data Transmit/Receive | Output , 3-state |
| **DEN** | Data Enable | Output , 3-state |
| **HOLD** | Hold request | Input |
| **HLDA** | Hold Acknowledgment | Output |
| **$\overline{INTA}$** | Interrupt Acknowledgment | Output |

**Table (3): Minimum mode Signals.**

- **Memory/IO (M/$\overline{IO}$ ):** This is a status line logically equivalent to S2 in maximum mode. When it is LOW, it indicates the CPU is having an I/O operation, and when it is HIGH, it indicates that the CPU is having a memory operation.
- **WRITE** ($\overline{WR}$): indicates that the processor is performing a write memory or write I/O cycle, depending on the state of the M/IO signal.
- **Interrupt Acknowledge ($\overline{INTA}$ ):** This signal is used as a read strobe for interrupt acknowledge cycles. i.e. when it goes low, the processor has accepted the interrupt.
- **Address Latch Enable (ALE) :**Itis an output signal provided by the 8086 and can be used to demultiplexed AD0 to AD15 in to A10 toA15 and D0 to D15. This signal is active high and is never tristated.
- **Data Transmit/Receive ($DT/\overline{R}$ ):** This output is used to decide the direction of data flow through the transceiver (bidirectional buffers). When **($DT/\overline{R}$ = 1)** the processor sends data out (transmitting), when **($DT/\overline{R}$ = 0)** the processor receiving data.
- **Data Enable (DEN):** activate external data bus buffers.

- **HOLD:** When an external device wants to take control of the system bus (Data, Address, Control), it signals to the 8086 by switching HOLD to logic 1.The hold input requests a direct memory access (DMA).
- **Hold Acknowledge :** The processor, after receiving the HOLD request, issues the hold acknowledge signal on HLDA pin, indicates that the 8086 has entered the hold.

## Maximum mode interface signals

| **Maximum mode Signals ( MN / $\overline{MX}$ =GND)** | | |
|---|---|---|
| **Name** | **Function** | **Type** |
| **$\overline{RQ}$ / $\overline{GT0}$ ,( $\overline{RQ}$ / $\overline{GT1}$ )** | Request/Grant bus access control | Bidirectional |
| **( $\overline{S2}$ , $\overline{S1}$ , $\overline{S0}$ )** | Status Lines | Output , 3-state |
| **$\overline{LOCK}$** | Bus priority lock control | Output , 3-state |
| **QS1 , QS0** | Queue Status | Output |

**Table (4): Maximum mode Signals.**

- **Request/Grant ( $\overline{RQ}$ / $\overline{GT0}$ ,( $\overline{RQ}$ / $\overline{GT1}$ ):**These lines are bidirectional, and are used to both request and grant a DMA operation in maximum mode.
- **$\overline{LOCK}$ :** This output pin indicates that other system bus master will be prevented fromgaining the system bus, while the LOCK signal is low. The LOCK signal is activated by the 'LOCK' prefix instruction and remains active until the completion of the next instruction.
- **Status Lines ( $\overline{S2}$ , $\overline{S1}$ , $\overline{S0}$ ):** These three bit are input to the external *bus controller* device**8288**, which decodes them to identify the type of next bus cycle. Table (5) shows the function of these status bits in maximum mode.

| $\overline{S2}$ | $\overline{S1}$ | $\overline{S0}$ | CPU Cycle | 8288 Command |
|---|---|---|---|---|
| 0 | 0 | 0 | Interrupt Acknowledge | $\overline{INTA}$ |
| 0 | 0 | 1 | Read I/O port | $\overline{IORC}$ |
| 0 | 1 | 0 | Write I/O port | $\overline{IOWC}$ , $\overline{AIOWC}$ |
| 0 | 1 | 1 | Halt | None |
| 1 | 0 | 0 | Code Access | $\overline{MRDC}$ |
| 1 | 0 | 1 | Read Memory | $\overline{MRDC}$ |
| 1 | 1 | 0 | Write Memory | $\overline{MWTC}$, $\overline{AMWC}$ |
| 1 | 1 | 1 | Passive | None |

**Table (5):** Bus Status Codes

- **Queue Status (QS1, QS2):**provide status to allow external tracking of the internal 8086 instruction queue. These pins are provided for access by the numeric coprocessor (8087). Table (6) shows the operation of the queue status bits.

| QS1 | QS0 | Indication |
|---|---|---|
| 0 | 0 | **No Operation (Queue is idle)** |
| 0 | 1 | **First Byte of the opcode from the queue** |
| 1 | 0 | **Empty Queue** |
| 1 | 1 | **Subsequent Byte from the Queue** |

**Table (6):Queue Status bits**

## SYSTEM CLOCK

- To synchronize the internal and external operations of the microprocessor a *clock* (CLK) input signal is used. The CLK can be generated by the 8284 clock generator IC.
- The 8086 is manufactured in three speeds: 5 MHz, 8 MHz and 10 MHz.
- For 8086, we connect either a 15-, 24- or 30-MHz crystal between inputs X1 and X2of the clock chip (see Fig. 4).
- The *fundamental crystal frequency* is divided by 3 within the 8284 to give either a 5-, 8- or 10-MHz clock signal, which is directly connected to the CLK input of the 8086.

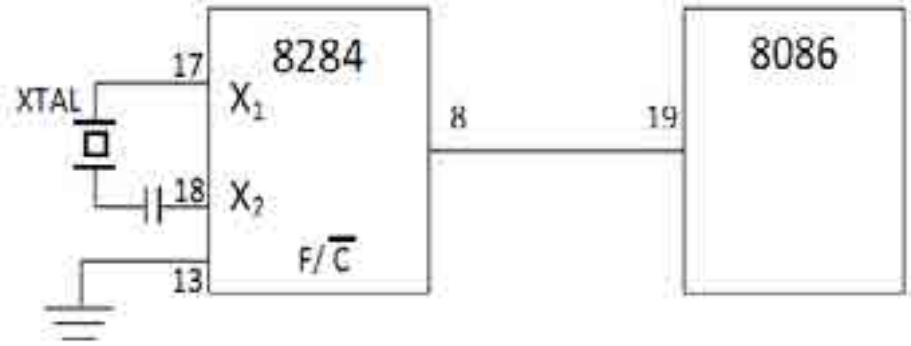


**Fig. 4: Connecting the 8284 to the 8086.**

## Bus cycle and time state

A *bus cycle* defines the basic operation that a microprocessor performs to communicate with external devices. Example of bus cycles are

- Memory read
- Memory write
- IO read
- IO write

The bus cycle of 8086 microprocessors consists of at least four clock periods (T1, T2, T3, and T4)

- During T1 the 8086 puts an address on the bus.
- During T2 the 8086puts the data on the bus (for write memory cycle) and maintained through T3 and T4

- During T2 the 8086puts the bus in high-Z state (for read cycle) and then the data to read must be available on the bus during T3 and T4.

These four clock states give a bus cycle duration of 125 ns × 4= 500 ns in an 8-MHz system.

## Idle States

If no bus cycles are required, the microprocessor performs what are known as *idle state*. During these states, no bus activity takes place. Each idle state is one clock period long, and any number of them can be inserted between bus cycles. Idle states are performed if the instruction queue inside the microprocessor is full and it does not need to read or write operands form memory.

## Wait States

Wait states can be inserted into a bus cycle. This is done in response to request by an event in external hardware instead of an internal event such as a full queue. The READY input of the 8086is provided specifically for this purpose. As long as READY is held at the 0 level, wait states are inserted between states T3 and T4 of the current bus cycle, and the data that were on the bus during T3 are maintained. The bus cycle is not completed until the external hardware returns READY back to the 1 logic level.

## Read Cycle

The read bus cycle begins with state T1. During this period, the 8086 output the 20bit address of the memory location to be accessed on its multiplexed address/data bus $AD_0$through $AD_{15}$and multiplexed lines $A_{16}/S_3$through $A_{19}/S_6$.Note that at the same time a pulse is also produced at ALE. The signal $\overline{\mathbf{BHE}}$ is also supplied with the address lines. (Figure 5 )

## Write Cycle

The write bus cycle is similar to the read bus cycle except that signal $\overline{\mathbf{WR}}$ instead of the signal $\overline{\mathbf{RD}}$ and signal $\mathbf{DT}/\overline{\mathbf{R}}$ is set to 1.

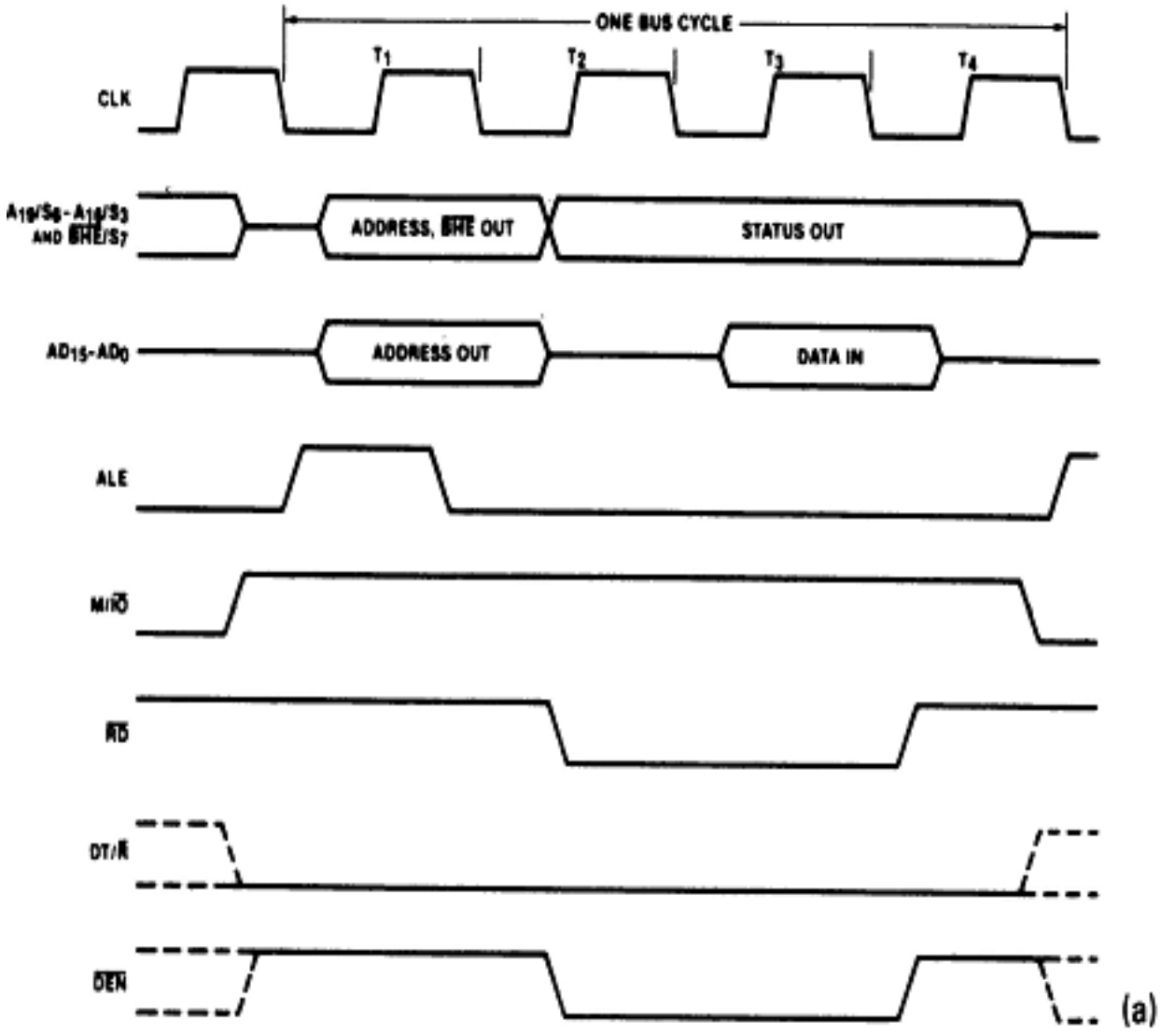


**Fig. 5: Minimum-mode memory read bus cycle of the 8086.**