Chapter 8 Summary: The 8086 Microprocessor and its Memory and Input/Output Interface

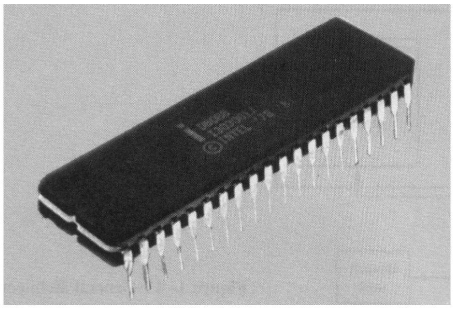


Figure 1-5 Intel Corporation's 8086 Microprocessor.

- The **8086**, announced in **1978**, was the **first 16-bit** microprocessor introduced by Intel Corporation.
- The **8086** is internally a **16-bit MPU** and **externally** it has a **16-bit data bus**. It has the ability to address up to **1 Mbyte** of memory via its **20-bit address bus**.
- In addition, it can address up to **64K of byte-wide input/output ports**.
- It is manufactured using **high-performance metal-oxide semiconductor (HMOS) technology**, and the circuitry on its chip is equivalent to approximately **29,000 transistors**.
- The 8086 is housed in a **40-pin dual in-line package**. The signals pinned out to each lead are shown in Fig. 8-1.
- The **address bus lines** A_0 through A_{15} and **data bus lines** D_0 through D_{15} are **multiplexed**. For this reason, these leads are labeled AD_0 through AD_{15}. By *multiplexed* we mean that the same physical pin carries an address bit at one time and the data bit at another time.

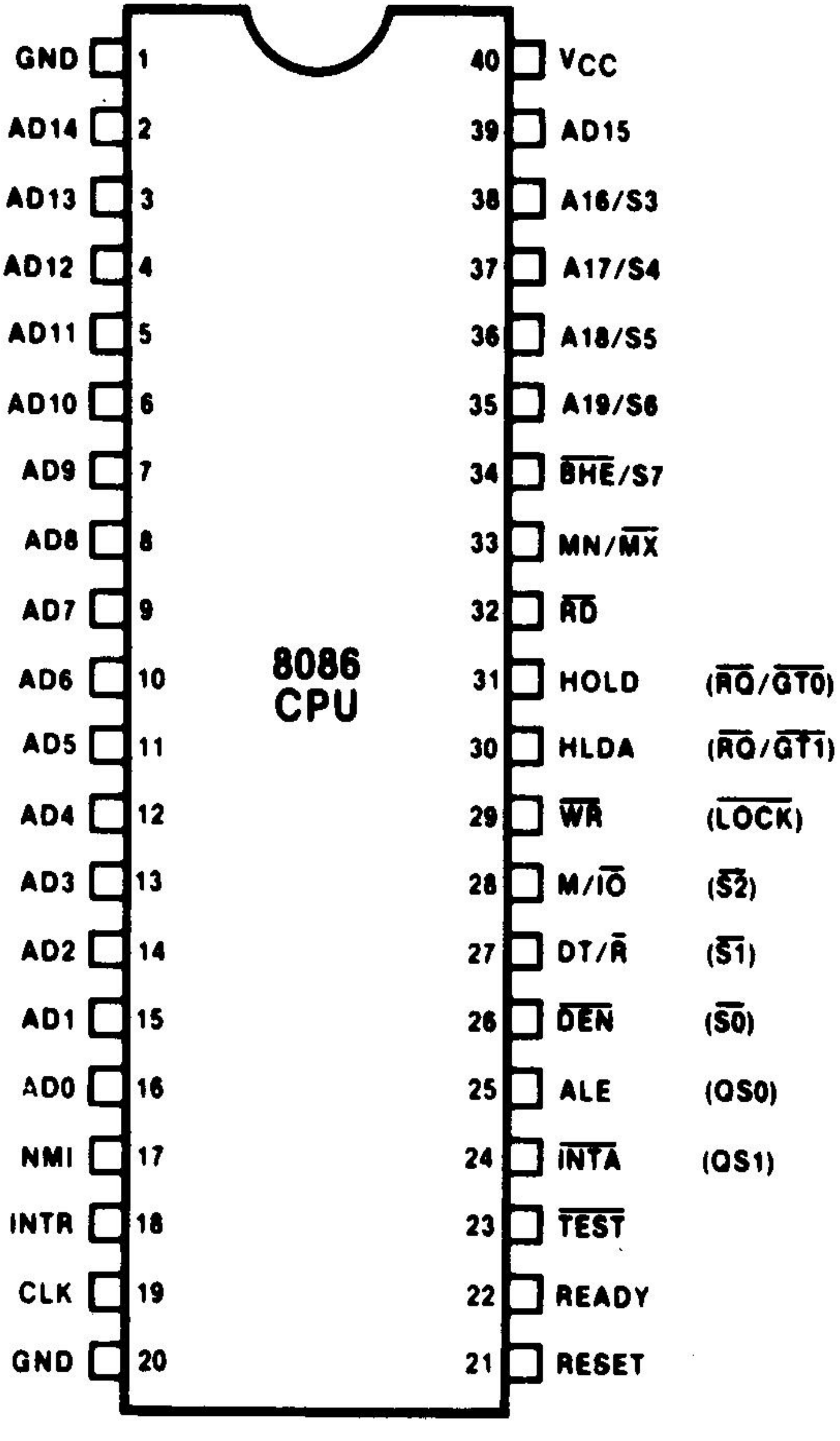


Figure 8-1 Pin layout of the 8086 Microprocessor.

8.2 MINIMUM-MODE AND MAXIMUM-MODE

- The **8086** can be configured to work in either of **two modes**:

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

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

- Depending on the **mode** of operation selected, the **assignments** for a number of the **pins** on the microprocessor package are **changed**. The **pin functions** specified in **parentheses** pertain to the **maximum-mode**.

- We will **only** discuss **minimum-mode** operation of the 8086. In minimum mode, the **8086** itself **provides** all the **control signals** needed to implement the memory and I/O interfaces (see Fig. 8-3). In **maximum-mode**, a separate chip (the **8288 Bus Controller**) is used to help in sending control signals over the shared bus (see Fig. 8-5).

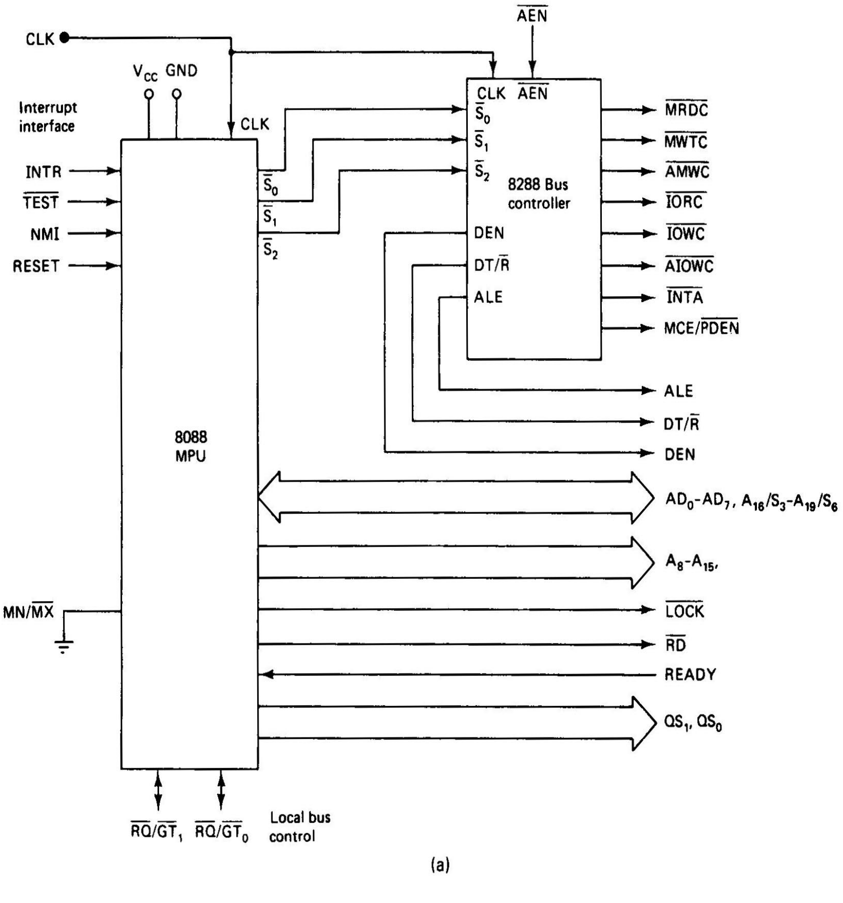


Figure 8–5 (a) 8088 maximum-mode block diagram. (b) 8086 maximum-mode block diagram.

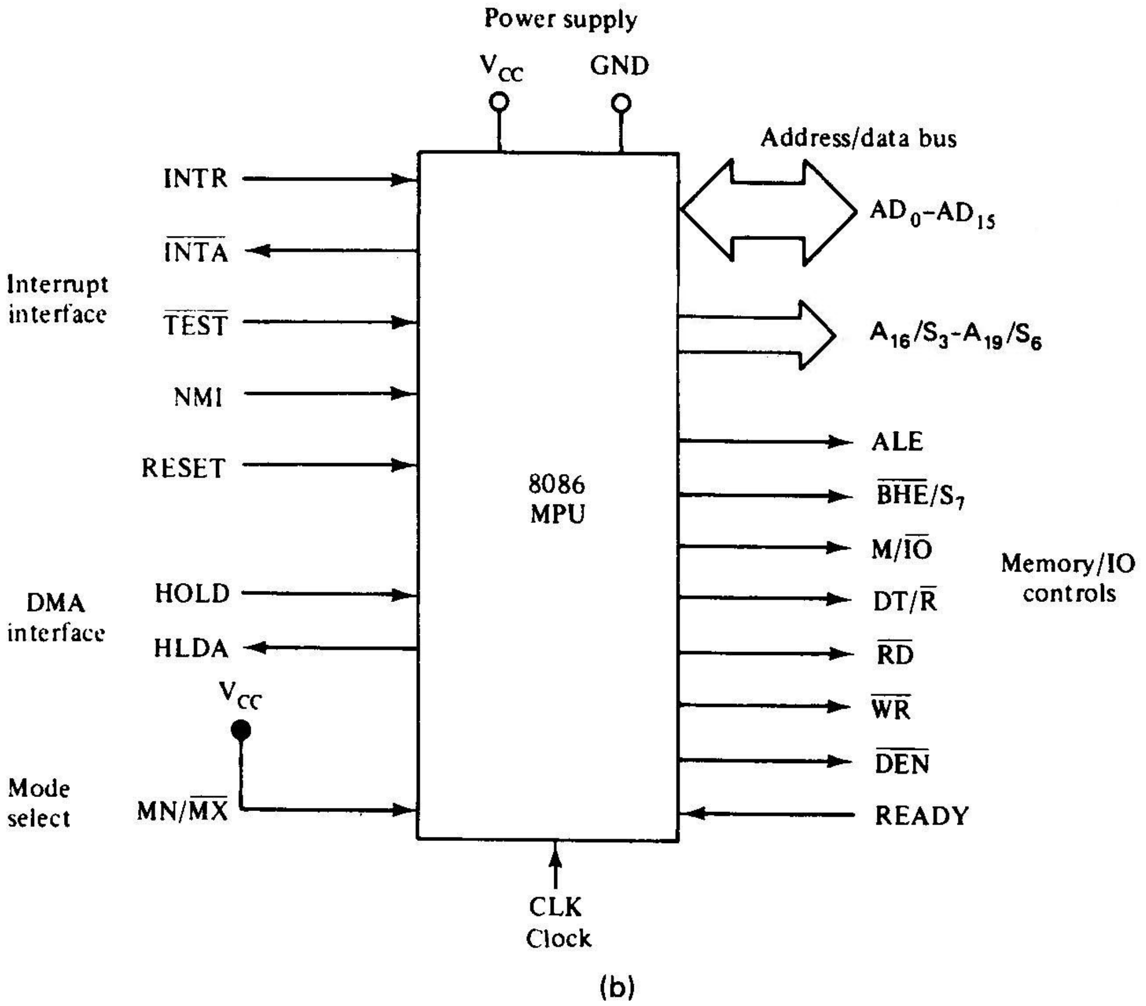


Figure 8–3 (a) Block diagram of the minimum-mode 8088 MPU. (b) Block diagram of the minimum-mode 8086 MPU.

8.3 SUMMARY OF MINIMUM-MODE INTERFACE SIGNALS

- **Address/Data Bus:** The address bus is **20 bits long** and consists of signal lines **A_0** (LSB) through **A_{19}** (MSB). However, only address lines **A_0** through **A_{15}** are used when accessing **I/O**.

- The **data bus** lines are **multiplexed** with address lines. For this reason, they are denoted as **AD_0** through **AD_{15}**. Data line **D_0** is the LSB.

- **Status Signals:** The four most significant address lines **A_{16}** through **A_{19}** of the 8086 are multiplexed with **status signals S_3** through **S_6**. These status bits are output on the bus at the same time that data are transferred over the other bus lines.

- **Control Signals:**
- When ***Address latch enable*** (**ALE**) is **logic 1** it signals that a **valid address** is on the bus. This address can be latched in external circuitry on the **1-to-0 edge** of the pulse at ALE.
- $\mathbf{M}/\overline{\mathbf{IO}}$ (*memory*/IO) tells external circuitry whether a memory or I/O transfer is taking place over the bus. **Logic 1** signals a **memory operation** and **logic 0** signals an **I/O operation**.
- $\mathbf{DT}/\overline{\mathbf{R}}$ (*data transmit/receive*) signals the **direction of data transfer** over the bus. **Logic 1** indicates that the bus is in the **transmit mode** (i.e., data are either written into memory or to an I/O device). **Logic 0** signals that the bus is in the **receive mode** (i.e., reading data from memory or from an input port).
- The *bank high enable* ($\overline{\mathbf{BHE}}$) signal is used as a **memory enable signal** for the **most significant byte** half of the data bus, $\mathbf{D_8}$ through $\mathbf{D_{15}}$.
- $\overline{\mathbf{WR}}$ (*write*) is switched to **logic 0** to signal external devices that **valid output data** are on the bus.
- $\overline{\mathbf{RD}}$ (*read*) indicates that the MPU is performing a **read of data** off the bus. During read operations, one other control signal, $\overline{\mathbf{DEN}}$ (*data enable*), is also supplied. It enables external devices to supply data to the microprocessor.
- The **READY** signal can be used to **insert wait states** into the bus cycle so that it is extended by a number of clock periods. This signal is supplied by a **slow memory or I/O subsystem** to signal the MPU when it is ready to permit the data transfer to be completed.
- **Interrupt Signals:**
- *Interrupt request* (**INTR**) is an **input** to the 8086 that can be used by an **external device** to **signal** that it needs to be **serviced**. **Logic 1** at INTR represents an active interrupt request.
- When the MPU **recognizes an interrupt request**, it indicates this fact to external circuits with logic 0 at the *interrupt acknowledge* (**INTA**) output.
- On the **0-to-1 transition** of *nonmaskable interrupt* (**NMI**), control is passed to a nonmaskable **interrupt service routine** at completion of execution of the current instruction. NMI is the interrupt request with highest priority and **cannot be masked by software**.

- The **RESET** input is used to provide a **hardware reset** for the MPU. Switching RESET to **logic 0** initializes the internal registers of the MPU and initiates a reset service routine.

- **DMA Interface Signals:**

- When an **external device** wants to **take control** of the **system bus**, it signals this fact to the MPU by switching HOLD to the **logic level 1**.

- When in the hold state, lines $\mathbf{AD_0}$ through $\mathbf{AD_{15}}$, $\mathbf{A_{16}/S_3}$ through $\mathbf{A_{19}/S_6}$, $\overline{\mathbf{BHE}}$, $\mathbf{M/\overline{IO}}$, $\mathbf{DT/\overline{R}}$, $\overline{\mathbf{WR}}$, $\overline{\mathbf{RD}}$, $\overline{\mathbf{DEN}}$ and **INTR** are all put in the **high-Z** state. The MPU signals external devices that it is in this state by switching **HLDA** to **1**.

8.6 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 X$_1$ and X$_2$ inputs of the clock chip (see Fig. 8-11). 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.

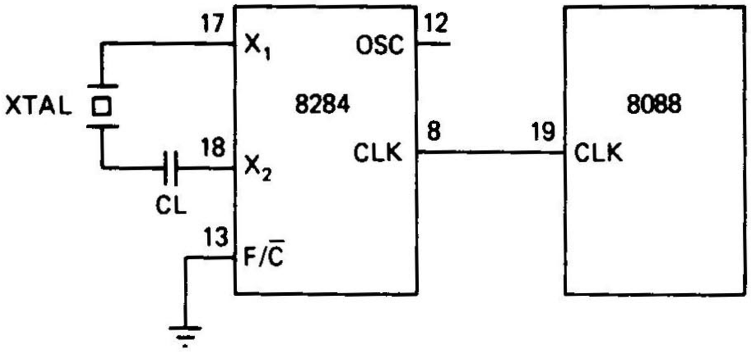


Figure 8–11 Connecting the 8284 to the 8088. (Reprinted with permission of Intel Corporation, © 1979)

8.7 BUS CYCLE AND TIME STATES

- A **bus cycle** defines the **sequence of events** when the MPU **communicates** with an external device, which starts with an **address** being output on the system bus followed by a **read or write data** transfer.

- Types of bus cycles:
    - **Memory Read Bus Cycle**
    - **Memory Write Bus Cycle**
    - **Input/Output Read Bus Cycle**
    - **Input/Output Write Bus Cycle**

- The **bus cycle** of the 8086 microprocessor consists of **at least four clock periods**. These four time states are called **T_1, T_2, T_3 and T_4**.

8.11 MEMORY READ AND WRITE BUS CYCLES

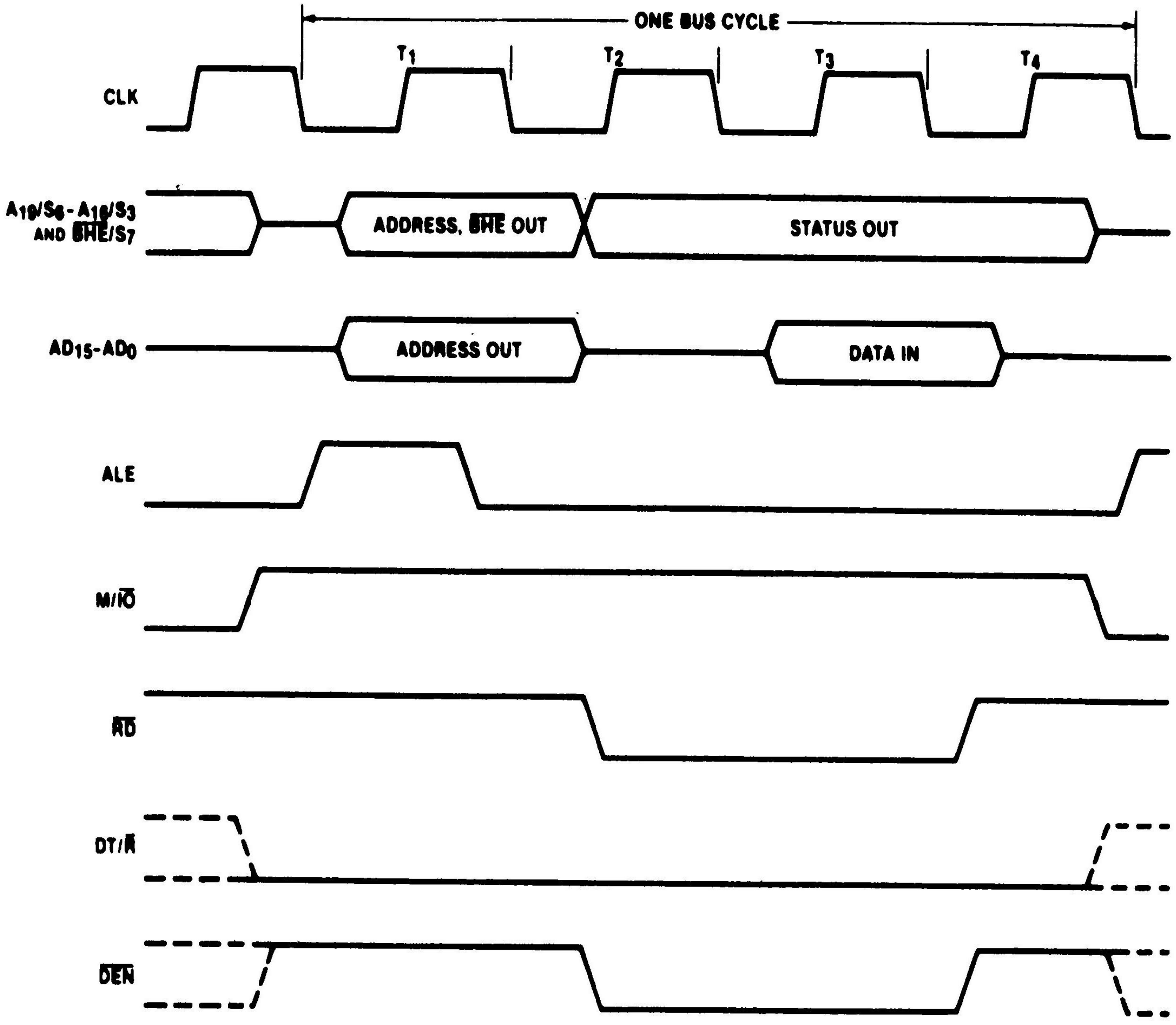
- **Fig. 8-22(a)** shows a **memory read cycle** of the **8086**:

- During **period T_1**,
    - The 8086 outputs the **20-bit address** of the memory location to be accessed on its multiplexed **address/data bus**. $\overline{\mathbf{BHE}}$ is also output along with the address during T_1.
    - At the same time a pulse is also produced at **ALE**. The **trailing edge** or the **high level** of this pulse is used to **latch** the address in external circuitry.
    - Signal $\mathbf{M/\overline{IO}}$ is set to **logic 1** and signal $\mathbf{DT/\overline{R}}$ is set to the **0 logic level** and both are maintained throughout all four periods of the bus cycle.

- Beginning with **period T_2**,
    - Status bits **S_3** through **S_6** are output on the upper four address bus lines. This status information is maintained through periods **T_3** and **T_4**.
    - On the other hand, address/data bus lines **AD_0 through AD_7** are put in the **high-Z state** during **T_2**.
    - Late in period **T_2**, $\overline{\mathbf{RD}}$ is switched to **logic 0**. This indicates to the memory subsystem that a read cycle is in progress. $\overline{\mathbf{DEN}}$ is switched to **logic 0** to enable external circuitry to allow the data to move from memory onto the microprocessor's data bus.

- During **period T_3**,
    - The memory must provide **valid data** during **T_3** and maintain it until after the processor terminates the read operation. The data read by the 8086 microprocessor can be carried over all **16 data bus** lines

- During **T$_4$,**
    - The 8086 switches $\overline{\mathbf{RD}}$ to the inactive **1 logic level** to terminate the read operation. $\overline{\mathbf{DEN}}$ returns to its inactive logic level late during **T$_4$** to disable the external circuitry.
- **Fig. 8-22(b)** shows a **memory write cycle** of the 8088:
- During **period T$_1$**,
    - The **address** along with $\overline{\mathbf{BHE}}$ are output and latched with the **ALE** pulse.
    - $\mathbf{M/\overline{IO}}$ is set to **logic 1** to indicate a memory cycle.
    - However, this time $\mathbf{DT/\overline{R}}$ is switched to **logic 1**. This signals external circuits that the 8086 is going to **transmit data** over the bus.
- Beginning with **period T$_2$**,
    - $\overline{\mathbf{WR}}$ is switched to **logic 0** telling the memory subsystem that a write operation is to follow.
    - The 8086 puts the **data** on the bus late in **T$_2$** and maintains the data valid through **T$_4$**. Data will be carried over all **16 data bus lines**.
    - $\overline{\mathbf{DEN}}$ enables the external circuitry to provide a path for data from the processor to the memory.



(a)

Figure 8-22(a) Minimum-mode memory read bus cycle of the 8086.

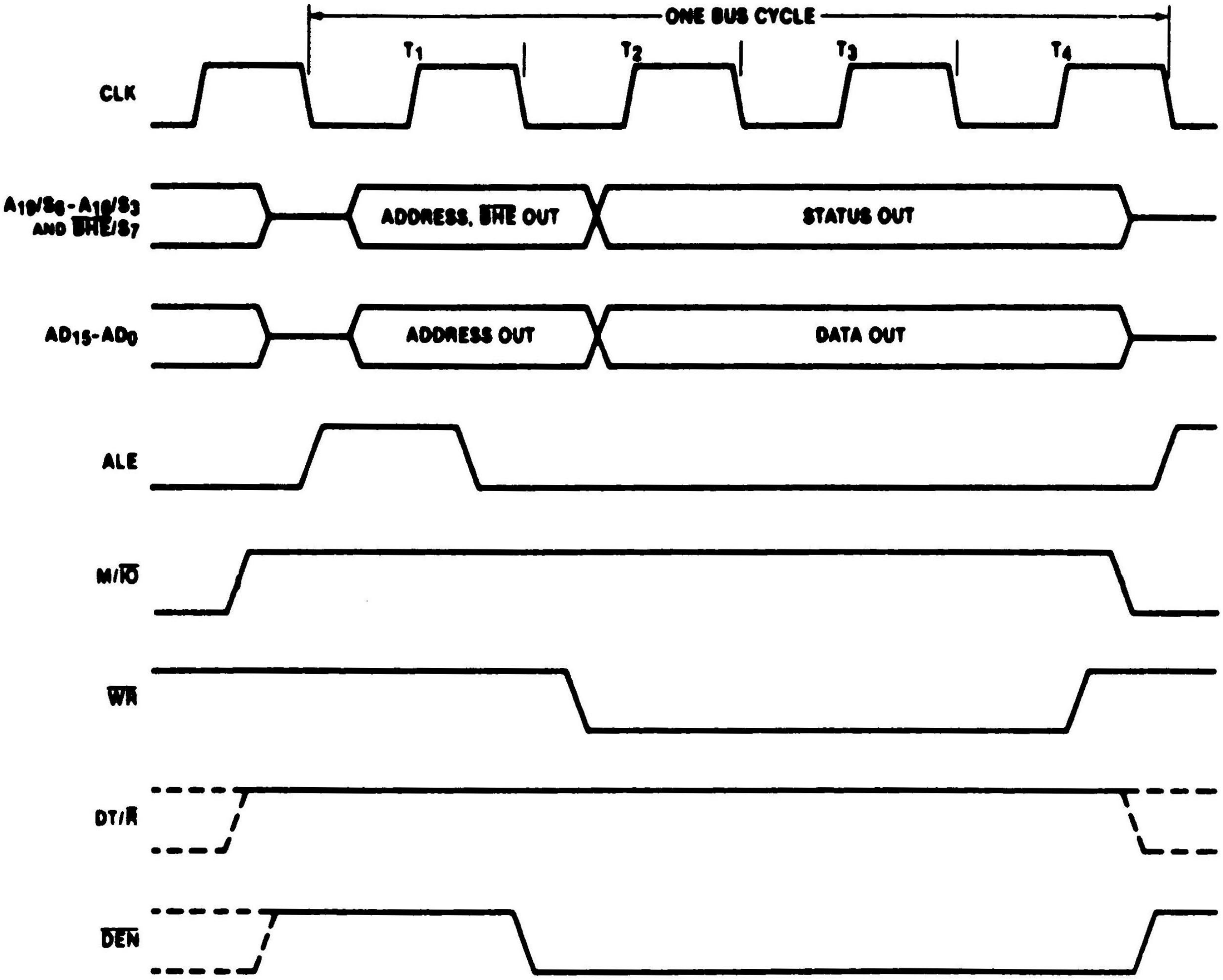


Figure 8-22(b) Minimum-mode memory write bus cycle of the 8086.

8.8 HARDWARE ORGANIZATION OF MEMORY ADDRESS SPACE

- The memory address space of the 8086-based microcomputers has different logical and physical **organizations** (see **Fig. 8-15**).

- **Logically**, memory is implemented as a single **1M × 8 memory chunk**. The byte-wide storage locations are assigned consecutive addresses over the range from 00000_{16} through $FFFFF_{16}$.

- **Physically**, memory is implemented as **two** independent **512Kbyte banks**: the **low (even) bank** and the **high (odd) bank**. Data bytes associated with an even address (00000_{16}, 00002_{16}, etc.) reside in the low bank, and those with odd addresses (00001_{16}, 00003_{16}, etc.) reside in the high bank.

- Address bits A_1 through A_{19} select the storage location that is to be accessed. They are applied to both banks in parallel. $\mathbf{A_0}$ and bank high enable ($\overline{\mathbf{BHE}}$) are used as **bank-select** signals.

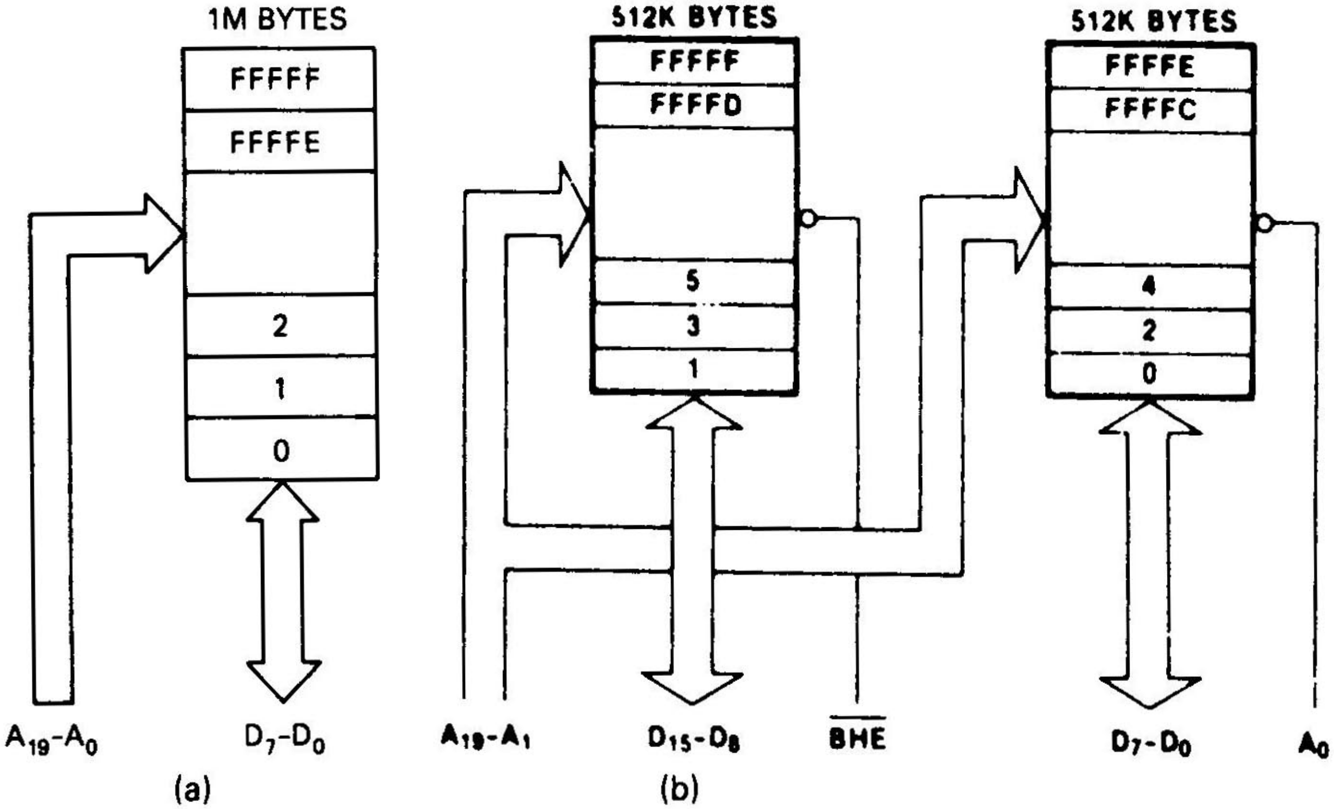


Figure 8-15 (a) Logical memory organization, and (b) Physical memory organization (high and low memory banks) of the 8086 microprocessor.

- **Each** of the memory **banks** provides half of the 8086's 16-bit data bus. The **lower** bank transfers bytes of data over data lines **D_0 through D_7**, while data transfers for a **high** bank use **D_8 through D_{15}**.

- The **8086 microprocessor** accesses memory as follows:

- **Fig. 8-17(a)** shows how a **byte-memory** operation is performed to **address X**, an **even**-addressed storage location. **A_0** is set to **logic 0** to enable the low bank of memory and $\overline{\mathbf{BHE}}$ to **logic 1** to disable the high bank. **Data** are transferred to or from the lower bank over data bus lines **D_0 through D_7**.

- **Fig. 8-17(b)** shows how a **byte-memory** operation is performed to an **odd-**addressed storage location such as **X + 1**. **A_0** is set to **logic 1** and $\overline{\mathbf{BHE}}$ to **logic 0**. This enables the high bank of memory and disables the low bank. Data are transferred over bus lines **D_8** through **D_{15}**. **D_8** represents the **LSB**.

- **Fig. 8-17(c)** illustrates how an **aligned word** (at even address X) is accessed. **Both** the high and low **banks** are accessed at the **same time**. Both **A_0** and $\overline{\mathbf{BHE}}$ are set to 0. This 16-bit word is transferred over the complete data bus **D_0** through **D_{15}** in just **one bus cycle**.

- **Fig. 8-17(d)** illustrates how a **misaligned word** (at address X + 1) is accessed. **Two bus cycles** are needed. During the first bus cycle, the byte of the word located at address X + 1 in the high bank is accessed over **D_8** through **D_{15}**. Even though the data transfer uses data lines **D_8** through **D_{15}**, to the processor it is the **low byte** of the addressed data word. In the **second memory bus cycle**, the even byte located at X + 2 in the low bank is accessed over bus lines **D_0** through **D_7**.

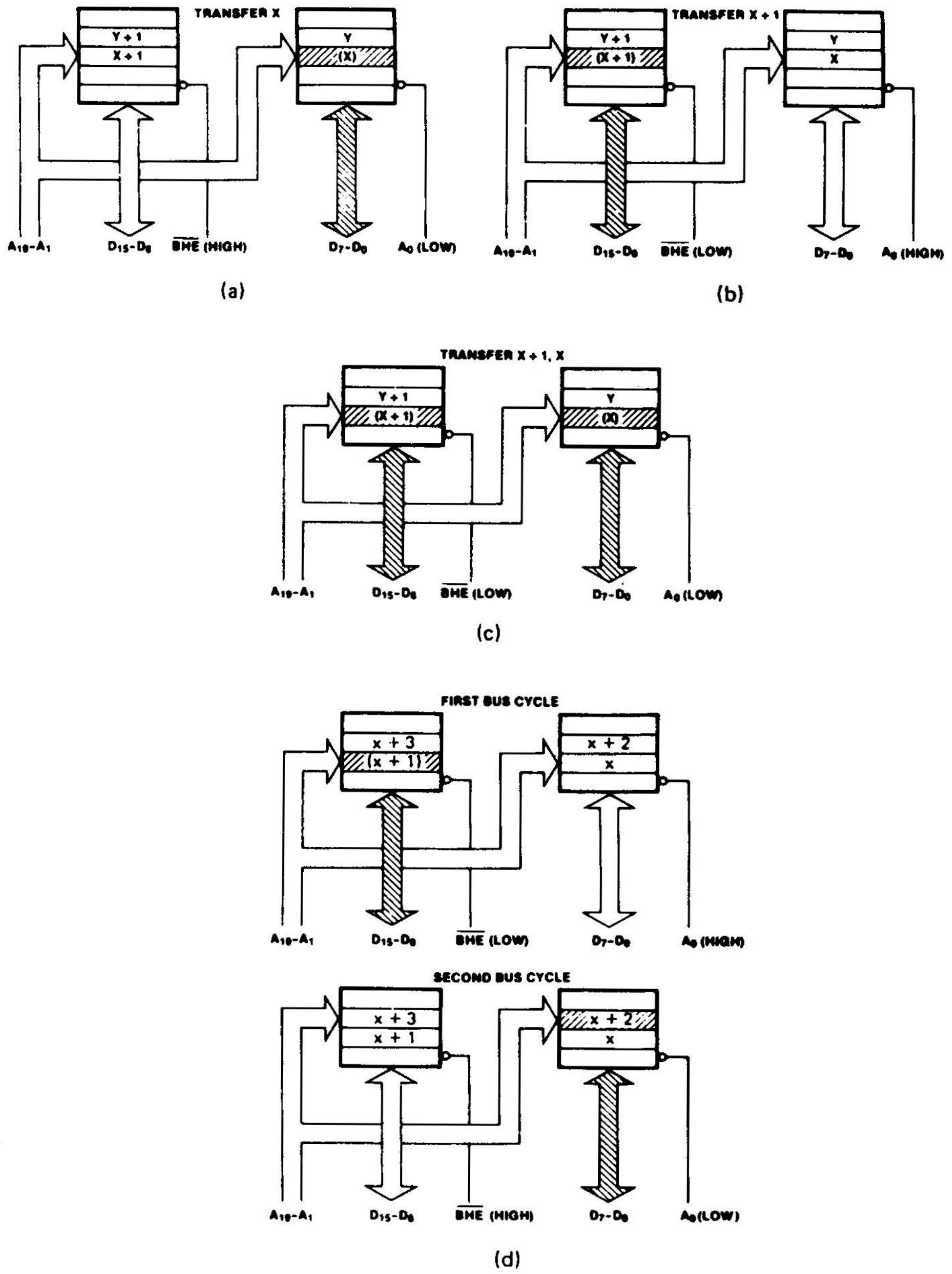


Figure 8–17 (a) Even-address byte transfer by the 8086. (Reprinted with permission of Intel Corporation, © 1979) (b) Odd-address byte transfer by the 8086. (Reprinted with permission of Intel Corporation, © 1979) (c) Even-address word transfer by the 8086. (Reprinted with permission of Intel Corporation, © 1979) (d) Odd-address word transfer by the 8086. (Reprinted with permission of Intel Corporation, © 1979)

8.14 TYPES OF INPUT/OUTPUT

- The 8086 employs two different types of input/output (I/O): **isolated I/O** and **memory-mapped I/O**. These I/O methods differ in how I/O ports are mapped into the 8086's address spaces. We will only consider isolated I/O.

- **Isolated Input/Output:** In this scheme, the I/O devices are treated **separate from** memory (see **Fig. 8-44**). I/O ports are organized as **bytes of data**; the memory address space contains 1M consecutive byte addresses in the range 00000_H, through $FFFFF_H$; and the I/O address space contains 64K consecutive byte addresses in the range 0000_H through $FFFF_H$.

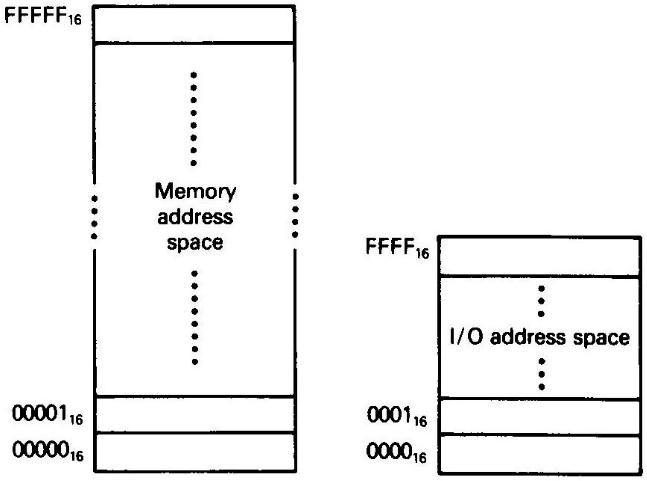


Figure 8–44 8088/8086 memory and I/O address spaces.

- **Fig. 8-45** shows that bytes of data in two consecutive I/O addresses could be accessed as **word-wide data**. For instance, I/O addresses 0000_H, 0001_H, 0002_H, and 0003_H can be treated as independent byte-wide I/O ports: **ports 0, 1, 2 and 3**, or as word-wide **ports 0 and 1**.

- For example, in **Emu8086** the Seven Segment Display was a **word-wide I/O** device that was assigned the addresses **199 and 200**, while the stepper motor was a **byte-wide I/O** device that is assigned the address **7**.

8.15 ISOLATED INPUT/OUTPUT INTERFACE

- The way in which the microprocessor deals with input/output circuitry is similar to the way in which it interfaces with memory circuitry. The only difference is that unlike memory, this time just the 16 least significant lines of the bus, AD_0 through AD_{15}, are in use (because I/O addresses are 16 bit long), and throughout the bus cycles, the $\mathbf{M/\overline{IO}}$ control signal is set to **0**.

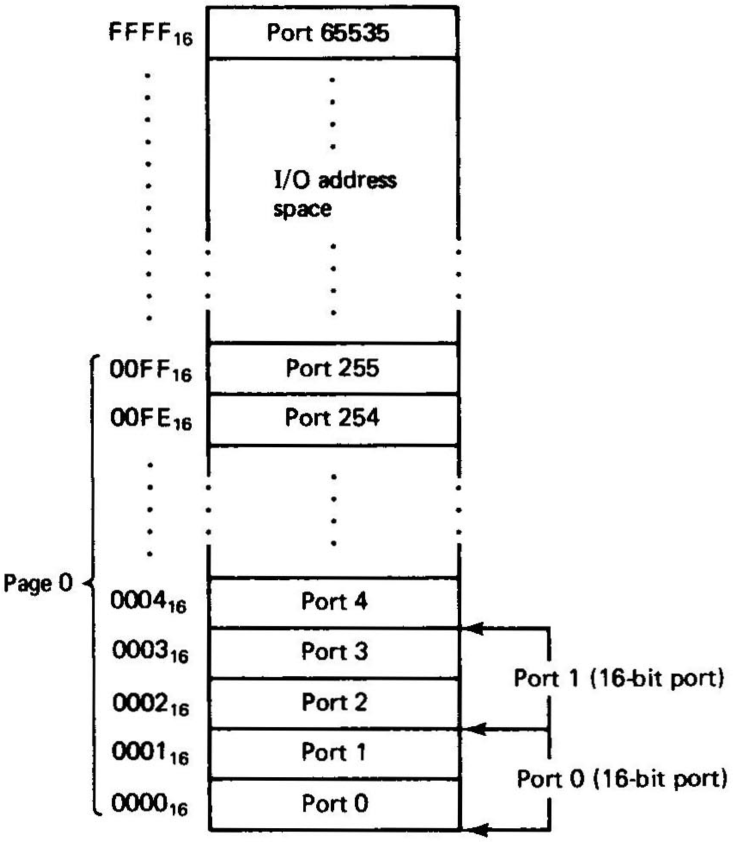


Figure 8-45 Isolated I/O ports.

8.18 NPUT/OUTPUT BUS CYCLES

- All the timing signals in the I/O read and write bus cycles other than the **M/$\overline{\text{IO}}$** are identical to those already described in the memory read/write bus cycle (See **Figs. 8-52** and **8-53**).

8.16 NPUT/OUTPUT DATA TRANSFERS

- Input/output data transfers in the 8086 microcomputers can be either **byte-wide** or **word-wide**.

- Data transfers to byte-wide I/O ports always require **one bus cycle**.

- Word data transfers between the 8086 and I/O devices are accompanied by the code A_0 **$\overline{\text{BHE}}$** = 00 and are performed over the complete data bus D_0 through D_{15}. To ensure that just **one bus cycle** is required for the word data transfer, word-wide I/O ports should be aligned at even-address boundaries.

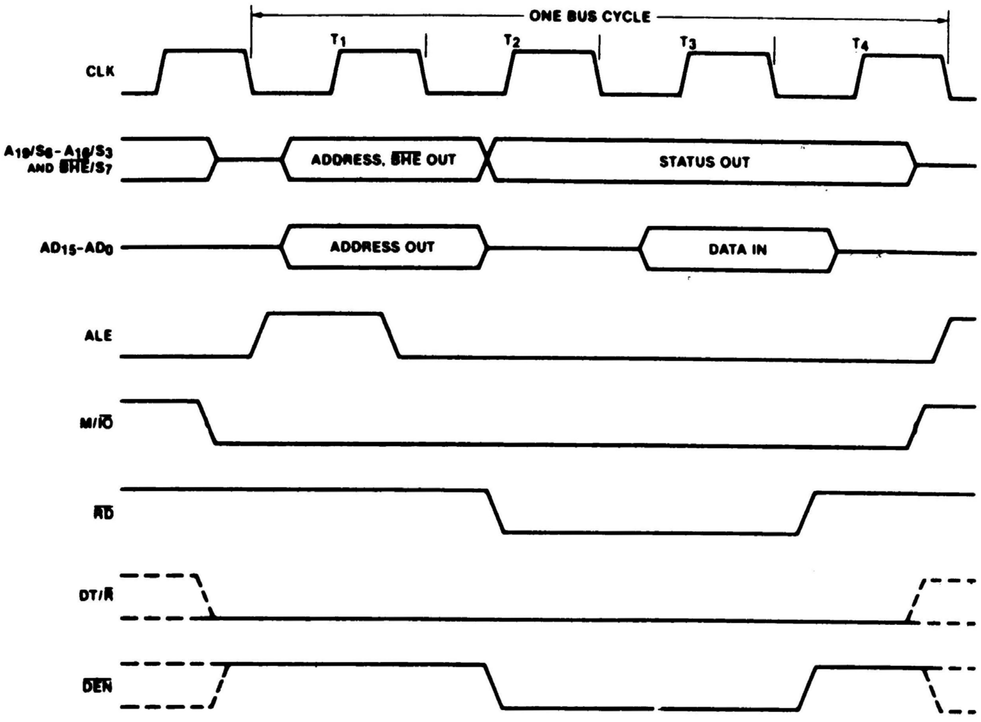


Figure 8–52 Input bus cycle of the 8086.

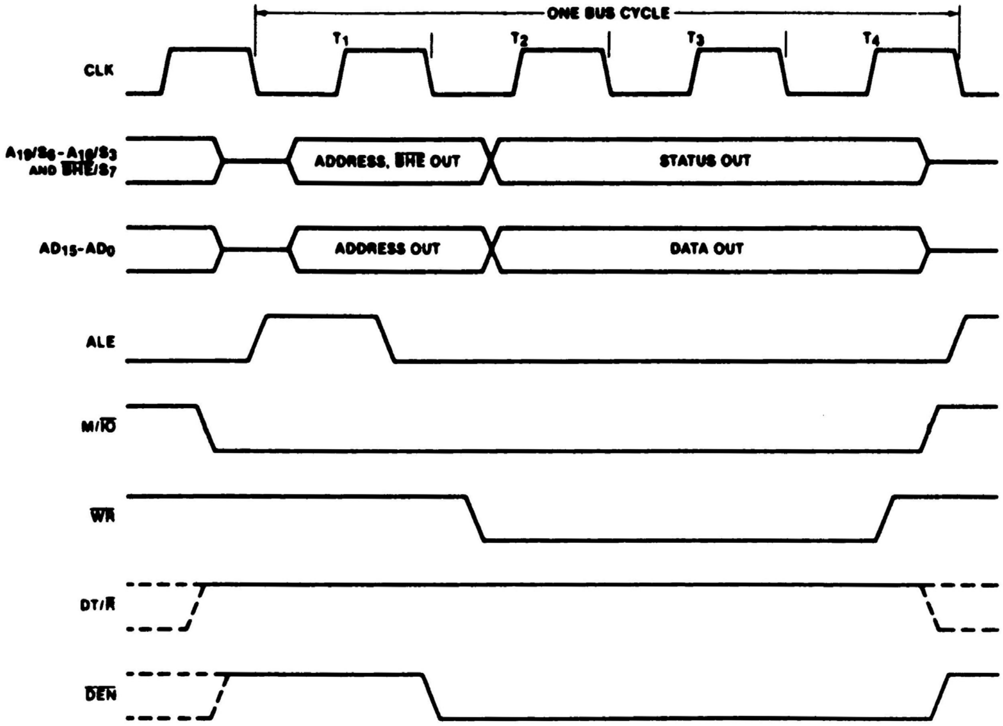


Figure 8–53 Output bus cycle of the 8086.

8.17 INPUT/OUTPUT INSTRUCTIONS

- For **isolated I/O**, special input and output instructions are used. These instructions are (IN) and (OUT), and they are described below:

Mnemonic	Meaning	Format	Operation
IN	Input direct	**IN AL, Address-8-bit**	Port → AL (Byte)
		IN AX, Address-8-bit	Port → AX (Word)
	Input indirect	**IN AL, DX**	Port → AL (Byte)
		IN AX, DX	Port → AX (Word)
OUT	Output direct	**OUT Address-8-bit, AL**	AL → Port (Byte)
		OUT Address-8-bit, AX	AX → Port (Word)
	Output indirect	**OUT DX, AL**	AL → Port (Byte)
		OUT DX, AX	AX → Port (Word)

- Byte transfers involve the AL register, and word transfers the AX Register.

- In a **direct I/O** instruction, the address of the I/O port is specified as part of the instruction. **Eight bits** are provided for this direct address. For this reason, its value is limited to the address range from **00_H to FF_H**. This range is referred to as **page 0** in the I/O address space.

- **Example:** `IN AL, 0FEh` causes the byte-wide I/O port at address FE_h to send its input to the AL register.

- **Example:** To output the data FF_h to a byte-wide output port at address AB_h of the I/O address space, we use:

```
MOV AL, 0FFh
OUT 0BAh, AL
```

- The **indirect I/O** instructions use a 16-bit address that resides in the DX register. The value in DX is not an offset. It is the actual address that is to be output on AD_0 through AD_{15}. Variable I/O instructions can access ports located anywhere in the 64K-byte I/O address space.

- **Example:** To input the contents of the byte-wide input port at $A000_h$ of the I/O address space into BL, we use

```
MOV DX, 0A000h
IN AL, DX
MOV BL, AL
```

- **Example:** To read data from two byte-wide input ports at addresses AA_h and $A9_h$ and the output the data as a word to the word-wide output port at address $B000_h$.

```
IN AL, 0AAh
MOV AH, AL
IN AL, 0A9h
MOV DX, 0B000h
OUT DX, AX
```