**8086 Hardware Specifications**

Table of Contents

[Minimum and Maximum Modes 4](#_Toc83226985)

[8288 Bus Controller 5](#_Toc83226986)

[Status Signals 6](#_Toc83226987)

[Pin Specifications 7](#_Toc83226988)

[7](#_Toc83226989)

[and 7](#_Toc83226990)

[to 7](#_Toc83226991)

[to 7](#_Toc83226992)

[7](#_Toc83226993)

[8](#_Toc83226994)

[and 8](#_Toc83226995)

[8](#_Toc83226996)

[9](#_Toc83226997)

[9](#_Toc83226998)

[9](#_Toc83226999)

[9](#_Toc83227000)

[10](#_Toc83227001)

[10](#_Toc83227002)

[10](#_Toc83227003)

We have previously seen that the Intel 8086 has **40 pins** in **Dual In-Line Package** (DIP). There is also a notch at the top to indicate orientation.



## Minimum and Maximum Modes

The 8086 microprocessor actually has two **modes** and depending on which mode is being used, the functionality of a few of the pins change, specifically pins 24 to 31.

The **minimum mode** is applicable for a **single processor system** printed on a **single printed circuit board** (PCB). This means that the microprocessor, memory and I/O device connections are all on the same PCB, making the communications between them very simple.

The **maximum mode** is applicable for more **complex systems** that have separate I/O and memory boards.

Which mode is being used is specified by **Pin 33**, the pin. A indicates that minimum mode is being used while a indicates that maximum mode is being used.

Using **maximum mode** allows us to have **multiple processors** as opposed to using minimum mode, which only allows a single processor. When using multiple processors, the additional processors are **co-processors**.

For example, we can have the 8087 processor as a co-processor to handle mathematical instructions, while other instructions are handled by the 8086 processor. This allows us to achieve **multiprocessing**. Working in the maximum mode when doing this gives us better performance. If we have a single processor, it means we have just the 8086 processor.

## 8288 Bus Controller

As can be seen, several pins available in the **minimum mode** are missing from the **maximum mode**. For some of these, the functionalities are handled by a special processor called the **8288 Bus Controller**.

These pins are:

* **Address Latch Enable** (ALE)
* **Data Enable** (), which allows data to be transmitted
* **Data Transmit or Receive** ()
* **Interrupt Acknowledgement** ()

The reason we need a separate IC to handle these is because we can have multiple processors in maximum mode. This means we need to also identify which processor is sending a particular signal, which is the work of the 8288 bus controller IC.

By separating out the work into different ICs, we are also improving **performance**, albeit in exchange for a more **complex** circuit, since each IC now needs to concentrate on a specific set of tasks.

## Status Signals

In the **minimum mode**, 8086 has **control signals** for **Memory or I/O** (), **Read** ) and **Write** (). Instead of using these, in **maximum mode**, we use **status signals**.

Each processor generates status signals using the pins , and .

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  | Interrupt Acknowledgement |
|  |  |  | Read I/O Port |
|  |  |  | Write I/O Port |
|  |  |  | Halt |
|  |  |  | Code Access |
|  |  |  | Read Memory |
|  |  |  | Write Memory |
|  |  |  | Inactive |

## Pin Specifications

### 

Pin 33 dictates whether the processor is behaving in **minimum** or **maximum** mode. A indicates maximum mode while a indicates minimum mode.

### and

Pin 40, , and Pins 20 and 1, both , deal with the **power supply**.

### to

These pins either form the **data bus** or a portion of the **address bus**.

### to

These pins, along with the to pins, form the **address bus**.

### 

Pin 25, **Address Latch Enable**, dictates whether the pins to are acting as the **data bus** or are working with the **address bus**. A indicates the former while a indicates the latter. The pin is only present in **minimum mode**.

### 

The **Clock** pin, Pin 19, provides basic timing for the processor and bus controller. Essentially, it alternatively flips between high and low, thus acting as a clock. In 8086, the frequency is or . For comparison, modern computers are almost never below . Note that the clock pin is **receiving** signals from an external oscillator. It is not providing them.

The clock is asymmetric with a  **duty cycle**. This refers to the duty cycle of the others pins. For example, consider the pin. This will not actually remain high or low for the entire clock cycle. Instead, if the signal is meant to be high, it will be high for most of the time and then will become low. With a duty cycle, the ALE pin will be high for of the time, which is why we decide that it is high.

### and

The **Interrupt Request** pin, Pin 18, is used for hardware interrupt requests (as opposed to software interrupt requests). If is high when the **Interrupt Flag** is set to , the **interrupt acknowledgement cycle** begins. While the interrupt is serviced, , Pin 24, is high. The pin is only present in **minimum mode**.

### 

The **Non-Maskable Interrupt**, Pin 17, is like the , except it does not check the **interrupt flag** or consider any **priority**. is still used.

### 

The pin, Pin 21, is held high for **four clock cycles** in order to reset the microprocessor. This causes any **present activity** to be immediately **terminated**.

### 

The pin, Pin 22, is used to force an **idle state**. If it is set to , the microprocessor goes into the idle state. If it is set to , the microprocessor operates normally. Before any execution has started, this pin is set to .

The pin, Pin 23, is an input that is tested by the instruction. A in this pin causes the instruction to function as , which stands for **no-operation**. This just means that the microprocessor is not doing any operation and is in the **waiting state**. A in this pin indicates that the microprocessor is performing some operation, which causes the instruction to wait till this pin becomes again. Note that by waiting, we mean that the microprocessor was executing something and is waiting in between the execution.

The **Bus High Enable** (), Pin 34, is used to enable use of the **higher part** of the **data bus** pins, to . Notice that it works with **logic** , so if it is set to , then the pins are enabled and vice versa.

### 

The **Data Enable** () pin, Pin 26, is set to **high** when data is placed on the **data bus**. Say the microprocessor wants to read or write data from a particular memory location. It will set the address on the address bus, will set the or pin and finally, when it is ready to read or write the data, it will set the pin. At that moment data will be placed on the data bus.

### 

The **Data Transmission or Read** () pin, Pin 27, is set to **high** if the microprocessor is **transmitting data** and set to **low** when the microprocessor is **receiving** data.

This may seem like the and the pins. The difference is that those pins are only used when **data** is being transmitted but the pin is used for all signals, including data.

### 

The **Memory or I/O** () pin, Pin 28, is set to **high** when the microprocessor is communicating with a **memory device** and set to **low** when it is communicating with an I/O device.