Programmable DMA Controller

Table of Contents

[DMA Controller 2](#_Toc106753120)

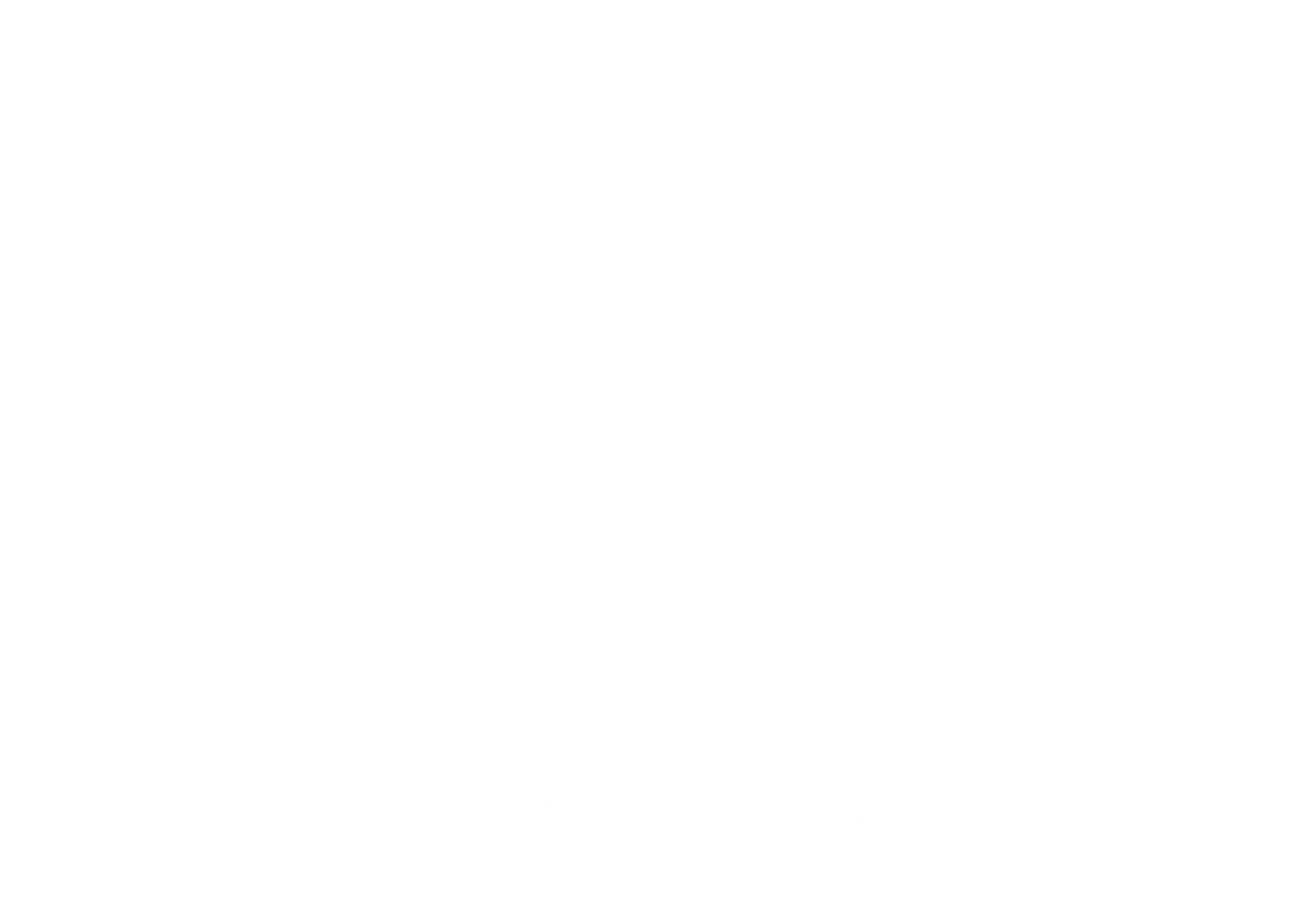
[8237 DMA Controller 4](#_Toc106753121)

[Control Signals 5](#_Toc106753122)

We have previously studied the differences between **Programmed I/O**, which involves the processor handling memory access events with mechanisms such as polling or interrupts, and **Direct Memory Access** (DMA). The major issue with programmed I/O is that it wastes a lot of the processor’s time, since I/O devices are very slow. DMA resolves this issue, freeing the processor from the responsibility and instead giving it to a **DMA Controller**. This increases the data transfer rate from or per byte to per byte. This makes DMA appropriate whenever large amounts of data need to be transferred between memory and an external I/O device.

## DMA Controller

The **DMA controller** can be found either internally or externally. These controllers need to perform functions that are very similar to a microprocessor. However, it is still not independent. It functions by taking instructions from the microprocessor. For example, to read data from an I/O device, the microprocessor must supply the DMA controller with the I/O device number, the main memory buffer address, the number of bytes to transfer, the direction of transfer, etc.

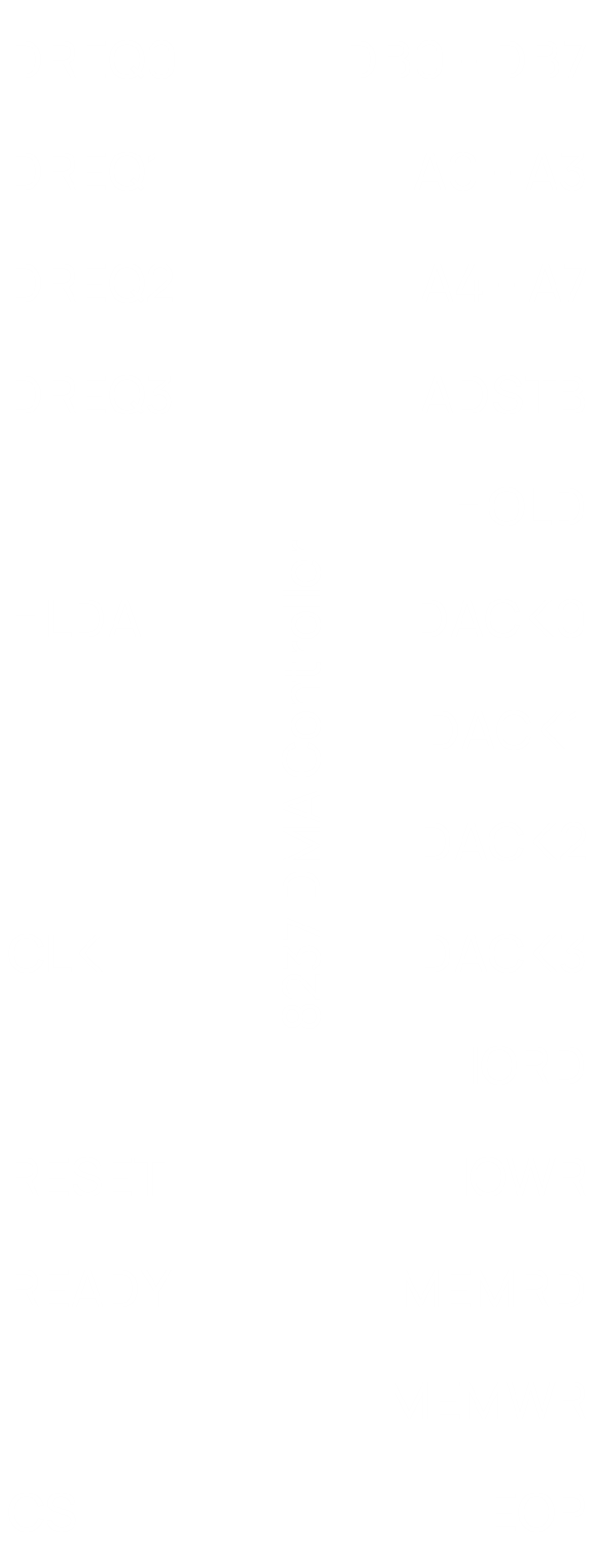


The steps to transfer data using DMA are as follows:

1. The I/O device sends a **DREQ** (DMA Request) to the DMA controller.
2. The DMA controller sends a **HOLD** signal to the CPU.
3. The CPU sends an **HLDA** signal to the DMA Controller.
4. The DMA Controller sends an **DACK** (DMA ACK) to the I/O device.
5. The **IORD** or **IOWR** signal is sent to the I/O device, instructing it to read/write from memory.
6. The **MEMRD** or **MEMWR** signal is sent to the memory, to inform it that data will be read or written to it.

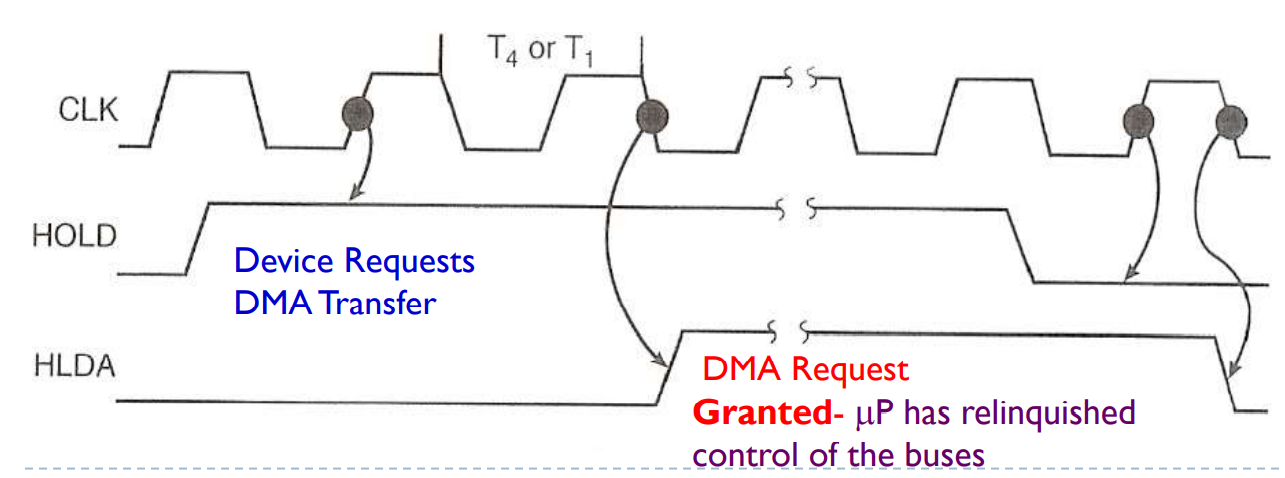
Note that MEMRD and IOWR work together while MEMWR and IORD work together.

## 8237 DMA Controller



The **8237 Programmable DMA Controller** was designed to work with the 8086 microprocessor. As can be seen, it is capable of handling up to 4 I/O devices simultaneously.

## Control Signals



The **HOLD** signal being held **high** indicates that a **DMA request** is being made by the DMA Controller. The microprocessor responds to this by suspending execution and placing the address, data and control buses at **high impedance state**. Once the DMA process has finished, the HOLD signal is held **low**.

If the DMA Controller requires just **one bus cycle**, the process is called **cycle stealing**. If more bus cycles are required, the process is handled separately and independently to the CPU’s execution.