8237 DMA Controller

Presented By:
Diwakar Yagyasen
Asst Prof, CSE, BBDNITM, Lucknow
Introduction

Direct Memory Access (DMA) allows devices to transfer data without subjecting the processor to a heavy overhead. Otherwise, the processor would have to copy each piece of data from the source to the destination. This is typically slower than copying normal blocks of memory since access to I/O devices over a peripheral bus is generally slower than normal system RAM. During this time the processor would be unavailable for any other tasks involving processor bus access. But it can continue to work on any work which does not require bus access. DMA transfers are essential for high performance embedded systems where large chunks of data need to be transferred from the input/output devices to or from the primary memory.
DMA Controller

A DMA controller is a device, usually peripheral to a CPU that is programmed to perform a sequence of data transfers on behalf of the CPU. A DMA controller can directly access memory and is used to transfer data from one memory location to another, or from an I/O device to memory and vice versa. A DMA controller manages several DMA channels, each of which can be programmed to perform a sequence of these DMA transfers. Devices, usually I/O peripherals, that acquire data that must be read (or devices that must output data and be written to) signal the DMA controller to perform a DMA transfer by asserting a hardware DMA request (DRQ) signal. A DMA request signal for each channel is routed to the DMA controller. This signal is monitored and responded to in much the same way that a processor handles interrupts.
When the DMA controller sees a DMA request, it responds by performing one or many data transfers from that I/O device into system memory or vice versa. Channels must be enabled by the processor for the DMA controller to respond to DMA requests. The number of transfers performed, transfer modes used, and memory locations accessed depends on how the DMA channel is programmed. A DMA controller typically shares the system memory and I/O bus with the CPU and has both bus master and slave capability. Fig.16.1 shows the DMA controller architecture and how the DMA controller interacts with the CPU.
DMA Controller ...

— In bus master mode, the DMA controller acquires the system bus (address, data, and control lines) from the CPU to perform the DMA transfers. Because the CPU releases the system bus for the duration of the transfer, the process is sometimes referred to as cycle stealing.

— In bus slave mode, the DMA controller is accessed by the CPU, which programs the DMA controller's internal registers to set up DMA transfers. The internal registers consist of source and destination address registers and transfer count registers for each DMA channel, as well as control and status registers for initiating, monitoring, and sustaining the operation of the DMA controller.
The DMA Controller Architecture
DMA Controller Operation
Steps in a Typical DMA cycle

Device wishing to perform DMA asserts the processors bus request signal.
1. Processor completes the current bus cycle and then asserts the bus grant signal to the device.
2. The device then asserts the bus grant ack signal.
3. The processor senses in the change in the state of bus grant ack signal and starts listening to the data and address bus for DMA activity.
4. The DMA device performs the transfer from the source to destination address.
5. During these transfers, the processor monitors the addresses on the bus and checks if any location modified during DMA operations is cached in the processor. If the processor detects a cached address on the bus, it can take one of the two actions:
   o Processor invalidates the internal cache entry for the address involved in DMA write operation
   o Processor updates the internal cache when a DMA write is detected
6. Once the DMA operations have been completed, the device releases the bus by asserting the bus release signal.
7. Processor acknowledges the bus release and resumes its bus cycles from the point it left off.
The 8237 Pin-out
The 8237 Architecture
The 82C37A direct memory access controller is designed to improve the data transfer rate in systems which must transfer data from an I/O device to memory, or move a block of memory to an I/O device. It will also perform memory-to-memory block moves, or fill a block of memory with data from a single location. Operating modes are provided to handle single byte transfers as well as discontinuous data streams, which allows the 82C37A to control data movement with software transparency. The DMA controller is a state-driven address and control signal generator, which permits data to be transferred directly from an I/O device to memory or vice versa without ever being stored in a temporary register. This can greatly increase the data transfer rate for sequential operations, compared with processor move or repeated string instructions. Memory-to-memory operations require temporary internal storage of the data byte between generation of the source and destination addresses, so memory-to-memory transfers take place at less than half the rate of I/O operations, but still much faster than with central processor techniques. The block diagram of the 82C37A is shown in Fig.16.6. The timing and control block, priority block, and internal registers are the main components. The timing and control block derives internal timing from clock input, and generates external control signals. The Priority Encoder block resolves priority contention between DMA channels requesting service simultaneously.
Questions

Q.1. Draw the architecture of 8237 and explain the various parts.

Ans. Refer slide 9.
Questions.

Q.2. Highlight on different modes of DMA data transfer. Which mode consumes the list power and which mode is the fastest?
Ans. DMA Transfer modes.

Single, block, and demand are the most common transfer modes. Single transfer mode transfers one data value for each DMA request assertion. This mode is the slowest method of transfer because it requires the DMA controller to arbitrate for the system bus with each transfer. This arbitration is not a major problem on a lightly loaded bus, but it can lead to latency problems when multiple devices are using the bus. Block and demand transfer modes increase system throughput by allowing the DMA controller to perform multiple DMA transfers when the DMA controller has gained the bus. For block mode transfers, the DMA controller performs the entire DMA sequence as specified by the transfer count register at the fastest possible rate in response to a single DMA request from the I/O device. For demand mode transfers, the DMA controller performs DMA transfers at the fastest possible rate as long as the I/O device asserts its DMA request. When the I/O device unasserts this DMA request, transfers are held off.