Direct Memory Access (DMA) controller design process

Simon Musgun[1,⋆]

[1] *San Francisco Bay University, 161 Mission Falls Ln, Fremont, USA*

Abstract. This academic essay discusses the design process of a Direct Memory Access (DMA) controller using the 8257DMA datasheet as a reference. The design process involves understanding the functionalities and features of the 8257DMA, determining the system requirements, selecting appropriate interfacing circuits, designing the control and status registers, and testing and verifying the system operation. The essay highlights the importance of proper design and testing to ensure the successful implementation of DMA in modern computing systems.

Introduction

Direct Memory Access (DMA) controllers have become essential in modern computer systems, providing a means of data transfer between input/output devices and memory without CPU intervention. The 8257DMA is a widely used integrated circuit that o*ff*ers e*ffi*cient DMA transfer operations. This academic essay discusses the design process of a DMA controller using the 8257DMA datasheet as a reference, focusing on the functional blocks of the chip and the Verilog modules that will be used in the design.

The 8257DMA chip comprises several functional blocks that are essential to its operation. These blocks include the Data Bus Bu*ff*er, Address Latch, Control Logic, Mode Set Logic, Request and Acknowledge, DMA Transfer Control, and Memory and I/O Control blocks.

The 8257 is programmable. Direct Memory Access (DMA) device which, when coupled with single Intel® 8212 I/O port device, provides complete four-channel DMA controller for use in Intel® microcomputer systems. After being initialized by software, the 8257 can transfer block of data, containing up to 16.384 bytes, between memory and peripheral device directly, without further intervention required of the CPU. Upon receiving DMA transfer request from an enabled peripheral, the 8257:

1 DMA chip functional blocks

The 8257DMA (Direct Memory Access) controller is an older DMA controller chip that was commonly used in earlier computer systems. It consists of several functional blocks that work together to manage data transfers between peripherals and memory.

These functional blocks work together to provide a flexible and e*ffi*cient mechanism for transferring data between peripherals and memory without CPU intervention. The 8257DMA controller is an older device, and modern DMA controllers may have additional functional blocks and features to support more advanced data transfer operations.

1.1 DMA Channels

The DMA channel module is responsible for managing a single DMA channel. It contains registers that control the transfer, including the source and destination addresses, transfer mode, and transfer size. The DMA channel module also contains a DMA request line that triggers the DMA transfer when activated.

It supports up to four independent DMA channels, each capable of transferring data independently from a different peripheral device to memory or vice versa. The 8257 DMA channels are (labeled CH-0 to CH-3). Each channel includes two sixteen-bit registers:

- DMA address register.
- terminal count register.

The DMA address register is a control register within the Direct Memory Access (DMA) controller that stores the starting address of the memory block to be transferred during a DMA operation. It is a 16-bit register and is used in conjunction with the DMA count register to specify the source and destination addresses of a DMA transfer.

The Terminal Count (TC) register is a 16-bit control register within the Direct Memory Access (DMA) controller that stores the maximum number of bytes to be transferred during a DMA operation. The TC register is used in conjunction with the DMA address register and the DMA count register to specify the source and destination addresses of a DMA transfer.

The value loaded into the low-order 14-bits of the terminal count register specifies the number of DMA cycles minus one before the Terminal Count (TC) output is activated.

⋆ e-mail: smusgun5544@student.sfbu.edu

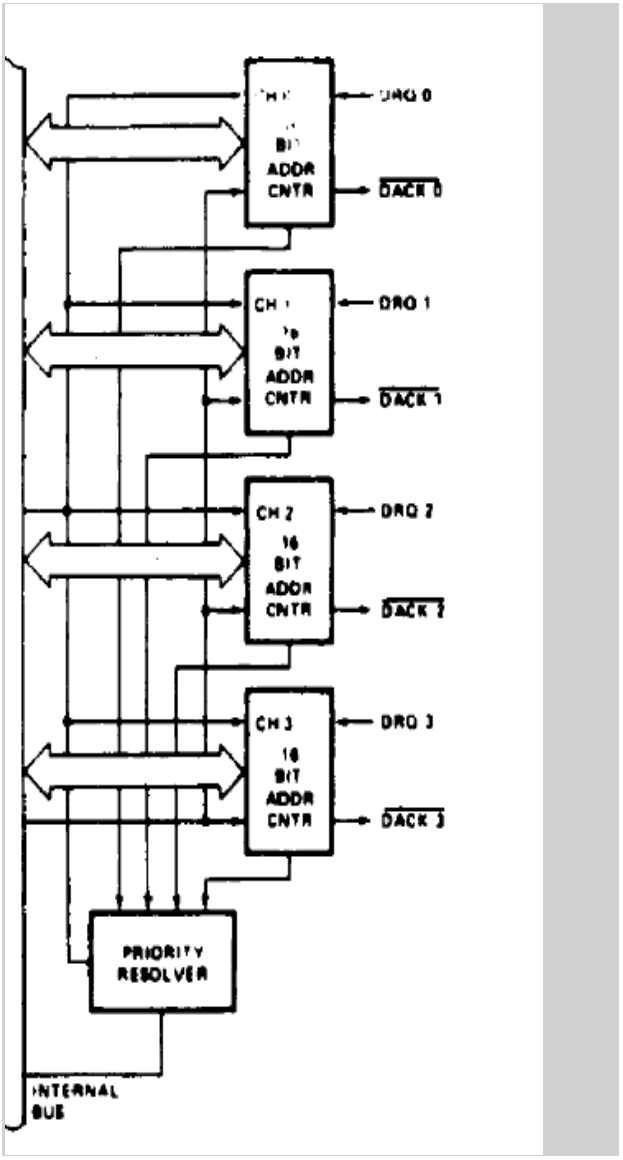

Figure 1. 8257 Block Diagram of DMA Channels

Table 1. DRQ line is associated with a specific channel

DRQ line	Channel
DRQ 0	Channel 0
DRQ 1	Channel 1
DRQ 2	Channel 2
DRQ 3	Channel 3

(DRQ 0 - DRQ 3)

DMA Request: These are individual asynchronous chan nel request inputs used by the peripherals to obtain DMA cycle. If not in the rotating priority mode then DRQ 0 has the highest priority and DRQ has the lowest.

(DACK 0 - DACK 3)

The DMA acknowledge (DACK) signal is an output signal from the 8257 DMA controller that is used to inform the I/O device that it has been selected for a DMA cycle.

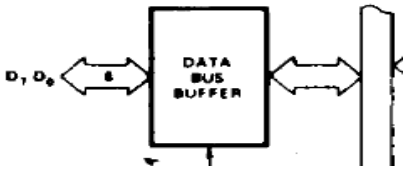

Figure 2. 8257 Data Bus Buffer

When an I/O device sends a DMA request signal to the controller, the controller selects the appropriate channel and sends a DACK signal to the device.

Once the DACK signal is active, the I/O device can begin transferring data to or from memory without the need for intervention from the CPU. The DMA controller manages the data transfer and updates the channel parameters (such as the address and count registers) as necessary.

1.2 Data bus buffer

The data bus buffer is an important functional block in the 8257 DMA controller that is responsible for interfacing the controller with the system data bus. It acts as a bidirectional buffer that allows data to be transferred between the DMA controller and the system bus.

The data bus buffer is made up of two sets of buffers: the input buffer and the output buffer. The input buffer is used to buffer data from the system bus that is being read by the DMA controller, while the output buffer is used to buffer data that is being written by the DMA controller to the system bus.

The input buffer consists of four tri-state buffers, one for each DMA channel. These buffers are enabled when the corresponding channel is selected for a DMA cycle, allowing data to be read from the system bus and buffered for use by the DMA controller.

Similarly, the output buffer also consists of four tri-state buffers, one for each DMA channel. These buffers are enabled when the corresponding channel is selected for a DMA cycle, allowing data to be written to the system bus by the DMA controller.

These are bi-directional three-state lines. when the 8257 is being programmed by the CPU. eight bits of data for DMA address register, terminal count register, or Mode Set register are received on the data bus. When the CPU reads DMA address register, terminal count register or the Status register, the data is sent to the CPU over the data bus.

1.3 Read/Write Logic

The read/write logic is a functional block in the 8257 DMA controller that controls the direction of data transfer during DMA cycles. It determines whether data is being transferred from the DMA controller to an I/O device or from the I/O device to the DMA controller.

The Read/Write Logic accepts the I/O Read (USE) or I/O Write (175OT) signal, decodes the least significant four address bits, (A0-A3), and either writes the contents of the data bus into the addressed register (if I/OW is true) or places the contents of the addressed register onto the data bus (if I/OR is true).

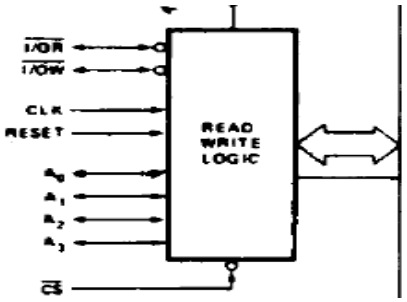

Figure 3. Read/Write Logic Function

I/O Read:

When an I/O read operation is initiated by the DMA controller, the channel register for the selected channel is first loaded with the appropriate parameters, including the starting memory address, the transfer length, and the transfer mode. The DMA controller then sends a request to the I/O device to initiate the data transfer.

I/O Write:

Once the I/O device has acknowledged the request and is ready to receive data, the DMA controller sends a signal to the read/write logic to enable the output buffer and disable the input buffer. The read/write logic then controls the tri-state buffers to allow data to be transferred from the DMA controller to the I/O device.

CLK and RESET:

The CLK signal is used by the various functional blocks of the DMA controller to ensure that they operate in a synchronized and coordinated manner. For example, the channel registers, read/write logic, and data bus buffer all use the CLK signal to ensure that they operate on the correct clock cycles and that data is transferred correctly between the DMA controller and the I/O device.

The reset signal is typically generated by an external device, such as a microprocessor or system controller. The exact timing and duration of the reset signal may vary depending on the specific system requirements, but it is typically a brief pulse that is sent to the DMA controller to initiate the reset sequence.

Address lines:

During a DMA transfer, the system controller or microprocessor provides the DMA controller with the starting address for the transfer, which is typically stored in the DMA address register. The DMA controller then uses the address lines to access the memory or I/O device at the specified address and transfer data to or from that device.

Chip Select:

The CS signal is generated by the DMA controller and is typically connected to the peripheral device's chip select input. When the DMA controller begins a data transfer to or from the peripheral device, it asserts the CS signal to select the device. The device responds by activating its own DMA acknowledge signal (DACK), which informs the DMA controller that the device is ready to receive or transmit data.

1.4 Control Logic

The control logic block in the 8257 DMA controller is responsible for coordinating and controlling the overall operation of the controller. It consists of a set of control registers, counters, and logic circuits that manage the data transfer process and ensure that the DMA controller operates correctly.

The control logic block includes several functional blocks, including the initialization and mode set block, the DMA request priority resolver block, the channel control block, and the master control block.

Address line and READY

These A4 up to A7 address lines are three-state outputs which constitute bits through of the 16-bit memory address generated by the 8257 during all DMA cycles.

The ready signal is used to coordinate the transfer of data between the DMA controller and the peripheral device. When the DMA controller is ready to transfer data, it asserts the ready signal, indicating to the peripheral device that it is ready to receive or send data. The peripheral device then responds by asserting the DMA request (DRQ) signal, requesting the transfer of data.

Hold Request and Hold acknowledge

When the DMA controller needs to perform a data transfer, it asserts the HRQ signal to request the CPU to release the control of the system bus. The CPU responds by de-asserting the Hold Acknowledge (HLDA) signal to indicate that it has released the control of the system bus.

Once the DMA controller has control of the system bus, it can perform data transfers without interference from the CPU. During the data transfer, the DMA controller may need to access shared resources on the system bus, such as the memory or an I/O device. If the CPU needs to regain control of the system bus during the data transfer, it can do so by asserting the HLDA signal, indicating to the DMA controller that it should release the

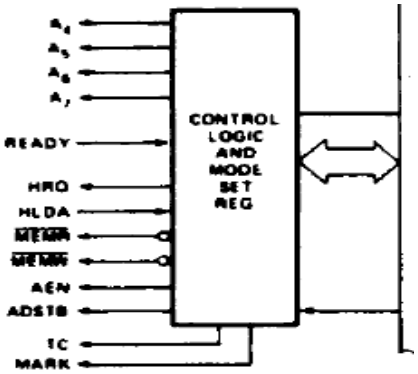

Figure 4. Control logic block

control of the system bus.

Memory Read and Memory write

During a memory read operation, the DMA controller uses the address lines and control signals to read the data from the specified memory location. The data is then transferred to the peripheral device using the data bus and control signals such as DRQ, DACK, and READY. Once the transfer is complete, the DMA controller updates the address register and the count register and may signal the peripheral device using the EOP signal.

Similarly, during a memory write operation, the DMA controller uses the address lines and control signals to write the data from the peripheral device to the specified memory location. The data is transferred from the peripheral device to the data bus and then written to the memory location using the control signals such as WR, RD, and READY. Once the transfer is complete, the DMA controller updates the address register and the count register and may signal the peripheral device using the EOP signal.

Address Strobe and Address enable

Address Strobe (AS) is a control signal used to indicate that the address on the address lines is valid and can be latched by the memory. The AS signal is generated by the DMA controller and is typically active low. When the AS signal goes low, the memory latches the address on the address lines and prepares to access the data at the specified memory location.

Address Enable (AE) is a control signal used to enable the memory for a read or write operation. The AE signal is typically active low and is used to indicate that the memory should perform a read or write operation based on the control signals such as read (RD) and write (WR) and the data present on the data bus.

Terminal Count and mark

Terminal Count (TC) is a control signal used to indicate that the DMA transfer has been completed. The DMA controller asserts the TC signal at the end of a transfer, indicating to the peripheral device that the transfer is complete. The peripheral device can then initiate another DMA transfer if required.

Modulo 128 Mark (M/128) is a control signal used for block transfer operations. When the DMA controller is programmed for block transfers, it automatically decrements the count value after each byte transfer. When the count value reaches 0, the DMA controller asserts the TC signal to indicate that the transfer is complete.

1.5 Mode Sat Register

the mode sat register is a controller that is used to program the DMA transfer mode. The MSR contains four bits that are used to configure the DMA transfer mode.

- The Memory-to-Memory Transfer Mode Select (D0) bit is used to select between a memory-to-memory transfer or an I/O-to-memory transfer. When D0 is set to 1, the DMA transfer is a memory-to-memory transfer. When D0 is set to 0, the DMA transfer is an I/O-to-memory transfer.
- The Block Transfer Mode Select (D1) bit is used to select between a single transfer or a block transfer. When D1 is set to 1, the DMA controller performs block transfers. When D1 is set to 0, the DMA controller performs single transfers.
- The Compressed Timing Select (D2) bit is used to select between compressed timing or normal timing. When D2 is set to 1, the DMA controller uses compressed timing. When D2 is set to 0, the DMA controller uses normal timing.
- The Transparent Mode Select (D3) bit is used to select between transparent mode or non-transparent mode. When D3 is set to 1, the DMA controller uses transparent mode. When D3 is set to 0, the DMA controller uses non-transparent mode.

1.6 Status Register

The status register is an 8-bit register in the 8257 DMA controller that provides information about the current status of the DMA transfer. It is a read-only register, which means that the CPU can only read the contents of this register and cannot write to it.

The TC status bits are set when the Terminal Count (TC) output is activated for that channel. These bits remain set until the status register is read or the 8257 is reset. The UPDATE FLAG, however, is not a*ff*ected by status register read operation.

2 functional test plan for the 8257DMA

- Register initialization test:

  This test ensures that all the internal registers are properly initialized upon power-up or reset. The test will involve reading the values of all the internal registers, such as the mode register, command register, address register, and terminal count register. The expected values should match the default values as specified in the datasheet.

- Data transfer test:

  This test ensures that the DMA controller is capable of transferring data between the memory and the I/O devices. The test will involve configuring the DMA controller in the appropriate mode and initiating a data transfer operation. The test will verify that the correct data is transferred and that the transfer is completed within the specified time.

- Interrupt generation test:

  This test ensures that the DMA controller is capable of generating interrupts upon completion of a data transfer operation. The test will involve configuring the DMA controller to generate interrupts and initiating a data transfer operation. The test will verify that the interrupt signal is generated upon completion of the transfer and that the interrupt is serviced by the processor.

- Channel arbitration test:

  This test ensures that the DMA controller is capable of arbitrating between multiple DMA requests and servicing them in the appropriate order. The test will involve configuring multiple channels with different priority levels and initiating DMA requests simultaneously. The test will verify that the DMA controller services the requests in the correct order based on their priority levels.

- Error handling test:

  This test ensures that the DMA controller is capable of handling errors during a data transfer operation. The test will involve inducing errors, such as parity errors, and verifying that the DMA controller is able to detect and handle the errors appropriately.

- Timing analysis test:

  This test ensures that the DMA controller is capable of meeting the timing requirements specified in the datasheet. The test will involve measuring the time taken for a data transfer operation and verifying that it meets the specified maximum transfer time.

- Boundary test:

  This test ensures that the DMA controller is capable of handling data transfers that cross memory or I/O device boundaries. The test will involve initiating data transfers that cross the boundary and verifying that the DMA controller is able to handle them correctly.

Summary

In summary, the 8257 Direct Memory Access (DMA) controller is a popular chip used to transfer data between a memory and an I/O device without involving the CPU. This chip consists of several functional blocks, including the DMA channels, address register, terminal count register, data bus buffer, control logic, and status register. Each of these blocks plays a crucial role in the DMA process and needs to be designed and tested properly.

The design process for the 8257 DMA controller involves the creation of Verilog modules for each functional block, as well as the integration of these modules into a complete system. This requires a thorough understanding of the chip's datasheet, including its pinouts, timing diagrams, and control signals.

To ensure that the design works as intended, a functional test plan must be developed and executed. This plan should include a range of tests to verify that each functional block operates correctly and that the DMA transfer process works as expected. Some of the tests that can be included in the plan are the data transfer test, status register test, terminal count test, and interrupt test.

Overall, the design and testing of the 8257 DMA controller require careful attention to detail and a good understanding of the chip's specifications. By following the proper design process and executing a thorough functional test plan, it is possible to create a reliable and efficient DMA controller for various applications.

References

[1] A. Mecke, I. Lee, J.R.B. jr., M.M.B. Holl, B.G. Orr, European Physical Journal E: Soft Matter **14**, 7 (2004)

[2] 8257/8257-5 programmable DMA controller dat -sheet

[3] https://www.alldatasheet.com/datasheet-pdf/pdf/127002/INTEL/8257.html

[4] Koti, J., & Kumar, G. M. Cost Efective and High Speed Design of DMA Memory Copy Accelerator.