International Journal of Current Engineering and Technology
E-ISSN 2277 – 4106, P-ISSN 2347 – 5161

Available at http://inpressco.com/category/ijcet

Research Article

Design and Implementation of Generic DMA using Vhdl

Anil Didariya*, Harshbardhan Singh Jasrotia, Rohan Gupta, Sanjeev Gurjar and Mr. Neeraj Tripathi

Electronics and Communication Department, Shri Mata Vaishno Devi University, Katra Jammu and Kashmir, India

Accepted 22 May 2016, Available online 26 May 2016, **Vol.6, No.3 (June 2016)**

Abstract

In this paper, a generic Direct Memory Access (DMA) Controller is designed which is dynamically reconfigurable and it has different control feature which is programmable. This DMA controller works in two different modes. In the first mode, there is fixed data bus(8 bit), address bus(16 bit) and word count register(16 bit) and in second mode , there is variable data bus(8,16,32,64 bit), address bus(16,24,32 bit) and word count register(8,16,24,32 bit) i.e. We can reconfigure this DMA controller just by changing some words or numbers in the existing VHDL code instead of writing or changing the whole code.

Keywords: *VHDL, Generic, DMA controller (DMAC), Finite State Machine etc .*

1. Introduction

In a computer system, DMA is a feature due to which the input/output devices can access the RAM of the computer independently of CPU. Without DMA, it is not possible for CPU to perform other tasks at the same time when the data is being transferred. But with DMA, it is possible for CPU to operate on other tasks during transfer of data. The CPU initiate the transfer of data. During the transfer of data between the I/O device and DMA channel , the CPU is independent to operate on other tasks. After the completion of transfer of data, an interrupt request from the DMA controller is received by the CPU. DMA is extremely important in those embedded systems where high performance, speed and multitasking is required because DMA makes the CPU free so that it can perform some other tasks which do not require data bus. DMA is generally used in those systems where a large chunks of data is transferred.

2. DMA operation

Whenever a DMA operation is to be performed, the following steps will have to be taken. The I/O device which desires direct memory access indicates its intentions by giving a request to DMA controller Having process this request the DMA controller indicates to the CPU through the HOLD signal the intention for DMA. The CPU immediately tristates its Address bus, Data bus and Control bus and responds with HOLD Acknowledge (HLDA) signal to the DMA controller. Having received the Hold acknowledge signal, the DMA controller gives a Data Acknowledge(DACK) to the peripheral device which requested DMA and then onwards allow the peripheral device to perform data transfer with respect to the memory without using CPU. Before DMA operation is initiated, the DMA controller will be in slave mode with respect to the CPU. After the transfer is complete, DMA controller sends a signal to CPU and the control is shifted to CPU again.

3. DMA channels

The DMA controller has four DMA channels viz CH-0, CH-1, CH-2 and CH-3. Two sixteen bit resisters are present in each DMA channel. There are (1) DMA address registers (2)Terminal count register.

Before enabling a channel, the registers (Both Address and Terminal count) must be initialized. In the DMA address register, the memory location address to be accessed is firstly loaded onto the DMA address register. The value of the Low significant 14 bits of the register tells the number of DMA cycles minus one for eg. If n is number of DMA cycles, then the low order 14 bits of the Terminal count register must be loaded with the value of n-1. The type of DMA operation is defined by the remaining two most significant bits of the Terminal count register. These two bits cannot be changed during a DMA cycle but it can be modified between DMA blocks. The basic function of the DMA channel is to accept a DMA Request (DRQn) input and provides a DMA Acknowledge (DACKn) Output.[CHEN Shuang-yan, WANG Dong-hui, HOU Chaohuan(2007)]

*Corresponding author: **Anil Didariya**

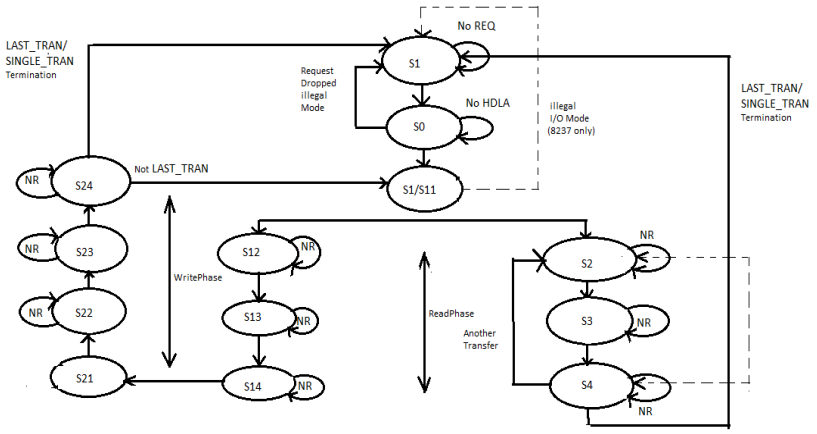

Fig.1 Finite state machine of proposed DMA Controller (DMAC)

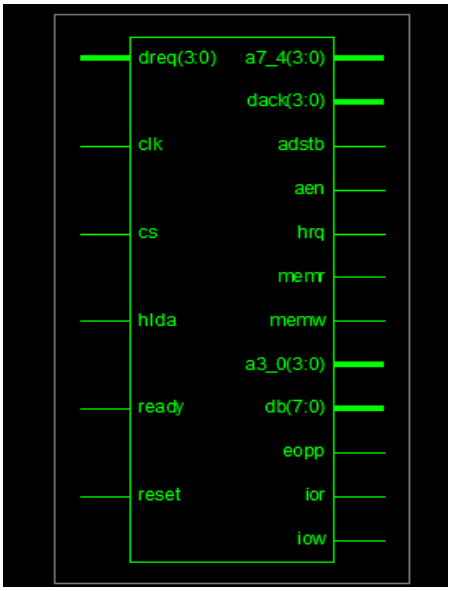

Fig.2 Pin diagram of proposed DMA

4. Signal pins used in DMA: The types of signals used in DMA controller are as follow:

4.1 DMA Request (DRQ0-DRQ3)

To acquire a DMA cycles, the peripheral uses these individual Asynchronous channel requests input (DRQ0-DRQ3). The priority sequence from highest to lowest are DRQ0-DRQ1-DRQ2-DRQ3 if it is not in rotating priority mode. By raising the request lines, a request can be generated and hold it high until the DMA acknowledge it. In Burst mode i.e for multiple DMA cycles, the request line is hold high until the DACK of the last cycle arrives [Vibhu Chinmay, Shubham Sachdeva (2014)]

4.2 DMA Acknowledge (DACK0 to DACK3)

It is an active low signal and it's work is to tell the peripheral devices connected to the channel that it is selected for the DMA cycle. The output DACK acts as a "chip select" for the peripheral device requesting service. This line goes inactive (high) and active (low) once for each byte transferred even if a Burst of data is transferred.

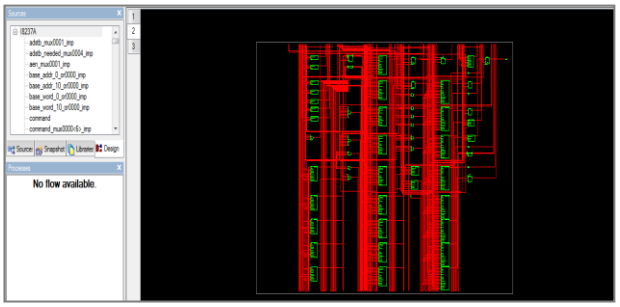

Fig.3 Internal diagram of proposed DMA

4.3 Data Bus Buffer

This eight bit bidirectional buffer interfaces the DMA controller to the data bus of the system.

4.4 Data Bus Lines (D0-D7)

These are three state lines and they are bidirectional. When the DMA controller is being programmed by the CPU, eight bits of data for mode set register, Terminal count register or DMA address register are received on the Data bus. Also when the CPU reads status register, DMA address register or Terminal count register then the data is send to CPU through data bus. At the beginning of a DMA cycle, the most significant eight bits of memory address will be transferred then the bus will be released to handle the memory data transfer during the DMA cycle.

4.5 Read/write Logic

The I/O write and memory read or I/O read and memory write signals are generated by the Read/write Logic during the direct memory access cycles. It also control the data associated with the peripheral that has been granted the DMA cycle.

4.6 I/O Read (I/OR)

It is an active low signal having bi-directional three state line. In "slave" mode, it acts as an input which allows the terminal count register or the upper/lower byte of a 16 bit DMA address register or 8 bit status register to be read. I/OR serves as a control output in "Master" mode which is used to access data from a peripheral device during the DMA write cycle.

4.7 I/O Write (I/OR)

It is also an active Low signal having bidirectional three state line. In slave mode, it acts as an which allows the constant of the data bus to be loaded into terminal count register or upper /lower byte if a 16 bit DMA address register or 8 bit status register. In "Master" mode, I/OR serves as a control output which allows data to be output to a peripheral during a DMA read cycle.

4.8 Chip Select (CS)

It is an active low input signal that can enables the I/O Read or I/O Write when 8237 is programmed when it is in the slave mode. Control signal gets automatically disabled in order to prevent the chip from selecting itself when it is performing the DMA operation.

4.9 Address Lines (A0-A3)

These are the least significant four address lines which are bidirectional. In the "Slave" mode, these are the inputs that select anyone of the registers to be read or write. In the "Master" mode, these are the outputs which constitutes the least four significant bits of 16 bit memory address.

4.10 Reset

An asynchronous input that disables all DMA channels thereby clearing the mode registers and control lines.

Address lines (A4-A7):

These are the four address lines which are three state output that constitutes four through 7 of the 16 bits memory address during all DMA cycles.

4.11 Ready

It is an asynchronous input that is used to elongate the memory read and write cycles with wait states.

4.12 Hold Request (HRQ)

It is a signal that is given by DMA telling it to relinquish the control of the address buses and control buses. [Vibhu Chinmay , Shubham Sachdeva (2014)]

4.13 Hold Acknowledge (HLDA)

It is an acknowledgement signal that is sent back to the DMA by CPU in response to the HOLD signal giving confirmation that it has relinquished the control of the address buses and the data buses.[Vibhu Chinmay , Shubham Sachdeva (2014)]

4.14 Memory Read (MEMR)

It is a active low three state output that is used to read the data from the address memory location when the DMA is performing the read cycle.

4.15 Memory Write (MEMW)

It is an active low three state output that is used to write the data from the address memory location when the DMA is performing the write cycle.

4.16 Address Strobe (ADSTB)

It is an output that strobes the most significant bit of the memory address from the data bus.

4.17 Address Enable (AEN)

It is an output that is used to float the system data bus and control bus. It is also used to disable the address bus by enabling the address bus drivers in systems in order to inhibit non DMA devices from responding when the DMA cycles is going on. It is used to isolate data bus from the system data bus [Rashmi misra, Rupal chauhan, Garima arora (2013)]

4.18 Terminal Count (TC)

This output tells which peripheral is selected if TC STOP Bit is high in the mode set register, the selected channels get disabled at the end of DMA cycle automatically.

5. Mode Set Register

By setting the mode set register, the four channels of DMA can be enabled or disabled. CPU program this

register once the DMA address register and terminal count register is accessed, they get initialized by the CPU. The mode set register gets cleared by giving the reset input and disables the all other options, including the channels and prevents the on junction of bus after power up. Four channels cannot be left enabled unless CPU gives a valid address register value and terminal count register value. The options that are enabled by varying mode section register are described below using figures.

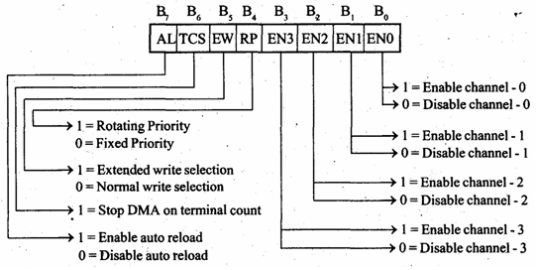

Fig.3 Diagram showing mode set register

5.1 Rotating Priority (BIT 4 of MODE REGISTER):

When the rotating priority bit is high, the priority of channels follows the circular sequence.

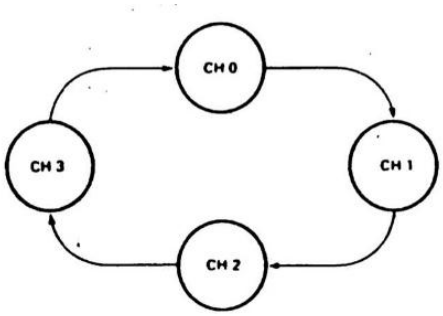

Fig.4 Diagram showing rotating bit priority

Once the DMA cycle is completed, the priority of every channel changes. It follows the following order. When the rotating priority bit is set low. Every DMA channel has a pre fixed priority. When the fixed priority is set then channel 0 will have the maximum priority but when the rotating priority is set then it does not follow the fixed pattern and alter the priority of channel after the terminal count of that channel reaches. The next channel, then reaches the highest priority and the channel that has just completed the transfer become the least priority.

5.2 Extended Write BIT 5

When the extended write bit is set high, the duration of Memory Write and I/O signal gets extended by activating these signals before the DMA cycle, If device cannot be accessed within the time, it return the signal " Not Ready" indicating to the DMAC. Although devices are fast enough to be accessed within the cycle without the use of wait state.

5.3 TC Bit 6(Terminal Count Stop Bit)

When the Terminal count stop bit is high, the channel gets disabled once the terminal count goes true and presents the further operation on that channel indicating that the data flow has been completed and the chance to the next channel should be given..The enable bit for that particular channel has to be set high in order to enable it again and proceed with the data flow.[Shi Bing, Ding Zhi-gang, Zhang Wei-hong(2009)]

5.4 Auto Load Bit 7

Auto load allows the channel 2 for repeated block operation without giving any software intervention. The channel 2 register get initialized as usual for first block. After the first block of the DMA cycle gets executed by channel 2 when TC output goes true. The parameters stored in channel 3 register will be transferred to the channel number 2 when the auto load bit is high the parameters stored in channel number 2 are automatically repeated in channel 3 when channel 2 is programmed.

6. Status Register

It is a 8 bit status register whose channels have reached TC and conclude the update flag.TC status bit gets high when the terminal count is activated for that particular channel. These Bits are high unless the status register is reset again and the update flag is however not affected by read operation of status register.

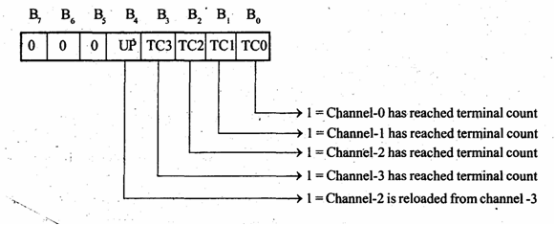

Fig.5 Diagram showing Status register

7. Operational Summary

DMA controller has 4 pairs of channel register. Every pair is provided with a 16 bit terminal count register and a 16 bit address register. The DMAC has also been provided with a 8 bit mode register. The DMAC has also been provided with a 8 bit mode register and a 8-bit status register. An I/O write input tells that the addressed register has to be programmed whereas an I/O read input tells that addressed register has to be read. The least significant address bits i.e A0-A2 tells specific registers to be accessed. when mode set or status register is accessed, then A0-A2 are all zero.[CHEN Shuang-yan, WANG Dong-hui, HOU Chaohuan(2007)].

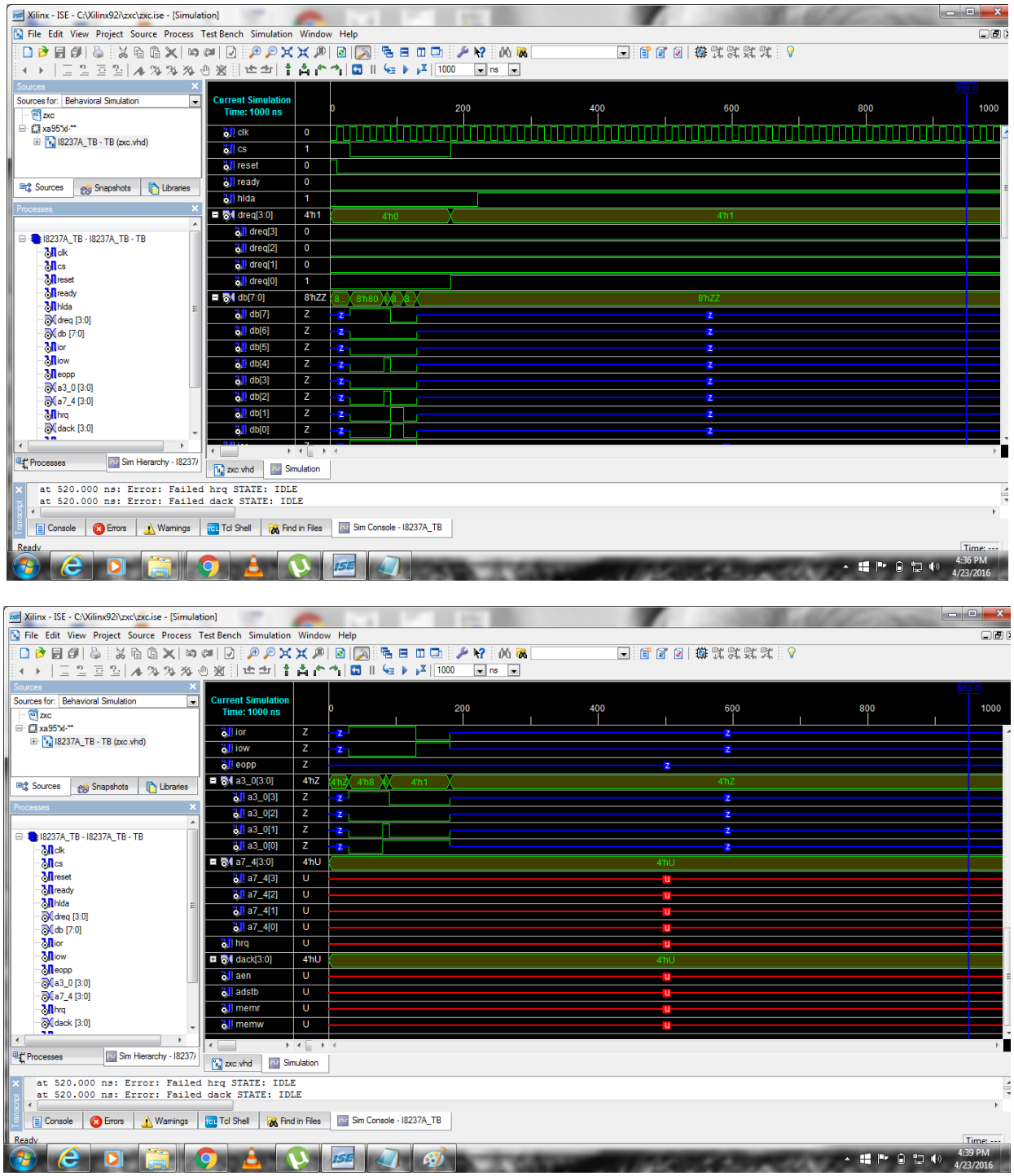

Fig.6 Simulation results of data transfer from memory to peripheral using DMA

8. Simulation Result Explanation

In the above simulation result diagrams, simulation for 200 nano second is shown. A clock of 50 Mega Hertz is used. Control signal, Reset signal and Ready signal is enabled. Since these signals are active low, we are enabling the channel 0 and auto priority is maintained through the program. A eight bit data is send through memory to peripheral through channel 0 by enabling the Read signal [Circuit design with VHDL by Volnei A. Pedroni].

Conclusion

Thus, it can be concluded that any kind of DMA can be made by using VHDL code. The above simulation results are just an example. The VHDL code is quite adaptable for making any kind of DMA and can be synchronize with the CPU. It provides flexibility for the choice of peripheral device also. In future application, the design will be implemented on FPGA. We have checked the proposed finite state machine model in this paper.

References

Circuit design with VHDL by Volnei A. Pedroni. VHDL Programming By Example doughlas perry.

Abdullah Aljumah, Mohammed Altaf Ahmed (2015), Design of High speed data transfer Direct Memory Access Controller for system on chip based Embedded products, *Ansinet journal of applied sciences*, volume 15(3), pp 576-581, 2015.

Vibhu Chinmay , Shubham Sachdeva (2014), A Review Paper on Design of DMA Controller Using VHDL, *IJIRT,* volume 1, issue6,ISSN : 2349-6002

Abdulraqeb Alnabihi & Prof Liu Yijun (2015), The Implementation of DMA Controller on Navigation baseband SoC, *Global Journal of Computer Science and Technology,* Volume 15, Issue 2, ISSN: 0975-4172.

A.J.C. van Gemund, B.R. Sodoyer (2006), Memory controller for a 6502 CPU in VHDL, *Final report for in3019p 'Bachelor project' TU Delft, faculty of EEMCS.*

Zhao Qiang(2014), The Design of DMA Controller based on AHB Bus Specification, Master Dissertation, Xidian University. 2014.

Shi Bing, Ding Zhi-gang, Zhang Wei-hong(2009), NAND Flash DMA Application Based on PXA3xx Processor, *Journal of Computer Applications*, Vol 29(8): 2136-2138.

Chen Shuang-yan, WANG Dong-hui, HOU Chaohuan(2007), Design and Implementation of a Configurable Multi-Channel DMA Controller Based on System, *Microelectronics & Computer,* Vol 24 (5): pp 48-51.

Rashmi misra, Rupal chauhan, Garima arora (2013), Design of DMA controller using VHDL, *International Journal of Engineering, Applied and Management Sciences Paradigms*, Vol. 02, Issue 01.