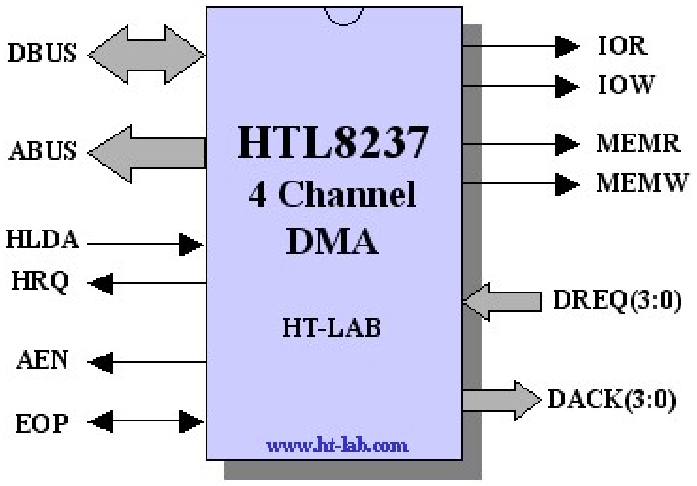
HTL8237

82C37 Compatible IP core



Version 1.3

1. Introduction

The **HTL8237** is a synchronous implementation of the industry standard 8237A Direct Memory Access Controller. The DMA controller is fully pin and software compatible with the 8237A and support four independently programmable channels. Each channel is capable of performing memory-to-memory, memory-to/from-peripheral and memory initialisation tasks. The DMA controller is written in vendor neutral VHDL and can be used for ASIC/FPGA implementations.

The **HTL8237** supports four DMA channels each with an address range of 64K bytes/words. An address can be auto incremented, decremented or put on hold to support initialisation tasks. A channel can be programmed to automatically re-initialise after each transfer. The DMA controller support 4 modes of operations, single, block, demand and cascade.
Single mode allows one transfer before returning the bus to the processor; this mode ensures that the processor can always access the bus between DMA transfers.
Block mode transfers a programmed number of words without returning the bus. During the transfer the bus request signal is kept asserted until all words have been transferred or the End-Of-Process(EOP) signal is asserted.
In demand mode the peripheral continues to transfer data until exhausted or when the preprogrammed number of transfers has been reached or when EOP signal is asserted.
The final mode is the cascade mode which is used to extend the number of channels by cascading additional 8237 controller(s).

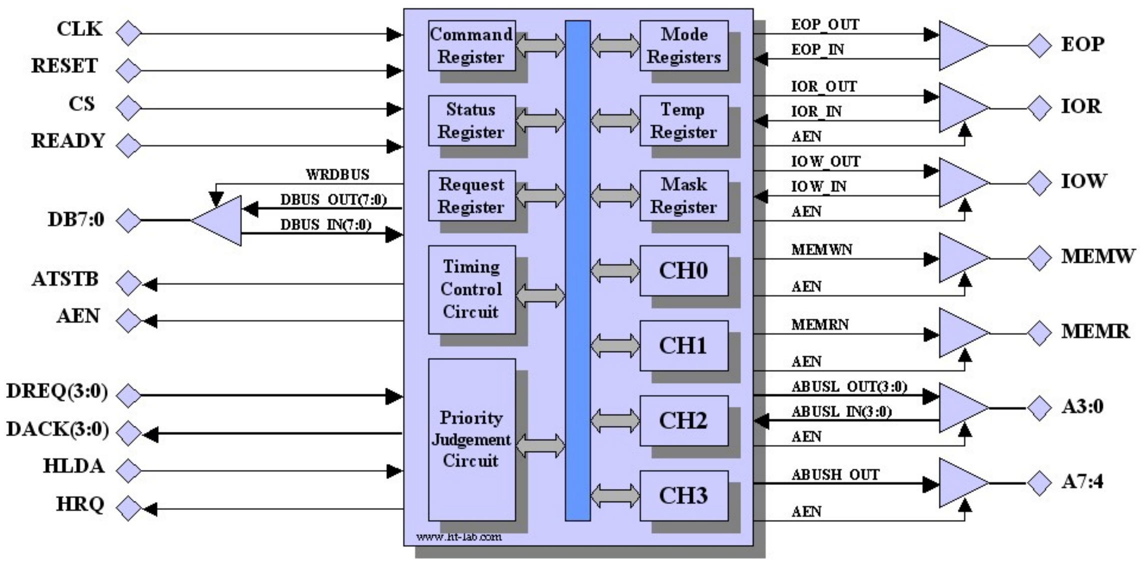


Figure 1: Internal structure of the HTL8237

2. Directory Structure

Directory	Contents
HTL8237\doc	Documentation and Datasheet
HTL8237\rtl	Synthesizable VHDL Core files
HTL8237\testbench	Testbench VHDL files
HTL8237\Simulation	Example script for Modelsim
HTL8237\Synthesis	Synthesis file list

3. Filelist

The HTL8237 synthesizable design is contained in 6 VHDL files, the dependency order is listed in the file fileList.txt. The following files are listed in the RTL directory.

Filename(s)	Contains
HTL8237_pkg.vhd	Definitions
dreqack_rtl.vhd	DMA Request and Acknowledge logic
fsm37_fsm.vhd	Finite State machine controlling the various blocks
channel_rtl.vhd	Logic/counters for each DMA channel (4 times instantiated)
blk37_struct.vhd	Embedded Top Level, no tri-states
HTL8237_struct.vhd	Top Level

The basic testbench for the HTL8237 consist of 5 files:

Filename(s)	Contains
sram_behavior	sram memory model Note1
utils.vhd	Various utility routines
ioblk_behavioral	I/O device simulation model
stimulus_behavioral	Stimulus generator
HTL8237_tb_struct.vhd	HTL8237 top level Testbench

Note1: Free download from http://tams-www.informatik.uni-hamburg.de/vhdl/models/sram/sram.html

4. Simulation

The HTL8237 is written in synthesizable VHDL and as such can be simulated by most if not all simulation tools.

An example simulation script is provided for Mentor Graphics' Modelsim. To run the simulation navigate to the Simulation directory and execute the **run.bat** file from within a DOSbox/CMD shell. Alternative, execute the **run.do** file from within the Modelsim GUI.

The output in both cases should be similar to the text shown below (only part of the output shown).

```
#  ======================= Init Master 8239 ====================================
#  ======================= Single Transfer Read Test ===========================
#  ----------- HTL8237 status before memory to IO transfer --------------------
#  CH0:A=0000 C=0000  CH1:A=0000 C=0000  CH2:A=0000 C=0004  CH3:A=0000 C=0000
#  MD0:D,A++,  ,VF    MD1:D,A++,  ,VF    MD2:S,A++,  ,RD    MD3:D,A++,  ,VF
#  Status=00 Mask=B Request=0 Command=00 Temp=00
#  Start DMA Test 0 Addr++, 5 bytes, address 10000-10004
#  DACK2 Asserted
#    IO DMA Write channel 2 H
#    CPU has access to the bus
#  DACK2 Asserted
#    IO DMA Write channel 2 e
#    CPU has access to the bus
#  DACK2 Asserted
#    IO DMA Write channel 2 l
#    CPU has access to the bus
#  DACK2 Asserted
#    IO DMA Write channel 2 l
#    CPU has access to the bus
#  DACK2 Asserted
```

The expected output is shown in the file expected_output.txt located in the Simulation directory.

4.1 HTL8237

The HTL8237 core consist of 3 main blocks, the channel block (instance ICH0,1,2,3) contains the address and count registers, DREQ/DACK logic (instance DRQA) and a finite state machine (instance

FSM). Surrounding the blocks are a few registers and multiplexers. The operation of the HTL8237 is best described with the FSM as the starting point as this blocks control all the various signals.
A DMA cycle starts when dma_req is asserted, the FSM jumps from Sidle to S1. If memory_to_memory mode is selected (signal mem_2_mem='1') the FSM will continue to jump to state S11 and performs 2 back to back memory cycles, for all other modes (Block/Single/Demand) the FSM jumps to state S1 and continues until S4. Bit 6 and 7 of the mode register can be found in signal dma_mode[1:0].
After each completed single transfer the FSM will assert the endbuscycle signal during state S4 (Block/Single/Demand) and state S24 (memory to memory). The endbuscycle signal is used to update the various address and count registers in the channel blocks. The endbuscycle is multiplexed onto a 4 bits vector (endbuscycle_s) depending on the current active channel (dma_chan). For a memory to memory transfer the endbuscycle signal controls both channel0 and channel1.
At the end of the block transfer when the channel word counter rolls over from 0000 to 0xFFFF the corresponding terminal count signal (tc0,1,2,3) is asserted. This signal is used to assert the EOP_out signal. The tc0,1,2,3 signals are also used to reset the various counters to their initial value if the autoinitialization bit is set in the channel's mode register. When the external EOP signal is asserted (active low) the FSM will sample the signal during S2/S4 for Block/Single/Demand transfers and S22/S24 for memory_to_memory transfers. Only during state S4 and S24 is the external EOP acted upon (FSM output signal eop_latched_cld). When both eop_latched_cld and endbuscycle are asserted the request register (request_reg_s located in the instance DRQA) is cleared.
Each channel instance has a borrowcarry signal which is asserted when the address register rolls over from 0 to 0xFF and from 0xFF to 0 (depending on the mode register Address increment/decrement select). This is then used in the FSM to jump from state S4/S24 to state S1/S11 which results in an ADSTB pulse.
If the mode register is programmed for Cascade mode (dma_mode="11") then the FSM jumps from the idle state Sidle to state S0 when a dma request is detected (dma_req='1'). The FSM then continues to state Scas when the CPU relinquishes the bus (HLDA is asserted) and stays in this state until HLDA is negated or the dma request is removed. During the Scas state no read/write signals or address strobe is generated, the controller will only act as a priority decoder for the various channels.
Further description of the various registers and modes can be found in the 82C37 datasheet.

4.2 Testbench

The testbench (testbench\HTL8237_tb_struct.vhd) instantiates two HTL8237 (Master/Slave), an I/O device simulation model (IO0), a stimulus generator (STIM) and two 128Kbyte SRAM memory models (MEM0/MEM1). The stimulus generator will generate a number of DMA requests scenarios and will output the HTL8237 status before and after each transfer.

```
  ----------- HTL8237 status before Memory to Memory transfer -----------------
#  CH0:A=0007 C=0004  CH1:A=001E C=0004  CH2:A=000B C=FFFF  CH3:A=000A C=0004
#  MD0:B,A--,AI,RD    MD1:B,A--,AI,WR    MD2:B,A++,  ,RD    MD3:S,A++,  ,RD
#  Status=00 Mask=E Request=0 Command=01 Temp=6C MEM2MEM
```

CHx:A=0007 C=0004   DMA Channel Address and Counter value

MDx*:<type>,<Addr>,<init>,<rw>* DMA Channel Mode

<type>	D=Demand mode, B=Block mode, S=Single Mode, C=Cascade Mode
<addr>	A--=Address Decrement, A++=Address Increment
<init>	AI=Autoinitialization Enable, blank=disabled
<rw>	RD=Memory Read IO Write, WR=Memory Write IO Read, VF=Verify
Status	Status Register
Mask	Mask Register
Request	Request Register
Command	Command Register
Temp	Memory to Memory Temporary Register

The last section decodes the Command Register, possible values are MEM2MEM, CH0Hold, Compressed Timing, Rotate Priority, Extended Write.

Some helper procedures are available to simulate an x86 IN (inport) and OUT (outport) port instruction. Example:

```
inport(REQUEST_REGISTER_C,data_s);
outport(BASE_COUNT2_C,X"04");
```

The Stimulus generator (instance STIM) works in combination with the IO device simulation model (Instance IO0). The testmode signal indicates which test is in progress. Each test is further subdivided into smaller test (minor changes) and is indicated by the loop variable “n”.

```
--===========================================================================
-- TestX
--
--  Test  Action
--    0  Addr++, 5 bytes, address 10000-10004
--    1  Addr++, 5 bytes, address 10000-10004, AutoInit
--    2  Addr--, 5 bytes, address 10004-10000, Extended write
--===========================================================================
for n in 0 to 2 loop
     various tests
end loop;
```

For some of the test the waveform is shown to highlight some timing issues. The waveforms are from Mentor Graphics Modelsim.

The stimulus generator starts by initializing the master or 1st level DMA controller in cascade mode. This means that the controller is only used to sort out priority, it does not output any R/W strobes or EOP signals.

```
#  ======================= Init Master 8239 ====================================
```

4.2.1 Single Transfer Read Test

testmode=”00001”

The Single Transfer mode transfers a single byte (or word) between memory and an I/O device. The 8237 DMA controller will release and re-acquire the CPU bus between each transfer. This mode is commonly-used by devices that cannot transfer the entire block of data in one go. The peripheral will request the DMA each time it is ready for another transfer. If the peripheral continuously assert the DREQ line (as used in the tester) then the 8237 will still give the bus back to the CPU between transfers.

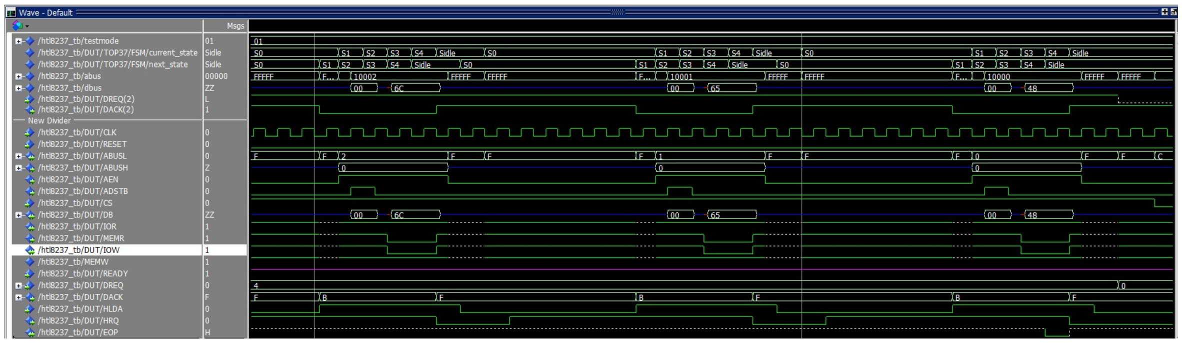
Note in the testbench output below the CPU regains access to the bus between each transfer.

```
#  ======================= Single Transfer Read Test ===========================
#  ----------- HTL8237 status before memory to IO transfer --------------------
#  CH0:A=0000 C=0000  CH1:A=0000 C=0000  CH2:A=0000 C=0004  CH3:A=0000 C=0000
#  MD0:D,A++,  ,VF    MD1:D,A++,  ,VF    MD2:S,A++,  ,RD    MD3:D,A++,  ,VF
#  Status=00 Mask=B Request=0 Command=00 Temp=00
#  Start DMA Test 0 Addr++, 5 bytes, address 10000-10004
#  DACK2 Asserted
#    IO DMA Write channel 2 H
#    CPU has access to the bus
#  DACK2 Asserted
#    IO DMA Write channel 2 e
#    CPU has access to the bus
#  DACK2 Asserted
#    IO DMA Write channel 2 l
#    CPU has access to the bus
#  DACK2 Asserted
#    IO DMA Write channel 2 l
#    CPU has access to the bus
#  DACK2 Asserted
#  EOP Asserted Channel 2
#    IO DMA Write channel 2 o
#    CPU has access to the bus
#  ----------- 82C37 status after memory to IO transfer ----------------------
#  CH0:A=0000 C=0000  CH1:A=0000 C=0000  CH2:A=0005 C=FFFF  CH3:A=0000 C=0000
#  MD0:D,A++,  ,VF    MD1:D,A++,  ,VF    MD2:S,A++,  ,RD    MD3:D,A++,  ,VF
#  Status=04 Mask=F Request=0 Command=00 Temp=00
#
#  ----------- HTL8237 status before memory to IO transfer --------------------
#  CH0:A=0000 C=0000  CH1:A=0000 C=0000  CH2:A=0000 C=0004  CH3:A=0000 C=0000
#  MD0:D,A++,  ,VF    MD1:D,A++,  ,VF    MD2:S,A++,AI,RD    MD3:D,A++,  ,VF
#  Status=00 Mask=B Request=0 Command=00 Temp=00
#  Start DMA Test 1 Addr++, 5 bytes, address 10000-10004, AutoInit
#  DACK2 Asserted
#    IO DMA Write channel 2 H
#    CPU has access to the bus
#  DACK2 Asserted
#    IO DMA Write channel 2 e
#    CPU has access to the bus
#  DACK2 Asserted
#    IO DMA Write channel 2 l
#    CPU has access to the bus
#  DACK2 Asserted
#    IO DMA Write channel 2 l
#    CPU has access to the bus
#  DACK2 Asserted
#  EOP Asserted Channel 2
#    IO DMA Write channel 2 o
#    CPU has access to the bus
#  ----------- 82C37 status after memory to IO transfer ----------------------
#  CH0:A=0000 C=0000  CH1:A=0000 C=0000  CH2:A=0000 C=0004  CH3:A=0000 C=0000
#  MD0:D,A++,  ,VF    MD1:D,A++,  ,VF    MD2:S,A++,AI,RD    MD3:D,A++,  ,VF
#  Status=04 Mask=B Request=0 Command=00 Temp=00
```

For autoinit the Channel2 address and count values are reset to the original value (0000,0004).

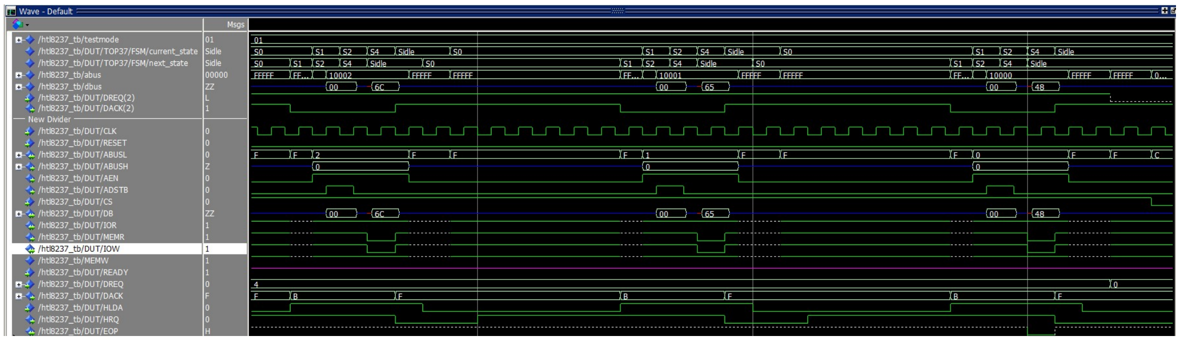
```
#
#  ----------- HTL8237 status before memory to IO transfer --------------------
#  CH0:A=0000 C=0000  CH1:A=0000 C=0000  CH2:A=0004 C=0004  CH3:A=0000 C=0000
#  MD0:D,A++,  ,VF    MD1:D,A++,  ,VF    MD2:S,A--,  ,RD    MD3:D,A++,  ,VF
#  Status=00 Mask=B Request=0 Command=20 Temp=00 Extended Write
#  Start DMA Test 2 Addr--, 5 bytes, address 10004-10000, Extended write
#  DACK2 Asserted
#    IO DMA Write channel 2 o
#    IO DMA Write channel 2 o
#    CPU has access to the bus
#  DACK2 Asserted
#    IO DMA Write channel 2 l
#    IO DMA Write channel 2 l
#    CPU has access to the bus
#  DACK2 Asserted
#    IO DMA Write channel 2 l
#    IO DMA Write channel 2 l
#    CPU has access to the bus
#  DACK2 Asserted
#    IO DMA Write channel 2 e
#    IO DMA Write channel 2 e
#    CPU has access to the bus
#  DACK2 Asserted
#    IO DMA Write channel 2 H
#  EOP Asserted Channel 2
#    IO DMA Write channel 2 H
#    CPU has access to the bus
#  ----------- 82C37 status after memory to IO transfer ----------------------
#  CH0:A=0000 C=0000  CH1:A=0000 C=0000  CH2:A=FFFF C=FFFF  CH3:A=0000 C=0000
#  MD0:D,A++,  ,VF    MD1:D,A++,  ,VF    MD2:S,A--,  ,RD    MD3:D,A++,  ,VF
#  Status=04 Mask=F Request=0 Command=20 Temp=00 Extended Write
```

For extended write the IOW and MEMW signals have the same durations as the IOR and MEMR signals, that is 2 clock cycles. The signals can be further extended using the READY input.



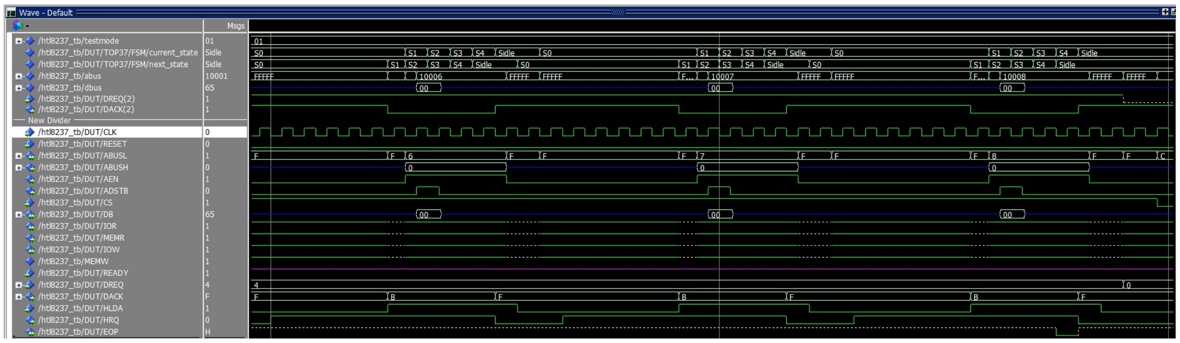
```
#  ----------- HTL8237 status before memory to IO transfer --------------------
#  CH0:A=0000 C=0000  CH1:A=0000 C=0000  CH2:A=0004 C=0004  CH3:A=0000 C=0000
#  MD0:D,A++,  ,VF    MD1:D,A++,  ,VF    MD2:S,A--,  ,RD    MD3:D,A++,  ,VF
#  Status=00 Mask=B Request=0 Command=08 Temp=00 Compressed Timing
#  Start DMA Test 3 Addr--, 5 bytes, address 10004-10000, Compressed timing, Autoinit
#  DACK2 Asserted
#    IO DMA Write channel 2 o
#    CPU has access to the bus
#  DACK2 Asserted
#    IO DMA Write channel 2 l
#    CPU has access to the bus
#  DACK2 Asserted
#    IO DMA Write channel 2 l
#    CPU has access to the bus
#  DACK2 Asserted
#    IO DMA Write channel 2 e
#    CPU has access to the bus
#  DACK2 Asserted
#  EOP Asserted Channel 2
#    IO DMA Write channel 2 H
#    CPU has access to the bus
#  ----------- 82C37 status after memory to IO transfer ----------------------
#  CH0:A=0000 C=0000  CH1:A=0000 C=0000  CH2:A=FFFF C=FFFF  CH3:A=0000 C=0000
#  MD0:D,A++,  ,VF    MD1:D,A++,  ,VF    MD2:S,A--,  ,RD    MD3:D,A++,  ,VF
#  Status=04 Mask=F Request=0 Command=08 Temp=00 Compressed Timing
```

For compressed timing the IOR and MEMR signals are reduced to 1 clock cycle, the whole DMA transfer takes 3 clock cycles (FSM state S3 is skipped).



```
#  ----------- HTL8237 status before memory to IO transfer --------------------
#  CH0:A=0000 C=0000  CH1:A=0000 C=0000  CH2:A=0004 C=0004  CH3:A=0000 C=0000
#  MD0:D,A++,  ,VF    MD1:D,A++,  ,VF    MD2:S,A++,  ,VF    MD3:D,A++,  ,VF
#  Status=00 Mask=B Request=0 Command=00 Temp=00
#  Start DMA Test 4 Addr++, 5 bytes, address 10000-10004, Verify transfer
#  DACK2 Asserted
#    CPU has access to the bus
#  DACK2 Asserted
#    CPU has access to the bus
#  DACK2 Asserted
#    CPU has access to the bus
#  DACK2 Asserted
#    CPU has access to the bus
#  DACK2 Asserted
#  EOP Asserted Channel 2
#    CPU has access to the bus
#  ----------- 82C37 status after memory to IO transfer ----------------------
#  CH0:A=0000 C=0000  CH1:A=0000 C=0000  CH2:A=0009 C=FFFF  CH3:A=0000 C=0000
#  MD0:D,A++,  ,VF    MD1:D,A++,  ,VF    MD2:S,A++,  ,VF    MD3:D,A++,  ,VF
#  Status=04 Mask=F Request=0 Command=00 Temp=00
```

During a verify transfer mode no read/write signals are asserted.



4.2.2 Single Transfer Write Test

testmode="00101"

This test is similar to the previous read test with the exception that we are now reading data from an IO device and writing it to memory.

```
#  ======================== Single Transfer Write Test ==========================
# 80100 : xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx
# 80110 : xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx
# 80120 : xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx
#  ----------- HTL8237 status before memory to IO transfer --------------------
#  CH0:A=0000 C=0000  CH1:A=0000 C=0000  CH2:A=0100 C=0004  CH3:A=0000 C=0000
#  MD0:D,A++,  ,VF    MD1:D,A++,  ,VF    MD2:S,A++,  ,WR    MD3:D,A++,  ,VF
#  Status=00 Mask=B Request=0 Command=00 Temp=00
#  Start DMA Test 0 Addr++, 5 bytes, address 80100-80104
#  DACK2 Asserted
#    IO DMA Read channel 2
#    IO DMA Read channel 2
#    CPU has access to the bus
#  DACK2 Asserted
#    IO DMA Read channel 2
#    IO DMA Read channel 2
#    CPU has access to the bus
#  DACK2 Asserted
#    IO DMA Read channel 2
#    IO DMA Read channel 2
#    CPU has access to the bus
#  DACK2 Asserted
#    IO DMA Read channel 2
#    IO DMA Read channel 2
#    CPU has access to the bus
#  DACK2 Asserted
#    IO DMA Read channel 2
#    IO DMA Read channel 2
#  EOP Asserted Channel 2
#    CPU has access to the bus
#  ----------- 82C37 status after memory to IO transfer ----------------------
#  CH0:A=0000 C=0000  CH1:A=0000 C=0000  CH2:A=0105 C=FFFF  CH3:A=0000 C=0000
#  MD0:D,A++,  ,VF    MD1:D,A++,  ,VF    MD2:S,A++,  ,WR    MD3:D,A++,  ,VF
#  Status=04 Mask=F Request=0 Command=00 Temp=00
#
# 80100 : 38 32 43 33 39 xx xx xx xx xx xx xx xx xx xx xx
# 80110 : xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx
# 80120 : xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx
#  ----------- HTL8237 status before memory to IO transfer --------------------
#  CH0:A=0000 C=0000  CH1:A=0000 C=0000  CH2:A=0100 C=0004  CH3:A=0000 C=0000
#  MD0:D,A++,  ,VF    MD1:D,A++,  ,VF    MD2:S,A++,AI,WR    MD3:D,A++,  ,VF
#  Status=00 Mask=B Request=0 Command=00 Temp=00
#  Start DMA Test 1 Addr++, 5 bytes, address 80100-80104, AutoInit
#  DACK2 Asserted
#    IO DMA Read channel 2
#    IO DMA Read channel 2
#    CPU has access to the bus
#  DACK2 Asserted
#    IO DMA Read channel 2
#    IO DMA Read channel 2
#    CPU has access to the bus
#  DACK2 Asserted
#    IO DMA Read channel 2
#    IO DMA Read channel 2
#    CPU has access to the bus
#  DACK2 Asserted
#    IO DMA Read channel 2
#    IO DMA Read channel 2
#    CPU has access to the bus
#  DACK2 Asserted
#    IO DMA Read channel 2
#    IO DMA Read channel 2
#  EOP Asserted Channel 2
#    CPU has access to the bus
#  ----------- 82C37 status after memory to IO transfer ----------------------
#  CH0:A=0000 C=0000  CH1:A=0000 C=0000  CH2:A=0100 C=0004  CH3:A=0000 C=0000
#  MD0:D,A++,  ,VF    MD1:D,A++,  ,VF    MD2:S,A++,AI,WR    MD3:D,A++,  ,VF
#  Status=04 Mask=B Request=0 Command=00 Temp=00
#
# 80100 : 38 32 43 33 39 xx xx xx xx xx xx xx xx xx xx xx
# 80110 : xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx
# 80120 : xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx
#  ----------- HTL8237 status before memory to IO transfer --------------------
#  CH0:A=0000 C=0000  CH1:A=0000 C=0000  CH2:A=0110 C=0004  CH3:A=0000 C=0000
#  MD0:D,A++,  ,VF    MD1:D,A++,  ,VF    MD2:S,A--,  ,WR    MD3:D,A++,  ,VF
#  Status=00 Mask=B Request=0 Command=20 Temp=00 Extended Write
#  Start DMA Test 2 Addr--, 5 bytes, address 80110-80110, Extended write
#  DACK2 Asserted
#    IO DMA Read channel 2
#    IO DMA Read channel 2
#    CPU has access to the bus
#  DACK2 Asserted
#    IO DMA Read channel 2
#    IO DMA Read channel 2
#    CPU has access to the bus
#  DACK2 Asserted
#    IO DMA Read channel 2
#    IO DMA Read channel 2
#    CPU has access to the bus
#  DACK2 Asserted
#    IO DMA Read channel 2
#    IO DMA Read channel 2
#    CPU has access to the bus
#  DACK2 Asserted
#    IO DMA Read channel 2
#    IO DMA Read channel 2
#  EOP Asserted Channel 2
#    CPU has access to the bus
#  ----------- 82C37 status after memory to IO transfer ----------------------
#  CH0:A=0000 C=0000  CH1:A=0000 C=0000  CH2:A=010B C=FFFF  CH3:A=0000 C=0000
#  MD0:D,A++,  ,VF    MD1:D,A++,  ,VF    MD2:S,A--,  ,WR    MD3:D,A++,  ,VF
#  Status=04 Mask=F Request=0 Command=20 Temp=00 Extended Write
#
# 80100 : 38 32 43 33 39 xx xx xx xx xx xx xx 39 33 43 32
# 80110 : 38 xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx
# 80120 : xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx
#  ----------- HTL8237 status before memory to IO transfer --------------------
#  CH0:A=0000 C=0000  CH1:A=0000 C=0000  CH2:A=0120 C=0004  CH3:A=0000 C=0000
#  MD0:D,A++,  ,VF    MD1:D,A++,  ,VF    MD2:S,A--,  ,WR    MD3:D,A++,  ,VF
#  Status=00 Mask=B Request=0 Command=08 Temp=00 Compressed Timing
#  Start DMA Test 3 Addr--, 5 bytes, address 80120-80120, Compressed timing, Autoinit
#  DACK2 Asserted
#    IO DMA Read channel 2
#    CPU has access to the bus
#  DACK2 Asserted
#    IO DMA Read channel 2
#    CPU has access to the bus
#  DACK2 Asserted
#    IO DMA Read channel 2
#    CPU has access to the bus
#  DACK2 Asserted
#    IO DMA Read channel 2
#    CPU has access to the bus
#  DACK2 Asserted
#    IO DMA Read channel 2
#  EOP Asserted Channel 2
#    CPU has access to the bus
#  ----------- 82C37 status after memory to IO transfer ----------------------
#  CH0:A=0000 C=0000  CH1:A=0000 C=0000  CH2:A=011B C=FFFF  CH3:A=0000 C=0000
#  MD0:D,A++,  ,VF    MD1:D,A++,  ,VF    MD2:S,A--,  ,WR    MD3:D,A++,  ,VF
#  Status=04 Mask=F Request=0 Command=08 Temp=00 Compressed Timing
#
# 80100 : 38 32 43 33 39 xx xx xx xx xx xx xx 39 33 43 32
# 80110 : 38 xx xx xx xx xx xx xx xx xx xx xx 39 33 43 32
# 80120 : 38 xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx
```

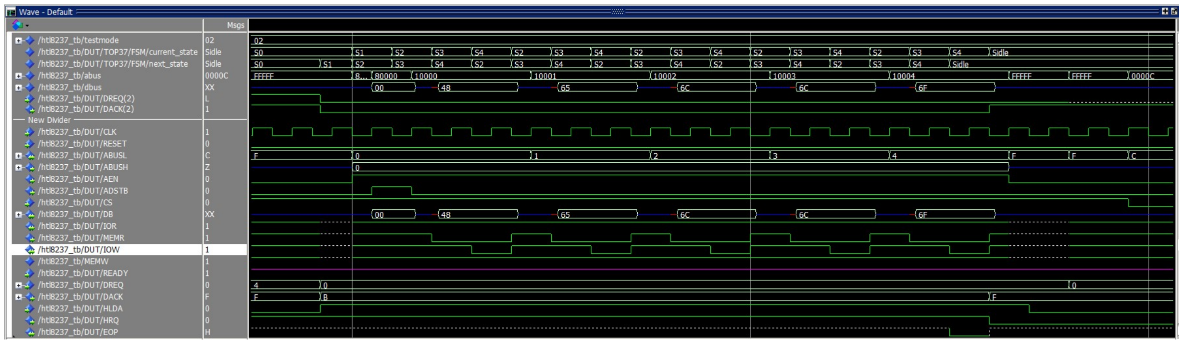
4.2.3 Block Transfer Read Test

testmode="00010"

In Block Transfer mode, the device is activated by DREQ or software request and continues making transfers until a TC, caused by word count going to 0xFFFF, or an external End of Process (EOP) is encountered. DREQ needs only be held active until DACK becomes active. An autoinitialization will occur at the end of the service if the channel has been programmed for that option.

```
#  ======================= Block Read Transfer Test ============================
#  ----------- HTL8237 status before memory to IO transfer --------------------
#  CH0:A=0000 C=0000  CH1:A=0000 C=0000  CH2:A=0000 C=0004  CH3:A=0000 C=0000
#  MD0:D,A++,  ,VF    MD1:D,A++,  ,VF    MD2:B,A++,  ,RD    MD3:D,A++,  ,VF
#  Status=00 Mask=B Request=0 Command=00 Temp=00
#  Start DMA Test 0 Addr++, 5 bytes, address 10000-10004
#  DACK2 Asserted
#    IO DMA Write channel 2 H
#    IO DMA Write channel 2 e
#    IO DMA Write channel 2 l
#    IO DMA Write channel 2 l
#    IO DMA Write channel 2 o
#  EOP Asserted Channel 2
#    CPU has access to the bus
#  ----------- 82C37 status after memory to IO transfer ----------------------
#  CH0:A=0000 C=0000  CH1:A=0000 C=0000  CH2:A=0005 C=FFFF  CH3:A=0000 C=0000
#  MD0:D,A++,  ,VF    MD1:D,A++,  ,VF    MD2:B,A++,  ,RD    MD3:D,A++,  ,VF
#  Status=04 Mask=F Request=0 Command=00 Temp=00
```

Note unlike the Single Transfer Mode the 8237 does not relinquish the bus back to the CPU between transfers. Thus the I/O device holds the bus until all data has been transferred.



```
#  ----------- HTL8237 status before memory to IO transfer --------------------
#  CH0:A=0000 C=0000  CH1:A=0000 C=0000  CH2:A=0000 C=0004  CH3:A=0000 C=0000
#  MD0:D,A++,  ,VF    MD1:D,A++,  ,VF    MD2:B,A++,AI,RD    MD3:D,A++,  ,VF
#  Status=00 Mask=B Request=0 Command=00 Temp=00
#  Start DMA Test 1 Addr++, 5 bytes, address 10000-10004, AutoInit
#  DACK2 Asserted
#    IO DMA Write channel 2 H
#    IO DMA Write channel 2 e
#    IO DMA Write channel 2 l
#    IO DMA Write channel 2 l
#    IO DMA Write channel 2 o
#  EOP Asserted Channel 2
#    CPU has access to the bus
#  ----------- 82C37 status after memory to IO transfer ----------------------
#  CH0:A=0000 C=0000  CH1:A=0000 C=0000  CH2:A=0000 C=0004  CH3:A=0000 C=0000
#  MD0:D,A++,  ,VF    MD1:D,A++,  ,VF    MD2:B,A++,AI,RD    MD3:D,A++,  ,VF
#  Status=04 Mask=B Request=0 Command=00 Temp=00
#
#  ----------- HTL8237 status before memory to IO transfer --------------------
#  CH0:A=0000 C=0000  CH1:A=0000 C=0000  CH2:A=0004 C=0004  CH3:A=0000 C=0000
#  MD0:D,A++,  ,VF    MD1:D,A++,  ,VF    MD2:B,A--,  ,RD    MD3:D,A++,  ,VF
#  Status=00 Mask=B Request=0 Command=20 Temp=00 Extended Write
#  Start DMA Test 2 Addr--, 5 bytes, address 10004-10000, Extended write
#  DACK2 Asserted
#    IO DMA Write channel 2 o
#    IO DMA Write channel 2 o
#    IO DMA Write channel 2 l
#    IO DMA Write channel 2 l
#    IO DMA Write channel 2 l
#    IO DMA Write channel 2 l
#    IO DMA Write channel 2 e
#    IO DMA Write channel 2 e
#    IO DMA Write channel 2 H
#    IO DMA Write channel 2 H
#  EOP Asserted Channel 2
#    CPU has access to the bus
#  ----------- 82C37 status after memory to IO transfer ----------------------
#  CH0:A=0000 C=0000  CH1:A=0000 C=0000  CH2:A=FFFF C=FFFF  CH3:A=0000 C=0000
#  MD0:D,A++,  ,VF    MD1:D,A++,  ,VF    MD2:B,A--,  ,RD    MD3:D,A++,  ,VF
#  Status=04 Mask=F Request=0 Command=20 Temp=00 Extended Write
#
#  ----------- HTL8237 status before memory to IO transfer --------------------
#  CH0:A=0000 C=0000  CH1:A=0000 C=0000  CH2:A=0004 C=0004  CH3:A=0000 C=0000
#  MD0:D,A++,  ,VF    MD1:D,A++,  ,VF    MD2:B,A--,AI,RD    MD3:D,A++,  ,VF
#  Status=00 Mask=B Request=0 Command=08 Temp=00 Compressed Timing
#  Start DMA Test 3 Addr--, 5 bytes, address 10004-10000, Compressed timing, Autoinit
#  DACK2 Asserted
#    IO DMA Write channel 2 o
#    IO DMA Write channel 2 l
#    IO DMA Write channel 2 l
#    IO DMA Write channel 2 e
#    IO DMA Write channel 2 H
#  EOP Asserted Channel 2
#    CPU has access to the bus
#  ----------- 82C37 status after memory to IO transfer ----------------------
#  CH0:A=0000 C=0000  CH1:A=0000 C=0000  CH2:A=0004 C=0004  CH3:A=0000 C=0000
#  MD0:D,A++,  ,VF    MD1:D,A++,  ,VF    MD2:B,A--,AI,RD    MD3:D,A++,  ,VF
#  Status=04 Mask=B Request=0 Command=08 Temp=00 Compressed Timing
```

4.2.4 Demand Transfer Read Test

testmode="00011"

Demand Mode will transfer data until DREQ is de-asserted, TC or an external EOP is detected. When DREQ is re-asserted the transfer resumes where it was suspended.

```
#  ======================= Demand Transfer Test ================================
#  ----------- HTL8237 status before memory to IO transfer --------------------
#  CH0:A=0000 C=0000  CH1:A=0000 C=0000  CH2:A=0000 C=0004  CH3:A=0000 C=0000
#  MD0:D,A++,  ,VF    MD1:D,A++,  ,VF    MD2:D,A++,  ,RD    MD3:D,A++,  ,VF
#  Status=00 Mask=B Request=0 Command=00 Temp=00
#  Start DMA Test 0 Addr++, 5 bytes, address 10000-10004
#  DACK2 Asserted
#    IO DMA Write channel 2 H
#    IO DMA Write channel 2 e
#    IO DMA Write channel 2 l
#    CPU has access to the bus
#  ----------- 82C37 status during memory to IO transfer --------------------
#  CH0:A=0000 C=0000  CH1:A=0000 C=0000  CH2:A=0003 C=0001  CH3:A=0000 C=0000
#  MD0:D,A++,  ,VF    MD1:D,A++,  ,VF    MD2:D,A++,  ,RD    MD3:D,A++,  ,VF
#  Status=40 Mask=B Request=0 Command=00 Temp=00
#  DACK2 Asserted
#    IO DMA Write channel 2 l
#    IO DMA Write channel 2 o
#  EOP Asserted Channel 2
#    CPU has access to the bus
#  ----------- 82C37 status after memory to IO transfer ----------------------
#  CH0:A=0000 C=0000  CH1:A=0000 C=0000  CH2:A=0005 C=FFFF  CH3:A=0000 C=0000
#  MD0:D,A++,  ,VF    MD1:D,A++,  ,VF    MD2:D,A++,  ,RD    MD3:D,A++,  ,VF
#  Status=04 Mask=F Request=0 Command=00 Temp=00
```

After 3 bytes has been transferred the IO device relinquished the bus, after a while new data is available which results in the DMA transfer to continue were it left off.

```
#  ----------- HTL8237 status before memory to IO transfer --------------------
#  CH0:A=0000 C=0000  CH1:A=0000 C=0000  CH2:A=0000 C=0004  CH3:A=0000 C=0000
#  MD0:D,A++,  ,VF    MD1:D,A++,  ,VF    MD2:D,A++,AI,RD    MD3:D,A++,  ,VF
#  Status=00 Mask=B Request=0 Command=00 Temp=00
#  Start DMA Test 1 Addr++, 5 bytes, address 10000-10004, AutoInit
#  DACK2 Asserted
#    IO DMA Write channel 2 H
#    IO DMA Write channel 2 e
#    IO DMA Write channel 2 l
#    CPU has access to the bus
#  ----------- 82C37 status during memory to IO transfer --------------------
#  CH0:A=0000 C=0000  CH1:A=0000 C=0000  CH2:A=0003 C=0001  CH3:A=0000 C=0000
#  MD0:D,A++,  ,VF    MD1:D,A++,  ,VF    MD2:D,A++,AI,RD    MD3:D,A++,  ,VF
#  Status=40 Mask=B Request=0 Command=00 Temp=00
#  DACK2 Asserted
#    IO DMA Write channel 2 l
#    IO DMA Write channel 2 o
#  EOP Asserted Channel 2
#    CPU has access to the bus
#  ----------- 82C37 status after memory to IO transfer ----------------------
#  CH0:A=0000 C=0000  CH1:A=0000 C=0000  CH2:A=0000 C=0004  CH3:A=0000 C=0000
#  MD0:D,A++,  ,VF    MD1:D,A++,  ,VF    MD2:D,A++,AI,RD    MD3:D,A++,  ,VF
#  Status=04 Mask=B Request=0 Command=00 Temp=00
#
#  ----------- HTL8237 status before memory to IO transfer --------------------
#  CH0:A=0000 C=0000  CH1:A=0000 C=0000  CH2:A=0004 C=0004  CH3:A=0000 C=0000
#  MD0:D,A++,  ,VF    MD1:D,A++,  ,VF    MD2:D,A--,  ,RD    MD3:D,A++,  ,VF
#  Status=00 Mask=B Request=0 Command=20 Temp=00 Extended Write
#  Start DMA Test 2 Addr--, 5 bytes, address 10004-10000, Extended write
#  DACK2 Asserted
#    IO DMA Write channel 2 o
#    IO DMA Write channel 2 o
#    IO DMA Write channel 2 l
#    IO DMA Write channel 2 l
#    IO DMA Write channel 2 l
#    IO DMA Write channel 2 l
#    CPU has access to the bus
#  ----------- 82C37 status during memory to IO transfer --------------------
#  CH0:A=0000 C=0000  CH1:A=0000 C=0000  CH2:A=0001 C=0001  CH3:A=0000 C=0000
#  MD0:D,A++,  ,VF    MD1:D,A++,  ,VF    MD2:D,A--,  ,RD    MD3:D,A++,  ,VF
#  Status=40 Mask=B Request=0 Command=20 Temp=00 Extended Write
#  DACK2 Asserted
#    IO DMA Write channel 2 e
#    IO DMA Write channel 2 e
#    IO DMA Write channel 2 H
#    IO DMA Write channel 2 H
#  EOP Asserted Channel 2
#    CPU has access to the bus
#  ----------- 82C37 status after memory to IO transfer ----------------------
#  CH0:A=0000 C=0000  CH1:A=0000 C=0000  CH2:A=FFFF C=FFFF  CH3:A=0000 C=0000
#  MD0:D,A++,  ,VF    MD1:D,A++,  ,VF    MD2:D,A--,  ,RD    MD3:D,A++,  ,VF
#  Status=04 Mask=F Request=0 Command=20 Temp=00 Extended Write
#
#  ----------- HTL8237 status before memory to IO transfer --------------------
#  CH0:A=0000 C=0000  CH1:A=0000 C=0000  CH2:A=0004 C=0004  CH3:A=0000 C=0000
#  MD0:D,A++,  ,VF    MD1:D,A++,  ,VF    MD2:D,A--,AI,RD    MD3:D,A++,  ,VF
#  Status=00 Mask=B Request=0 Command=08 Temp=00 Compressed Timing
#  Start DMA Test 3 Addr--, 5 bytes, address 10004-10000, Compressed timing, Autoinit
#  DACK2 Asserted
#    IO DMA Write channel 2 o
#    IO DMA Write channel 2 l
#    IO DMA Write channel 2 l
#    CPU has access to the bus
#  ----------- 82C37 status during memory to IO transfer --------------------
#  CH0:A=0000 C=0000  CH1:A=0000 C=0000  CH2:A=0001 C=0001  CH3:A=0000 C=0000
#  MD0:D,A++,  ,VF    MD1:D,A++,  ,VF    MD2:D,A--,AI,RD    MD3:D,A++,  ,VF
#  Status=40 Mask=B Request=0 Command=08 Temp=00 Compressed Timing
#  DACK2 Asserted
#    IO DMA Write channel 2 e
#    IO DMA Write channel 2 H
#  EOP Asserted Channel 2
#    CPU has access to the bus
#  ----------- 82C37 status after memory to IO transfer ----------------------
#  CH0:A=0000 C=0000  CH1:A=0000 C=0000  CH2:A=0004 C=0004  CH3:A=0000 C=0000
#  MD0:D,A++,  ,VF    MD1:D,A++,  ,VF    MD2:D,A--,AI,RD    MD3:D,A++,  ,VF
#  Status=04 Mask=B Request=0 Command=08 Temp=00 Compressed Timing
```

4.2.5 Priority Test

testmode="00100"

The 8237 support 2 types of priority, Fixed priority were channel 0 has the highest priority followed by channels 1,2 and 3 (lowest) and Rotate priority which gives the last serviced channel the lowest priority. Rotate priority can be used to prevent any channel from blocking lower priority channels, each channel will get the same slice of bus bandwidth.
In the test below all channels are requesting the bus at the same time.

```
#  ======================= Priority Test ===============================
#  ----------- 82C37 status before Priority test --------------------------
#  CH0:A=0002 C=0004  CH1:A=0004 C=0004  CH2:A=0008 C=0004  CH3:A=000A C=0004
#  MD0:S,A++,  ,RD    MD1:S,A++,  ,RD    MD2:S,A++,  ,RD    MD3:S,A++,  ,RD
#  Status=00 Mask=0 Request=0 Command=00 Temp=00
#  Start Fixed Priority DMA Test 0 Addr++, 5 bytes, address 10000-10004
#  DACK0 Asserted
#  DACK0 Asserted
#  DACK0 Asserted
#  DACK0 Asserted
#  DACK0 Asserted
#  EOP Asserted Channel 0
#  DACK1 Asserted
#  DACK1 Asserted
#  DACK1 Asserted
#  DACK1 Asserted
#  DACK1 Asserted
#  EOP Asserted Channel 1
#  DACK2 Asserted
#  DACK2 Asserted
#  DACK2 Asserted
#  DACK2 Asserted
#  DACK2 Asserted
#  EOP Asserted Channel 2
#  DACK3 Asserted
#  DACK3 Asserted
#  DACK3 Asserted
#  DACK3 Asserted
#  DACK3 Asserted
#  EOP Asserted Channel 3
#  ----------- 82C37 status after Priority test -----------------------------
#  CH0:A=0007 C=FFFF  CH1:A=0009 C=FFFF  CH2:A=000D C=FFFF  CH3:A=000F C=FFFF
#  MD0:S,A++,  ,RD    MD1:S,A++,  ,RD    MD2:S,A++,  ,RD    MD3:S,A++,  ,RD
#  Status=0F Mask=F Request=0 Command=00 Temp=00
```

Channel 0 is serviced first (highest priority) followed by channels 1 to 3.

```
#  ----------- 82C37 status before Priority test --------------------------
#  CH0:A=0002 C=0004  CH1:A=0004 C=0004  CH2:A=0008 C=0004  CH3:A=000A C=0004
#  MD0:S,A++,  ,RD    MD1:S,A++,  ,RD    MD2:S,A++,  ,RD    MD3:S,A++,  ,RD
#  Status=00 Mask=0 Request=0 Command=10 Temp=00 Rotate Priority
#  Start Rotate Priority DMA Test 1 Addr--, 5 bytes, address 10000-10004
#  DACK0 Asserted
#  DACK1 Asserted
#  DACK2 Asserted
#  DACK3 Asserted
#  DACK0 Asserted
#  DACK1 Asserted
#  DACK2 Asserted
#  DACK3 Asserted
#  DACK0 Asserted
#  DACK1 Asserted
#  DACK2 Asserted
#  DACK3 Asserted
#  DACK0 Asserted
#  DACK1 Asserted
#  DACK2 Asserted
#  DACK3 Asserted
#  DACK0 Asserted
#  EOP Asserted Channel 0
#  DACK1 Asserted
#  EOP Asserted Channel 1
#  DACK2 Asserted
#  EOP Asserted Channel 2
#  DACK3 Asserted
#  EOP Asserted Channel 3
#  ----------- 82C37 status after Priority test ----------------------------
#  CH0:A=0007 C=FFFF  CH1:A=0009 C=FFFF  CH2:A=000D C=FFFF  CH3:A=000F C=FFFF
#  MD0:S,A++,  ,RD    MD1:S,A++,  ,RD    MD2:S,A++,  ,RD    MD3:S,A++,  ,RD
#  Status=0F Mask=F Request=0 Command=10 Temp=00 Rotate Priority
```

Each channel gets equal access to the bus.

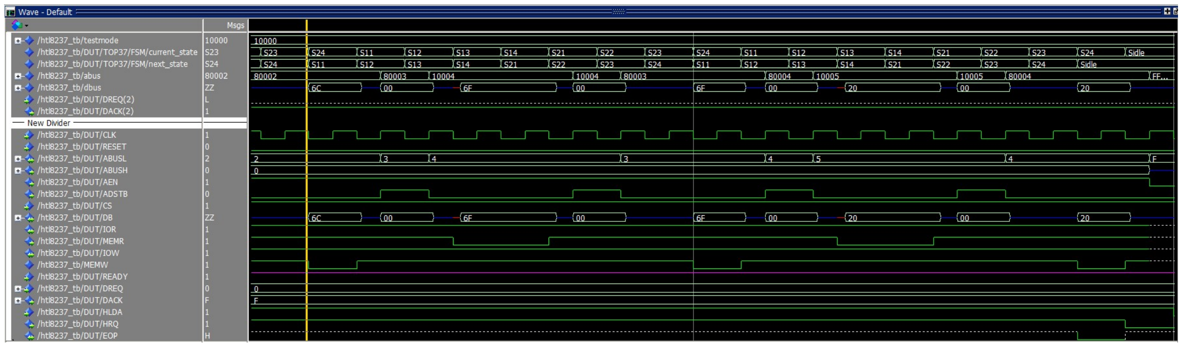
4.2.6 Memory to Memory Test

testmode=”10000”

Memory to Memory transfer uses Channel0 for reading and Channel1 for writing. During the transfer the data is stored in a temporary register (Temp). In the test below data is transferred from 0x10000 to 0x80000.

```
#  ======================== Memory to Memory Test ============================
# 10000 : 48 65 6C 6C 6F 20 57 6F 72 6C 64 00 00 00 00 00
# 10010 : 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
# 10020 : 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
#  Test 0 Address 10001 to 80000, Normal Write, CH0++, CH1++
#  EOP Memory to Memory Transfer
#  ----------- 82C37 status after Memory transfer ----------------------------
#  CH0:A=0006 C=FFFF  CH1:A=0005 C=FFFF  CH2:A=000D C=FFFF  CH3:A=000F C=FFFF
#  MD0:B,A++,  ,RD    MD1:B,A++,  ,WR    MD2:S,A++,  ,RD    MD3:S,A++,  ,RD
#  Status=02 Mask=E Request=0 Command=01 Temp=20 MEM2MEM
# 80000 : 65 6C 6C 6F 20 xx xx xx xx xx xx xx xx xx xx xx
# 80010 : xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx
# 80020 : xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx
```

As shown in the Modelsim waveform below a Memory to Memory transfer takes 8 stages through the finite state machine. During the first state (states S11 to S14) the data is transferred from memory to the temporary register and transferred from the temporary register back to memory during stages S21 to S24.
Memory initialization (see test3) can be achieved by setting the hold enable bit (bit 1) in the command register. Channel0 will then keep the address constant but channel1 is still changing. Hence any data channel0 is pointing to will be written to all channel1 addresses.



```
#  Test 1 Address 10007 to 8001E, AutoInit, CH0--, CH1--
#  EOP Memory to Memory Transfer
#  ----------- 82C37 status after Memory transfer ----------------------------
#  CH0:A=0002 C=FFFF  CH1:A=001E C=0004  CH2:A=000D C=FFFF  CH3:A=000F C=FFFF
#  MD0:B,A--,AI,RD    MD1:B,A--,AI,WR    MD2:S,A++,  ,RD    MD3:S,A++,  ,RD
#  Status=02 Mask=C Request=0 Command=01 Temp=6C MEM2MEM
# 80000 : 65 6C 6C 6F 20 xx xx xx xx xx xx xx xx xx xx xx
# 80010 : xx xx xx xx xx xx xx xx xx xx xx 6C 6F 20 57 6F xx
# 80020 : xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx
#
#  Test 2 Address 1000A to 80007, Extended write, CH0--, CH1++
#  EOP Memory to Memory Transfer
#  ----------- 82C37 status after Memory transfer ----------------------------
#  CH0:A=0005 C=FFFF  CH1:A=000C C=FFFF  CH2:A=000D C=FFFF  CH3:A=000F C=FFFF
#  MD0:B,A--,  ,RD    MD1:B,A++,  ,WR    MD2:S,A++,  ,RD    MD3:S,A++,  ,RD
#  Status=02 Mask=E Request=0 Command=21 Temp=57 MEM2MEM Extended Write
# 80000 : 65 6C 6C 6F 20 xx xx 64 6C 72 6F 57 xx xx xx xx
# 80010 : xx xx xx xx xx xx xx xx xx xx 6C 6F 20 57 6F xx
# 80020 : xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx
#
#  Test 3 Address 10005 to 80000, Hold CH0, CH1++, fill with 20
#  EOP Memory to Memory Transfer
#  ----------- 82C37 status after Memory transfer ----------------------------
#  CH0:A=0005 C=FFFF  CH1:A=0005 C=FFFF  CH2:A=000D C=FFFF  CH3:A=000F C=FFFF
#  MD0:B,A--,  ,RD    MD1:B,A++,  ,WR    MD2:S,A++,  ,RD    MD3:S,A++,  ,RD
#  Status=02 Mask=E Request=0 Command=03 Temp=20 MEM2MEM CH0Hold
# 80000 : 20 20 20 20 20 xx xx 64 6C 72 6F 57 xx xx xx xx
# 80010 : xx xx xx xx xx xx xx xx xx xx 6C 6F 20 57 6F xx
# 80020 : xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx
#
#  Test 4 Address 10000 to 8001F, Hold CH0, CH1--, fill with 48
#  EOP Memory to Memory Transfer
#  ----------- 82C37 status after Memory transfer ----------------------------
#  CH0:A=0000 C=FFFF  CH1:A=001A C=FFFF  CH2:A=000D C=FFFF  CH3:A=000F C=FFFF
#  MD0:B,A++,  ,RD    MD1:B,A--,  ,WR    MD2:S,A++,  ,RD    MD3:S,A++,  ,RD
#  Status=02 Mask=E Request=0 Command=03 Temp=48 MEM2MEM CH0Hold
# 80000 : 20 20 20 20 20 xx xx 64 6C 72 6F 57 xx xx xx xx
# 80010 : xx xx xx xx xx xx xx xx xx xx 6C 48 48 48 48 48
# 80020 : xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx
```

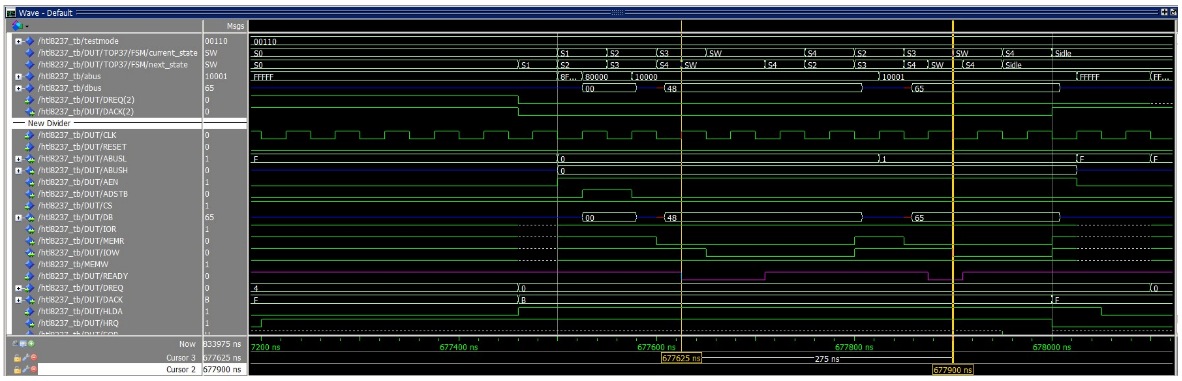
4.2.7 READY input signal Test

testmode="00110" to “01001”

Each memory/IO read and write cycle can be extended using the READY input pin. The READY pin is sampled during the transition from state S3 to S4 and for the Memory to Memory transfers from S13 to S14 (memory read) and S23 to S23 (memory write).

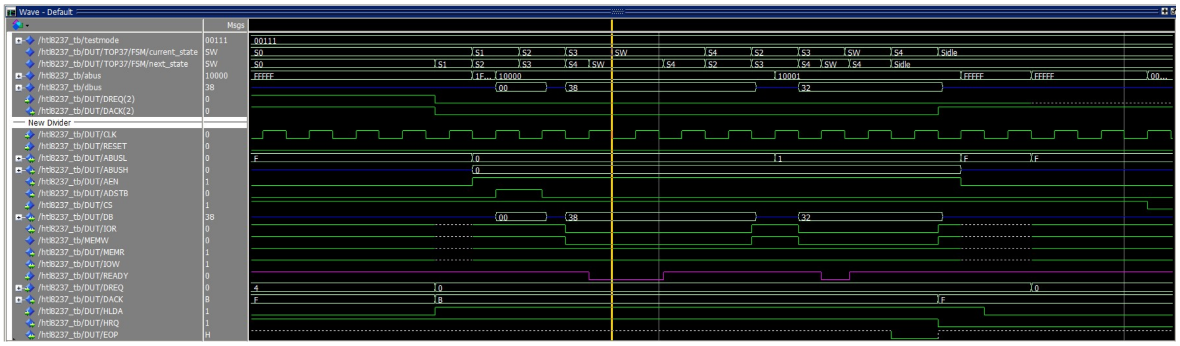
```
#  ======================= READY input Test ============================
#  Start DMA Test 0 Addr++, 2 bytes, address 10000-10004, assert READY for memory read
#  DACK2 Asserted
#    IO DMA Write channel 2 H
#    IO DMA Write channel 2 H                extra waitstates
#    IO DMA Write channel 2 H                extra waitstates
#    IO DMA Write channel 2 e
#    IO DMA Write channel 2 e                extra waitstates
#  EOP Asserted Channel 2
#    CPU has access to the bus
```

In the testcase below the READY pin is asserted for 2 clock cycles during the memory read and 1 clock cycle for the IO write. The READY pin is sampled at the end of S3.



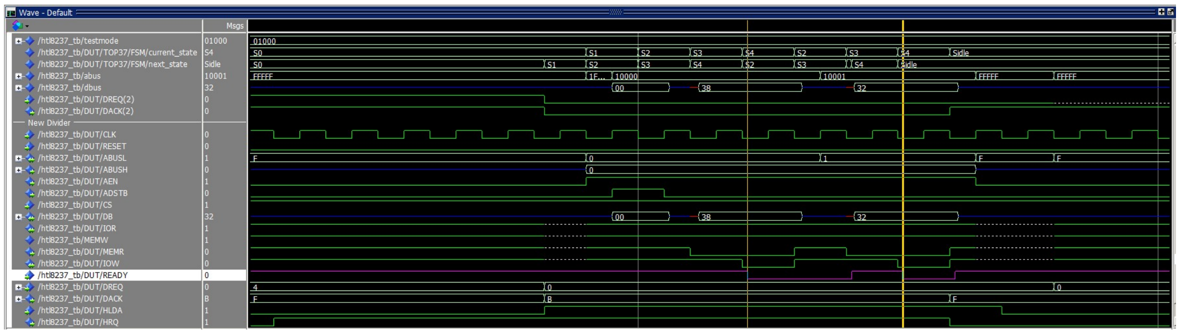
```
#  Start DMA Test 1 Addr++, 2 bytes, address 10000-10004, assert READY for memory write
#  DACK2 Asserted
#    IO DMA Read channel 2
#    IO DMA Read channel 2
#    IO DMA Read channel 2
#    IO DMA Read channel 2
#    IO DMA Read channel 2
#    IO DMA Read channel 2
#    IO DMA Read channel 2
#  EOP Asserted Channel 2
#    CPU has access to the bus
#
#  Start DMA Test 2 Addr++, 2 bytes, address 10000-10004, Ext. Write, assert READY for memory
write
#  DACK2 Asserted
#    IO DMA Read channel 2
#    IO DMA Read channel 2
#    IO DMA Read channel 2
#    IO DMA Read channel 2
#    IO DMA Read channel 2
#    IO DMA Read channel 2
#    IO DMA Read channel 2
#  EOP Asserted Channel 2
#    CPU has access to the bus
```

For Extended Write the IOW signal has the same extended length as the MEMR signal.



```
#  Start DMA Test 3 Addr++, 2 bytes, address 10000-10004, assert READY for IO write
#  DACK2 Asserted
#    IO DMA Write channel 2 8
#    IO DMA Write channel 2 2
#  EOP Asserted Channel 2
#    CPU has access to the bus
```

During Test3 the READY signal is asserted during S4, in this case the READY signal is not recognized and hence ignored.



```
#
#  Start DMA Test 4 Addr++, 2 bytes, address 10000-10004, assert READY for IO read
#  DACK2 Asserted
#    IO DMA Read channel 2
#    IO DMA Read channel 2
#    IO DMA Read channel 2
#    IO DMA Read channel 2
#    IO DMA Read channel 2
#    IO DMA Read channel 2
#    IO DMA Read channel 2
#  EOP Asserted Channel 2
#    CPU has access to the bus
```

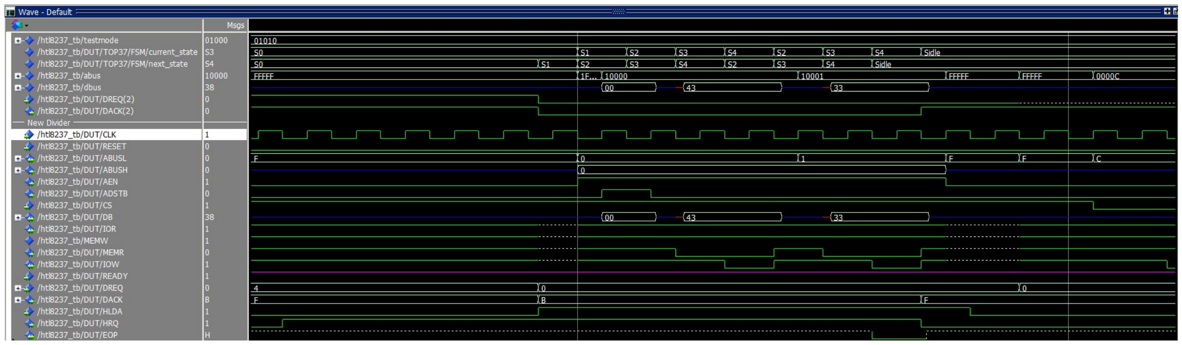
4.2.8 EOP input Test

testmode="01010"

When an EOP pulse occurs, whether internally or externally generated, the 8237 will terminate the service, and if autoinitialize is enabled, the base registers will be written to the current registers of that channel. The mask bit and TC bit in the status word will be set for the currently active channel by EOP unless the channel is programmed for autoinitialize. In that case, the mask bit remains clear.

```
  ======================= EOP input Test =================================
#  ----------- HTL8237 status before memory to IO transfer --------------------
#  CH0:A=0000 C=FFFF  CH1:A=001A C=FFFF  CH2:A=0000 C=000A  CH3:A=000F C=FFFF
#  MD0:B,A++,  ,RD    MD1:B,A--,  ,WR    MD2:B,A++,  ,RD    MD3:S,A++,  ,RD
#  Status=04 Mask=B Request=0 Command=00 Temp=48
#  Start DMA Test 0 Addr++, 11 bytes, address 10000-10040, Issue EOP after byte 2, READ
#  DACK2 Asserted
#    IO DMA Write channel 2 C
#    IO DMA Write channel 2 3
#  EOP Asserted Channel 2
#    CPU has access to the bus
#  ----------- 82C37 status after memory to IO transfer ----------------------
#  CH0:A=0000 C=FFFF  CH1:A=001A C=FFFF  CH2:A=0002 C=0008  CH3:A=000F C=FFFF
#  MD0:B,A++,  ,RD    MD1:B,A--,  ,WR    MD2:B,A++,  ,RD    MD3:S,A++,  ,RD
#  Status=04 Mask=F Request=0 Command=00 Temp=48
```

EOP is asserted during the second transfer.



```
#  ----------- HTL8237 status before memory to IO transfer --------------------
#  CH0:A=0000 C=FFFF  CH1:A=001A C=FFFF  CH2:A=0000 C=000A  CH3:A=000F C=FFFF
#  MD0:B,A++,  ,RD    MD1:B,A--,  ,WR    MD2:B,A++,AI,RD    MD3:S,A++,  ,RD
#  Status=00 Mask=B Request=0 Command=00 Temp=48
#  Start DMA Test 1 Addr++, 11 bytes, address 10000-10040, Issue EOP after byte 2, autoinit,
READ
#  DACK2 Asserted
#    IO DMA Write channel 2 C
#    IO DMA Write channel 2 3
#  EOP Asserted Channel 2
#    CPU has access to the bus
#  ----------- 82C37 status after memory to IO transfer ----------------------
#  CH0:A=0000 C=FFFF  CH1:A=001A C=FFFF  CH2:A=0000 C=000A  CH3:A=000F C=FFFF
#  MD0:B,A++,  ,RD    MD1:B,A--,  ,WR    MD2:B,A++,AI,RD    MD3:S,A++,  ,RD
#  Status=04 Mask=B Request=0 Command=00 Temp=48
#
#  ----------- HTL8237 status before memory to IO transfer --------------------
#  CH0:A=0000 C=FFFF  CH1:A=001A C=FFFF  CH2:A=0000 C=000A  CH3:A=000F C=FFFF
#  MD0:B,A++,  ,RD    MD1:B,A--,  ,WR    MD2:B,A++,  ,RD    MD3:S,A++,  ,RD
#  Status=00 Mask=B Request=0 Command=00 Temp=48
#  Start DMA Test 2 Addr++, 11 bytes, address 10000-10040, Issue Spurious EOP to Master after
byte 2, READ
#  DACK2 Asserted
#    IO DMA Write channel 2 C
#    IO DMA Write channel 2 3
#    IO DMA Write channel 2 l         EOP asserted to Master but is ignored
#    IO DMA Write channel 2 l
#    IO DMA Write channel 2 o
#    IO DMA Write channel 2
#    IO DMA Write channel 2 W
#    IO DMA Write channel 2 o
#    IO DMA Write channel 2 r
#    IO DMA Write channel 2 l
#    IO DMA Write channel 2 d
#  EOP Asserted Channel 2
#    CPU has access to the bus
#  ----------- 82C37 status after memory to IO transfer ----------------------
#  CH0:A=0000 C=FFFF  CH1:A=001A C=FFFF  CH2:A=000B C=FFFF  CH3:A=000F C=FFFF
#  MD0:B,A++,  ,RD    MD1:B,A--,  ,WR    MD2:B,A++,  ,RD    MD3:S,A++,  ,RD
#  Status=04 Mask=F Request=0 Command=00 Temp=48
```

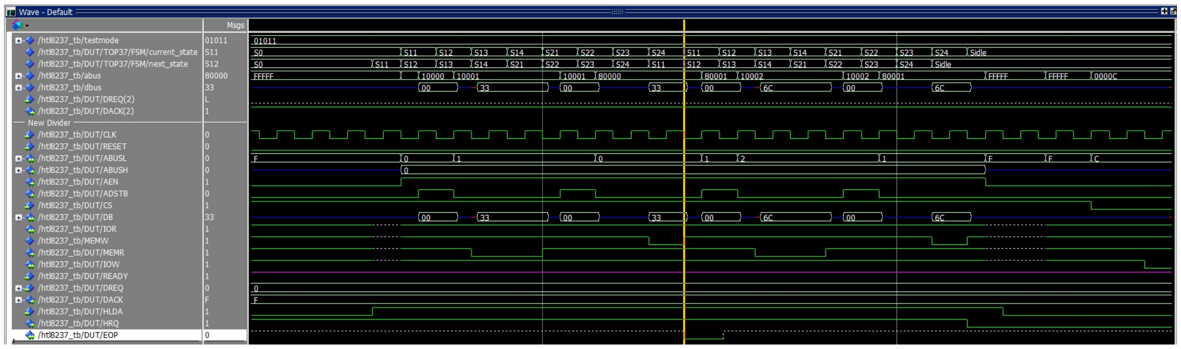
Note an EOP issued to the Master controller is ignored as it is in Cascade mode

4.2.9 EOP input during Memory to Memory transfer Test

testmode=”01011”

When an EOP pulse is asserted during a Memory to Memory transfer the transfer is completed before termination. EOP is sampled during state S11 to S12 and state S21 to S22, the transfer is terminated after S24.

```
#  ======================== EOP Memory to Memory Test ========================
# 10000 : 43 33 6C 6C 6F 20 57 6F 72 6C 64 00 00 00 00 00
# 10010 : 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
# 10020 : 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
#  ----------- HTL8237 status before Memory to Memory transfer -----------------
#  CH0:A=0001 C=0004  CH1:A=0000 C=0004  CH2:A=000B C=FFFF  CH3:A=000F C=FFFF
#  MD0:B,A++,  ,RD    MD1:B,A++,  ,WR    MD2:B,A++,  ,RD    MD3:S,A++,  ,RD
#  Status=00 Mask=E Request=0 Command=01 Temp=48 MEM2MEM
#  Test 0 Address 10001 to 80000, Normal Write, CH0++, CH1++
# External EOP asserted!
#  EOP Memory to Memory Transfer
#  ----------- 82C37 status after Memory to Memory transfer ----------------------
#  CH0:A=0003 C=0002  CH1:A=0002 C=0002  CH2:A=000B C=FFFF  CH3:A=000F C=FFFF
#  MD0:B,A++,  ,RD    MD1:B,A++,  ,WR    MD2:B,A++,  ,RD    MD3:S,A++,  ,RD
#  Status=02 Mask=E Request=0 Command=01 Temp=6C MEM2MEM
# 80000 : 33 6C 20 20 20 xx xx 64 6C 72 6F 57 xx xx xx xx
# 80010 : xx xx xx xx xx xx xx xx xx xx 6C 48 48 48 48 48
# 80020 : xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx
```



The EOP is asserted during the beginning of the second memory to memory read operation (S11) which results in termination after the transfer has completed (state S24). Also note the DACK output signals are not asserted during a memory to memory transfer.

```
#  ----------- HTL8237 status before Memory to Memory transfer -----------------
#  CH0:A=0007 C=0000  CH1:A=001E C=0004  CH2:A=000B C=FFFF  CH3:A=000F C=FFFF
#  MD0:B,A--,AI,RD    MD1:B,A--,AI,WR    MD2:B,A++,  ,RD    MD3:S,A++,  ,RD
#  Status=00 Mask=E Request=0 Command=01 Temp=6C MEM2MEM
#  Test 1 Address 10007 to 8001E, AutoInit, CH0--, CH1--
# External EOP asserted!
#  EOP Memory to Memory Transfer
#  ----------- 82C37 status after Memory to Memory transfer ----------------------
#  CH0:A=0005 C=FFFE  CH1:A=001E C=0004  CH2:A=000B C=FFFF  CH3:A=000F C=FFFF
#  MD0:B,A--,AI,RD    MD1:B,A--,AI,WR    MD2:B,A++,  ,RD    MD3:S,A++,  ,RD
#  Status=02 Mask=C Request=0 Command=01 Temp=57 MEM2MEM
# 80000 : 33 6C 20 20 20 xx xx 64 6C 72 6F 57 xx xx xx xx
# 80010 : xx xx xx xx xx xx xx xx xx xx 6C 48 48 57 6F 48
# 80020 : xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx
```

Note only channel1 is reloaded after an EOP pulse.

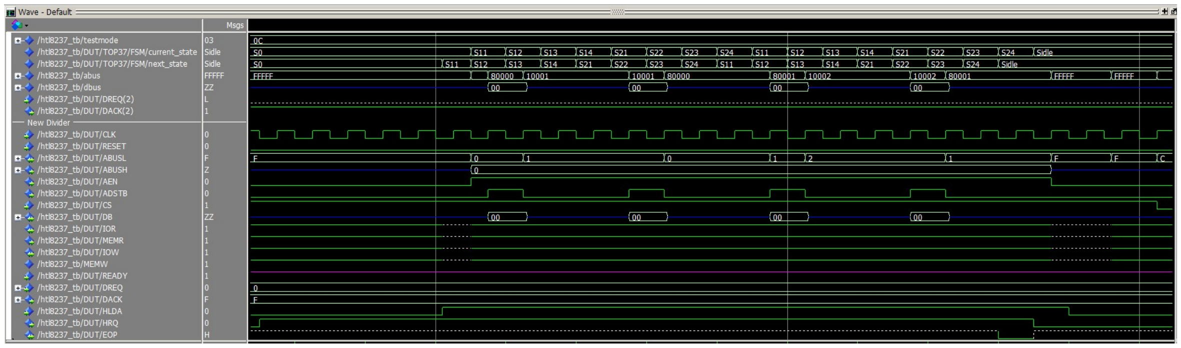
4.2.10 Memory to Memory in Verify mode with READY input Test

testmode=”01100”, “01101”

Verify mode result in pseudo transfers where all I/O and memory read/write control signals are kept inactive. The READY input is ignored in verify transfers.

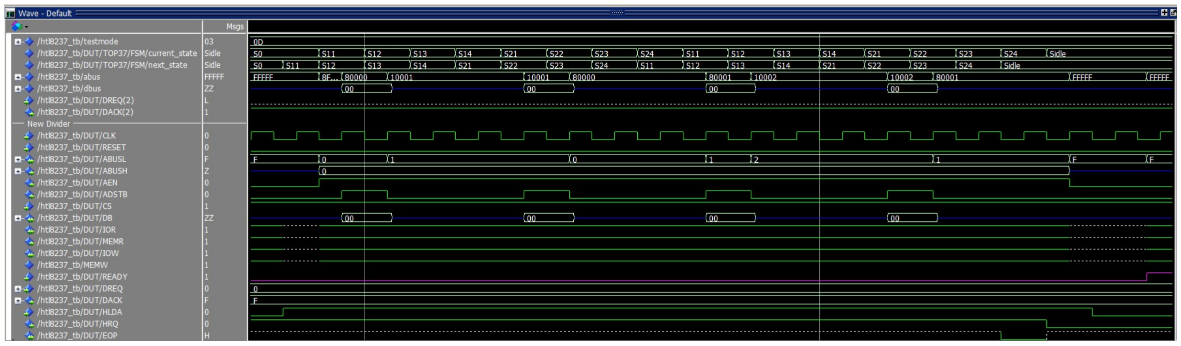
```
#  ======================== Memory to Memory Verify/Ready Test ===================
# 80000 : 20 20 20 20 20 xx xx 64 6C 72 6F 57 xx xx xx xx
# 80010 : xx xx xx xx xx xx xx xx xx xx 6C 48 48 57 43 43
# 80020 : xx xx xx xx xx xx xx xx xx xx xx xx xx xx 6F 6C
#  ----------- HTL8237 status before Memory to Memory transfer -----------------
#  CH0:A=0001 C=5555  CH1:A=0000 C=0001  CH2:A=000B C=FFFF  CH3:A=000F C=FFFF
#  MD0:B,A++,  ,VF    MD1:B,A++,  ,VF    MD2:B,A++,  ,RD    MD3:S,A++,  ,RD
#  Status=00 Mask=E Request=0 Command=01 Temp=43 MEM2MEM
#  Test 0 Address 10001 to 80000, Verify, CH0++, CH1++
#  EOP Memory to Memory Transfer
#  ----------- 82C37 status after Memory to Memory transfer ----------------------
#  CH0:A=0003 C=5553  CH1:A=0002 C=FFFF  CH2:A=000B C=FFFF  CH3:A=000F C=FFFF
#  MD0:B,A++,  ,VF    MD1:B,A++,  ,VF    MD2:B,A++,  ,RD    MD3:S,A++,  ,RD
#  Status=02 Mask=E Request=0 Command=01 Temp=43 MEM2MEM
# 80000 : 20 20 20 20 20 xx xx 64 6C 72 6F 57 xx xx xx xx
# 80010 : xx xx xx xx xx xx xx xx xx xx 6C 48 48 57 43 43
# 80020 : xx xx xx xx xx xx xx xx xx xx xx xx xx xx 6F 6C
```

Note the lack of MEMR/MEMW signals in the screenshot below



```
#  ----------- HTL8237 status before Memory to Memory transfer -----------------
#  CH0:A=0001 C=5555  CH1:A=0000 C=0001  CH2:A=000B C=FFFF  CH3:A=000F C=FFFF
#  MD0:B,A++,  ,VF    MD1:B,A++,  ,VF    MD2:B,A++,  ,RD    MD3:S,A++,  ,RD
#  Status=00 Mask=E Request=0 Command=01 Temp=43 MEM2MEM
#  Test 0 Address 10001 to 80000, Verify+READY, CH0++, CH1++
#  EOP Memory to Memory Transfer
#  ----------- 82C37 status after Memory to Memory transfer ----------------------
#  CH0:A=0003 C=5553  CH1:A=0002 C=FFFF  CH2:A=000B C=FFFF  CH3:A=000F C=FFFF
#  MD0:B,A++,  ,VF    MD1:B,A++,  ,VF    MD2:B,A++,  ,RD    MD3:S,A++,  ,RD
#  Status=02 Mask=E Request=0 Command=01 Temp=43 MEM2MEM
# 80000 : 20 20 20 20 20 xx xx 64 6C 72 6F 57 xx xx xx xx
# 80010 : xx xx xx xx xx xx xx xx xx xx 6C 48 48 57 43 43
# 80020 : xx xx xx xx xx xx xx xx xx xx xx xx xx xx 6F 6C
```

Negating READY will not affect the transfer timing as shown below (purple line).



5. Synthesis

The HTL8237 can be synthesized using any modern synthesis tool. An in order file list is provided in the synthesis directory.

6. Pin Description

Symbol	Type	Function
CLK	I	Clock Input Signal, note logic is clocked on the falling edge
RESET	I	Asynchronous reset input Note1
CS	I	Active low chip select input
DREQ	I	Synchronous DMA request input Note2
READY	I	Active high READY signal to extend R/W timesNote2,3
HRQ	O	Active high Hold Request output
HLDA	I	Active high Hold Acknowledge input
DACK	O	DMA Acknowledge output
DB	I/O	Bidirectional Data Bus

IOR	I/O	Active low IO Read
IOW	I/O	Active low IO Write
EOP	I/O	Active low End of Process Note2,3
ABUSL	I/O	Address bits A[3-0]
ABUSH	O	Address bits A[7-4]
AEN	O	Address Enable output
ADSTB	O	Address Strobe output
MEMR	O	Active low Memory Read output
MEMW	O	Active low Memory Write output
NC	O	Pin5 on the original VLSI 8237 is a no connect (tri-stated)

Note1: For some FPGA types, like Xilinx, a synchronous reset is more efficient.
Note2: Metastability prevention circuit (2/3 serial FF's) might be required.
Note3: Connect to VCC when not used

6.1 RESET Signal

RESET is an active high asynchronous reset signal. No reset synchronization circuit (asynchronously asserted, synchronously negated) is available but can easily be added to the blk37 reset input. Note that all internal registers are asynchronously reset, for some FPGA vendors like Xilinx a synchronous reset is more efficient.

6.2 READY Signal

READY is a synchronous input signal used to extend the read/write strobes. The READY signal is only sampled at specific times during a DMA transfer. The signal should be tied to Vcc if not used.
The READY signal is "sampled" by the FSM during the transition from S3 to S4, S13 to S14 and from S23 to S24. If the signal is asynchronous then a metastability input protection circuit must be added. In addition, the FSM must be modified to take into account the additional delay input.

6.3 EOP Signal

EOP is an active low bidirectional End Of Process signal. The HTL8237 will pulse the output low when the terminal count of any of the channels has occurred. External circuitry can pull this signal low to terminate the DMA cycle. The EOP signal is asserted in the last clock cycle of the DMA transfer. The EOP is asserted low whilst the read/write cycle is in progress (as per the original VLSI device). As an input signal the EOP is sampled and latched during FSM states S2, S12, S22 and acted upon during states S4 and S24. Note that EOP is assumed to be a synchronous input signal.

7. Differences

The following lists the main differences between the HTL8237 and the original VLSI Intel 82C37A

1. The HTL8237 is not timing compatible with the original VLSI device.
2. The HTL8237 tries to be bus cycle compatible with the original VLSI device; however, small differences might occur [tbc].
3. Unlike the original VLSI device the HTL8237 clock cannot be stopped. The original VLSI device can be reconfigured with the clock stopped.

8. History

Version	Date	Changes
1.0	20/01/2005	First Version
1.1	07/04/2007	EOP latching extended to S11/S12/S13, S21/S22/S23.
1.2	08/03/2016	First/Last FF logic changed to avoid glitches
1.3	20/11/2023	Cleaned and uploaded to github

9. License

See the LICENSE file for details.

Trademarks