HUNT ENGINEERING
Chestnut Court, Burton Row,
Brent Knoll, Somerset, TA9 4BP, UK
Tel: (+44) (0)1278 760188,
Fax: (+44) (0)1278 760199,
Email: sales@hunteng.co.uk
www.hunteng.co.uk
www.hunt-dsp.com

The HERON FPGA7 Example2

Rev 1.2 R. Williams 29-04-05

The HERON-FPGA7 is a module that has a Virtex-II Xilinx FPGA and 256Mbytes of SDRAM memory.

Most users will use the HERON-FPGA7 to provide a custom storage capability using the SDRAM to store large amounts of data. The module may also use the FPGA to process the data stored in the SDRAM.

Whatever the intended application of the HERON-FPGA7, it is useful to be able to test all SDRAM memory locations to check the integrity of that memory. Example2 performs this test using a combination of different data patterns that are read and written to all locations of memory.

History

Example revision 1.0	26-09-03	Developed from Example2 for the HERON-FPGA5
Example revision 1.1	17-12-03	Changes made to ensure memory test runs in all slots
Example revision 1.2	29-04-05	Removed reference to specific ISE versions

What the Bit-stream Does

Example2 (Memory Test) for the HERON-FPGA7 is supplied on the HUNT ENGINEERING CD, and web site. The FPGA source code is supplied along with bit-streams that can be loaded directly onto the HERON-FPGA7. There is also an example program that can be used on the Host PC.

This example can be used as it is, or if it suits your needs can be used as a starting place for creating your own memory test. Please note, when using the bit-streams and host program together, you will first need to configure your FPGA module using the HERON-FPGA programming tool before running the example host program.

If you make changes to the project and re-build it you can change the functionality to be whatever you want, but if you use the supplied bit-stream you need to know what it is doing. This document describes that for you.

The standard clock soldered to “User Osc3” of the HERON-FPGA7 is 100Mhz.

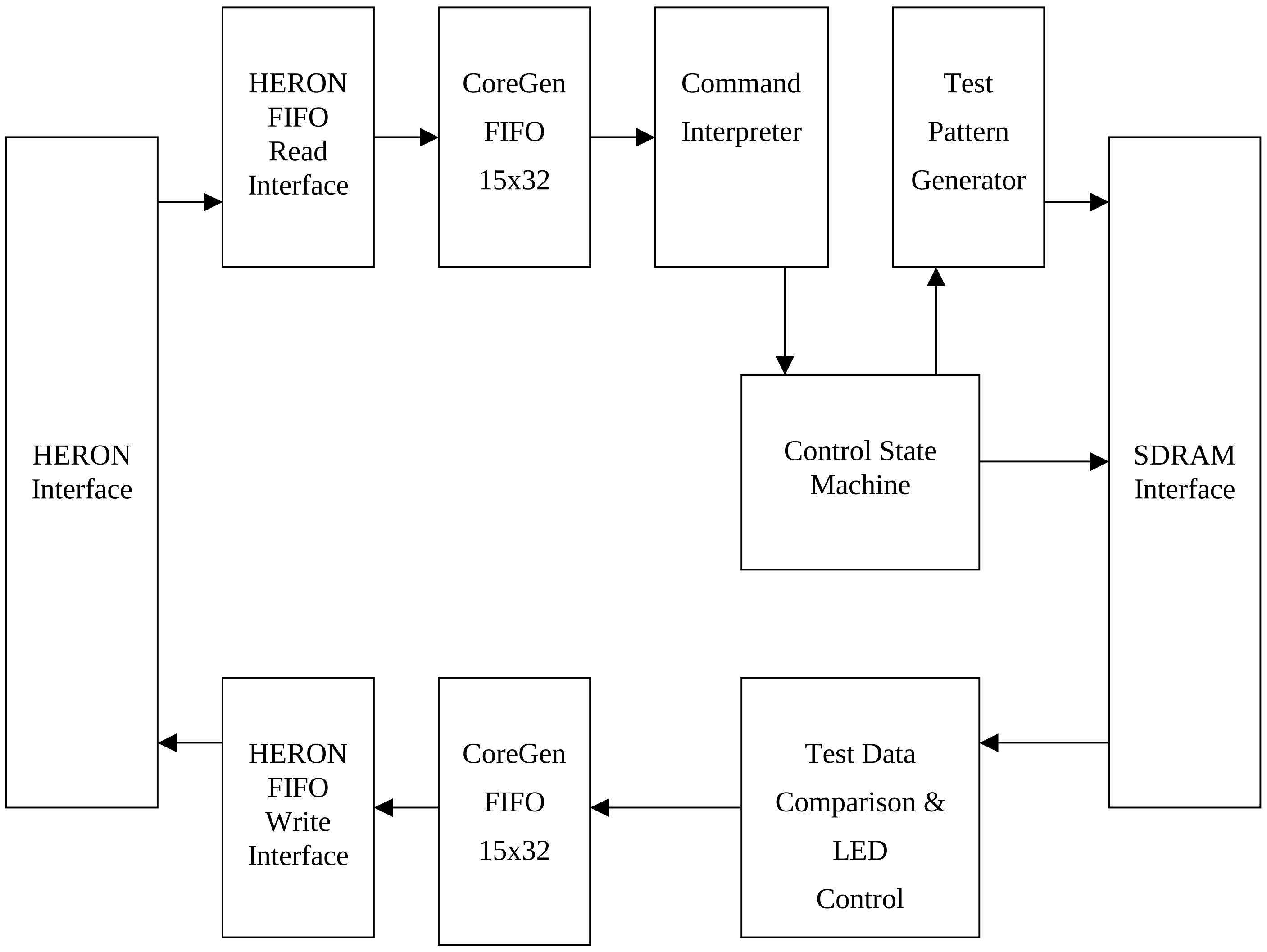
Example2 always uses this clock to drive the fundamental clock source of the SDRAM. This clock source is synthesized up to 133MHz inside the SDRAM interface component of the Hardware Interface Layer.

This clock is also used to drive the FIFO clock directly, or to drive the FIFO clock after being divided by 2.

For an HEPC8 the HERON FIFO clocks must be driven at less than 60Mhz, whereas for the HEPC9 they must be driven at between 60Mhz and 100Mhz. For this reason there are different bit-streams supplied for HEPC8 and HEPC9 as follows. There are also several options depending on the FPGA fitted to the module. The valid FPGA device options are xc2v3000, xc2v6000 and xc2v8000:

2v3000ff1152.hcb	HERON-FPGA7 fitted to HEART based carrier e.g HEPC9 (100Mhz FIFO clocks)
2v3000ff1152_pc8.hcb	HERON-FPGA7 fitted to HEPC8 (50Mhz FIFO clocks)
2v6000ff1152.hcb	HERON-FPGA7 fitted to HEART based carrier e.g HEPC9 (100Mhz FIFO clocks)
2v6000ff1152_pc8.hcb	HERON-FPGA7 fitted to HEPC8 (50Mhz FIFO clocks)
2v8000ff1152.hcb	HERON-FPGA7 fitted to HEART based carrier e.g HEPC9 (100Mhz FIFO clocks)
2v8000ff1152_pc8.hcb	HERON-FPGA7 fitted to HEPC8 (50Mhz FIFO clocks)

FUNCTIONAL BLOCK DIAGRAM



Operation of the Control State Machine

The heart of Example2 is the Control State Machine that controls how an iteration of the memory test is performed.

A single iteration of the memory test involves reading and writing the SDRAM memory with one particular test pattern, and is triggered by sending one word over HERON Input FIFO 0. For one iteration, the memory can be written only, read only, or both written and then read. For one iteration one of six test patterns can be selected.

The function of the control word data bits that are sent through the HERON Input FIFO is defined in the table below:

Control Word Data Bit	Function
3 to 0	Test Pattern Select(3 downto 0)
4	Memory Write Enable
5	Memory Read Enable
31 to 6	Not used, set to zero

The first four bits of the control word define which of six test patterns to use for the current iteration of the memory test. The definition of these bits is shown in the table below.

Test Pattern Select	Pattern Used
0000	All test pattern data bytes set to 00h
0001	All test pattern data bytes set to FFh
0010	All test pattern data bytes set to 55h
0011	All test pattern data bytes set to AAh
0100	Incrementing count, starting with value 0
1000	Shift left, one bit high, starting with value 1

For one iteration of the memory test a single word must be sent to HERON Input FIFO 0. This word will set the test pattern and memory read and memory write enables. The control state machine will then, if the Memory Write Enable is set high, write all 64 Mlocations of the SDRAM using the test pattern selected. (Note, the SDRAM memory is organised as 32-bit wide, therefore 256 Mbytes of storage is presented as 64Mlocations of 32-bits).

When the write process has completed, the control state machine will then read all 64Mlocations of the SDRAM, if the Memory Read Enable has been set high. All data read from the SDRAM is checked against the expected data, and is output through HERON Output FIFO 0.

For each iteration of the memory test, LEDs 0 to 3 represent the setting of Test Pattern Select bits 0 to 3. If during a memory read operation the data check detects an error, all of these LEDs will become illuminated. Once illuminated due to a data error, they will remain illuminated until the next system reset.

LED 4 will always flash to indicate that the system FIFO clock FCLK_G is running.

Typical Use of the Example Bit-stream

The typical use of the example bit-stream is to read and write all memory locations with all of the test patterns. Doing so will perform a thorough check of the SDRAM.

To perform such a test the following sequence of events should be followed:

1. Load the bit-stream for the Memory-Test (Example2)
2. Send one word to HERON Input FIFO 0 with the value `030h`. This will test with 0's.
3. Receive 67108864 words from HERON Output FIFO 0. Check all values are `00000000h`.
4. Send one word to HERON Input FIFO 0 with the value `031h`. This will test with 1's.
5. Receive 67108864 words from HERON Output FIFO 0. Check all values are `FFFFFFFFh`.
6. Send one word to HERON Input FIFO 0 with the value `032h`. This will test with 5's.
7. Receive 67108864 words from HERON Output FIFO 0. Check all values are `55555555h`.
8. Send one word to HERON Input FIFO 0 with the value `033h`. This will test with A's.
9. Receive 67108864 words from HERON Output FIFO 0. Check all values are `AAAAAAAAh`.
10. Send one word to HERON Input FIFO 0 with the value `034h`. This will test with a count.
11. Receive 67108864 words from HERON Output FIFO 0. Check all values against a count.
12. Send one word to HERON Input FIFO 0 with the value `038h`. This will test with a 'walking one'.
13. Receive 67108864 words from HERON Output FIFO 0. Check all values against left shift of one.
14. Check that the LEDS do not indicate an error condition was detected by the data check.

The above sequence is performed by the 'mem_test' host example program that is provided with the example VHDL project and bit-streams with the exception of step 1. In order to configure your FPGA module you will first need to use the HERON-FPGA programming tool using one of the supplied bit-streams for Example 2.

Where are the Bit-streams and Example Program?

The bit streams for this example can be found on the HUNT ENGINEERING CD in the directory `\fpga\fpga7v1\Memory_Test(ex2)`. The name of the bitstream file reflects the Xilinx FPGA part number and the Carrier board type, as explained earlier in this document.

The host example program can be found in the '`host`' sub-directory.

An easier way to navigate to the correct directory is to select the "Files" link next to the "SDRAM Memory Test" link under the IP sections of the CD browser.

The source files for the FPGA example can be found in the '`src`' sub-directory. The sources in the '`\fpga7v1\common`' directory are also required. A project for the Xilinx ISE development tools is provided for you in the '`ISE`' sub-directory.

Host Based Example Program

HEART Carrier like HEPC9

If you have a HEART based module carrier like the HEPC9, you can run the Host based example code with your HERON-FPGA7 module fitted to any slot. Then you need to configure the HEART connections between the module and the Host using HeartConf. This is done from inside the host example program for you using the ‘network’ file located in the same directory as the host example executable.

Please note, depending on the slot in which your HERON-FPGA7 module has been placed, you may need to edit the network file to correctly define its location.

The line shown below defines, amongst other things, the slot in which the FPGA module has been placed. The last item in the line is the HERON-ID of the module and must match the slot number. In the line below, we can see that the module has been declared as being in slot 1 of the carrier card.

```
fpga 0      module  root            0x01
```

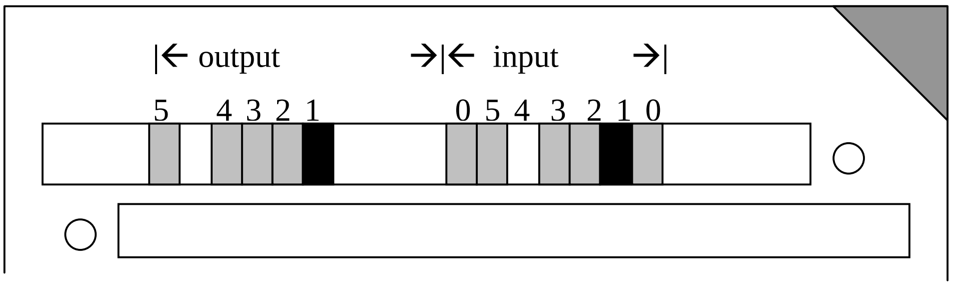
Please adjust this line according to the placement of your HERON-FPGA7.

HEPC8 Carrier

If you have an HEPC8, then you must fit your HERON-FPGA7 module to the first HERON slot as this is the only slot that has a FIFO connection to the Host machine.

The example bit-streams are configured to use HERON Input FIFO 0 and HERON Output FIFO 0. The host FIFO connection for the module in slot 1 is FIFO 1. In order to make the correct FIFO connection you will need to use the default routing jumpers to connect FIFO 0 to FIFO 1.

The default routing jumpers are provided by HERON modules. These are the longer pins on the topmost HERON connector of the module. These pins protrude above the HERON module when it is fitted to the module carrier, to which jumper links can be fitted.



These jumpers are used to select the default routing of FIFO 0 on the Carrier card.

The exact use of these jumpers is defined by the HERON module carrier, so you should reference the user manual for the Carrier card you are using, but the numbering of these “default routing” jumpers is defined in the HERON spec, and shown above.

For this example you will need to fit jumpers as shown in the diagram in order to connect FIFO 0 in and out of the HERON-FPGA7 to FIFO 1 of the HEPC8 carrier.

Host Example: What it does

We supply a pre-compiled Windows program for the PC, which can be used to communicate with the HERON-FPGA7 to run the standard memory test sequence described earlier in this document.

The program is called 'mem_test.exe' and is located in the 'host' sub-directory of Example2. This example program is intended to be used with the example2 bit-stream to provide a confidence check of the SDRAM memory fitted to your HERON-FPGA7.

The 'mem_test' program is supplied as an executable, but we have also included the Microsoft Visual C/C++ 6 project for this program. This allows you to change the program and re-compile as you want.

After you have loaded the correct bit stream into your HERON-FPGA7 module, the "DONE" LED should be switched off showing that the configuration was successful. Also The USER LED4 should flash about once per second, showing that the FIFO clocks are properly running in the FPGA. If the DONE LED is off, but the LED is not flashing it may be because the Delay Locked Loop (DLL) used in the FIFO clock circuit of the FPGA needs to be reset. You can do this using the utility under "programs → HUNT ENGINEERING → API board RESET" if you want, but such a reset will also be made when you start the memory test program.

Host Example: Using it

First the correct example2 bit-stream for your module should be programmed into the FPGA using the standard Windows FPGA programmer found under "Programs → HUNT ENGINEERNG → Program HERON FPGA". Which bit-stream to choose is discussed in the above sections and depends on the carrier type you have.

If you don't know how to load a bit-stream onto the HERON-FPGA7, please review example1 once more. Verify that after the loading process the "Done" LED goes off. Finally, execute the 'mem_test.exe' program.

The memory test program will perform five loops of the memory test sequence described earlier in this document. For each loop, the SDRAM memory is both written and read with each of the six test patterns.

The first pattern is data with all bits set to low, the second pattern is data with all bits set high, the third pattern is the value 55555555h and the fourth pattern is the value AAAAAAAAh.

The fifth pattern is a count that increments by 1 for each new value, starting with the value 0. The sixth and final pattern is a 'walking one'. Starting with the value 00000001h, the value is left shifted by one bit for each new value. After the value 80000000h, the next value returns to 00000001h.

For each test pattern, the host program sends one word to HERON input FIFO 0 of the HERON-FPGA7. This word will select both a write and read of memory, along with the test pattern to be used. The host program then expects to receive 67108864 words from HERON output FIFO 0 of the HERON-FPGA7. The data received is checked against what is expected.

As soon as a data error is detected, the program displays the data error and waits for a key-press.

If all five loops are completed without error the program displays success and terminates.

Host Example: Changing and Building it

The project can be copied from the CD to your hard drive. Then each file needs to have the permissions changed so that they are not read only. The project should build.

Now you can make changes to the file 'mem_test.c' as you wish.

FPGA Example Code

You should understand the HUNT ENGINEERING VHDL support for HERON modules before looking at this section. If you do not then please review Example1 again (the 'Getting Started' example for FPGA modules).

As you are expected to be already familiar with Example1, this section only discusses the points that are unique to example2.

In the USER-AP entity for Example2, contained in the file 'user_ap2.vhd', there are options that allow you to select the FIFO clock frequency. Just as with Example1, you can select if the 100Mhz input clock is divided by 2 to provide the FIFO clock frequency. This allows example2 to be able to clock the FIFOs at 50Mhz (suitable for HEPC8) or 100Mhz (suitable for HEPC9).

You must also remember to set the HIGH_FCLK_G option to show if this clock is higher or lower than 60Mhz.

For example2 the correct options for an HEPC8 are:

DIV2_FCLK	FCLK_G_DOMAIN	HIGH_FCLK_G	HIGH_FCLK_RD	HIGH_FCLK_WR
True	True	False	n/a	n/a

This will set both input and output FIFOs to be clocked at 50Mhz.

For example2 the correct options for an HEPC9 are :-

DIV2_FCLK	FCLK_G_DOMAIN	HIGH_FCLK_G	HIGH_FCLK_RD	HIGH_FCLK_WR
False	True	True	n/a	n/a

This will set both input and output FIFOs to be clocked at 100Mhz.

You need to consider the timing constraints that are defined in the '.ucf' file for your design. Actually if you use a time specification that is more strict than needed there is no problem, so the standard '.ucf' file have the FIFO clocks specified as 100Mhz, along with the SDRAM clock defined as 133MHz. If the project builds (as Example2 does) with this specification it is still guaranteed to work at lower clock speeds. If you add new clock nets into your design then you need to add new timing constraints into your design.