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 FPGA12 Example2

Rev 1.0 R. Williams 14-02-06

The HERON-FPGA12 is a module that has a Virtex-4 FX Xilinx FPGA and 128Mbytes of DDR SDRAM memory. The Virtex-4 FX FPGA includes a Power PC processor core.

Users can use the HERON-FPGA12 to provide a custom storage capability using the DDR SDRAM to store large amounts of data. The module may also use the FPGA and Power PC processor core to process the data stored in the DDR SDRAM.

Whatever the intended application of the HERON-FPGA12, it is useful to be able to test all DDR 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	14-02-06	Developed from Example2 for the HERON-FPGA9

What the Bit-stream Does

Example2 (Memory Test) for the HERON-FPGA12 is supplied on the HUNT ENGINEERING CD, and web site. The FPGA source code is supplied along with a bit-stream that can be loaded directly onto the HERON-FPGA12. 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.

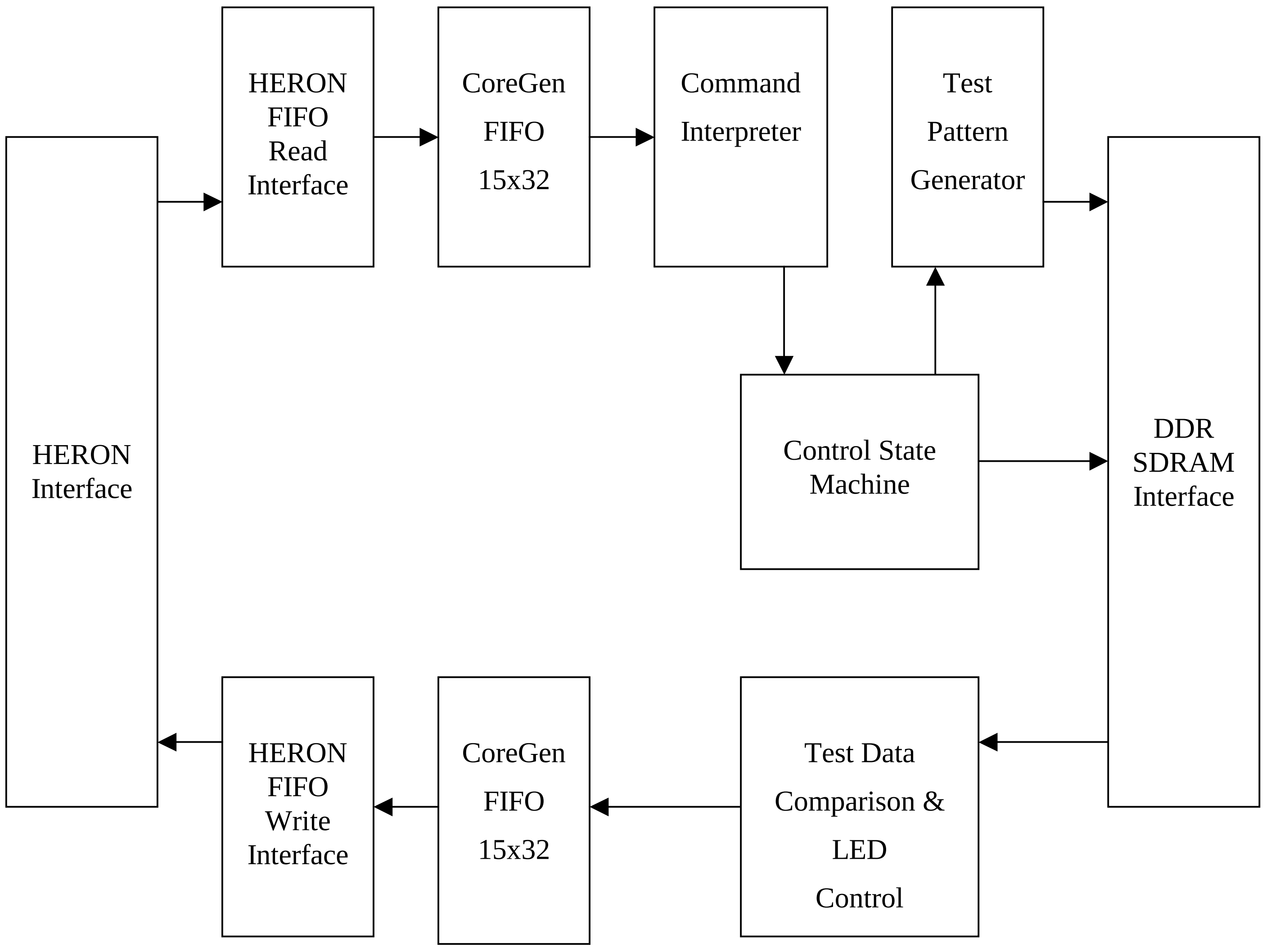
The HERON-FPGA12 is fitted with a 50MHz oscillator which is used to generate a 200MHz clock source for the FPGA. The 200MHz clock source is used to generate all DDR SDRAM clock signals. This is automatically handled by the Hardware Interface Layer.

The HERON-FPGA12 is also fitted with a 100MHz oscillator on OSC3. This clock is used to drive the FIFO clocks directly.

The following bit-stream is supplied as part of the example.

4vfx12ff668.hcb	HERON-FPGA12 fitted to HEART based carrier e.g. HEPC9 (100Mhz 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 DDR 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 32 Mlocations of the DDR memory using the test pattern selected. (Note, the memory is organised as 32-bit wide, therefore 128 Mbytes of storage is presented as 32Mlocations of 32-bits).

When the write process has completed, the control state machine will then read all 32Mlocations of the DDR memory, if the Memory Read Enable has been set high. All data read from the memory 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 DDR 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 33554432 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 33554432 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 33554432 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 33554432 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 33554432 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 33554432 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\fpga12v1\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 '`\fpga12v1\common`' directory are also required. A project for the Xilinx ISE development tools is provided 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-FPGA12 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-FPGA12 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-FPGA12.

Host Example: What it does

We supply a pre-compiled Windows program for the PC, which can be used to communicate with the HERON-FPGA12 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 DDR SDRAM memory fitted to your HERON-FPGA12.

The 'mem_test' program is supplied as an executable, but we have also included the C source 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-FPGA12 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-FPGA12, 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 DDR 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-FPGA12. 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 33554432 words from HERON output FIFO 0 of the HERON-FPGA12. 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.

For example2 the correct options for a HEPC9 are:

FCLK_G_DOMAIN
True

This will set both input and output FIFOs to be clocked by the same 100Mhz clock signal.

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 signals defined as 200MHz. 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.