Applicationguide, DDRControllerTest

This design demonstrates the functionality of the onboard DDR RAM memory. The PCI Express functionality is not used.
One of the requirements for the board is to be able to store 100 MB of data. The XC3S1000 chip used on the board only has 54 kB blockram available, and therefore another solution is needed. The Spartan3 PCI Express Starter kit is mounted with two 512 Mb DDR RAM chips, providing 128 MB of memory with a 32 bit data interface. The chips used are Micron Technology MT46V32M16-6[1].
The ISE project can be found in the folder "*Source\FPGA_Design\DDRControllerTest*" on CD1.

The DDRController

As with all DRAM, the onboard memory needs a controller to handle refreshing, precharging, and similar. Xilinx provides the Memory Interface Generator (MIG) IPcore to generate such a controller.
MiG is started by adding a new IP source in Xilinx, and then choosing *Memories & Storage Elements -> Memory Interface Generators -> MiG v2.0*.

Generating the DDRController

The core comes with parameters for several RAM chips, but only for speedgrades -75 and -5b, not -6 which is used on the board[2]. Therefore a custom part needs to be created, which can be done from within MiG. All the necessary timings and parameters can be found in table 19 in the datasheet for the memory chips. The datawidth needs to be 32 bits, and the frequency should be adjusted to whatever is used in the design. Here, the design is generated with an internal DCM, but depending on the application, an external DCM might be more suitable.
Almost all connections to the memory chips on the board are located in bank0 and bank1 on the FPGA, with the exception of the clock feedback pin, which is located in bank5. Therefore these should be used when asked for bank assignments in MiG.
MiG will then generate a folder with two designs in the project folder. An example design with a testbench, and a user design without a testbench.

The example design

The DDRControllerTest design implements the example design, with a few modifications:

- MiG generates the designs to use a differential clock input. As no differential clocks are available on the Spartan3 PCI Express Starter kit, this is changed to a single-ended clock. This is done in the *infrastructure_top0* module, by exchanging the instantiation of the differential input clock buffer:

```
IBUFGDS_LVDS_25 lvds_clk_input (
   .I(sys_clk),
   .IB(sys_clk_b),
   .O(sys_clk_ibuf) );
```

  To a single-ended input clock buffer:

```
IBUFG #(.IOSTANDARD("DEFAULT")) ibufg_inst (
   .O(sys_clk_ibuf),
   .I(sys_clk) );
```

[1] A datasheet can be found in the *Datasheet* folder on CD1.
[2] The speed grade is not printed directly on the RAM chips, but can be found using the Micron FBGA Part Marking Decoder found here: http://www.micron.com/support/designsupport/tools/fbga/decoder

Additionally, the references to the *sys_clk_b* signal have been removed from the design, as they are not needed anymore.

- For some reason the design uses two multiplexed clock buffers for the internal DCM clock outputs. This has lead to quite a few problems with the design becoming unroutable, so these are changed to standard clock buffers instead. This is done in the *clk_dcm0* module by exchanging the multiplexed clock buffer instantiations:

```
BUFGMUX BUFG_CLK0 (
   .O(clk0_buf),
   .I0(clk0dcm),
   .I1(clk0dcm),
   .S(1'b0) );
BUFGMUX BUFG_CLK90 (
   .O(clk90_buf),
   .I0(clk90dcm),
   .I1(clk90dcm),
   .S(1'b0) );
```

To standard clock buffers:

```
BUFG BUFG_CLK0 (
   .O(clk0_buf),
   .I(clk0dcm) );
BUFG BUFG_CLK90 (
   .O(clk90_buf),
   .I(clk90dcm) );
```

- Some of the timing constraints generated by MiG refer to nets that are not present after synthesis. It is unknown what the reason for this is, but might be due to a different level of optimization or effort being used. Some of the nets appear to exist in the design with a different name, and in these cases, their name in the constraints file has been corrected. All corrections can be seen in the constraints file for the project.

When placing and routing the design, two of the generated time constraints are not met. One of these appears to be due to the clock feedback pin being placed in a bank on the opposite side of the rest of the DDR controller. There is not much to do about this, as it would require a change of the board PCB.

Testing

The built-in testbench continuously writes and reads back a number of values to the memory. It has three single-bit outputs, *init_done*, *data_valid* and *error*, which are connected to three of the leds on the Spartan3 PCI Express Starter kit. Their functions are:

- *error:* Connected to led0 in the led bank. This is activated if a mismatch is encountered between the values read and those written to the memory. During correct operation this led should be off.
- *data_valid:* Connected to led1 in the led bank. This is activated whenever there is valid data present on the databus. During operation this should be on rather faintly, as valid data is only present during a limited time of the read/write cycles.
- *init_done:* Connected to led2 in the led bank. This is activated when the initialization of the memory chips has been completed, and should therefore be on at all times during operation.

The design is connected to the 50 MHz clock on the Spartan3 PCI Express Starter kit. Correct operation is verified as the error led stays off, and additionally by using chipscope to monitor and make sure that the values written match those that are read back.

According to the datasheet for the memory chips the valid clock range is between 75 and 133 MHz (with CAS Latency = 2), so even though the memory appears to work correctly at 50 MHz, a 125 MHz oscillator[1] has been mounted on the empty oscillator socket on the Spartan3 PCI Express Starter kit. The design did not work at this clock speed though. The error led was activated, and by using chipscope, it became clear that the values on the databus were not stabilizing quickly enough. This could probably be due to either the unmet or the non-working time constraints, but this has not been investigated further.

Usage

To run the test, the design can simply be loaded into the FPGA (synthesize/implement if necessary). The testbench will start automatically, and the onboard leds will display the results.
To use the DDR controller in a design with the PCI Express functionality, there are a few things needing to be done:

- The DDR controller and the PIPE core currently use up all the clocklines in the Spartan3, and it is thus not possible to include for instance a chipscope with these two cores. It might however be possible to optimize a few of these clocklines away from the DDR controller, but this has not been investigated further.
- A thorough testing should be performed to make sure that the memory chips work as intended at the frequency applied, if using a clock speed below 75 MHz.

[1] The oscillator is made by Rakon, product id: LF SPXO009440, Farnell product id: 9713620.