

BEE Mux-Demux Implementation
For Connecting Subsystems
On 2 FPGAs

15 July 2002

Hans-Martin Bluethgen

hmb@eecs.berkeley.edu

1 Scope

In BEE's local mesh, i.e. the interconnections between adjacent FPGAs, 48 wires are available for routing. If a multiple-FPGA system requires more wires signals can be multiplexed as long as the overall data throughput doesn't exceed about 50 MHz x 48. But it is difficult to synchronize the multiplexer on the transmitter side with the demultiplexer on the receiver side, for two reasons:

1. FPGAs on BEE are programmed one-at-a-time. Therefore, subsystems on these FPGAs start at different times which is different from the Simulink behaviour.
2. The System Generator designs on a FPGA cannot be reset to their initial state right after programming.

Synchronization between multiplexer and demultiplexer has therefore to be taken special care of. The system described here (see Figure 1) provides a solution for this problem.

All necessary files for this system can be found in file:\\hit\designs\MCMA\Blocks\BEE\mux_test.

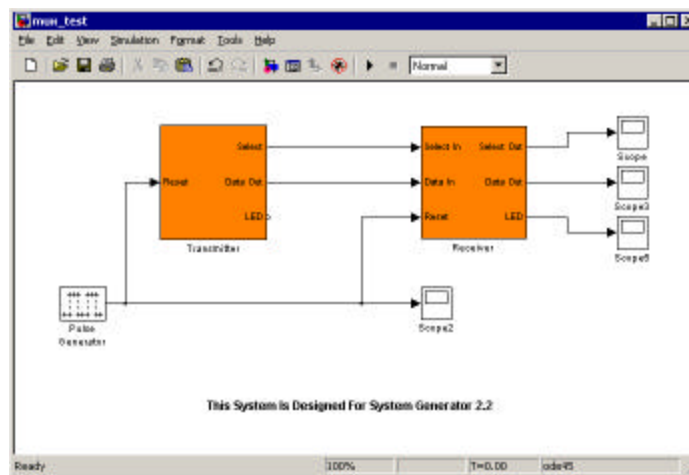


Figure 1: 2-FPGA System

The system is partitioned into two subsystems each of which has to be programmed on a specific FPGA on BEE. The assignment is:

- Transmitter on FPGA 14
- Receiver on FPGA 10

Both transmitter and receiver have been implemented with a ChipScope core.

2 Implementation

Figure 2 shows the transmitter side where the data is multiplexed. 'Counter1' and the shift register subsystem generate 8 parallel 4-bit signals with clock rate f_s . The data needs to be up-sampled to $8f_s$ because the 'Mux' block requires that all input signals have the same sampling rate. The signal transitions after the up-sampling block are still aligned to the rising edge of the slower clock f_s . In the 'Sync' block the data is latched if 'Counter' is 7. One clock cycle later 'Counter' wraps around and outputs signal 0 through 7 to the multiplexed bus.

Note:

The 'Sync' block is necessary because it is not possible to align the rising edge of f_s

exactly with the reset of 'Counter' by using an external reset. After starting a Simulink simulation they are aligned even across multiple FPGAs. After programming an FPGA they are also aligned within one FPGA but not across several FPGAs. After resetting the FPGA also the alignment within one FPGA is lost because the external reset cannot reset the clock generator core inserted by System Generator together with the 'Counter' reset. The necessary port is not accessible from the Simulink level.

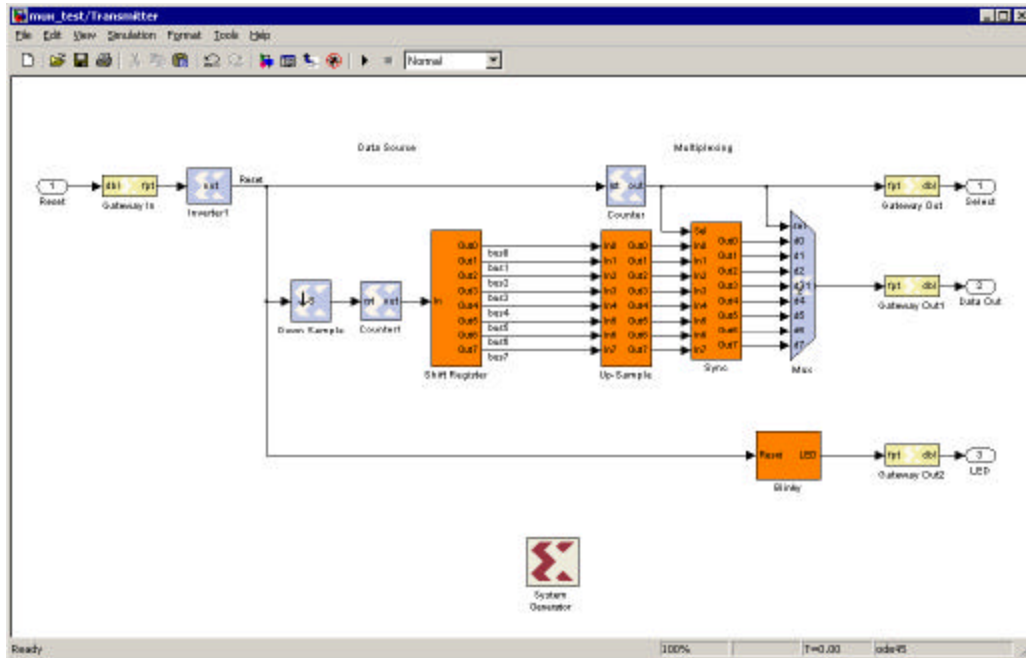


Figure 2: Transmitter Side

The 'Mux' select signal is transmitted to the receiver along with the multiplexed data. There it is used to demultiplex the signal and latch it synchronously to the higher clock rate $8f_s$ in the 'sync' block. The down-sample block latches it again with the slower clock rate f_s .

This 8x4 bit multiplexer implementation costs 58 slices. The demultiplexer implementation costs 95 slices. Multiplexer and demultiplexer can run at clock frequencies of about 100 MHz and 80 MHz, respectively.

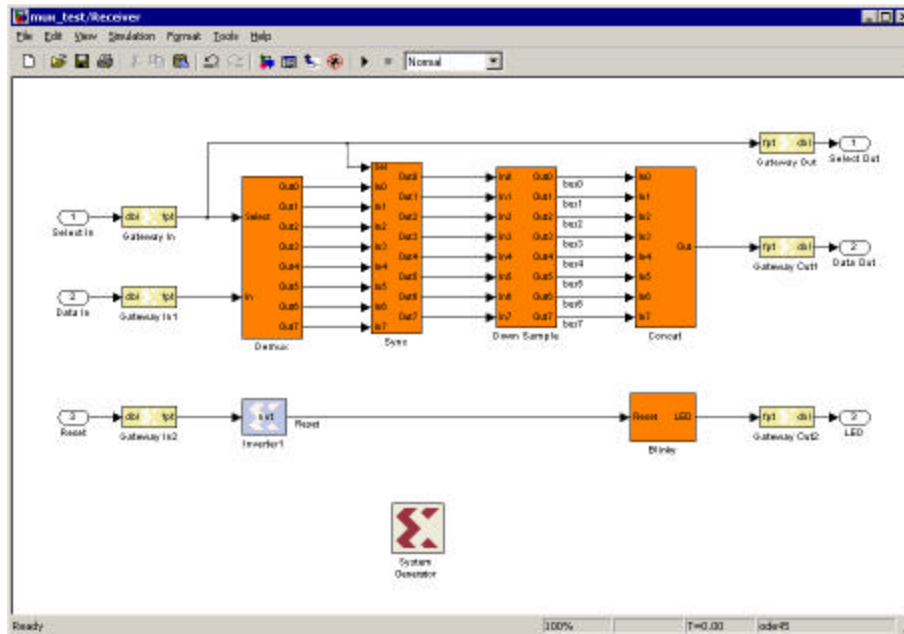


Figure 3: Receiver Side

3 Starting the System

To run the system do the following:

1. Log into bee_unit1
2. Make sure the files Tx.bit and Rx.bit are uploaded on bee_unit1
3. Set BEE clock frequency to e.g. 10 MHz
4. Program FPGA 14 with Tx.bit
5. Program FPGA 10 with Rx.bit

The User LEDs of the two FPGAs should be flashing now.

4 Looking Into The system With ChipScope

Each subsystem contains a ChipScope core which samples interesting signals inside the system. On the transmitter side these signals are

1. The 4-bit buses bus0 through bus7 before they enter the multiplexer
2. The multiplexed 4-bit output signal
3. The multiplexer select signal
4. External reset

On the receiver side the signals are

1. The 4-bit buses bus0 through bus7 after demux and down-sampling
2. The demux select signal
3. External reset

To connect to these cores log into the Stinger terminal server and run the ChipScope Analyzer. For each subsystem there is a project file stored in file:\hit\designs\MCMA\Blocks\BEE\mux_test\ChipScope.

If the project file is loaded before connecting to the respective core the waveform window shows the correct signal and bus names for the sampled signals.

