Creating Finite State Machines

Using True Dual-Port Fully Synchronous SelectRAM Blocks

Create very dense, high-performance, highly efficient designs that require no logic resources.

by Edgard Garcia
Senior Engineer, Multi Video Designs
edgard.garcia@mvd-fpga.com

The latest Virtex, Virtex-E, and Spartan-II FPGA families offer a broad range of unique features, including block RAM, that give you dramatic speed and density improvements. The dedicated RAM blocks allow you to build fast and dense bidirectional data buffers and FIFOs, with built-in data width conversion. This RAM can also be used to implement very fast and efficient sequencers and Finite State Machines (FSMs), which frees your logic gates for other tasks.

A well known approach to building sequencers consists of a ROM-based design with output registers. The same method can apply to the Virtex 4K-bit block SelectRAM™ which incorporates output registers. You can use a single 4K-bit RAM block as a 512 x 8 clocked ROM, to implement a very fast FSM working at more than 150 MHz, and it uses no CLBS. You can implement the following, for example:

- 16 states + 4 additional outputs and 5 inputs + Enable and Synchronous Reset.
- 32 states + 3 additional outputs and 4 inputs + Enable and Synchronous Reset.

Design Example

The following example shows how to implement FSMs or sequencers with a single 4K-bit block SelectRAM. The same method can easily be expanded to more complex designs that could require two or more blocks.

Synchronous FSMs and sequencers have some important characteristics in common:

- They are clocked by a single clock.
- A feedback path allows you to (partially) define what the next step will be.
- They may need a clock enable to suspend the operations.
- They must have a reset to go back to a predefined state.

Figure 1 shows a typical FSM or sequencer logic diagram.

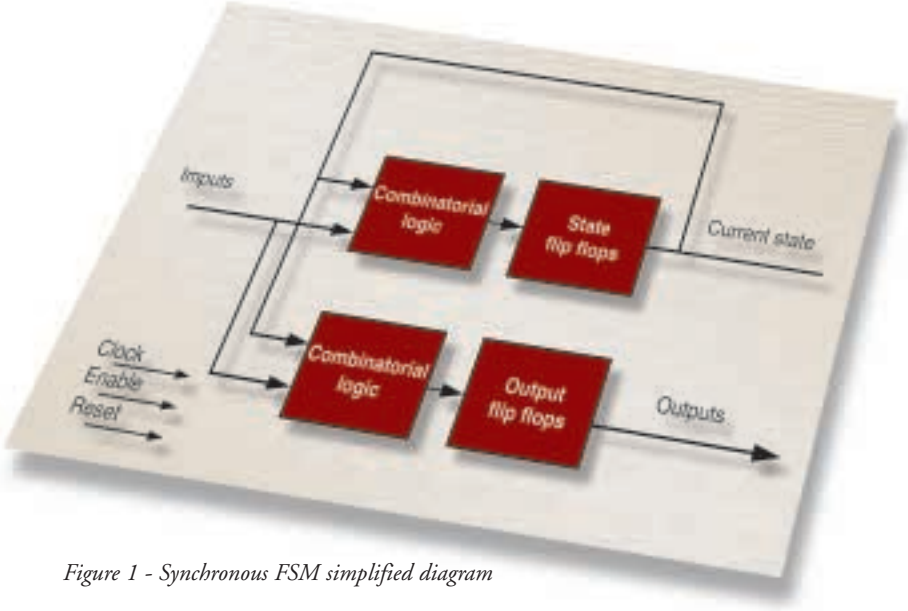

Figure 1 - Synchronous FSM simplified diagram

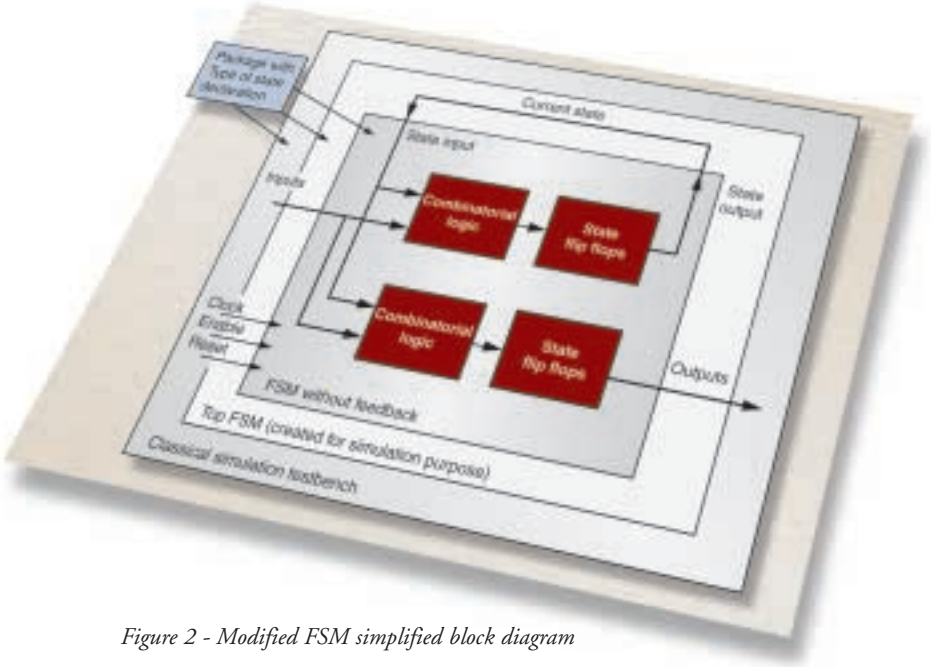

Figure 2 - Modified FSM simplified block diagram

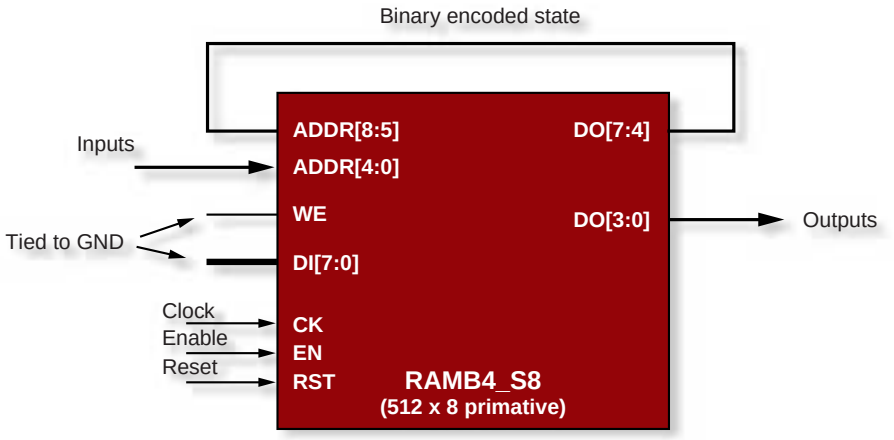

Figure 3 - Using fully synchronous RAM blocks for FSM implementation

Simulation

Converting the FSM behavior to a truth table can be a very tedious and time consuming task; debugging and modifying the design can turn into a nightmare.

The method used here takes advantage of the modern design entry, simulation, synthesis, and implementation tools, combining their complementary respective power. The goal is to use a VHDL simulator to automatically generate the constraint file for initialization of the block SelectRAM used as ROM.

One of the problems encountered when simulating a design with VHDL, is that an output can't be forced to any logic level. An easy way to avoid this inconvenience consists of breaking the feedback loop, which allows you to enter patterns into the inputs, including current and illegal states. Figure 2 illustrates one way of designing the FSM VHDL code so it can be simulated more easily. A top-level file can provide the feedback for a classical simulation of the FSM.

Optimization

If the number of inputs (including binary encoding states feedback) is nine or less, and the number of outputs (including binary encoding states) is eight or less, a single 4K-bit block SelectRAM can be applied by using structural VHDL with the scheme shown in Figure 3.

RAM Initialization

By initializing the contents of the memory with the appropriate values, the behavior of any synchronous FSM can be reproduced. The initialization of the RAM block is done by an NCF (constraint) file that will be used by synthesis and Xilinx implementation tools. A very easy way to initialize the memory with the correct values is to make an automatic generation of the NCF file, by using a VHDL simulator and another testbench.

Consider the behavioral VHDL code of the FSM without feedback. A simple 9-bit pseudo counter (generated by a testbench) can provide all the 512 possible states of the inputs, including illegal states. Each associated result (state output and FSM outputs) can

thus be converted to a text file obeying the NCF file format. See Figure 4.

Resources Required

Table 1 summarizes some examples of typical FSMs or sequencers in terms of performance and logic resources. All these designs can be implemented with a single RAM block, but the same method can easily expanded to more complex functions, providing similar improvement. As you can see, a RAM-based FSM is much faster and uses no logic resources.

True Dual-Port RAM Advantages

Block SelectRAM provides true dual port capability; a single location can be read at the same time by the two ports. Therefore, a single block can be used to implement two identical synchronous FSMs, with separate inputs, synchronous reset, and clock enable; you can also implement separate clocks, if needed. Figure 5 shows the architecture of a dual FSM using a single dual-port block SelectRAM. Table 2 shows a dual FSM implementation comparison.

Conclusion

The Virtex architecture provides powerful features and flexibility, allowing you to create very dense and high performance designs by taking advantage of the innovative features such as block SelectRAM. By combining the power of the Xilinx architecture and implementation tools, with the associated VHDL synthesizers and simulators, your design productivity is greatly improved, and complex designs can be easily implemented or modified.

For more information, please e-mail: Edgard.garcia@mvd-fpga.com

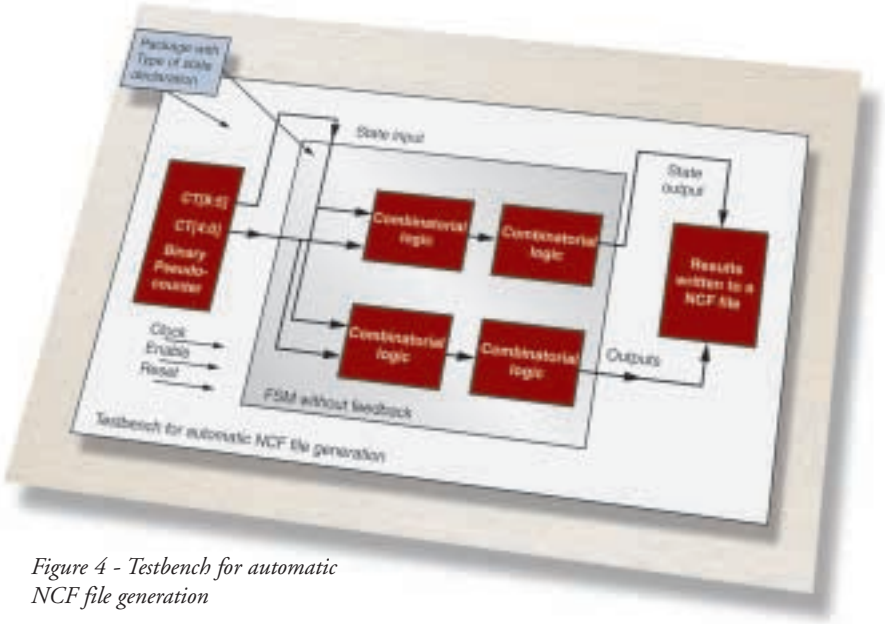

Figure 4 - Testbench for automatic NCF file generation

Implementation mode	Number of FFs	Number of Slices	RAM Blocks	Speed*
One Hot Encoding	20/35	30...60	–	80...110Mhz
Binary Encoding	8/8	25...50	–	70...100Mhz
SelectRAM block (binary encoding)	0	0	1	150 Mhz

• Results for a single 16 (or 32) state synchronous FSM with Enable, synchronous Reset, 5 (4) inputs and 4 (3) outputs.

Table 1 - Single FSM implementation comparisons

Implementation mode	Number of FFs	Number of Slices	RAM Blocks	Speed*
One Hot Encoding	40/70	60...120	–	80...110Mhz
Binary Encoding	16/16	50...100	–	70...100Mhz
SelectRAM block (binary encoding)	0	0	1	140 Mhz

*Dual 16 (or 32) states synchronous FSM with Enable, synchronous Reset, 5 (4) inputs and 4 (3) outputs.

Table 2 - Dual FSM implementation comparisons

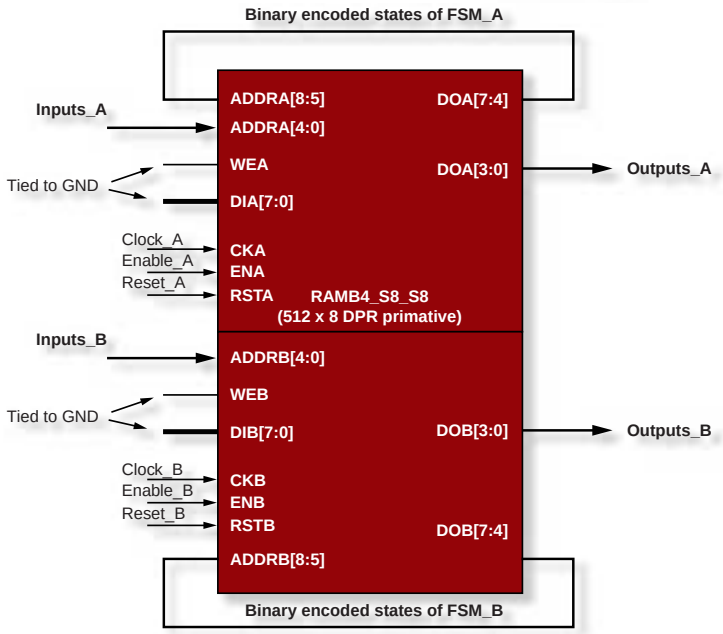

Figure 5 - Dual FSM implementation

About Multi Video Designs

MVD is a design and training center, specializing in Xilinx FPGA/CPLD designs and Hardware Description Languages. Consulting services and on site classes are offered in France, neighboring countries, and South America, in French, Spanish, and Portuguese. More information on our activity as well as the source code of some examples are available on our website, at: www.mvdfpga.com/training/VHDL_examples.htm