Moving the M8051Ewarp ASIC Core to a Virtex FPGA

Mentor Graphics implements their M8051Ewarp™ core in a Virtex FPGA.

by Kevin Rowley
Design Engineer, Mentor Graphics - IP division
kevin_rowley@mentorg.com

There is an ever increasing demand for digital IP cores that have been proven in FPGAs as well as ASIC test silicon. This is due to the obvious prototyping advantages and increasing market share of FPGAs. In this article I explain the strategy we used and the results we obtained when implementing the Mentor Graphics M8051Ewarp™ core, which was originally created for ASIC development, in Xilinx Virtex™ technology.

Test Strategy

The usual method we use to prove the functionality of the M8051Ewarp core in ASIC gates is to envelope the synthesized core in a wrapper which has instantiated all the necessary components for simulation of the core. These components typically are program memory, data memory, and extra peripherals. However, after FPGA place and route the core will have I/O pads on its interface.

The delays inherent in these pads preclude the usual ASIC test method and a new synthesizable wrapper is required which, after place and route, will have the M8051Ewarp core, memories, and required peripherals in a single netlist which can then be simulated. The synthesizable wrapper we used is illustrated in Figure 1.

The most important point about Figure 1 is that all memories are on-chip, which means there are no I/O pads between the M8051Ewarp core and its memory modules, and thus delays on inputs and outputs to and from memory are avoided. The test code for the core is contained in the Program ROM. Also shown are the data memory RAM block and the internal data memory (IRAM). The peripherals are the wait-state generator necessary for the test suite and the External Special Function Registers (ESFRS) also needed by the test suite.

Data captured from the wrapper is written to a simulation-listing file for comparison

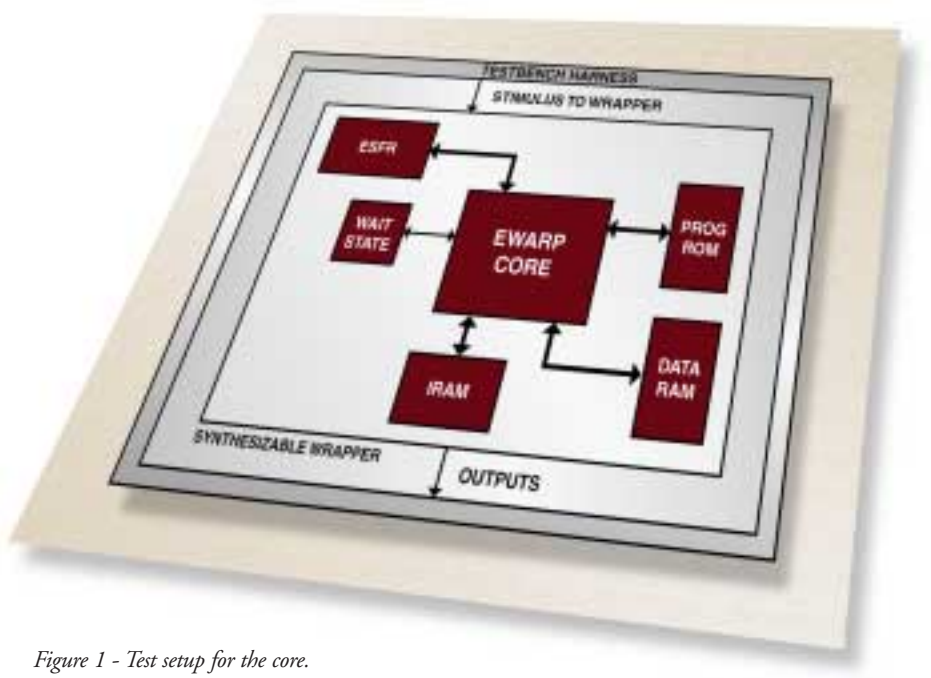

Figure 1 - Test setup for the core.

to reference listings. The test bench is a self-checking type which will setup and initialize the M8051Ewarp core and then check the program execution (using the test program stored in ROM).

Memory Requirements

Successful simulation of the system test suite required the following memory :

- 4k bytes of synchronous ROM.
- 1k bytes of synchronous single-port RAM.
- 256 bytes of dual-port synchronous RAM.

The CORE Generator tool from the Xilinx 2.1i Alliance Series software suite was used to design and generate the memory modules.

Specifying Memory Contents

The RAM cores had all their contents initialized to zero by the CORE Generator, however the ROM module had to have test opcodes stored in it for simuation. There are two ways of specifying ROM contents with the CORE Generator:

- .MIF file - Requires a line for each ROM location but also each byte has to be in binary format.
- .COE file - Allows the ROM locations to be specified in hex format.

Because the .COE format is closer to Intel hex format we decided to use a .COE file to specify the ROM. First however it was necessary to write a special program which would take the Intel hex format file for the test opcodes and convert it into a .COE file for the CORE Generator.

The final COE file looked like the following (abridged) :

```
Component_Name=crom;
Data_Width=8;
Depth=4096;
Radix=16;
Default_Data=0;
Memory_Initialization_Vector =
01,
80,
00,
c2,
a8,
...
```

The CORE Generator then generated an EDIF netlist with this .COE file for the ROM; it also generated EDIF netlists for the RAM modules. The EDIF netlists would be dragged-in later at the place and route stage for the wrapper.

Synthesis Strategy

We used Mentor Graphics Leonardo™ for synthesis. The target operating speed of the M8051Ewarp core in the Virtex FPGA was 30 Mhz. Therefore, the synthesis constraints were setup accordingly. The part we targeted was the XCV200BG352. To synthesize the wrapper we first synthesized the M8051Ewarp core separately and then read it in, during the synthesis of the wrapper, as a separate file.

Wrapper Synthesis

For synthesis, all the memory modules are treated as "black boxes." In addition to producing an EDIF file of the synthesized wrapper for place and route, synthesis also produced an .NCF file which Xilinx place and route uses to determine the timing constraints of the circuit. The timing analysis by Leonardo produced the following results:

```
Clock Frequency Report

Clock        : Frequency
-----------------------------------------------------------------

SCLK         : 33.8 MHz
CCLK         : 33.4 MHz
PCLK         : 33.4 MHz


Critical Path Report

There are no paths that violate user specified
options or constraints
And the expected area utilization report:

****************************************
Device Utilization for v200bg352
****************************************
Resource        Used  Avail Utilization
-----------------------------------------------------------------
IOs          97   260   37.31%
Function Generators  3653  4704  77.66%
CLB Slices       1827  2352  77.68%
Dffs or Latches    653  4704  13.88%
-----------------------------------------------------------------
```

This shows that the wrapper fitted into the XCV200 device and, according to Leonardo, was expected to run correctly at 30 Mhz.

Place and Route of the Wrapper

Because several runs were required to pass static timing during synthesis and place and route, it was necessary to automate the place and route stage using the following script :

```
ngdbuild -p v200bg352-6 ewarp_f.edf ewarp_f.ngd

map ewarp_f

par -d 1 -ol 5 -pl 5 -rl 5 -w ewarp_f ewarp_f_out.ncd ewarp_f.pcf

trce ewarp_f_out ewarp_f.pcf -v 3 -o ewarp_f_out

ngdanno ewarp_f_out ewarp_f.ngm

ngd2ver ewarp_f_out -w
```

The file ewarp_f.edf is the EDIF file for the wrapper after synthesis. Note "ngdanno" produces the SDF file which must be back-annotated with the FPGA netlist for gate-level simulation. The "ngd2ver" program produces a verilog netlist of the SimPrims primitives for verilog gate-level simulation. The FPGA device utilization figures we achieved are detailed in the output file from the "par" program:

```
Device utilization summary:

   Number of External GCLKIOBs   3 out of 4   75%
   Number of External IOBs      94 out of 260  36%

   Number of BLOCKRAMs          11 out of 14  78%
   Number of SLICEs           2073 out of 2352 88%

   Number of GCLKs               3 out of 4   75%
   Number of TBUFs              16 out of 2464  1%
```

The "trce" program listed static timing information and constraints applied for place and route. This produced a lot of violations on paths which upon closer inspection turned out to be multi-cycle path exceptions. This was because multi-cycle path exceptions setup for the M8051Ewarp synthesis were not included in the .NCF file generated from wrapper synthesis. Some effort was needed to examine all violating paths from "trce" and make sure they were path exceptions.

We found that the maximum speed with the netlist was 31.25 Mhz. Above this speed the setup time required for PROGA feeding into the ROM was being violated;

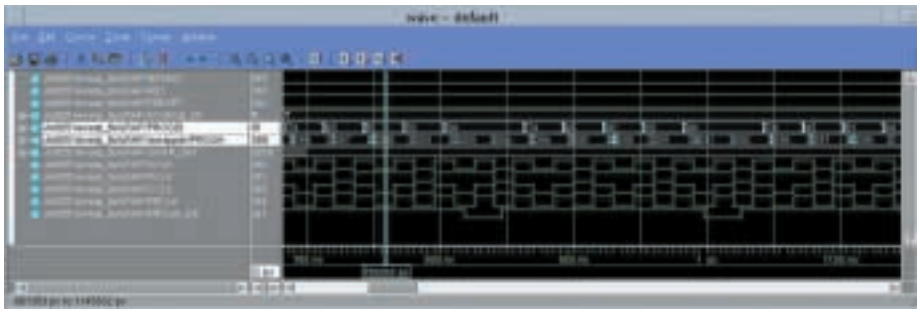

Figure 2 - Waveform view.

Verification

Illustrated in Figure 2 is a waveform view of the M8051Ewarp core cycling through the test code stored on ROM at 30 Mhz. Program opcode can be seen coming into M8051Ewarp core on the PROGDI input. The program address is shown in signal PROGA. The test code was setup by the COE file at ROM generation.

Running the complete simulation through 4k bytes of test code results in 12100 vectors. If the core has simulated correctly, it will finish at PROGA equal to FFF hex. If there has been a problem then the simulation will not reach program address FFFh and it will be stuck in a loop at an earlier program address and will terminate after a certain time-out period.

The output delays on some of the M8051Ewarp core output ports exceeded the strobe period at 30 Mhz. Usually core outputs are strobed out to a listing file twice per clock cycle, the strobe period being just less that half a clock cycle. Therefore, since some output delays exceeded the strobe period it was not possible to do a straight unix "diff" between the Virtex netlist simulation listing and the reference listing for this test. However inspection of the listing file from simulation and comparison with the assember code listing for the test showed the circuit to be functioning correctly.

PROGA was becoming valid too late before the next clock cycle.

Conclusion

We successfully placed and routed a complete M8051Ewarp core plus memory and peripherals on a Xilinx Virtex device and got it to work at speeds up to 31.25 Mhz. The key to our success was using the on-chip memory of the Xilinx part, and using the Xilinx CORE Generator software to design the memories. Synthesis was performed by Leonardo and the post-synthesis EDIF netlist of the M8051Ewarp wrapper was placed and routed by Xilinx 2.1i Alliance Series software. The resulting verilog netlist and SDF file, after place and route, ran through the test suite successfully at the required speed.

For more information on the M8051Ewarp core see the Mentor Graphics website at: www.mentor.com/inventra/8051e_warp.html

References

[1] ASIC/FPGA market share , www.xilinx.com, June 2000