Odin – A Verilog RTL synthesis tool for heterogeneous FPGAs

Peter Jamieson and Jonathan Rose

The Edward S. Rogers Sr. Department of Electrical and Computer Engineering - University of Toronto

Introduction

Modern heterogeneous FPGAs (like that pictured in Figure 1) contain “hard” specific-purpose structures such as memories and multipliers in addition to the completely flexible “soft” programmable logic and routing. These hard structures provide major benefits, and reduce the logic density gap between Field-Programmable Gate Arrays (FPGAs) and Application Specific Integrated Circuits (ASICs) from 35 times to 18 times [3].

Our work focuses on how to efficiently use these hard structures from both an architectural and CAD mapping perspective.

The focus here is presenting our public-domain open source Verilog RTL Synthesis tool, called *Odin*, and the algorithms that allow it to flexibly target different heterogeneous structures. Our goals for this work are:

1. Front-end tool to map to hard circuits on FPGA
2. Achieve comparable results to Industrial Front-end Synthesis
3. Deliver open source for academic community

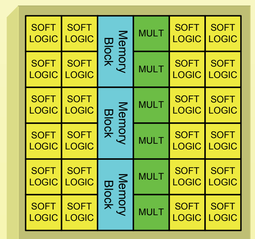

Figure 1

Front End Synthesis and CAD flow

Odin, takes a Verilog design as input and converts it into a flattened structural netlist. This conversion process includes elaboration (Icarus [2]) and partial-mapping stages, where the final netlist consists of primitive gates and complex logic functions targeting a heterogeneous FPGA.

This netlist can be passed into various CAD flows including Quartus, ModelSim, and a VPR CAD flow, although the latter cannot include hard circuits. Figure 2 shows both how Odin is included as part of an FPGA CAD flow and the major stages of the Odin tool.

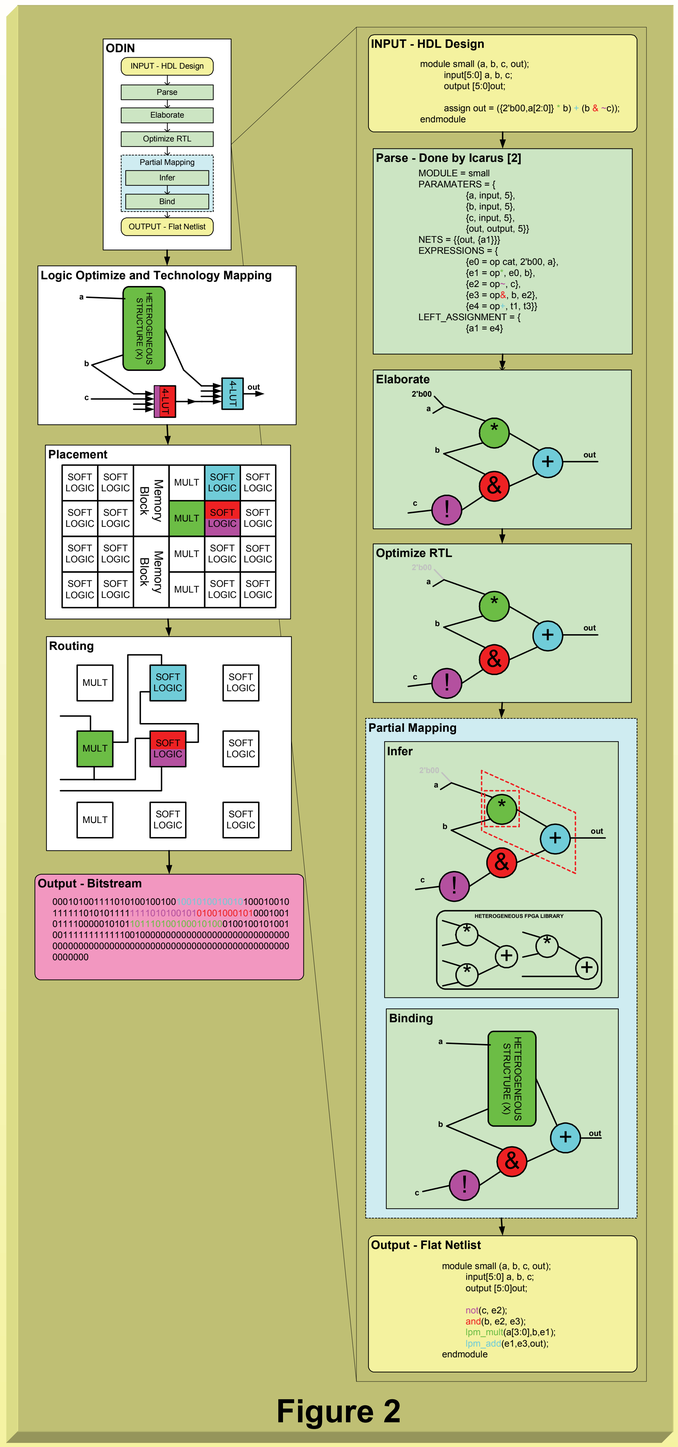

Figure 2

Mapping Techniques

To create a tool that achieves quality of results comparable to an industrial front-end synthesis tool, we achieve this using several key optimizations.

These include:

1. Partial Mapping (Figure 2)
2. Arithmetic optimizations (Figure 3)
3. Finite state machine re-encoding to one-hot
4. Multiplexer collapsing (Figure 4)

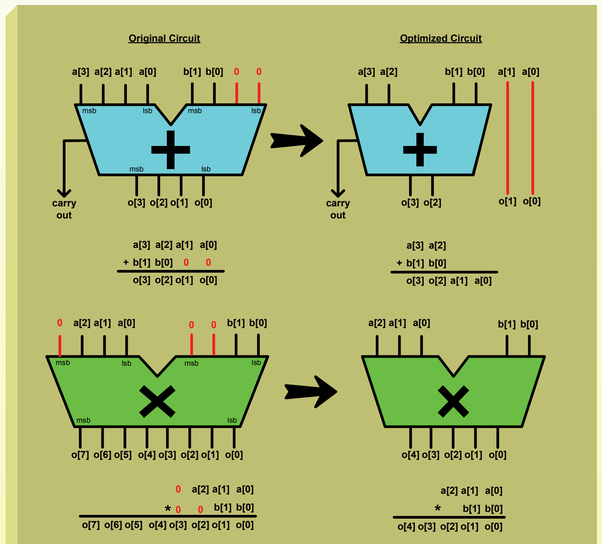

Figure 3

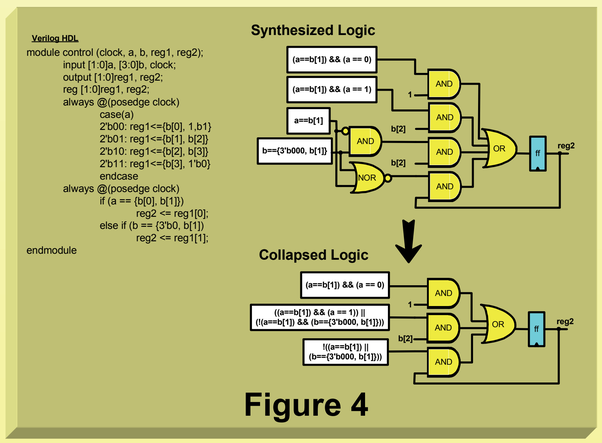

Figure 4

Results

Ran our benchmarks through both a full Quartus flow and Odin joined with Quartus [1] to compare area and speed results. Table 1 shows a summary of our results on the Stratix I FPGA, and the results are very close!

	Number of Logic Elements			Speed in MHz		
Designs	**A** - Quartus	**B** - Odin with Quartus	Ratio (**B/A**)	**E** - Quartus	**F** - Odin with Quartus	Ratio (**E/F**)
fft_258_6	2374	3190	1.34	101.76	146.16	0.70
iir1	289	501	1.73	84.99	82.53	1.03
iir	297	338	1.14	115.50	109.16	1.06
fir_3_8_8	84	84	1.00	251.48	251.93	1.00
fir_24_16_16	1598	1591	1.00	83.62	75.04	1.11
fir_scu_rtl	998	1099	1.10	149.75	117.63	1.27
diffeq_f_systemC	221	271	1.23	45.08	41.11	1.10
diffeq_paj_convert	512	369	0.72	37.67	29.86	1.26
sv_chip1	17765	17145	0.97	116.42	122.31	0.95
sv_chip2	35554	36194	1.02	48.17	49.74	0.97
sv_chip2_no_mem	34379	33803	0.98	52.99	56.56	0.94
rt_raygentop	2622	2679	1.02	134.86	127.24	1.06
rt_raygentop_no_mem	2118	2815	1.33	134.33	136.92	0.98
rt_top	25056	28653	1.14	45.15	52.05	0.87
rt_top_no_mem	21557	21317	0.99	47.78	53.60	0.89
oc45_cpu	2191	2969	1.36	86.43	62.94	1.37
reed_sol_decoder1	1151	1186	1.03	86.07	85.14	1.01
reed_sol_decoder2	1799	2037	1.13	68.26	57.74	1.18
md	10542.6	14867	1.41	41.98	34.94	1.20
cordic_8_8	591	838	1.42	212.12	256.44	0.83
cordic_18_18	2830	4104	1.45	166.95	222.72	0.75
MAC1	2864	2759	0.96	107.04	91.60	1.17
MAC2	9828	9625	0.98	82.92	65.76	1.26
CRC33_D264	102	102	1.00	0.00	0.00	0.00
des_area	1481	1523	1.03	235.31	194.37	1.21
des_perf	4592	5924	1.29	199.56	199.44	1.00
sv_chip0	12433	12729	1.02	NA	120.16	NA
sv_chip0_no_mem	7281	7122	0.98	162.81	146.20	1.11
sv_chip3_no_mem	170	134	0.79	321.00	328.94	0.98
rt_frambuf_top	546	729	1.34	120.15	129.79	0.93
rt_frambuf_top_no_mem	766	880	1.15	124.80	123.09	1.01
rt_boundtop	1519	2266	1.49	187.20	134.28	1.39
		Average	1.12		Average	1.04

Conclusions

Odin is an open source front-end synthesis tool that uses a few optimization techniques to closely approach the quality of results for front-end industrial tools.

References

1 Altera, *Quartus II Handbook, Volumes 1, 2, and 3*, 2004.

2 “ICARUS Verilog at www.icarus.com/eda/verilog/.”

3 I. Kuon and J. Rose, “Measuring the gap between FPGA s and ASICs,” in *FPGA’06*, Feb 2006, pp. 21-30

For further information

Please contact *jamieson@eecg.toronto.edu*

Download:http://www.eecg.toronto.edu/~jayar/software/odin/index.html

Acknowledgments

ROGERS ELECTRICAL COMPUTER ENGINEERING UNIVERSITY of TORONTO

NSERC CRSNG