Tuning of Highly Configurable Architectures

David Sheldon, Frank Vahid*,

Department of Computer Science and Engineering
University of California, Riverside
* *Also with the Center for Embedded Computer Systems at UC Irvine*

This work was supported in part by the National Science Foundation, the Semiconductor Research Corporation, and by hardware and software donations from Xilinx

UNIVERSITY OF CALIFORNIA RIVERSIDE

Current Platforms

- Soft-core FPGA processors
  - Microblaze
  - Nios
- Hard-core processors
  - Tensilica
  - Arm
- Other domains
  - Cache
  - FPGA architecture
  - Platform tuning

Current Methods for customization

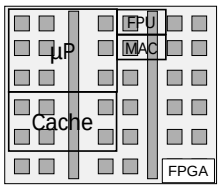
- Very few commercial tools for tuning of these architectures
- What tools that do exist are targeted for only a specific problem
  - (only tuning on the MCORE cache)

Previous Work

- Most of the research in this area is targeted at a specific type of device.
  - Givargis, T., F. Vahid. Platune: A Tuning Framework for System-on-a-Chip Platforms.
  - Kumar, R., Dean Tullsen, and Norman Jouppi. "Core Architecture Optimization for Heterogeneous Chip Multiprocessors".
  - Moyer, B., Tune *Multicore Hardware for Software*.
  - Mohanty, S., Prasanna, V. K., Neema, S., and Davis, J. Rapid design space exploration of heterogeneous embedded systems using symbolic search and multi-granular simulation.
  - Sekar, K., Kanishka Lahiri, Sujit Dey. Dynamic Platform Management for Configurable Platform-Based System-on-Chips.
  - Sheldon, D., R. Kumar, R. Lysecky, F. Vahid, D. Tullsen. Application-Specific Customization of Paramaterized FPGA Soft-Core Processors.
  - Yiannacouras, P., J. G. Steffan, and J. Rose. Application-Specific Customization of Soft Processor Microarchitecture.

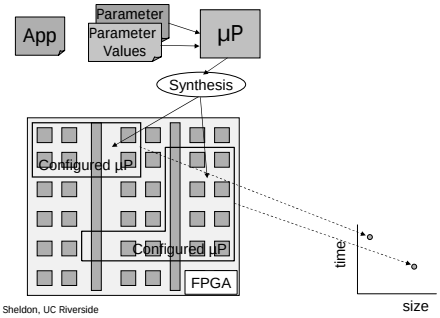
FPGA Soft Core Processors

- Soft Core Processors can have configurable options
  - Datapath units
  - Cache
  - Bus architecture
- Current commercial FPGA Soft-Core Processors
  - Xilinx Microblaze
  - Altera Nios

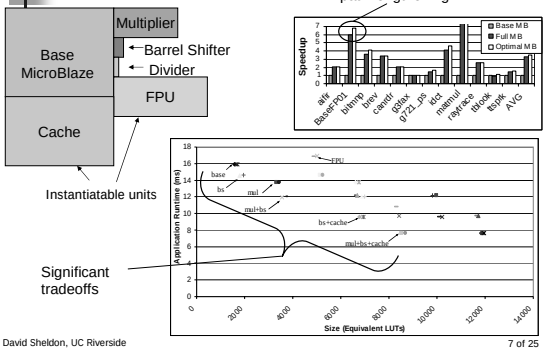


Goal

- Goal: Tune FPGA soft-core microprocessor for a given application

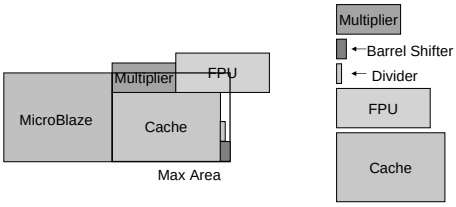


Microblaze – Xilinx FPGA Soft-Core



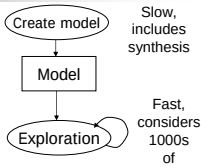
Constraints on Configurations

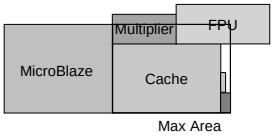
- Size constraints may prevent use of all possible units



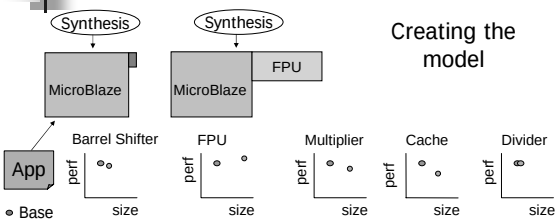
Approach 1: Traditional CAD Techniques

- Create a model of the problem
- Solve model with extensive search heuristics
- We will model this problem as a 0-1 knapsack problem





Approach 1: Traditional CAD Techniques

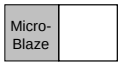


	BS	FPU	MUL	DIV	CACHE
Perf increment	1.1	0.9	1.2	1.0	1.3
Size increment	1.4	2.7	1.8	1.1	1.6
Perf/Size	0.96	0.34	0.63	0.93	0.80

Approach 1: Traditional CAD Techniques

- 0-1 knapsack model
  - Object's benefit = Unit's performance increment / size increment
  - Object's weight = Unit's Size
  - Knapsack's size constraint = FPGA size constraint

	BS	FPU	MUL	DIV	CACHE
Perf increment	1.1	0.9	1.2	1.0	1.3
Size increment	1.4	2.7	1.8	1.1	1.6
Perf/Size	0.96	0.34	0.63	0.93	0.80



Approach 1: Traditional CAD Techniques

- Solved the 0-1 knapsack problem using established methods
  - Toth, P., Dynamic Programming Algorithms for the Zero-One Knapsack Problem. Computing 1980
- Running time
  - 6 Microblaze configuration synthesis runs to create model
  - O(n*p) to solve model
    - n is the number of factors
    - p is the available area
    - Negligible (seconds) compared to synthesis runtimes (~hour)

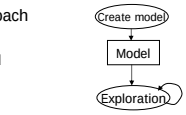
Approach 1: Traditional CAD Techniques

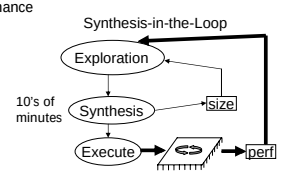
- Problems
  - 100's of target FPGAs
    - Different hard core resources (multiplier, block RAM)
  - Model approach *estimates* size and performance for two or more units
    - MUL speedup 1.3, DIV speedup 1.6 → estimate MUL+DIV speedup 1.9
    - May really be 1.7
  - Model inaccuracies may be large

Device	LUTs	PPCs
XC2V3000	21504	0
XC2VP2	2816	0
XC4VLX80	71680	0
XC4VLX15	12288	0
XC2S300E	6140	0
XC2V4000	46080	0
XC2VP40	38784	2
XC4VSX25	20480	0
XC4VSX55	30720	0
XC4VFX20	17088	1
XC2S150E	3456	0
XC2VP30	27392	2
XC4VLX60	53248	0
XC2S600E	13824	0
XC2VP20	18560	2
XC2V500	6144	0
XC2VPX70	66176	2
XC4VLX40	36864	0
XC2V6000	67584	0
XC4VF160	50560	2
XC4VFX100	84352	2
XC2VP4	6016	1
XC2VP70	66176	2

Approach 2: Synthesis-in-the-Loop

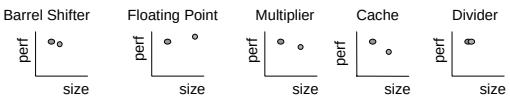
- Problem with traditional CAD approach
  - 100's of target FPGAs
  - Model approach *estimates* size and performance for two or more units
  - Model inaccuracies may be large
- Solution – Synthesis in the loop
  - No abstract model
  - Guided by actual size and performance data
  - But slow – can only explore a few configurations





Approach 2: Synthesis-in-the-Loop

- First pre-analyze units to guide heuristic
  - Same calculations as when creating model for knapsack



	BS	FPU	MUL	DIV	CACHE
Perf increment	1.1	0.9	1.2	1.0	1.3
Size increment	1.4	2.7	1.8	1.1	1.6
Perf/Size	0.96	0.34	0.63	0.93	0.80

Approach 2: Synthesis-in-the-Loop

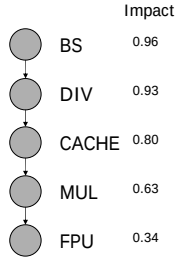
- Build "impact-ordered tree" structure
  - Tree is specific to given application

	BS	FPU	MUL	DIV	CACHE
Perf/Size	0.96	0.34	0.63	0.93	0.80

Sort

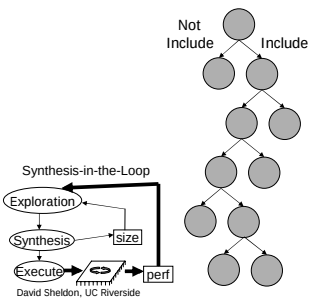
	BS	DIV	CACHE	MUL	FPU
Perf/Size	0.96	0.93	0.80	0.63	0.34

Application Specific Impact-ordering



Approach 2: Synthesis-in-the-Loop

- Run tree-based search heuristic



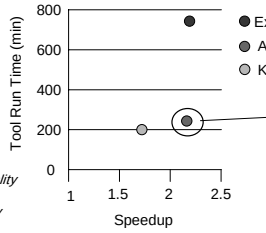
	Perf/Size	Useful
BS	0.96	Yes
DIV	0.93	No
CACHE	0.80	No
MUL	0.63	Yes
FPU	0.34	No

Comparison of Approaches

- Approach 1 – Traditional CAD
  - 6 synthesis runs to build model
  - O(np) knapsack solution
  - Examines thousands of configurations during exploration
- Approach 2 – Synthesis in the loop
  - 11 synthesis runs (6 pre-analysis, 5 exploration)
  - Examines (at most) 5 configurations during exploration

Results

- 10 EEMBC and Powerstone benchmarks
  - aifir, BaseFP01, bitmnp, brev, canrdr, g3fax, g721_ps, idct, matmul, tblook, ttsprk
  - Average results shown, on Virtex 2 Pro, for particular size constraint

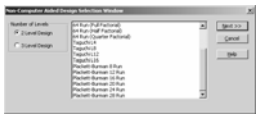


Application-specific impact-ordered tree approach yields near-optimal results in acceptable tool runtime

Knapsack sub-optimality due to multi-unit estimation inaccuracy

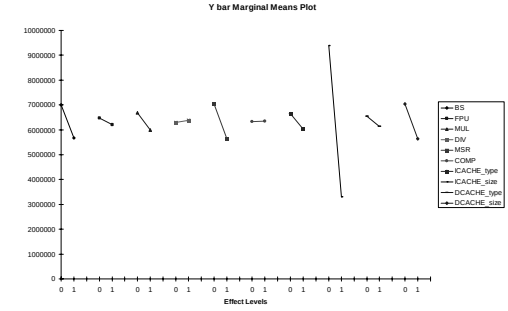
Design of Experiments (DoE)

- Tool selected set of runs
- 12 runs
- Both the main effects and the interactions are now known



Factor	A	B	C	D	E	F	G	H	I	J		Cycles
Row #	BS	FPU	MUL	DIV	MSR	COMP	ICACHE_type	ICACHE_size	DCACHE_type	DCACHE_size		Y1
1	0	0	0	0	0	0	0	0	0	0		12696265
2	0	0	0	0	0	0	1	1	1	1		3544216
3	0	0	1	1	1	0	0	0	1	1		8262214
4	0	1	0	1	1	0	1	1	0	0		3808647
5	0	1	1	0	1	1	0	1	0	1		2860019
6	0	1	1	1	0	1	1	0	1	0		10818171
7	1	0	1	1	0	0	1	1	0	1		2644509
8	1	0	1	0	1	1	1	0	0	0		8046171
9	1	0	0	1	1	1	0	1	1	0		3601399
10	1	1	1	0	0	0	0	1	1	0		3308450
11	1	1	0	1	0	1	0	0	0	1		9208946
12	1	1	0	0	1	0	1	0	1	1		7276392

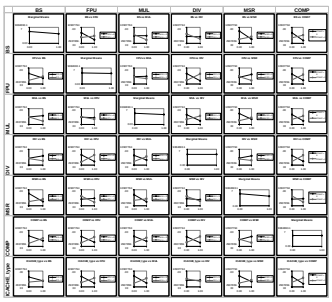
Design of Experiments (DoE)



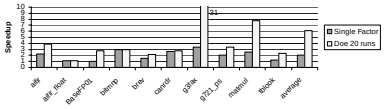
Regression Anaylsis

Y-hat Model		Cycles				Equiv LUTs			
Factor	Name	Coeff	P(2 Tail)	Tol	Active	Coeff	P(2 Tail)	Tol	Active
Const		6339616.6	0.0311			54650.5	0.0024		
A	BS	-658838.8	0.2799	1	X	331.17	0.3485	1	X
B	FPU	-126179.1	0.7538	1	X	1859.92	0.0707	1	X
C	MUL	-349694.3	0.4616	1	X	1083.50	0.1172	1	X
D	DIV	51031.1	0.8961	1	X	62.667	0.8083	1	X
E	MSR	-697142.9	0.2663	1	X	-0.83333	0.9974	1	X
F	COMP	6870.42	0.9859	1	X	217.33	0.4765	1	X
G	ICACHE_type	-316598.9	0.4931	1	X	-403.83	0.2851	1	X
H	ICACHE_size	-3045076.6	0.0546	1	X	3805.33	0.0337	1	X
I	DCACHE_type	-204476.3	0.6286	1	X	-186.17	0.5257	1	X
J	DCACHE_size	-706900.6	0.2630	1	X	3601.67	0.0356	1	X
R^2		0.9913				0.9987			
Adj R^2		0.9046				0.9861			
Std Error		1073326.1934				699.1712			
F		11.4319				79.1900			
Sig F		0.2265				0.0872			
F_{LOF}		NA				NA			
Sig F_{LOF}		NA				NA			
Source		SS	df	MS		SS	df	MS	
Regression		131699005162031.0	10	13169900516203.1		387112792.7	10	38711279.3	
Error		1152020917366.1	1	1152020917366.1		488840.3	1	488840.3	
$Error_{Pure}$		NA	0	NA		NA	0	NA	
$Error_{LOF}$		NA	0	NA		NA	0	NA	
Total		132851034279391.0	11			387601633.0	11		

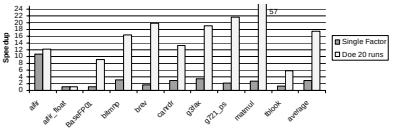
Interaction Plots



- 10,000 equiv LUT size constraint



- 30,000 equiv LUT size constraint



Conclusions

- Design of Experiments (DoE) is able to find a very good solution to configuring a Soft-core processor.
  - No domain knowledge is required to create this experiment
  - Easy to define and run with a proper tool package.
- This technique can be applied to many different domains of tuning a given architecture.