



## Hybrid FPGAs - Architecture, Tools and Floating Point Applications

*Philip Leong<sup>1,2</sup>, Chun Hok Ho<sup>2</sup>, Chi Wai Yu<sup>2</sup>, Wayne Luk<sup>2</sup>, Steve Wilton<sup>3</sup>*

<sup>1</sup> Department of Computer Science and Engineering, Chinese University of Hong Kong

<sup>2</sup> Department of Computing, Imperial College London

<sup>3</sup> Department of Electrical and Computer Engineering, University of British Columbia

28 August 2007

1

## Overview

1. Introduction
2. Virtual Embedded Blocks
3. Hybrid FPGAs
4. Conclusion

2

## Introduction

- Fixed point computation dominant in FPGA designs due to speed and efficiency
- Floating point FPGAs suitable for DSP and HPC applications
- Increased parallelism and higher memory bandwidth
- What modifications are required to FPGA architectures for floating point?

3

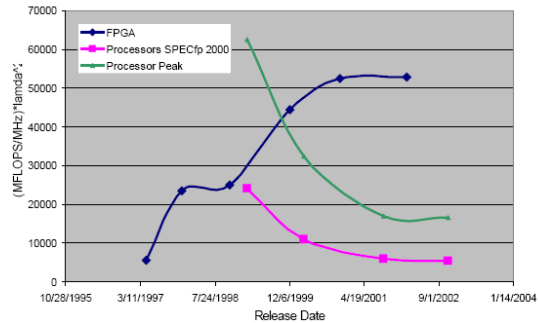
## Key results

- VEB method for studying embedded blocks
- Domain-specific hybrid FPGA architecture
  - reconfigurable resources: multiple granularity
  - customised for different applications
  - model based on current FPGAs and CAD tools
- Hybrid FPGA for floating point applications
  - parameterised coarse-grained block
  - 6 Benchmarks, compared with Virtex-II device
  - 18 times area reduction, 2.5 times speedup

4

## Computational Density

Normalized Performance Comparison



January 12, 2004

BWRC, UC Berkeley 5

## Microprocessors/DSPs vs FPGAs

### FPGA Strengths

- More parallelism
- Higher computational density
- Lower power consumption
- Higher memory bandwidth, direct control of accesses
- Can be fault tolerant

### Weaknesses

- Long wordlengths
- Floating point
- Low clock frequency
- Run out of resources
- Design time
- Legacy code

6

## Floating point FPGAs

### Commercial FPGA Weaknesses

- Long wordlengths
- Low clock frequency
- Run out of resources
- Design time

### Domain-specific FPGA for FP

- Hardwired FPUs
- Reconfigurable fabric
- Dynamic Reconfiguration

### Advantages

- More transistors for computing than a uP
- Better FP perf than FPGA/uP
- Development time reduced as designers do not need to deal with fixed point quantisation issues
- External memory often bottleneck, FPGAs offer potentially higher bandwidth (multiple channels) as well as custom control of cache
- Branch mispredictions don't cause tens of cycles to recover

7

## Overview

1. Introduction
2. Virtual Embedded Blocks
3. Hybrid FPGA
4. Conclusion

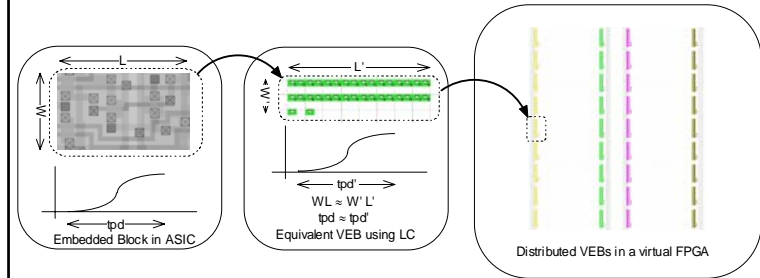
8

## Virtual Embedded Blocks

- Use existing tools to study effects of embedded elements in FPGAs
- Use commercial design flows
  - Closer to real architectures
  - Optimisations such as retiming
  - Greatly reduced development time
- Explore technology trends based on systematic variation of VEB parameters in applications.

9

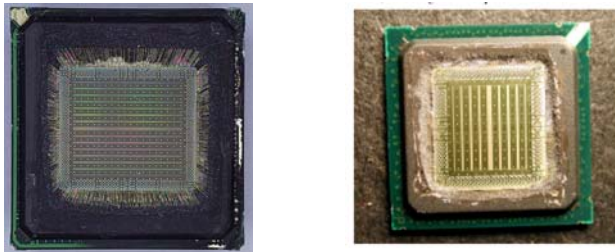
## Virtual Embedded Blocks: VEB



- Dummy blocks
  - model coarse-grained unit's area and delay
- Timing analyser
  - estimate performance, e.g. fine-to-coarse delay

## Area Model

- Use logic cells to model real ASIC embedded blocks
- Estimate LC area (normalised to feature size) from die photos



11

## Area Model

- Estimates of logic cell area including configuration bit, buffer and interconnect overheads.
- Area based on estimate that 70% of the total die area for logic cells, the other area being for pads, block memories, multipliers etc.
- Normalised to feature size

Device	LCs/CLB $L$	Area/CLB $A (\mu\text{m}^2)$	Feature Size $f (\mu\text{m})$	Normalised LC area $(N = A/Lf^2)$
Apex 20K400E [8]	10	63161	0.18	195,000
Virtex E [8]	4	35462	0.18	267,000
Virtex II 3000 [9]	8	$71,429 \times 0.7$	0.12	434,000
Virtex II 1000 [10]	8	$72,782 \times 0.7$	0.12	442,000

12

## Area Model

- Area model can be used to compare logic cell area to any embedded block
  - Logic Cell (LC): 442, 000 = 1 LC
  - Multiplier: 2, 751, 000 (normalised) ~ 6 LC
  - Blue-gene floating point unit (FPU) ~ 570 LC

13

## Delay Model

- Match delays using LC
  - Use adder carry chain to model the delay
- For small blocks, may fail to match both area and delay

14

## Benchmarks

- Large unsigned multiplier
  - Formed by cascading embedded block multiplier
  - 34-bit (mul34), 68-bit (mul68), 136-bit (mul136)
- Brace, Gatarek and Musiela (BGM) framework
  - Price interest rate derivative using Monte Carlo simulation
  - A good example for retiming because of its complexity
- Fly compiler
  - Allow a single description to generate a circuit with different arithmetic operations
  - Applications include
    - Digital sine cosine generator (dscg)
    - ordinary differential equation solver (ode)
    - 3x3 matrix multiplication (mm3)
    - Finite impulse response filter (fir4)
    - Butterfly stage for discrete Fourier transform (bfly)
- 18-bit fixed point and double precision floating point circuits generated

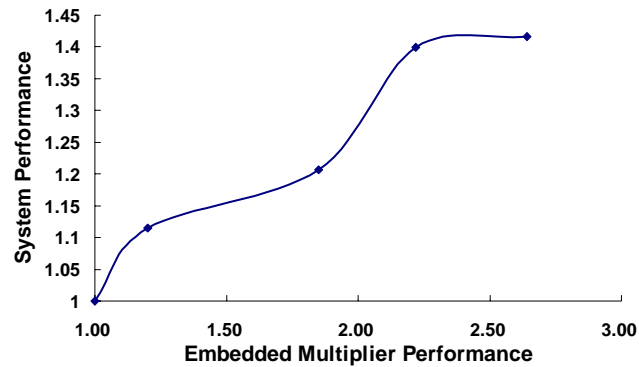
15

## Verification using Embedded Multipliers

	EM delay (ns)	VEB delay (ns)	Overall Diff (%)
DSCG	4.599	4.981	8
FIR4	4.616	4.794	2
ODE	4.402	4.539	3
MM3	4.859	4.815	1
BFLY	5.668	5.224	8
MUL34	11.191	11.287	1
MUL68	12.553	14.099	11
MUL136	14.632	13.248	10
BGM	14.055	13.866	1
BGM (retimed)	11.594	11.602	0

16

## Effect of Faster Multipliers (BGM)



17

## Overview

1. Introduction
2. Virtual Embedded Blocks
3. Hybrid FPGA
4. Conclusion

18

## Floating Point FPGAs

- Simplest architecture would be island-style FPGA+FPU
  - Measure its performance using VEB approach
- Improved method is a hybrid FPGA

19

## Embedded FPU

- For comparison with hybrid FPGA
- Embedding a floating point unit
  - FPU delay and area based on published Blue Gene data
  - 700MHz,  $4.26 \text{ mm}^2 = 570 \text{ LCs}$
  - For FPGAs, reduce latency and clock frequency by a factor of 5: 140 MHz, one cycle latency
- Explore the speedup by increasing the performance of floating point unit
  - Tested on floating-point butterfly circuit (bfly)

20

## Benchmark Circuits

- 6 benchmark circuits
  - DSP computation kernels: e.g. bfly
  - linear algebra: e.g. matrix multiplication
  - complete application: e.g. bgm
- Circuits: partitioned to control + datapath
  - control: vendor tools to fine-grained units
  - datapath: manually map to coarse-grained units
- Comparison
  - directly synthesized to Xilinx Virtex-II devices

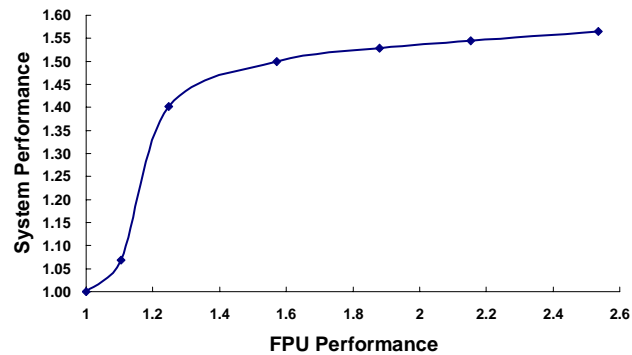
21

## Effect of FPU on System Performance

	FPGA		VEB		Reduction Factor	
	size (LC)	delay (ns)	size (LC)	delay (ns)	size	delay
dscg	19006	22.711	3420 + 940	8.807	4.4	2.6
fir4	20590	23.545	3990 + 996	9.539	4.1	2.5
ode	13984	17.756	2850 + 870	8.525	3.8	10.4*
mm3	17236	19.320	2850 + 2390	8.587	3.3	11.3*
bfly	25640	20.245	4560 + 3424	8.821	3.2	2.3
Geometric Mean:					3.7	4.4

22

## System Perf vs FPU Perf



23

## Observations

- Embedded FPUs: 4x speed and density
- Retiming important (20% diff in speed)
- Inefficiencies when 1-bit LUTs used to process 64-bit datapaths
- Coarser-grained blocks may offer higher performance

24

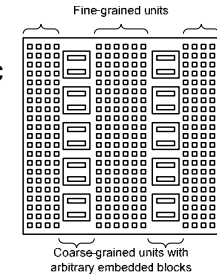
## Domain-specific Hybrid FPGAs

- Reconfigurable resources of multiple granularity
- Parameterised
- Domain-specific

25

## Hybrid FPGA: architecture

- Most digital circuits
  - datapath: regular, word-based logic
  - control: irregular, bit-based logic
- Hybrid FPGA
  - customised coarse-grained block: domain-specific requirements
  - fine-grained blocks: existing FPGA architecture
  - good match to computing applications for a given domain



26

## Analysis

- Modelling
  - synthesizable coarse-grained fabric (VEB)
- Evaluation
  - place & route benchmarks on hybrid FPGA
  - measure speed and area
- Exploration
  - cover large design space of architectures

27

## Floating point hybrid FPGA

- Coarse-grained units
  - dominated by multiplication and addition
  - IEEE double precision floating point
  - 64-bit datapaths much more efficient: shared routing and configuration, specialised logic
- Fine-grained units
  - implement control logic, state machine
  - based on Xilinx Virtex-II

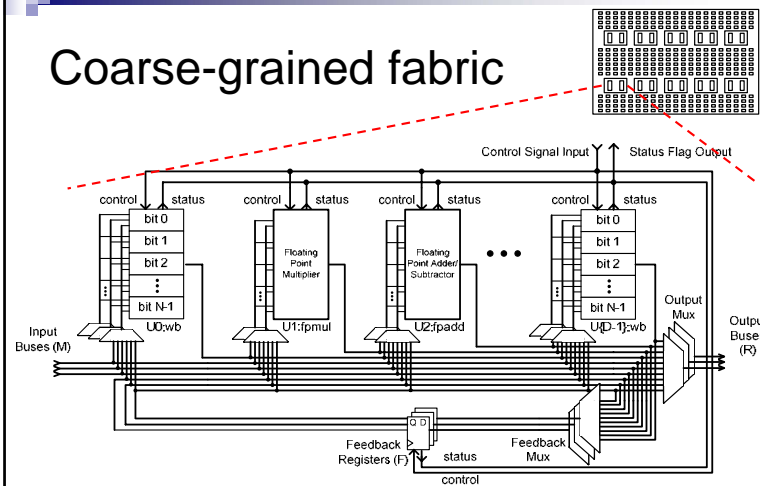
28

## Coarse-grained fabric design

- Coarse-grained block synthesized
  - ASIC delay/size parameters from standard cells
- HDL allows parameterisation of architecture
  - number of embedded floating point operators
  - number of feedback registers, etc

29

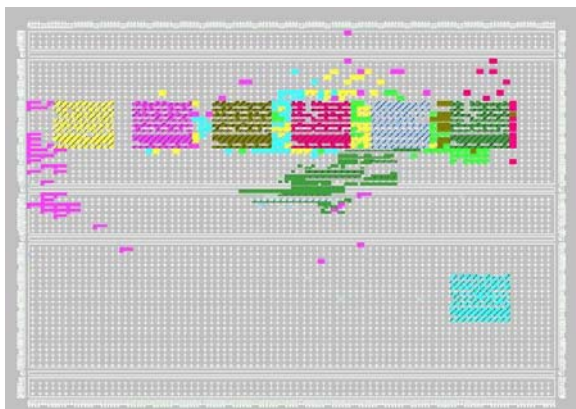
## Coarse-grained fabric



D=9, M=4, R=3, F=3, 2 add, 2 mul: best density over benchmarks

30

## Example floorplan: bgm



31

## Hybrid FPGA Results

	Floating Point hybrid FPGA		XC2V3000-6		Area (times)	Delay (times)
	Area (slices)	Delay (ns)	Area (slices)	Delay (ns)		
bfly	565	9.02	13733	24.57	<b>24.3</b>	<b>2.72</b>
dscg	661	10.11	9614	22.78	<b>14.5</b>	<b>2.25</b>
fir4	371	9.06	11290	23.68	<b>30.4</b>	<b>2.61</b>
mm3	642	8.90	8889	23.4	<b>13.8</b>	<b>2.63</b>
ode	545	9.74	8238	21.93	<b>15.1</b>	<b>2.25</b>
bgm	1810	10.00	30207	24.34	<b>16.7</b>	<b>2.43</b>
				Geometric Mean	<b>18.3</b>	<b>2.48</b>

32

## Conclusion

- Domain-specific hybrid FPGAs
  - fine-grained + synthesizable coarse-grained units
  - explore customisations using existing FPGA tools
- Domain-specific floating point FPGA
  - 18 times area reduction
  - 2.5 times speedup
  - greatly improved area over island-style+FPU

33

## Future work

- Estimating routing overhead
- Explore different coarse-grained units for
  - floating point operations
- Automated design tools
  - mapping to coarse-grained units
  - partitioning between coarse and fine-grained units
- Other domain-specific applications
  - scientific computing
  - imaging
  - networking

34

## References

- C.H. Ho et. al. "Domain-specific hybrid FPGA: architecture and floating point applications." *FPL*, pp. 196–201, 2007.
- C.H. Ho et. al. "Virtual embedded blocks: A methodology for evaluating embedded elements in FPGAs." *FCCM*, pp. 35–44, 2006.

35