Resource to Performance Tradeoff Adjustment for Fine-Grained Architectures —A Design Methodology

Fahad Islam Cheema, Zain-Ul-Abdin,
Professor Bertil Svensson

Halmstad University, Halmstad, Sweden
Fahad.Cheema@hh.se

Overview

- Motivation
  - Computation intensive algorithms
  - Fine grained architectures
- Problem Definition
  - Resource to Performance Tradeoffs
    - Hardware/logic gates to performance tradeoffs
    - Memory to performance tradeoffs

Experimental Setup

- Computation intensive algorithms
  - Interpolation Kernels
- Fine Grained Architecture
  - FPGA
- Fine Grained Parallelism
  - Mitrion virtual processor
    - Extract fine grained parallelism
  - Mitrion-C high level language (HLL)
- Hardware Platform
  - Cray XD1 with Vertex-4

3

Mitrion Parallel Architecture

- Mitrion Virtual Processor (MVP)
  - Fine-Grained, Soft-Core Processor
  - Almost 60 IP blocks defined in HDL [1]
  - Non von-neuman architecture
- Mitrion-C
  - HLL for FPGA
  - Data dependence instead of order-of-execution
  - Parallelism Language Constructs [2]
  - Pipelining

4

Interpolation Kernels

- What is interpolation
  - Process of calculating new values within the range of available values [3]
- Cubic interolation
- Bicubic interpolation
  - Applying cubic in 2D
  - 5 cubic kernels

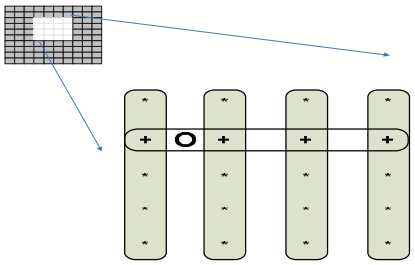

Figure-1: 2D Interpolation

5

Main Idea

- Parallelism Levels
  - Bit Level Parallelism (BLP)
  - Kernel Level Parallelism (KLP)
  - Problem Level Parallelism (PLP)
- Mitrion SDK extracts maximum possible parallelism within kernel
  - Based on data dependence

```
#define x_int  2
P03_lv = for(i in <0 .. 10>)
{
    int:16 d0 = x_int - x0;
    int:16 d1 = x_int - x1;
    int:16 d2 = x_int - x2;
    int:16 d3 = x_int - x3;
    int:16 p01 = (y0*d1 - y1*d0)  / (x0 - x1);
    int:16 p12 = (y1*d2 - y2*d1)  / (x1 - x2);
    int:16 p23 = (y2*d3 - y3*d2)  / (x2 - x3);
    int:16 p02 = (p01*d2 - p12*d0) / (x0 - x2);
    int:16 p13 = (p12*d3 - p23*d1) / (x1 - x3);
    int:16 p03 = (p02*d3 - p13*d0) / (x0 - x3);
}p03;
```

Block 1

Figure-2: Cubic interpolation kernel

6

Main idea (Conti.)

- Maximum parallelism at one level is not ultimate solution
  - Customized parallelism at different levels
  - Can better adjust Resource-performance tradoffs
    - Gates-performance tradeoff
  - Parallelization Levels
    - Single Kernel (SKZ)
    - Cross Kernel (CKZ)
    - Multi-SKZ
    - Multi/CKZ

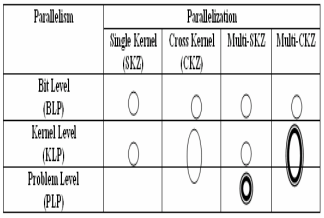

Figure-3: Parallelism and Parallelization Levels

7

Parallelization Levels

- Single Kernel Parallelization (SKZ)
  - Only kernel level parallelism (KLP)
  - Manually define data independent block

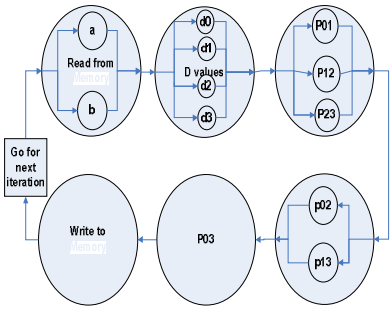

Figure-4: SKZ

8

Parallelization Levels (Conti.)

- Cross Kernel Parallelization (CKZ)
  - Extend kernel by Mixing more than one kernels
  - Replicate computation intensive data independent blocks
  - Resource computation balance

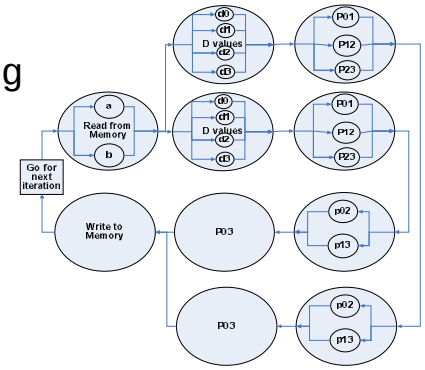

Figure-5: CKZ

Parallelization Levels (Conti.)

- Multi-SKZ
  - Replicate kernels which already have SKZ

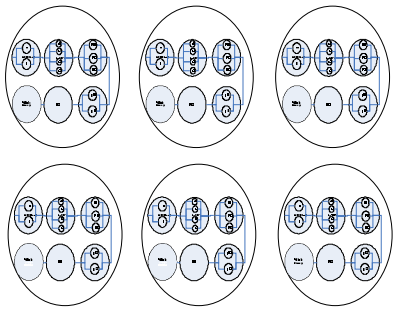

Figure-6: Multi-SKZ

Parallelization Levels (Conti.)

- Multi-CKZ
  - Replicate kernels which already have CKZ

Figure-6: Multi-CKZ

11

Results

Design Approach	Execution Time (ms)	Flip Flops (%)	# Memory banks	p	r	k	m	# Points computed in Parallel
Non-equidistant Cubic								
SKZ	10	20	1	6	-	-	-	1
CKZ	2.5	68	4	6	6	4	-	4
Multi-SKZ	1. 53	73	1	6	-	-	8	< 8
Multi-CKZ	0.19	74	4	6	4	4	20	< 80
Equidistant Bi-cubic								
SKZ	10	4	1	8	-	-	-	1
CKZ	2.5	16	4	8	8	4	-	4
Multi-SKZ	0.33	43	1	8	-	-	47	< 47
Multi-CKZ	0.17	55	4	8	6	4	24	< 96

Table-1 : Results

12

Conclusions

- Specific conclusions
  - For very limited resources, SKZ is better
  - CKZ is better for applications with high unbalanced computation distribution
  - SKZ and CKZ are better for large size applications
  - Multi-CKZ can provide high level of parallelism at cost of design complexity
  - Multi-SKZ and Multi-CKZ are attractive for small size Real-Time applications
- Using parallelization levels
  - Can adjust trade-offs
  - Can achieve highly custom parallelism
- Mix of parallelization levels can produce
  - Application-specific parallelism
  - Resource-specific parallelism

Future Work

- Automation of parallelization levels
- Parallelization levels to deal with other tradeoffs
- Generalized parallelization levels for all application
- Generalized parallelization levels for graphical processors to adjust tradeoffs
  - Floating point and accuracy

References

[1] Stefan Möhl, “The Mitrion Virtual Processor, Using FPGAs in HPC” Sixteenth ACM/SIGDA International Symposium on FPGAs <http://www.ece.wisc.edu/~kati/fpga2008/fpga2008%20workshop%20-%2005%20Mitrionics%20-%20Mohl.pdf> Date 14-05-2009

[2] “Mitrion User Guide”, Copyright © 2005 - 2008 by Mitrionics AB. <http://www.mitrionics.com/?page=developers_resources> Date 03-03-2009

[3] William H. Press Brian P. Flannery, Saul A. Teukolsky William T.Vetterling “Numerical Recipes, Art of Scientific Computing”, Cambridge University Press

15

Tack

(Hope you enjoyed)