Reconfigurable Computing

ICEMI 2013, Harbin China

18 August 2013

Philip Leong 梁恒惠 (philip.leong@sydney.edu.au)
School of Electrical and Information Engineering,
The University of Sydney

Wang Shaojun 王少军 (wangsj@hit.edu.cn)
School of Electrical Engineering and Automation
Harbin Institute of Technology
Introduction to Reconfigurable Computing

› FPGAs
› Reconfigurable computing
› Applications
› Case studies
Most electronics rely on application-specific ICs (ASICs) for perf, cost and P
A generalised ASIC
- Logic blocks for digital operations
- Programmable interconnect for routing
Arbitrary digital circuits can be implemented
Functionality downloaded to FPGA memory (in seconds)

Source: Steve Wilton (UBC)
FPGA Embedded Blocks

Source: Xilinx

Hard IP blocks for widely-used functions: faster, more efficient, lower power
Careful choice: every user must pay for these functions, whether used or not
### FPGA Families

Xilinx 7-series FPGAs, 28nm

<table>
<thead>
<tr>
<th>Feature</th>
<th>740</th>
<th>1,920</th>
<th>3,600</th>
</tr>
</thead>
<tbody>
<tr>
<td>DSP Slices</td>
<td>740</td>
<td>1,920</td>
<td>3,600</td>
</tr>
<tr>
<td>DSP Performance (symmetric FIR)</td>
<td>930GMACs</td>
<td>2,845GMACs</td>
<td>5,335GMACs</td>
</tr>
<tr>
<td>Transceiver Count</td>
<td>16</td>
<td>32</td>
<td>96</td>
</tr>
<tr>
<td>Transceiver Speed</td>
<td>6.6Gb/s</td>
<td>12.5Gb/s</td>
<td>28.05Gb/s</td>
</tr>
<tr>
<td>Total Transceiver Bandwidth (full duplex)</td>
<td>211Gb/s</td>
<td>800Gb/s</td>
<td>2,784Gb/s</td>
</tr>
<tr>
<td>Memory Interface (DDR3)</td>
<td>1,066Mb/s</td>
<td>1,866Mb/s</td>
<td>1,866Mb/s</td>
</tr>
<tr>
<td>PCI Express® Interface</td>
<td>x4 Gen2</td>
<td>x8 Gen2</td>
<td>x8 Gen3</td>
</tr>
<tr>
<td>Analog Mixed Signal (AMS)/XADC</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Configuration AES</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>I/O Pins</td>
<td>500</td>
<td>500</td>
<td>1,200</td>
</tr>
</tbody>
</table>
Zynq (ARM+ Reconfigurable Fabric)
ASIC vs FPGA Cost

Crossover volume increases with decreasing feature size.
ASIC Development Costs Today

Sources: Altera
Return on Investment Analysis

**New Product Business Plan Version 1.0**

- **ASIC / ASSP (65nm) Development Cost**: $55M
- **20% of Revenue on R&D**
- **Revenue Target**: $275M
- **$10 to $50 Device ASP**
- **Unit Volume Req’d**: 5Mu to 27Mu

Very Few High Volume Applications Justify ASIC / ASSP Development

Source: Altera
Typical High Performance Commercial Applications

**Application**
- Optical Transport OTU Transponder
- 40GbE/100GbE Switch
- Radar

**Requirements**
- >350 MHz performance
- 28 Gbps transceivers
- 10GBASE-KR backplane support
- High-performance on-chip memory
- High-performance and flexible memory controller
- Hard system-level IP for bandwidth
- High precision DSP

**Solution**

**Process:** 28HP
- >350 MHz performance
- Lowest power in its class
- Up to 1.1M LEs on a monolithic die

**Transceiver:** 14.1 Gbps/28 Gbps

**Product Architecture:**
- Soft memory controller supports 800MHz DDR3 DIMM
- 2,560 M20K memory blocks
- 54x54 variable precision DSP

**System IP:**
- PCIe Gen3 x8, 40 GbE/100 GbE, Interlaken

Source: Altera
FPGAs allow Dynamic Reconfiguration
Although a general-purpose CPU can do anything, its power consumption is poor.

A general-purpose CPU handles light control tasks.

The VME handles heavy work that would consume power efficiently.

Transforms into optimal special-purpose circuits while running

Therefore, it is flexible and low power!
Flexibility vs Performance

The Age Requires Programmability
Product diversification and multifunctionality
Multiformat support

Flexible hardware
The hardware configuration can be modified by software

Sony provides stamina

- Good
- Optimal special-purpose HW
- VME
- CPU + VME
- Stamina
- Power efficiency
- Performance
- Bad

Programmability (flexibility)
Architectural Choices

- General-purpose processor
  - M
  - M$
  - ALU
  - Bus

- Dedicated accelerators
  - M
  - $P
  - AC1
  - AC2
  - Bus

- Application-specific processor
  - M
  - M1
  - M2

- Reconfigurable processor
  - M
  - RC1
  - RC2
  - RC3

Source: Rabaey UCB
Approximately three orders of magnitude in inefficiency from general-purpose to dedicated!

Benchmark numbers @ 1999

- **Embedded Processors**
  - SA110: 0.4 MIPS/mW

- **DSPs**
  - 3 MOPS/mW

- **ASIPs**

- **Reconfigurable Processor/Logic**
  - Pleiades: 10-80 MOPS/mW

- **Dedicated HW**

Source: Rabaey UCB
FPGA vs DSP and CPU Cost Comparison

Berkeley BEE2 cost comparison (FPGA, DSP1, DSP2, uP)
Traditionally designed using ASIC development tools
- VHDL/Verilog very low level

Recent advances
- Vivado HLS
- OpenCL

Extensive module generators and libraries e.g. filters, fft, floating-point, maths coprocessors, soft processors, network controllers, memory controllers, I/O controllers …

Still an active research topic
Comparison of FPGAs with uP and ASIC

› Compared with uP and DSP
  - higher speed, lower power, smaller variance in execution time
  - Longer development times, higher cost per unit

› Compared with ASICs
  - Lower initial cost

› Rides Moore’s Law, development costs amortised over users
  - Faster time to market, lower risk
  - Can be customised to problem in ways not possible with ASICs
Overview

› FPGAs
› Reconfigurable computing
› Applications
› Case studies
Reconfigurable Computing

› Application of FPGA devices to computing problems
FPGAs allow computational problems to be accelerated through:
- Parallelism
- Customisation
- Integration
- Do what would take many cycles on uP in fewer cycles (instruction level parallelism)
- Do many independent tasks/threads/processes in parallel (multiprocessor)
- Tradeoff latency with throughput by doing things in stages (pipelining)
Parallelism Example

- Microprocessor: data passed sequentially to computing unit
- FPGA & ASIC: spatial composition of parallel computing units (multiple muls, pipelining)
- E.g. 4-tap FIR filter, FPGA 1 output per cycle, uP takes multiple cycles
- Lower power and higher speed

Source: DeHon “The Density Advantage of Configurable Computing”
More specific functions can be implemented more efficiently

Too expensive to design ASIC to perform very specialised function

FPGAs can be heavily customised due to their programmability i.e. only do one thing efficiently

- Tradeoffs between speed and accuracy can be exploited, on uP, only get single or double; char, short or long

- General operators can be replaced with specific ones

E.g. Chip which only encrypts for a specific password
Networking, chip IO and computation on same device
- Reduction of buffering can help latency
- Single chip operation massive interconnect within chip exploited
- Multiple (small) memories within FPGA offer enormous memory bandwidth
Overview

› FPGAs
› Reconfigurable computing
› Applications
› Case studies
› Vehicle Control Module uses Virtex-II devices
  - gearbox, differential, traction control, launch control and telemetry
› High speed real-time control and DSP application

Source: BMW Williams
Compact Muon Solenoid

- $10^{15}$ collisions per second
- Few interesting events ~ 100 Higgs events per year
- 1.5Tb/s real-time DSP problem
- More than 500 Virtex and Spartan FPGAs used in real-time trigger

Source: Geoff Hall, Imperial College
Square Kilometre Array (SKA) will be one of the largest and most ambitious international science projects ever devised (€1.5 billion).

CSIRO Developing Australian SKA Pathfinder (ASKAP), a $150M next-generation radio telescope using FPGA technology for the data collection & processing

Source: CSIRO
Other RC Applications

› Applications suited to acceleration
  - seismic processing astrophysics FFT
  - adaptive optics (transforming to frequency domain and removing telescope image noise)
  - biotech applications such as BLAST, Smith Waterman and HMM
  - financial modeling
  - electronic measurement and instruments

› Functions well suited to FPGA acceleration
  - searching & sorting
  - signal processing (audio/video/image manipulation)
  - encryption
  - error correction
  - coding/decoding
  - packet processing
  - random-number generation for Monte Carlo simulations

Source: cray.com
Overview

› FPGAs
› Reconfigurable computing
› Applications
› **Case studies**
Challenges for modern instruments:

1. Powerful data processing
2. Lower response times
3. Updatable
4. Low power consumption
The benefits of FPGAs:
1. Implementing custom functionality
2. Faster I/O response times
3. Rapid prototyping and verification
4. Eliminating the expense of re-design and maintenance
5. Exceeds the computing power of digital signal processors
Case Study A: FPGA for Instrument Design

- FlexRIO from NI
- ZT8440 Series RF/IF Digitizer
- M9703A AXIe digitizer from Agilent
- High Speed Subsystem from Teradyne
Case Study A: FPGA for Instrument Design

FlexRIO from NI

Adapter modules: analog and digital I/O
FPGA modules: digital signal processing and onboard DRAM
Case Study A: FPGA for Instrument Design

FPGA-Based Feedback Control

Real-time 3D OCT Imaging System

Source: NI
Data from Automatic Test Systems (ATS)

Data processing → Fault Prediction & Diagnosis

**AR(p) Model**

\[ X_t = c + \sum_{i=1}^{p} \varphi_i X_{t-i} + \varepsilon_t. \]

**MA(q) Model**

\[ X_t = \varepsilon_t + \sum_{i=1}^{q} \theta_i \varepsilon_{t-i} \]

**ANN**

**SVM**

PHM: Prognostic and Health Management
Case Study B: PHM with FPGAs

PHM of Aircraft: More than 2000 Prognostic Candidates!

- Actuator Leakage and Wear
- Engine
- Power and Cooling: Turbo Machine Life, Oil Condition, Oil Servicing and Filter Condition
- Heat Exchangers
- Generator Oil Level: Hydraulic Filters, Pump, and Hydraulic Fluid Level
- Battery
- Oxygen Generator
- Nitrogen Generator and Filter
- Landing Gear and Arresting Hook: Structure fatigue life
- Landing Gear Strut: Pressure and Fluid Level
- Rotary Actuator Wear
- Nose Wheel Stability
Challenges in Computing

- **High Efficiency Computing**
  - Real-Time, On-line, Low Latency
  - Large Data sets or Data Streams
  - Computationally Intensive Algorithms

- **Low Power Embedded Computing**
  - Energy Consumption Limit
  - Space Saving

- **Flexible Computing**
  - Algorithm Update
  - Parameter Resetting
The pseudo code of the SW-KRLS algorithm

Initialize $K_0 = (1+c)I$ and $K_0^{-1} = I/(1+c)$.

for $n = 1, 2, \ldots$ do
  Get $\tilde{K}_n$ from $K_{n-1}$ with Eq.(1)
  Calculate $\tilde{K}_{n-1}$ with Eq.(2)
  Get $K_n$ with Eq.(3)
  Calculate $K_n^{-1}$ with Eq.(4)
  Get the updated solution $\alpha_n = K_n^{-1}Y_n$
end for

Computation complexity: $O(N^2)$. 

\[ \tilde{K}_n = \begin{bmatrix} K_{n-1} & k_n(x_n) \\ k_n(x_n)^T & k_{nn} + c \end{bmatrix} \]  

(1)

\[ \tilde{K}_{n-1} = \begin{bmatrix} K_{n-1}^{-1}(I + bb^T K_{n-1}^TG) & -K_{n-1}^{-1}bg \\ -(K_{n-1}^{-1}b)^T g & g \end{bmatrix} \]  

(2)

where $b = k_{n-1}(x_n)$, $d = k_{nn} + c$, $g = (d - b^T K_{n-1}^{-1}b)^{-1}$

\[ K_n = \begin{bmatrix} k_{n-N,n-N} + c & p^T \\ p & \tilde{K}_{n-1} \end{bmatrix} \]  

(3)

where $p = [k(x_{n-N}, x_{n-N+1}), \ldots, k(x_{n-N}, x_{n-1})]^T$

\[ K_n^{-1} = G - \frac{ff^T}{e} \]  

(4)

where $\beta_n^{(i)} = \begin{bmatrix} e \\ f \\ G \end{bmatrix}$
Case Study B: PHM with FPGAs

FPGA Based Microcoded SW-KRLS Vector Processor

Altera Stratix V FPGA

Source: Dr. Pang Yeyong
### Case Study B: PHM with FPGAs

<table>
<thead>
<tr>
<th>Platform</th>
<th>Power (mW)</th>
<th>Latency (μS)</th>
<th>Energy (10^-5 J)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Our processor</td>
<td>26880</td>
<td>28</td>
<td>75</td>
</tr>
<tr>
<td>FPGA soft processor (NIOS II)</td>
<td>15120</td>
<td>58428</td>
<td>88344</td>
</tr>
<tr>
<td>DSP (TI TMS320C6748)</td>
<td>2025</td>
<td>54926</td>
<td>111123</td>
</tr>
<tr>
<td>CPU (Intel)</td>
<td>36818</td>
<td>238</td>
<td>876</td>
</tr>
</tbody>
</table>

- Implementation of a time series prediction in single precision floating point on Altera Stratix V FPGA with 127-sample window (recent paper under review available on request)
- This is a compact implementation designed for embedded applications. A version which can operate at network speed is in development.
Conclusion

- uPs are the most flexible technology but performance (speed and power) is relatively low
- FPGAs provide
  - Easy interfacing with hardware (tighter coupling than GPUs)
  - Parallelism
  - Have become large enough to implement DSP and ML algorithms
- Are very suited to implementing data acquisition, digital signal processing and machine learning in electronic measurement and instruments