Multi-core RISC Processor Design & Implementation Demonstration Viva

Ben Lancaster

201280376 ELEC5881M - Main Project

July 7, 2019

B. Lancaster

Introduction

Top Leve Design

Multi-core Functionalit

Results

- 1 Introduction
 Why Multi-core?
 Why RISC?
- 2 Top Level Design
 Overview
 Memory Map
 Interconnect
 Interrupts
- Multi-core Functionality HW/SW Requirements Context Identification

- 4 ResultsResults 1
- 5 Conclusion
 Accomplishments
 Future Improvements

B. Lancaster

Introduction Why Multi-core?

Why RISC?

Design

Multi-core Functional

Results

nesuits

. . .

1 Introduction Why Multi-core? Why RISC?

2 Top Level Design

Multi-core Functionality

4 Results

Why Multi-core?

Main Project

B. Lancaster

Introduction
Why Multi-core?

Why RISC?

Design

Function

Results

Conclusion

Block Title

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Why RISC?

Main Project

B. Lancaster

Introduction
Why Multi-core?
Why RISC?

Top Level Design Multi-core

Functionality

Results

B. Lancaster

1 Introduction

Introduction

Top Level

Design Overview

Overview Memory Map

nterconnect

Multi-core

Functionality

Results

Conclusion

2 Top Level Design

Overview

Memory Map

Interconnect

Interrupts

3 Multi-core Functionality

4 Results



Main Project

B. Lancaster

Introduction

Top Leve Design Overview

Memory Ma Interconnec

Multi-core

Results

Conclusion

What this project produces:

System-on-Chip with multi-processor functionality
 Tested on FPGA hardware with 1-96 CPU cores.

Main Project

B. Lancaster

Introduction

Top Level Design Overview Memory Map Interconnect

Multi-core Functionalit

Results

Conclusion

What this project produces:

- System-on-Chip with multi-processor functionality
 Tested on FPGA hardware with 1-96 CPU cores.
- Custom 16-bit RISC CPU
 With interrupts and its own Instruction Set Architecture (ISA).

Main Project

B. Lancaster

Introductio

Design
Overview
Memory Map
Interconnect
Interrupts

Multi-core Functionality

Results

Conclusion

What this project produces:

- System-on-Chip with multi-processor functionality
 Tested on FPGA hardware with 1-96 CPU cores.
- Custom 16-bit RISC CPU
 With interrupts and its own Instruction Set Architecture (ISA).
- Aimed at Design Engineers, not end users
 Project is provided as source code/design files for Design Engineers to customise and implement in hardware themselves.

Main Project

B. Lancaster

Introductio

Design
Overview
Memory Map
Interconnect
Interrupts

Multi-core Functionality

Results

Conclusion

What this project produces:

- System-on-Chip with multi-processor functionality
 Tested on FPGA hardware with 1-96 CPU cores.
- Custom 16-bit RISC CPU
 With interrupts and its own Instruction Set Architecture (ISA).
- Aimed at Design Engineers, not end users
 Project is provided as source code/design files for Design Engineers to customise and implement in hardware themselves.
- Software/Assembly compiler
 PRCO304 programming language/Intel assembly syntax.

Top Level Hierarchy

Main Project

B. Lancaster

Introduction

Top Level

Design

Memory Map Interconnect Interrupts

Multi-core

Regulte





Memory Map

Main Project

B. Lancaster

Introduction

....

Design

Overview

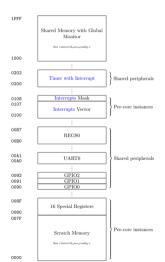
Memory Map

Interconnec

Multi coro

Functionali

Results



- Shared Memory with Global Monitor
- Timer with Interrupt
- Per-core Interrupt Vector and Mask
- Shared Register Set
- UART Transceiver
- Multiple GPIO ports
- Per-core scratch memory
- Per-core Special Registers
- Customisable by designers



Interconnect

Main Project

B. Lancaster

Introduction

Top Level Design Overview

Memory Map

Interconnect

Interrupts

Multi-core Functionality

Results

Interrupts

Main Project

B. Lancaster

Introduction

Top Level

Overview

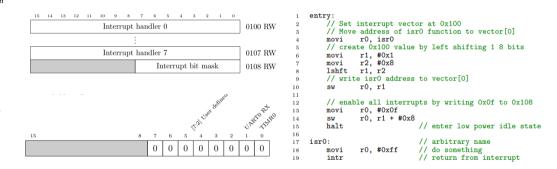
Memory Map

Interrupts

Functionali

Results

Conclusio



Demo: 2 Core LED toggle (GPIO0) with TIMR0 1s interrupt (interrupts_2.s)

Timer Interrupt Example

Main Project

B. Lancaster

Introduction

Top Level Design

Memory Map Interconnect Interrupts

Multi-core

ricounto

870 000 ns 3,870.000 ns 2,500 ns 5,500 ns Name Value 3.000 ns 3.500 ns 4.000 ns 4.500 ns .5.000 ns 0 / 0 / 0 / 0 / 0 0 V 0 V 0 V 0 V 0 /0 V0 V0 V0 V0 > 16 r_pc[15:0] 0010 0010 0010 0010 VO > W r instr[15:0] 4000 4000 4000 4000 4000 > W regs[0:7][15:0] 0010.0100.0008.0 0010.0100.0008.0000.0000.0000 > COALU V ∭ INT > W ints[7:0] 01 > W ints vector[127:0] 0000000000000000 > w ints_mask[7:0] 0 f 14 w intr M has int 18 int pending W int pending ack 14 regs_use_int > " MMU ≥ 16 C1 TIMRO W out M S PSELX 14 we

Figure: TIMR0 1us interrupt with context switching

B. Lancaster

Introduction

Top Leve Design

Multi-core Functionality

HW/SW Requirements Context Identification

Results

Conclusion

1 Introduction

2 Top Level Design

- Multi-core Functionality HW/SW Requirements Context Identification
- 4 Results
- 6 Conclusion

Main Project

B. Lancaster

Introduction

Top Leve Design

Multi-core Functionality HW/SW Requirements Context Identification

Results

Conclusion

Hardware:

 Bus Arbitration (scheduling: priority, rotating, etc.)

Main Project

B. Lancaster

Introductio

Top Leve Design

Functionality
HW/SW
Requirements
Context Identification

Desults

Conclusion

Hardware:

- Bus Arbitration (scheduling: priority, rotating, etc.)
- Atomic functions
 (atomic versions of load/store to prevent race conditions)

Main Project

B. Lancaster

Introductio

Top Leve Design

HW/SW Requirements

Context Identification

Results

Conclusio

Hardware:

- Bus Arbitration (scheduling: priority, rotating, etc.)
- Atomic functions
 (atomic versions of load/store to prevent race conditions)
- Per-core instruction memory

Main Project

B. Lancaster

Introductio

Top Leve Design

HW/SW Requirements

Context Identification

Hesuit

Conclusio

Hardware:

- Bus Arbitration (scheduling: priority, rotating, etc.)
- Atomic functions
 (atomic versions of load/store to prevent race conditions)
- Per-core instruction memory
- Per-core context-switching for interrupt handling

Main Project

B. Lancaster

Introductio

Top Leve Design

HW/SW Requirements

Context Identification

Results

Conclusio

Hardware:

- Bus Arbitration (scheduling: priority, rotating, etc.)
- Atomic functions
 (atomic versions of load/store to prevent race conditions)
- Per-core instruction memory
- Per-core context-switching for interrupt handling

Software:

 Semaphores/Mutexes (exclusive memory access)

Main Project

B. Lancaster

Introductio

Top Leve Design

HW/SW Requirements

Context Identification

Results

Conclusio

Hardware:

- Bus Arbitration (scheduling: priority, rotating, etc.)
- Atomic functions
 (atomic versions of load/store to prevent race conditions)
- Per-core instruction memory
- Per-core context-switching for interrupt handling

- Semaphores/Mutexes (exclusive memory access)
- Thread synchronisation (memory barriers)

Main Project

B. Lancaster

Introductio

Top Leve Design

HW/SW Requirements

Context Identification

Results

Conclusio

Hardware:

- Bus Arbitration (scheduling: priority, rotating, etc.)
- Atomic functions
 (atomic versions of load/store to prevent race conditions)
- Per-core instruction memory
- Per-core context-switching for interrupt handling

- Semaphores/Mutexes (exclusive memory access)
- Thread synchronisation (memory barriers)
- Context identification
 What core am I?
 How many cores?
 How much memory?

Context Identification

Main Project

B. Lancaster

Introduction

Top Leve Design

Multi-core Functionality

Context Identification

Context identificatio

Results

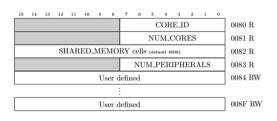


Figure: Special Registers 0x0080 to 0x008F

```
entry:
    // get core idx 0x80 in r7
    movi
            r7, #0x80
    lw
            r7. r7
    // Branch away if not core 0
            r7, r0
    cmp
    movi
            rO, exit
    br
            r0. BR_NE
    // Core 0 only instructions
            r0, r0
    nop
            r0, r0
    gon
            r0 . r0
    gon
exit:
    halt
            r0, r0
```

B. Lancaster

Introduction

Top Level Design

Multi-core Functionality

Results

Results 1

Conclusion

1 Introduction

2 Top Level Design

3 Multi-core Functionality

4 Results Results 1

Results

Main Project

B. Lancaster

Introduction

Top Level Design

Multi-core Functionality

Results 1

Results 1

Main Project

B. Lancaster

Introduction

Top Level Design

Multi-core Functionality

Results

Results 1

B. Lancaster

Introduction

Introduction

Design

2 Top Level Design

Multi-core Functionalit

3 Multi-core Functionality

Conclusion

Accomplishments
Future Improvements

4 Results

5 Conclusion Accomplishments Future Improvements

Accomplishments

Main Project

B. Lancaster

Introductio

Top Leve Design

Multi-core Functionali

Result

Conclus

Accomplishments

 Near complete System-on-Chip design with various peripherals Timers, GPIO, UART, Registers, Memory

- Common multi-thread/core synchronisation primitives
 Semaphores, Mutexes, Memory Barriers, Atomic Instructions
- AMBA APB bus interface with Global Monitor Timers, GPIO, UART, Registers, Memory
- Working shared bus arbitration
 Schedules access to shared resources
- Working FPGA implementation for a 96 core design Nearly fills Cyclone V FPGA on the DE1-SoC
- Interrupts with hardware context-switching Low latency to react to interrupt
- Acknowledges design limitations and attempts to overcome LUT resources, block memories, power and temperature requirements

Future Improvements

Main Project

B. Lancaster

Introductio

Top Leve Design

Multi-core Functionalit

Result

Conclusio

Accomplishments

Working Global Reset
 Timers, GPIO, UART, Registers, Memory

- Common multi-thread/core synchronisation primitives
 Semaphores, Mutexes, Memory Barriers, Atomic Instructions
- AMBA APB bus interface with Global Monitor Timers, GPIO, UART, Registers, Memory
- Working shared bus arbitration
 Schedules access to shared resources
- Working FPGA implementation for a 96 core design Nearly fills Cyclone V FPGA on the DE1-SoC
- Interrupts with hardware context-switching Low latency to react to interrupt
- Acknowledges design limitations and attempts to overcome LUT resources, block memories, power and temperature requirements

