RISC-V based core as a soft processor in FPGAs
Chowdhary Musunuri
Sr. Director, Solutions & Applications
Microsemi
chowdhary.musunuri@microsemi.com
Agenda

- A brief introduction to FPGAs
- A few examples of applications built on FPGAs
- RISCV as a soft processor core in FPGAs
What is an FPGA?

- An FPGA is an integrated circuit that can be configured to emulate any digital circuit as long as there are enough resources.
- An FPGA can be seen as an array of configurable logic elements connected through programmable routing interconnects.
Simplified Logic Element Structure

Look-Up Table (LUT)

Configuration bits

MUX

© 2017 Microsemi Corporation. Company Proprietary
Translating a design to an FPGA

- Design is captured in RTL code and/or in vendor provided design entry tools

- Design is run through vendor provided CAD tools

- Program design in FPGA
  - Reconfigurable FPGAs can be programmed multiple times for different functionality or fixing bugs
ASIC gives high performance at the cost of flexibility
A general purpose Processor is very flexible but not tuned to the application
An FPGA gives a nice compromise between application tuned performance and re-programmable flexibility
SoC FPGA Design Suite

FPGA Design Flow

- **Libero SoC**
  - Design
  - IP Catalog
  - Smart Design
  - Simulation
  - Synthesis
  - Layout
  - Timing Analysis
  - Power Analysis
  - Hardware Debug

Embedded Design Flow

- Soft Console IDE
- Firmware Catalog
- Sample Projects
- Compiler
- Debugger

Graphical Configurators & Applications

Programmer

Demo/Eval Board

© 2017 Microsemi Corporation. Company Proprietary
A few SOC FPGA application examples
- RISCV soft processor can be used for configuration of peripherals and internal functional IP blocks

- 12G QoS Switching in FPGA with Hierarchical Queues
- 30% Lower total power
- Instant ON FPGA can also be used for system management
Wireless HetNet: RRH & BBU

- RISCV soft processor subsystem can be used for configuration of peripherals and internal functional blocks

✓ Signal processing capabilities with hardened preadders ideal for supporting low/mid bandwidth DFE (4x4x60MHz) and Baseband Processing

✓ Ideal for up to 12.5G CPRI and JESD204b interfacing

✓ Up to 50% lower total power

✓ Best Security and SEU immune FPGA fabric
Power Generation Control

- Lowest power
- Instant on control functions
- Security
- Image Processing is done in the FPGA logic hardware
  - Embedded FPGA Mathblocks, PCIe Subsystem & DDR support
- Soft RISCV processor subsystem is used for configuration of peripherals and internal functional blocks
RV32IM RISCV Soft processor

RISCV soft processor on Microsemi® PolarFire™ FPGA
What is RISC-V

- A New free and open ISA developed at UC Berkeley
  - RISC-V is “Instruction Set Architecture” (ISA). Not a processor
  - The micro architecture implementations can be open or proprietary
  - Goal is to encourage both open-source & proprietary implementations of the RISC-V ISA specification
  - Designed for Research, Education & Commercial use

- Four base integer ISA variants
  - RV32I, RV64I, RV32E, RV128I (32,64,128bit machines)

- Standard Extensions
  - M: Integer Multiply/Divide
  - A: Atomic Memory Operations
  - F: Single Precision FP
  - D: Double Precision FP
  - G: IMAFD, General Purpose ISA
  - Q: Quad Precision Floating Point
  - C: 16-bit compressed instruction (RV32C, RV64C)
RISC-V Soft Processor on PolarFire™ FPGA

CoreRISCv_AXI4

- Supports the RISC-V standard RV32IM ISA
- Integrated 8Kbytes instruction cache and 8Kbytes data cache
- Two external AXI interfaces for IO and memory
- Supports up to 31 programmable interrupts
- Debug unit with a JTAG interface
- Best suited for low to mid range microcontroller applications
CoreRISCV_AXI4

- **Processor Core**
  - Based on the E31 Coreplex core by SiFive
  - Provides a single hardware thread
  - Machine-mode privileged architecture
  - Supports the RISCV standard RV32IM ISA
- **Two External AXI Interfaces**
  - AXI memory Interface
    - Cached access to instruction & data memory
  - AXI I/O interface
    - Un-Cached access to I/O peripherals
- **Memory System**
  - First level Instruction Cache
    - 8KB, Direct Mapped with 64 bytes line size, single clock cycle access latency
  - First Level Data Cache
    - 8KB, Direct Mapped with 64 bytes line size
    - Access Latency is two clock cycles for full words and three clock cycles for smaller quantities
  - Un-Cached memory access for I/O
### Interrupt Sources

- **Local Interrupts**
  - Wired directly to the CPU internally
  - Standard Software Interrupts
    - (Traps, Exceptions)
    - Timer Interrupt

- **Global Interrupts**
  - Routed via Platform Level Interrupt Controller
  - Supports up to 31 external interrupt sources
  - All external interrupts are single priority level at priority 1
    (External interrupts in the system can be connected here)

### JTAG Interface

- Industry standard 1149.1 JTAG interface
- Supports Interactive debug
- Supports Hardware Breakpoints (Max:2)
- Accessible via Microsemi FlashPro5 JTAG programmer/debugger
SoftConsole IDE

- Softconsole 5.0/5.1 (beta)
  - SC5.0/SC5.1 works with Flashpro5.0 JTAG debugger
  - SC5.0 supports specific versions of Ubuntu, Red hat/Centos and OpenSuse
  - SC5.1 (beta) supports windows 7

- Firmware project structure
  - riscv-hal – startup code, hardware abstraction layer, interrupt management
  - Drivers - Drivers for peripherals e.g. UART, I2C and SPI
  - riscv_hal is available on github
CoreRISCV_AXI system

- CoreBootStrap is configurable hardware boot loader
- At POR CoreBootStrap asserts PROC_RESET and holds RISCV in reset
- Copies executable binary from SPI Flash memory to internal RAM and releases the PROC_RESET
  - Image can reside in external SPI flash or internal NVM memory
- RISCV executes the code from internal RAM
Microsemi RISCV on GitHub

- [https://github.com/RISCV-on-Microsemi-FPGA](https://github.com/RISCV-on-Microsemi-FPGA)
  - Documentation
  - Example Design Projects
Questions?
Thank You