Building a custom Risc-V open source FPGA SOC

Lawrie Griffiths

lawrie.griffiths@ntlworld.com

Introduction

- Risc-V
- System on a Chip
- Open source hardware
- Open source tools
- VexRiscv, SaxonSoc
  - Charles Papon, @Dolu1990
- Blackice Mx
  - Alan Wood, @folknology

Risc-V

- Open source Instruction Set Architecture (ISA)
- Full-featured mature ISA
- Supported by Linux
- Supported by real-time Operating Systems
  - e.g. Zephyr
- Increasing industry support
- GCC and other tools support
- Rival to ARM

System-on-a-Chip (SOC)

- CPU plus peripherals on one chip
- There is one in your phone
- There is one in a Raspberry Pi
- Microcontrollers are simple SOCs

Risc-V implementations

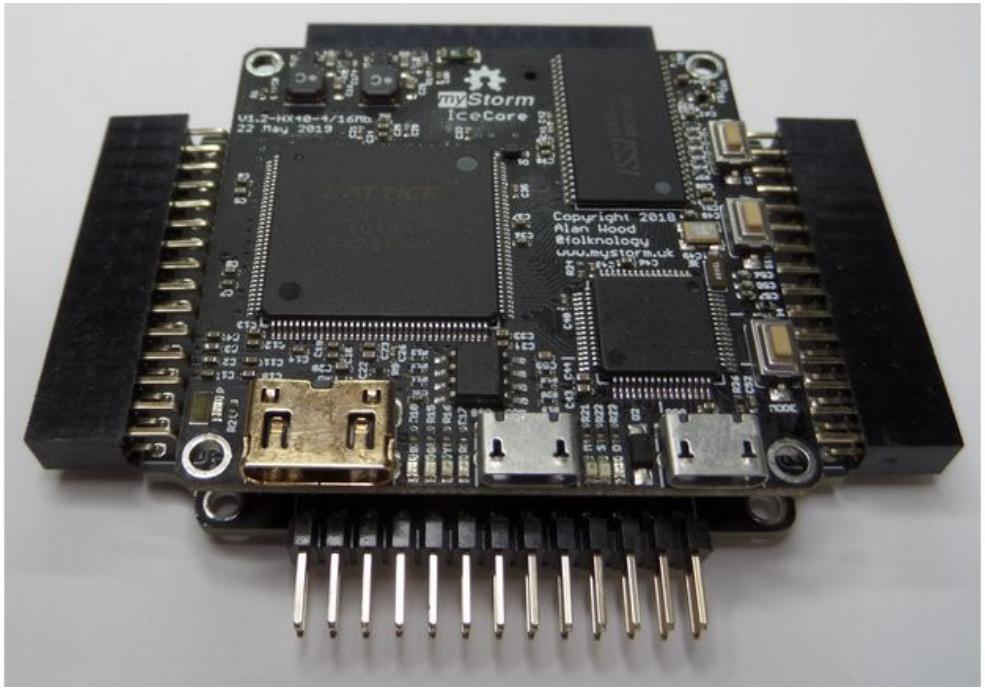
- Sifive
- Picorv32
- Icicle
- VexRiscv
- Lots of others
  - see https://github.com/riscv/riscv-cores-list

Why build a custom FPGA SOC

- Open source all the way down
- Exactly those peripherals that you want
- As many of each peripheral as you want
- Hardware implementations - no bit-banging
- Simpler software device drivers
- Much faster peripherals like LED panels
- Complete control

My choices

- VexRiscv
- SaxonSoc
- Blackice Mx board



Why VexRiscv

- Fastest 32-bit FPGA implementation
- Most configurable implementation
- Full featured, e.g. supervisor mode and MMU
- Will run Linux
- Written in SpinalHDL

Why Blackice Mx

- Open source Ice40 board with SDRAM
- Scalable and modular with carrier boards
- Compatible with Linux-capable BlackEdge
- HDMI support
- Lots of I/O
- .. but also runs on many other boards
  - like TinyFPGA BX, iCEBreaker, Fomu, etc.

Why SaxonSoc

- Same author as VexRiscv (Charles Papon)
- Written in SpinalHDL
- Supports many peripherals and memory types
- Will run Linux
- Banana Memory Bus and APB3 peripheral bus
- Concise description of custom SOCs
- Generation of BSPs
- Experience from Murax and Briey SOCs
- SaxonSoc is currently experimental

SpinalHDL

- Uses Scala language
  - designed to support Domain Specific Languages
- Runs in Java VM
- Much nicer semantics than Verilog
- Much less error-prone than Verilog
- Real software-engineering tool
- Rivals: Chisel and Python Migen with Litex

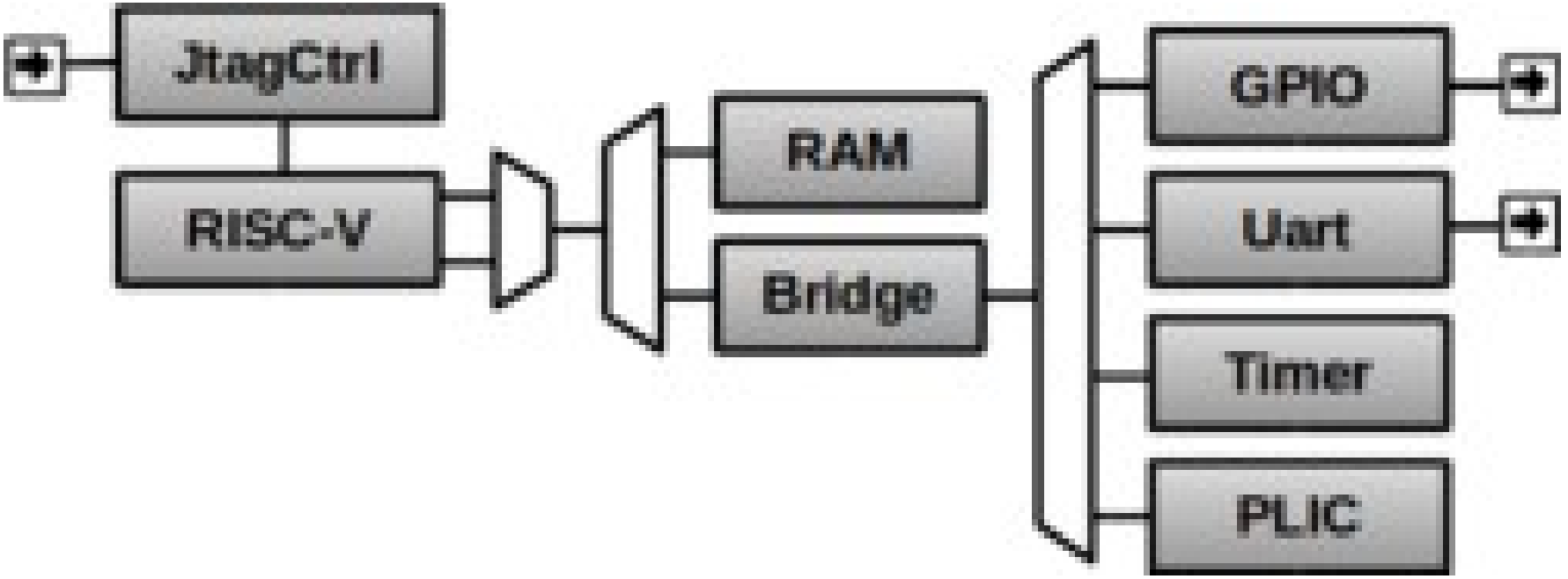
Installation on Linux

- Install Icestorm
- Install dfu & latest firmware for Blackice board
- Install SpinalTemplateSbt
- Install Risc-V compiler
- Install SaxonSoc Bmb branch
  - VexRiscv and SpinalHDL are submodules
- Build software and SOC with make files
- Can also use IDEs
  - e.g. IntelliJ and Visual Studio

Open Source toolchain

- SpinalHDL generates Verilog
- Yosys synthesises netlist
- Nextpnr place and route for specific FPGA chip
- Icepack produces ice40 bitstream

SaxonSoc architecture



SaxonSoc code extract

```
//Add components
val ramA = BmbOnChipRamGenerator(0x80000000l)
val uartA = Apb3UartGenerator(0x10000)
val gpioA = Apb3GpioGenerator(0x00000)
val spiA = Apb3SpiMasterGenerator(0x20000)
val spiB = Apb3SpiMasterGenerator(0x30000)

// Configure components
ramA.size.load(8 KiB)
ramA.hexInit.load("software/standalone/readSdcard/build/readSdcard.hex")
uartA.parameter load UartCtrlMemoryMappedConfig(
  baudrate = 115200,
  txFifoDepth = 32,
  rxFifoDepth = 32
)
spiA.parameter load SpiMasterCtrlMemoryMappedConfig(
  SpiMasterCtrlGenerics(
    dataWidth = 8,
    timerWidth = 32,
    ssWidth = 1
  )
)
```

SaxonSoc variants for Blackice Mx

- BlackiceMxSocMinimal - blinkAndEcho
- BlackiceMxSocSdram - writeFlash utility
- BlackiceMxXip - blinkAndEcho from flash
- BlackiceMxZephyr - runs Zephyr OS
- BlackiceMxArduino - supports Arduino IDE
  - based on f32c/arduino
- Produce your own

Links

- SaxonSoc
  - https://github.com/SpinalHDL/SaxonSoc/tree/Bmb
- VexRiscv
  - https://github.com/SpinalHDL/VexRiscv/
- SpinalHDLTemplateSbt
  - https://github.com/SpinalHDL/SpinalTemplateSbt/
- myStorm forum
  - https://forum.mystorm.uk/t/running-saxonsoc-on-blackice-mx/687

Demo of Zephyr OS

- Philosophers sample
  - https://twitter.com/i/status/1181939354308169739

Questions

- ?