SCRx family of the RISC-V compatible processor IP by Syntacore

Alexander Redkin
Executive director

Embedded World
February 2019
Syntacore introduction

IP company, founding member of RISC-V foundation

Develops and licenses state-of-the-art RISC-V cores
- Initial line is available and shipping to customers
- 3+ years of focused RISC-V development
- Core team comes from 10+ years of highly-relevant background
- SDKs, samples in silicon, full collateral

Full service to specialize CPU IP for customer needs
- One-stop workload-specific customization for 10x improvements
  - with tools/compiler support
- IP hardening at the required library node
- SoC integration and SW migration support
Company background

Est 2015
HQ in EU (Cyprus)
- R&D offices in St.Petersburg and Moscow
- Representatives in EMEA, APAC

Team background:
- 10+ years in the corporate R&D (major semi MNC)
- Developed cores and SoC are in the mass productions
- 15+ tapeouts, 180..14nm

Expertise:
- Low-power and high-performance embedded cores and IP
- ASIP technologies and reconfigurable architectures
- Architectural exploration & workload characterization
- Compiler technologies
SCRx baseline cores 2019

- **SCR1 (RV32I|E) MC**: 64b
- **SCR3 (RV32IMC) A**: 64b
- **SCR4 (RV64IMC) DA**: 64b
- **SCR5 (RV64IMC) FDA**: 64b
- **SCR7 (RV64IMCFDA)**

**Performance**
- **RTOS**: SCR1, SCR3, SCR4, SCR5
- **Linux/Full OS**: SCR7

**Area, power**
- **Shipping**: New 2019
## SCRx features at glance

<table>
<thead>
<tr>
<th>Features</th>
<th>SCR1+</th>
<th>SCR3</th>
<th>SCR4</th>
<th>SCR5</th>
<th>SCR7</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Baseline cores:</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Configurable and extensible</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Width</strong></td>
<td>32bit</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>64bit</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>ISA</strong></td>
<td>RV32</td>
<td></td>
<td>RV64I</td>
<td>RV64I</td>
<td>RV64I</td>
</tr>
<tr>
<td></td>
<td>[MC]</td>
<td>[AFD]</td>
<td>[AFD]</td>
<td>[AFD]</td>
<td>[AFD]</td>
</tr>
<tr>
<td><strong>Pipeline type</strong></td>
<td>In-order</td>
<td>In-order</td>
<td>In-order</td>
<td>In-order</td>
<td>Superscalar</td>
</tr>
<tr>
<td><strong>Pipeline, stages</strong></td>
<td>2-4</td>
<td>3-5</td>
<td>3-5</td>
<td>7-9</td>
<td>10-12</td>
</tr>
<tr>
<td><strong>Branch prediction</strong></td>
<td>Static BP, RAS</td>
<td>Static BP, RAS</td>
<td>Static BP, RAS</td>
<td>Static BP, BTB, BHT, RAS</td>
<td>Dynamic BP, BTB, BHT, RAS</td>
</tr>
<tr>
<td><strong>Execution priority levels</strong></td>
<td>Machine</td>
<td>User, Machine</td>
<td>User, Machine</td>
<td>User, Supervisor, Machine</td>
<td>User, Supervisor, Machine</td>
</tr>
<tr>
<td><strong>Extensibility/customization</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Execution units</strong></td>
<td>MU/DIV</td>
<td>area-opt</td>
<td>hi-perf</td>
<td>hi-perf</td>
<td>hi-perf</td>
</tr>
<tr>
<td></td>
<td>FPU</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>TCM</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>L1$ w/ECC</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>L2$ w/ECC</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>MPU</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Memory subsystem</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>MMU, virtual memory</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Debug</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>HW BP</td>
<td>1-2</td>
<td>1-8 adv ctrl</td>
<td>1-8 adv ctrl</td>
<td>1-8 adv ctrl</td>
</tr>
<tr>
<td></td>
<td>Performance counters</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Interrupt Controller</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>IRQs</td>
<td>8-32</td>
<td>8-1024</td>
<td>8-1024</td>
<td>8-1024</td>
</tr>
<tr>
<td></td>
<td>Features</td>
<td>basic</td>
<td>advanced</td>
<td>advanced</td>
<td>advanced+</td>
</tr>
<tr>
<td><strong>SMP support</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>I/F options</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>AHB</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>AXI4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ACE</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Download SCR1+ datasheet at www.github.com/syntacore SCR1+*
SCR1 overview

Compact MCU core for deeply embedded applications and accelerator control

- RV32I|E[MC] ISA
- 2 to 4 stages pipeline
- M-mode only
- Optional configurable IPIC
  - 8..32 IRQs
- Optional integrated Debug Controller
  - OpenOCD compatible
- Choices of the optional MUL/DIV unit
  - Area- or performance- optimized
- Open sourced under SHL-license (Apache 2.0 derivative)
  - Unrestricted commercial use allowed

- High quality free MCU IP
- In the top System Verilog Github repos in the world
- Best-effort support provided, commercial offered
SCR1 overview cont

<table>
<thead>
<tr>
<th>Performance*, per MHz</th>
<th>DMIPS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>-O2</td>
</tr>
<tr>
<td></td>
<td>-best**</td>
</tr>
<tr>
<td>Coremark</td>
<td>-best**</td>
</tr>
</tbody>
</table>

* Dhrystone 2.1, Coremark 1.0, GCC 7.1 BM from TCM
** -O3 -funroll-loops -fpeel-loops -fgcse-sm -fgcse-las -flto

What's new:
- Extensive user guide and quick start collateral
  - works out-of-the-box in all major sims
- Verilator support (version 3.922 and later)
- More tests/sample: RISC-V compliance, others
- Regular talk at ORCONF
- Updated and maintained

Synthesis data:
Minimal RV32EC config: 11 kGates
Default RV32IMC config: 32 kGates
Range 10..40+ kGates

250+ MHz @ tsmc90lp {typical, 1.0V, +25C}
SCR1 SDK

https://github.com/syntacore/scr1-sdk

Repository content:
- docs - SDK documentation
- fpga - SCR1 SDK FPGA projects
- images - precompiled binary files
- scr1 - SCR1 core source files
- sw – sample SW projects

Supported platforms:
- Digilent Arty and Nexys 4 (Xilinx)
- Terasic DE10-Lite and Arria V GX starter (Intel)

Software:
- Bootloader
- Zephyr OS
- Tests/sample apps
- Pre-built GCC-based toolchain (Win/Linux)

Fully open designs and pre-build images for a quick start
SCR3: 32 or 64 bit

High-performance multicore capable MCU-class core

- RV32I[MCA] or RV64I[MCA] ISA
- Machine and User privilege modes
- Optional MPU (Memory Protection Unit)
- Optional Tightly Coupled Memory (TCM), L1 caches ECC/parity
- 32|64bit AHB or AXI4 external interface
- Optional high-performance or area-optimized MUL/DIV unit
- Integrated IRQ controller and PLIC
- Advanced debug with JTAG i/f
- Multicore configs up to 4 SCRx cores
  - SMP and heterogeneous
  - with memory coherency

<table>
<thead>
<tr>
<th></th>
<th>DMIPS</th>
<th>RV32</th>
<th>RV64</th>
</tr>
</thead>
<tbody>
<tr>
<td>Performance*, per MHz</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-O2</td>
<td>1.86</td>
<td>1.97</td>
<td></td>
</tr>
<tr>
<td>-best**</td>
<td>2.937</td>
<td>3.27</td>
<td></td>
</tr>
<tr>
<td>Coremark</td>
<td>3.30</td>
<td>3.40</td>
<td></td>
</tr>
</tbody>
</table>

* Dhrystone 2.1, Coremark 1.0, GCC 8.1 BM from TCM
** -O3 -funroll-loops -fpeel-loops -fgcse-sm -fgcse-las -flto
SCR4: 32 or 64 bit

High-performance multicore capable MCU core with FPU

- RV32IMCF[DA] or RV64IMCF[DA] ISA
- U- and M-mode
- Configurable advanced BP, fast MUL/DIV
- Integrated IRQ controller and PLIC
- 32|64bit bit AHB or AXI4 external interface
- Optional MPU, TCM, L1 caches w/ECC
- Advanced debug controller with JTAG
- Configurable SP or DP FPU
  - IEEE 754-2008 compliant
- Multicore configs up to 4 SCRx cores
  - SMP and heterogeneous
  - with memory coherency

Performance*, per MHz

<table>
<thead>
<tr>
<th></th>
<th>DMIPS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>RV32</td>
</tr>
<tr>
<td>-O2</td>
<td>1.86</td>
</tr>
<tr>
<td>-best**</td>
<td>2.96</td>
</tr>
<tr>
<td>Coremark</td>
<td>3.30</td>
</tr>
<tr>
<td>-best**</td>
<td>1.22</td>
</tr>
<tr>
<td>DP Whetstone</td>
<td></td>
</tr>
</tbody>
</table>

* Dhrystone 2.1, Coremark 1.0, GCC 8.1 BM from TCM
** -O3 -funroll-loops -fpeel-loops -fgcse-sm -fgcse-las -flto
SCX5: 32 or 64 bit

Efficient entry-level APU/embedded core
- RV32IMC[AFD] or RV64IMC[AFD] ISA
- Multicore configs up to 4 SCRx cores
  - SMP and heterogeneous
- Advanced BP (BTB/BHT/RAS)
- IRQ controller (integrated and PLIC)
- M-, S- and U-modes
- Virtual memory support, full MMU
- L1, L2 caches with coherency, atomics, ECC
- High performance double-precision FPU
- Linux and FreeBSD support
- 1GHz+ @28nm
- Advanced debug with JTAG i/f

<table>
<thead>
<tr>
<th>Performance*, per MHz</th>
<th>DMIPS</th>
<th>Coremark</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>-O2</td>
<td>-best**</td>
</tr>
<tr>
<td>RV32</td>
<td>1.60</td>
<td>2.48</td>
</tr>
<tr>
<td>RV64</td>
<td>1.70</td>
<td>2.62</td>
</tr>
<tr>
<td>best**</td>
<td>2.83</td>
<td>3.02</td>
</tr>
</tbody>
</table>

* Dhrystone 2.1, Coremark 1.0, GCC 8.1 BM from TCM
** O3-funroll-loops ·ifpeel-loops ·fgcse-sm ·fgcse-ias ·flto
RV64 SCR7

Efficient mid-range application core

- RV64GC ISA
- Multicore configs up to 8, later 16 cores
- Flexible uarch template, 10-12 stage pipeline
- Initial SCR7 configuration (Q1’19):
  - Decode and dispatch of up to two instructions per cycle
  - Out-of-order issue of up to four micro-ops
  - Out-of-order completion, in-order retirement
- M-, S- and U-modes
- Virtual memory support, full MMU
- 16-64KB L1, up to 2MB L2 cache with ECC
- 1.2GHz+ @28nm
- Advanced debug with JTAG i/f

<table>
<thead>
<tr>
<th>Performance*, per MHz</th>
<th>DMIPS</th>
<th>Coremark</th>
</tr>
</thead>
<tbody>
<tr>
<td>-O2 best**</td>
<td>2.75</td>
<td>3.01</td>
</tr>
<tr>
<td>-best**</td>
<td>5.00</td>
<td></td>
</tr>
</tbody>
</table>

* Preliminary data, 2-way implementation, Dhrystone 2.1, Coremark 1.0, GCC 8.1 BM
** O3 -funroll-loops -fpeel-loops -fgcse-sm -fgcse-las -flto
Fully featured SW development suite

Stable IDE in production:
- GCC 8.1
- GNU Binutils 2.31.0
- Newlib 3.0
- GNU GDB 8.0.50
- Open On-Chip Debugger 0.10.0
- Eclipse 4.9.0

Hosts: Linux, Windows
Targets: BM, Linux (beta)

Also available:
- LLVM 5.0
- CompCert 3.1
- 3rd party vendors in 2019

Simulators:
- Qemu
- Spike
- 3rd party vendors

JTAG-based debug solutions:
Supports: Segger J-link, Olimex ARM-USB-OCD family, Digilink JTAG-HS2, Lauterbach trace32
SCRx SDK

Stable Eclipse/gcc based toolchain with IDE:
- GCC 8.1
- GNU Binutils 2.31.0
- Newlib 3.0
- GNU GDB 8.0.50
- Open On-Chip Debugger 0.10.0
- Eclipse 4.9.0

HW platform based on standard FPGA dev.kits
- Multiple boards supported (Altera, Xilinx)
- Low-cost 3rd party JTAG tools
- Open design for easy start

SW:
- Bootloader
- OS: Zephyr/FreeRTOS/Linux
- Application samples, tests, benchmarks

Extensibility/customization: how it works

Dynamic power

Customized core

Full energy

General-purpose core

Full energy

Processing time
Workload-specific customization

Extensibility features:
- Computational capabilities
  - New functions using existing HW
  - New Functional Units
- Extended storage
  - Mems/RF, addressable or state
  - Custom AGU
- I/O ports
- Specialized system behavior
  - Standard events processing
  - Custom events

Domain examples:
- Computationally intensive algorithms acceleration
- Specialized processors (including DSP)
- High-throughput applications
  - Wire Speed Processing/DPI/Real-time/Comms
SCRx extensibility example

Custom ISA extension for AES & other crypto kernels acceleration for SCR5

- **Data**
  - RV32G – FPGA-based devkit, g++ 5.2.0, Linux 4.6, optimized C++ implementation
  - RV32G + custom – same + intrinsics
  - Core i7 6800K @ 3.4GHz, g++ 5.4.0, Linux 64, optimized C++ implementation
- **60..575x** speedup @ modest area increase: 11.7% core, 3.7% at the CPU cluster level

<table>
<thead>
<tr>
<th>Platform</th>
<th>Fmax, MHz</th>
<th>Encoding throughput, MB/s</th>
<th>Normalized per MHz, MB/s</th>
<th>RV32G + custom speed-up</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Crypto-1</td>
<td>Crypto-2</td>
<td>AES-128</td>
</tr>
<tr>
<td>RV32G</td>
<td>20</td>
<td>0.025</td>
<td>0.129</td>
<td>0.238</td>
</tr>
<tr>
<td>RV32G + custom</td>
<td>20</td>
<td>14.375</td>
<td>15.188</td>
<td>14.502</td>
</tr>
<tr>
<td>Core i7</td>
<td>3400</td>
<td>79.115</td>
<td>235.343</td>
<td>335.212</td>
</tr>
<tr>
<td>Core i7 + NI</td>
<td>3400</td>
<td></td>
<td>3874.552</td>
<td>0.13957</td>
</tr>
</tbody>
</table>

Disclaimer: Authors are aware AES allows for more efficient dedicated accelerators designs, used as example algorithm.

Details in paper @EW2018 conference
Conclusion

- Syntacore offers high-quality RISC-V compatible CPU IP
  - Founding member, fully focused on RISC-V since 2015
  - Silicon-proven and shipping to customers
  - Turnkey IP customization services

- Stop by our stand for demos, including 28nm SCR4 silicon running Zephyr RTOS and Quad-core 64bit SCR5 Linux system running GNU Debian Linux
Thank you!