

Syntacore open-source and commercial RISC-V solutions



Copyright © 2021 Syntacore. All trademarks, product, and brand names belong to their respective owners.

Alexander Redkin **Executive director**

RISC-V Days Tokyo2021Autumn







Outline

- Company intro
- RISC-V compatible IP
- Customization services
- Evaluation





Copyright © 2021 Syntacore. All trademarks, product, and brand names belong to their respective owners.

2

Syntacore introduction

Semiconductor IP company, founding member of RISC-V foundation

Develops and licenses state-of-the-art RISC-V cores

- Immediately available, silicon-proven and ships in volume
- 7+ years of focused RISC-V experience
- Core team comes from 10+ years of highly-relevant background
- SDKs, samples in silicon, full collateral lacksquare

- Turnkey service to specialize CPU IP for customer requirements • 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, ~100 EE

HQ in EU (Cyprus)

- R&D offices in St. Petersburg, Moscow, Russia
- Full-time staff and representatives in APAC, EMEA, US

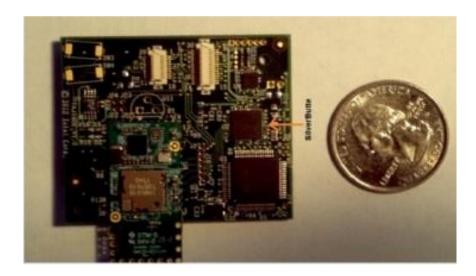
Team background:

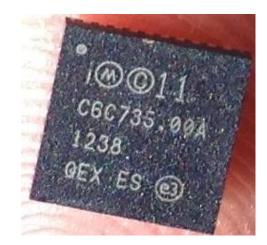
- 10+ years in the corporate R&D (major semi MNC)
- Developed cores and SoC are in the mass productions

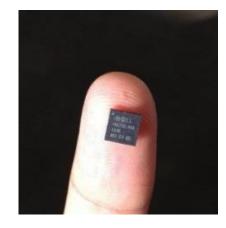
Expertise:

- High-performance and low-power embedded cores and IP
- ASIP technologies and reconfigurable architectures
- Architectural exploration & workload characterization
- Compiler technologies













Some current results

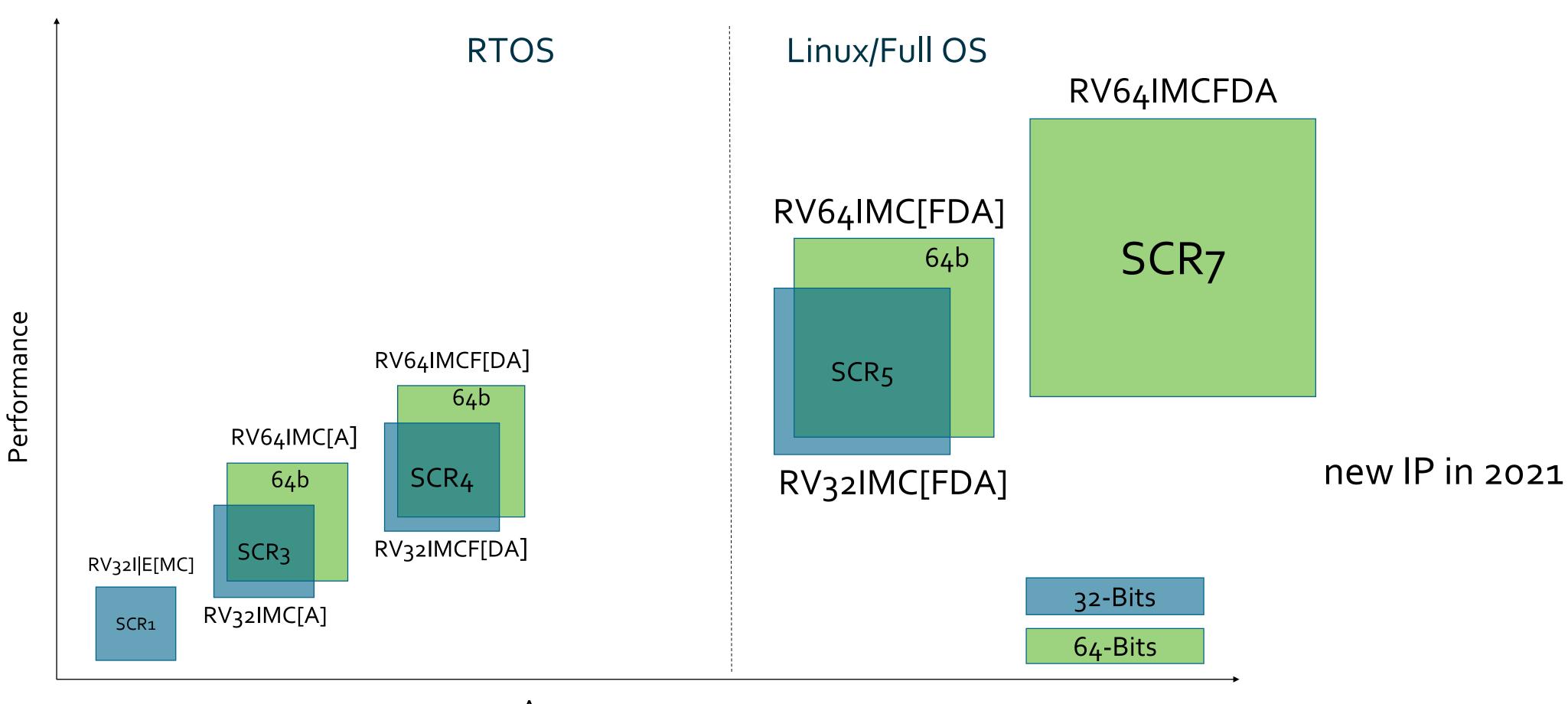
- State-of-the-art RISC-V CPU IP line with competitive features
 - commercially deployed in SoCs up to 5nm
 - first RISC-V client silicon in 2016, first RISC-V Linux-capable IP in 2016, in full-wafer from 2017
- MPWs and full-wafer production at the clients. Projects examples:
 - ✓ 56-cores heterogeneous SoC @7nm (64bit heterogeneous, NuMA, sophisticated system-level customization)
 - active battery-less SoC @22nm (SCR1, power-optimization, clock/power domains, ntv-ready)
- Customers in APAC, EMEA, US References available







Current product line













State-of-the art RISC-V CPU IP

eatures			m	RTOS/ Bare Metal		Linux/ "Full" OS			
catales			SCR1*	SCR3	SCR4	SCR5	SCR7		
\ A (i alth		32bit	•	•	•	•			
Width		64bit		•	•	•	•		
ISA			RV32IJE[MC]	RV[32 64]IMC[A]	RV[32 64]IMCF[AD]	RV[32 64]IMC[AFD]	RV64IMCAFD		
Pipeline type			In-order	In-order	In-order	In-order	Superscalar		
Pipeline, stages			2-4	3-5	3-5	7-9	10-12		
Branch prediction			STATIC BE RAS STATIC BE RAS		Static BP, BTB, BHT, RAS	Dynamic BP, BTB, BHT, RAS			
Execution priority levels			Machine	User, Machine	User, Machine	User, Supervisor, Machine	User, Supervisor, Machine		
Extensibility/c	Extensibility/customization		•	•	•	•	•		
Execution	MUL/DIV	area-opt	•	0	0				
units		hi-perf	0	•	•	•	•		
	F	PU			•	•	•		
	line i	/ECC parity]	0	0	0	0	0		
Mamony	LI\$[w/l	ECC[parity]		0	0	•	•		
Memory subsystem	L2\$ [w/ECC]				0	0		
5465/546111	M	1PU		•	•	•	•		
	MMU, virt	tual memory				•	•		
	Integrated	JTAG debug	•	•	•	•	•		
Debug	H۱	∧ BP	1-2	1-8 adv ctrl	1-8 adv ctrl	1-8 adv ctrl	1-8 adv ctrl		
	Performar	nce counters	0	0	0	0	0		
Interrupt	IF	RQs	8-32	8-1024	8-1024	8-1024	8-1024		
Controller	Fea	atures	basic	advanced	advanced	advanced+	advanced+		
SMP support				up t	o 4 cores with coher	ency	up to 8-16 cores		
	A	HB	•	0	0	0			
I/F options	A	×14	0	•	•	•	•		
	A	ACE					0		



Copyright © 2021 Syntacore. All trademarks, product, and brand names belong to their respective owners.

Baseline cores:

- Clean-slate designs in System Verilog
- Configurable and • extensible
- 100% compatible with major EDA flows



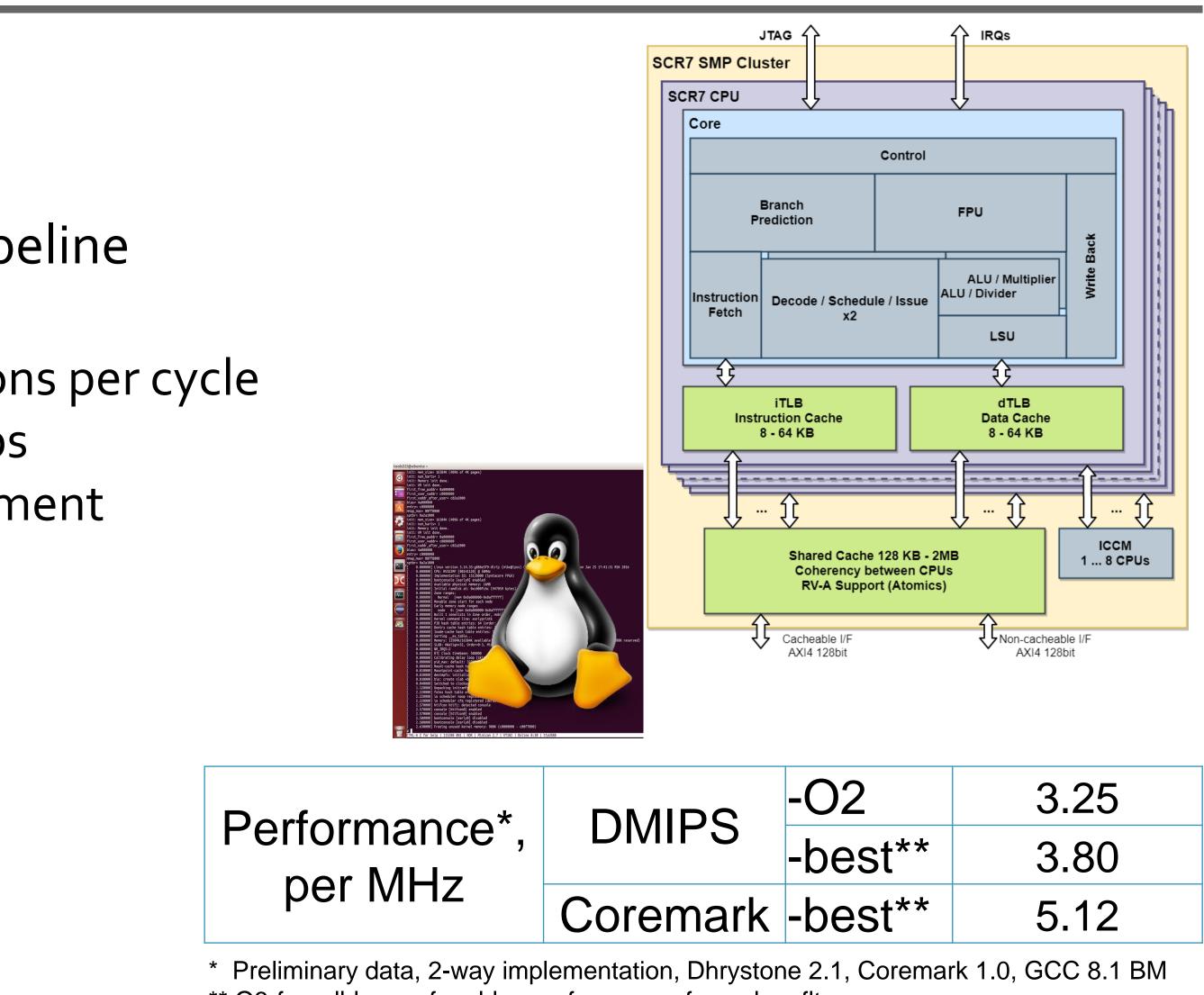


RV64 SCR7

Efficient mid-range application core

- RV64GCISA
- SMP up to 8, later 16 cores
- Flexible uarch template, 10-12 stage pipeline
- Stable SCR7 in production:
 - Decode and dispatch 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, Linux
- 16-64KB L1, up to 2MB L2 cache with ECC
- 1.5 GHz+ @28nm
- Advanced debug with JTAG i/f





** O3-funroll-loops -fpeel-loops -fgcse-sm -fgcse-las -flto

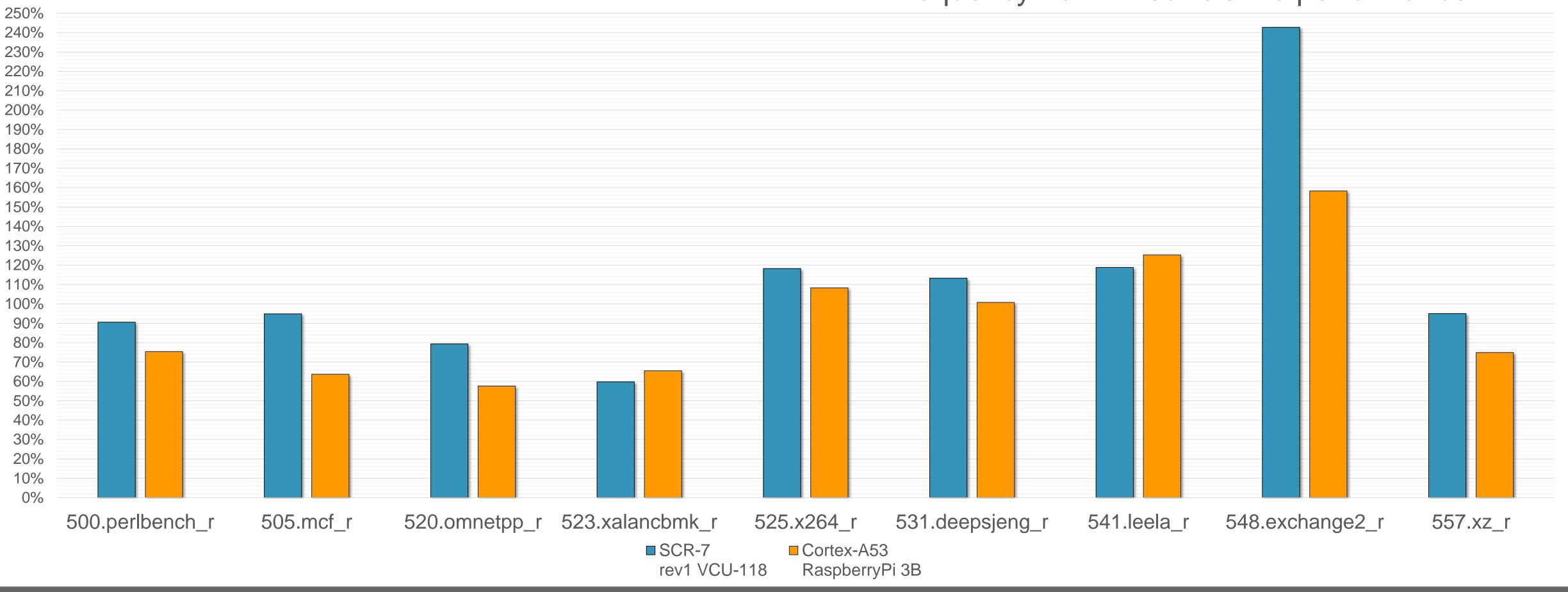






SCR7 SpecInt 2017

• Geomean +19.83% vs cortex a-53







SpecInt-2017 (group "refrate") Frequency normilized relative performance

Sk







SCR7 FPGA-based SDK

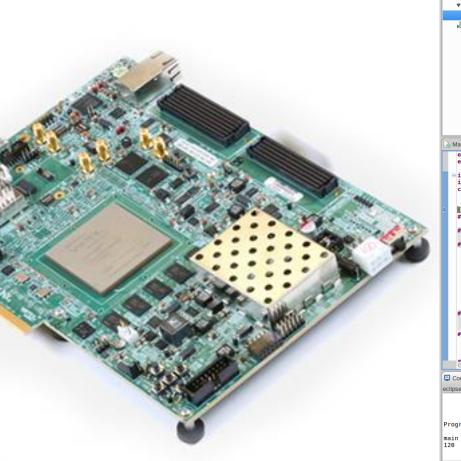
Fully-integrated system based on the off-the-shelf Xilinx VCU118 dev.kit: https://www.xilinx.com/products/boards-and-kits/vcu118.html

- Quad-core, 4GB RAM, up to 100-150 MHz, 1GB Ethernet, storage
- Boots upstream Linux kernel 4.19 (5.15 soon), Debian
- Integrated toolchain with IDE (supports Linux targets debug) \bullet
 - Windows: <u>https://yadi.sk/d/S1Ub16jKX2xLwQ</u>
 - Linux: https://yadi.sk/d/8ZsMgUx381GKiw

HTG-960 based (VU19P) dev.kit:

http://www.hitechglobal.com/Boards/VirtexUltraScale+_VU19P_Board.htm





File Edit Source Refactor Navigate Search Project Run Window Help	
[C ▼ [] (b) ≧]] \$ ▼ • • • • • •] × • • □ ■ # ≥ • • ± = 5 ∞] @ @ Ø ▼]	▲ ★ 월 ★ 월 ★ ♥ ♥ ♥ ♡ ▼
	i i i i i i i i i i i i i i i i i i i
Makefile C dhy_lo 23 g or. 5	□ □ 📴 Outline 😭 Multicore Visualizer 🚟 Disassembly 🕄 Enter location here 🔹
<pre>extern void Proc_7(One_Fifty, One_Fifty, One_Fifty *);</pre>	120 {
<pre>extern void Proc_8(Arr_l_Dim, Arr_2_Dim, int, int); eint main (argc, argv) int argc; char *argv[]; /* main and Proc_0 in the Ada version */ /* Main and Proc_0 in the Ada version */ int Number_Of_Runs; #else /* SELF_TIMED */ int duration; #endif /* SELF_TIMED */ One_Fifty Int_l_Loc; ReG One_Fifty Int_l_Loc; One_Fifty Int_l_Loc; One_Fifty Int_l_Loc; One_Fifty Int_l_Loc; Str_30 Str_l_Loc; Str_30 Str_2_Loc; /* Initializations */ #if 0 Next Ptr Glob = (Rec Pointer) malloc (sizeof (Rec_Type)); #else static Rec_Type globl, glob2; Next Ptr Glob = &glob1 Ptr Glob = &glob1 Ptr Glob = &glob2 *endif *endif *endif *endif *endif</pre>	<pre>main: addi sp.sp.192 e000005C0: sw ra,188(sp) 148 e000005d0: la8 e000005d0: la6 strcpy (Fr Glob->variant.var_1.Str_Comp, e00005d0: sw sz,1180(sp) e0000560: sw sz,1180(sp) e0000560: lu a2,0x2 l10 e0000560: lu a2,0x2 l11 e000005f0: lu a2,0x2 l12 e000005f0: lu s7, 152(a6) e000005f0: lw s7, 152(a6) e000005f0: lu s7, 152(a6) e000005f0: lu s7, 152(a6) e000005f0: lu s7, 152(a6) e000005f0: li se strcpy (Str_1 Loc, "DHRYSTONE PROGRAM, 1'ST STRING"); e000005f0: lu s7, 152(a6) e000005f0: lu s7, 152(a6) e000005f0: lu s7, 152(a6) e000005f0: lu s7, 152(a6) e000005f0: li se s2, 172(sp) e000006f0: sw s5, 164(sp) e000006f0: sw s5, 164(sp) e0000006f0: sw s5, 164(sp) e000000000000000000000000000000000000</pre>
Console 23 Tasks Problems C Executables Memory eclipse-dhy21 Default (GDB Hardware Debugging) /opt/riscv32g/bin/riscv32-unknown-elf-gdb (7.11.50.20160212) Program received signal SIGINT, Interrupt.	Sec. 2017 (2017)
main (argc=0, argv=0x0) at src/dhry_1.c:120 120 { 	





_	_

			E	苏	۶D	ebu	g	v	"
<u> </u>	x			€		٩L	~	-	
	 				-	-	-	-	n
•	\$ 1	1 ち			3	Ľ	~	-	
								ſ	
								1	
								0	
1 6	3	.	-1	۵	v	5	~	-	
		100		_					
									n
									Ш
									U

Fully featured SW development suite

Stable IDE in production:

- GCC 10.2
- 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

Also available:

- LLVM 5.0
- CompCert 3.1
- 3rd party vendors



Simulators:

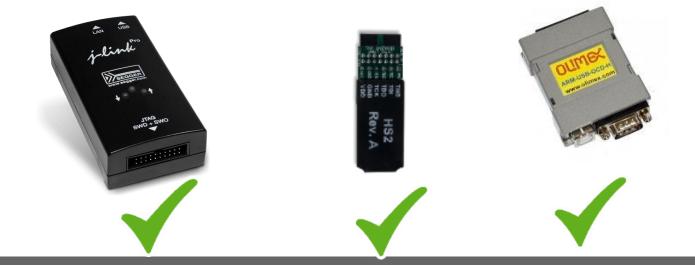
- Qemu
- Spike
- 3rd party vendors

Copyright © 2021 Syntacore. All trademarks, product, and brand names belong to their respective owners.

	Navigate Search Project Run Window Help ▼] 🧶 🔗 ▼] ∮] ∮ ♥ ∯ ♥ ⊕ ▼ ⇔ ♥ □ 🗖 🕞 Makefile 📝 drny_1.c 🕱 💙 Scrt.S		· · · · · · · · · · · · · · · · · · ·
 ▼ ²/₆ eclipse-dhry21 ▶ ³/₆ Includes ▶ ²/₆ build 	<pre>extern Boolean Func 2(Str 30, Str 30); extern void Proc_6(Enumeration, Enumeration *); extern void Proc_6(Enumeration, Enumeration *); extern void Proc_8(Arr_1Dim, Arr_2Dim, int, int); extern void Proc_8(Arr_1Dim, Arr_2Dim, int, int); eint main (aroc. arov)</pre>		
🔻 🗁 src	File Edit Source Refactor Navigate Search Project Run Window Help		
 c dhry_1.c c dhry_2.c] C + Q () + Q + Q + [] + () = () = () + () = () + (🖹 🏂 Debug 👻 "
dhry.h	★ Debug X ★ If i→ マロ(▼ E eclipse-dhry21 Default [GDB Hardware Debugging]	D M∾ Variables 💁 Breakpoints 🛛 📲 Registers 🛋 Modules	× 🔆 🍪 🗠 🗶 🗉 🗆 🚖 🕴 🎽 🗖
Makefile	▼ 🔐 dhry21.elf		
	▼ m [®] Thread #1 (Suspended : Signal : SIGINT:Interrupt) ≡ main() at dhry 1.c.120 0x5cc		
	j inding at unity_110.120 00000 pm // pm		
			\cap
			U
E Outline 🕱	Makefile (2 dhry_1c 23 3 crt.S	□ □ E Outline 😭 Multicore Visualizer 🏧 Disassembly 🕱	Enter location here 🔹 🗞 🏠 🕵 📑 🛃 🌄
⊟ ↓ª _Z ≷ √° €	<pre>extern void Proc 7(One Fifty, One Fifty *);</pre>	120 {	
 SCCSid : char[] stdio.h 	<pre>extern void Proc_8(Arr_1_Dim, Arr_2_Dim, int, int);</pre>	<pre>main: 000005cc: addi sp,sp,-192</pre>	
stdlib.h	⊖int main (argc, argv) int argc;	000005d0: sw ra,188(sp) 148 strcpy (Ptr_Glob->variant.var_1.Str_Comp,	
string.h dhry.h	<pre>char *argv[]; /* main program, corresponds to procedures */ /* Main and Proc 0 in the Ada version */</pre>	- 000005d4: addi ra,gp,1108 000005d8: lui a6,0x2	
Run_Index : unsigned long		120 { 000005dc: { sw s1,180(sp)	
Ptr_Glob : Rec_Pointer	<pre>#ifdef SELF_TIMED int Number_Of_Runs; #else '* SELF TIMED */</pre>	000005e0: sw s7,156(sp) 000005e4: sw s8,152(sp)	
 Next_Ptr_Glob : Rec_Point Int_Glob : int 	<pre>#etse /* SELF TIMED */ #endif /* SELF TIMED */</pre>	150 strcpy (Str_1_Loc, "DHRYSTONE PROGRAM, 1'ST ST 000005e8: lui a2,0x2	RING");
Bool_Glob : Boolean	REG One Fifty Int_Loc; REG One Fifty Int_2_Loc;	141 Ptr_Glob = &glob2 000005ec: addi s8,ra,48 148 strcov (Ptr Glob->variant.var 1.Str Comp.	
 Ch_1_Glob : char Ch_2_Glob : char 	One_Fifty Int_3_Loc; REG char Ch Index;	000005f0: lw s7,-152(a6)	U
Arr_1_Glob : int[]	Enumeration Enum Loc; Str_30 Str_1Loc;	000005f4: addi a3,a6,-152 150 strcpy (Str_1_Loc, "DHRYSTONE PROGRAM, 1'ST ST 000005f8: addi a5,a2,1328	RING");
Arr_2_Glob : int[][]	Str_30 Str_2_Loc;	141 Ptr Glob = &Glob2 000005fc: sw 58,1096(gp)	
Reg : Boolean	/* Initializations */ #if 0	146 Ptr_Glob->variant.var_1.Enum_Comp = Ident_ 00000600: li s8,2	3;
# REG Ø Reg : Boolean	<pre>Next_Ptr_Glob = (Rec_Pointer) malloc (sizeof (Rec_Type)); Ptr_Glob = (Rec_Pointer) malloc (sizeof (Rec_Type));</pre>	120 { 00000604: sw s0,184(sp)	
💋 time_info : struct tms	<pre>#else static Rec_Type glob1, glob2;</pre>	00000608: sw s2,176(sp) 0000060c: sw s3,172(sp)	
	Next_Ptr_Glob = &glob1 Ptr_Glob = &glob2	00000610: sw s5,164(sp) 00000614: sw s6,160(sp)	
# Too_Small_Time	enndi F		
	Console X 2 Tasks Problems Executables Memory		
📄 🕈 🈂 eclipse-dhry21	eclipse-dhry21 Default [GDB Hardware Debugging] /opt/riscv32g/bin/riscv32-unknown-elf-gdb (7.11.50.20160212)		
			\bigcap
	Program received signal SIGINT, Interrupt.		
	main (argc=0, argv=0x0) at src/dhry_1.c:120 120 {		
	120 [
	[] n°		
		Writable Smart Inser	1 120:1

JTAG-based debug solutions:

Supports: Segger J-link, Olimex ARM-USB-OCD family, Digilink JTAG-HS2, more vendors soon



Sk







Wide support by 3rd party tools and SW vendors

• Lauterbach Trace32

https://www.lauterbach.com/frames.html?pro/pro___syntacore.html

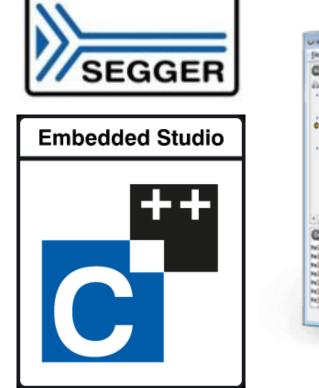


TRACE 32°

Debugger for RISC-

Segger Embedded Studio

https://wiki.segger.com/Syntacore_SCR1_SDK_Arty



	a statute finited	wercial License (Ropped) Build Debug Target	Jook Winds		-	_
a Bat Hen Jan	up Bedes notes		-3	* *	Registers 1	8
stocet, tri Jan	stu, 585.4			c1 d1	Groups 4	0 0 -
	to \$void main()				Name	Value
void min(void) {					- Charles	-
Let is					# CPU	R.
for (1 + 4; 1 4	1945 (++) (1975 38797, 11)				en 🐻	810000
					🗧 e 1	8x2000
an (1					e r2	2×0000
) which cour					📕 e3	810000
						State of the second sec
,				10	84	0xfefe
3	tot of file		••/		0 15	0xfefe
	tet of file		"/	1	6	0xfefe 0xfefe
·	be ef fin		"/		0 15	0xfefe 0xfefe 0xfefe
	und of file		"/		15 16 17	Oxfefe Oxfefe Oxfefe Oxfefe
-		1	"/		r5 r6 r7	0xfefe 0xfefe 0xfefe
a a a a a a a a a a a a a a a a a a a	De el Fia	(IIII)	"/		1010	0xfefe 0xfefe 0xfefe 0xfefe 0xfefe
The starting of		(111)	"/		r5 r6 r7 r8 r10 r11 r12	0xfefe 0xfefe 0xfefe 0xfefe 0xfefe 0xfefe
The World of		(100) 6 ≠ 5	"/		r5 r6 r7 r8 r9 r10 r11 r12 t0(r1	0xfefe 0xfefe 0xfefe 0xfefe 0xfefe 0xfefe 0xfefe 0xfefe
The North of Control o		GESTER S ► of Function			r5 r6 r7 r8 r9 r10 r11 r12 sp(r1 1r(r2	0xfefe 0xfefe 0xfefe 0xfefe 0xfefe 0xfefe 0xfefe 0xfefe 0xfefe 0xfefe
The Name of All States of All All All All All All All All All Al		California California Functions Protonantia	Catt	*	r5 r6 r7 r8 r9 r10 r11 r12 s0(r1 lr(r1 pc(r1	0xfefe 0xfefe 0xfefe 0xfefe 0xfefe 0xfefe 0xfefe 0xfefe 0xfefe 0xfefe 0xfefe 0xfefe 0xfefe 0xfefe 0xfefe
The series of the series of the series of the series of the series of the series of the series of		GESTER S ► of Function	Call J	w ddress	r5 r6 r7 r8 r9 r10 r11 r12 sp(r1 1r(r5 pc(r2) pc(r2)	0xfefe 0xfefe 0xfefe 0xfefe 0xfefe 0xfefe 0xfefe 0xfefe 0xfefe 0x62000 4) 0x0000 5)
an and a second		(111)	"/		r5 r6 r7 r8 r9 r10 r11	0x 0x 0x 0x 0x 0x 0x 0x
		California California Functions Protonantia	Call J	w Iddness 00278	r5 r6 r7 r8 r9 r10 r11 r12 s0(r1 lr(r1 pc(r1	0xfefe 0xfefe 0xfefe 0xfefe 0xfefe 0xfefe 0xfefe 0xfefe 0xfefe 0xfefe 0xfefe 0xfefe 0xfefe 0xfefe 0xfefe





IAR Embedded Workbench

https://www.iar.com/iar-embedded-workbench/#!?architecture=RISC-V

Noricipace			Registers 1 • # X Disassembly								
Debug	~	main() f()	(find registed)	×	Go to	 Memory 	(*				
Files Sesieve - Debug Beieve.c Output Beieve.mep Central Sieve.out	•	<pre>/* - SIENE.C - The benchmark C source function. SName: VJ_JJF VJ_JJD VJ_JJC S '/ einclude (stdio.h) /* Erstorthener Sieve Prime Number Program in C from Byte Jan 1983, */ edefine TROE 1 edefine TROE 1 edefine SIEE = ================================</pre>	CPU registers ZERO RA SP GP TP T0 T1 T2 S0/FP S1 A0 A1 A2 A3 A4 A5 A6 A7	Value 0x0000000 0x20001EX 0x00002FF0 0x00000000	Disassembly 20000226 20000228 fm 20000220 20000230 20000230 20000236 for (1 20000238 for (1 20000238 for (1 20000238 for (1	972E 00070023 r (k = 1 + prime 95E2 r (k = 1 + prime 6709 177D FEESC6E3 mit++ ** 0405 * 0 i <= SIZE 0505 * 0 i <= SIZE 0505 * 0 i <= SIZE 6589 15FD FAESSAE3 (flags[i])	c.add sb k <= SIZE c.add c.add blt primms fou c.addi	a4. a1 zero. k += pr) a1. a2 k += pr) a4. a1. a4. s0. a0. a1. a1. a1. a1. a1. a1. a1. a1	100) 0x2000 ; -0x1 ; 0x1 ; 0x1 ; 0x2000 ; -0x1 ; -0x1 ; -0x1 ;	Inn - Offs: Inn 1 Inn 1 Inn 8 Inn - Offs:	2192 -1 2192 -1
seve emmail/O Output 10 iterations	← a x Log lie: Of	<pre>register int 1, k; int prime, count, iter: printf(*10 iterations/s*): for (iter = 1; iter <= 10; iter++) /* do program 10 times */ { count = 0;</pre>	S2 S3 S4 S5 S6 S7 S8 S9 S10 S11 T3 T4 T5 T6 PC	0x00000000 0x00000000 0x00000000 0x000000	2000246 200024A 2000250 2000250 2000254 2000254 2000254 10 2000256 2000256 2000266 2000266 2000266	00450593 95AA 0005C503 0FF5F593 D1F5 004505B3 00350613 (k = 1 + prime 000505B3 B7F1 ("Nd primes \n". C022 20000537 28950513	addi c add lbu andi c begz .* twice addi c.j connt) c.swsp lui addi	a1. a1, a1. a0 a1. a1. a1. a1. a1. a1. s-28 index +) a1. a0. a s2. a1. b ** prise s0. co. 0x20 a0. a0.	03 0x0(a) 0x1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	4 ; Im) ; Im F ; Im Offs: 0 ; Im Offs: n 10th Imn 0 Imn: 5 9 ; Im	un: 4 un: 0 un: 25 un: 3 un: 3 un: 5000 336070
c Input Dtl codes Buffer scen	S Options	Timeline • # X	CYCLECOUNTER CCTIMER1 CCTIMER2 CCSTEP	91216 91216 91216 66594	2000026E) 20000272 20000274 20000276 20000276 2000027C 2000027C 2000027C 20000282 20000284 20000288 20000284	1C8000EF 40F2 4462 6105 9002 3031 6920 6574 6172 6974 0A736E5F 2500 2064	call20 c.lwsp c.addil6ap c.jr c.jal c.flw c.flw c.flw c.flw c.flw p c.flw jal c.fld c.fld	printf ra. w0. 00 ra \$-2036 fs0. fa3. fs2. fa3. t3. \$+223 fs0. fs1. fs1.	0x1C : 0x18 : 220 : 0x50(a) 0x4C(a) 0x1C : 0x54(a) 1398 :	Inn: 3 Offs: () : In () : In Inn: 2 () : In Offs: () : In	14 12 12 13 14 15 15 15 15 15 15 15 15 15 15 15 15 15



Copyright © 2021 Syntacore. All trademarks, product, and brand names belong to their respective owners.



12



IP collateral (what is included)

Standard core package (SCR7)

- RISC-V compatible core
 - RV64GC ISA
 - RTL (encrypted for evaluation stage), suitable for simulation and synthesis
 - Netlist for the required FPGA devices (Xilinx/Altera)
- Simulation and verification environment
 - Testbench, Integration verification environment
 - Architectural and compliance tests suites (pre- and post-si)
- Synthesis support harness
 - sample scripts, SDC/timing constraints for the required flow
- Reference instantiation examples
- Back-end support @ required process node (PDK access to be provided)
 - Full cycle: synthesis, floor-planning, netlist verification, PaR/CTS/timing closure, DRC, FEV, DFT)
- Support for 1 tapeout up to a year is included

Tools (pre-built & sources)

- GCC based toolchain
 - complier, debugger, linker, functional simulator, binutils, newlib, openocd
- Eclipse-based IDE (Linux, Windows)



FPGA-based SDK

- Sample FPGA project (open design)
- pre-build FPGA and SW images

SW:

- First stage bootloader (SC-BL)
- Linux for the SDK board, including BSP
- Tests/application samples

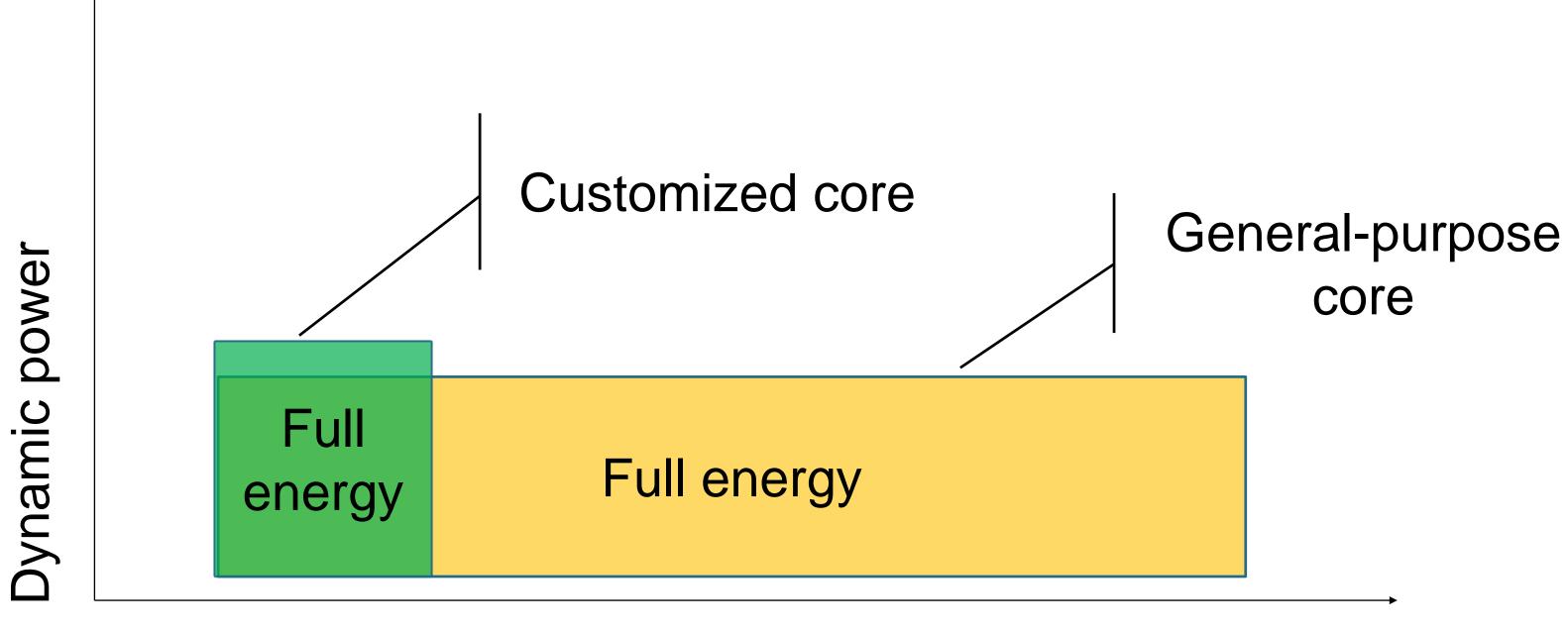
Documentation

- SCRx quick-start guide (user manual)
- SCRx EAS (External architecture specification)
- SCRx ISM (Instruction set manual)
- SCRx SDK guide
- Integration verification environment guide
- Tools guide (IDE & CLI)





Extensibility/customization: how it works



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
 - CV/ML/AI
 - Wireless/Comms
 - Network/DPI/real-time processing





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 68ooK @ 3.4GHz, g++ 5.4.o, Linux 64, optimized C++ implementation
- 60..575x speedup (a) modest area increase: 11.7% core, 3.7% at the CPU cluster level

		Encodir	ng throughput,	MB/s	Normali	zed per MHz	z, MB/s		RV32G +	
Platform	Fmax, MHz	Crypto-1	Crypto-2	AES-128	Crypto-1	Crypto-2	AES-128			speed
RV32G	20	0.025	0.129	0.238	0.00125	0.00645	0.0119	575.00	117.74	60
RV32G + custom	20	14.375	15.188	14.502	0.71875	0.7594	0.7251			
Core i7	3400	79.115	235.343	335.212	0.02327	0.06922	0.09859	30.89	10.97	-
Core i7 + NI	3400			3874.552			1.13957			

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









Getting access/evaluation

SCR1

- SHL-licensed with unrestricted commercial use allowed
 - Commercial SLA-based support is available

SCR 3|4|5|7

• Full package* access is available after simple evaluation agreement

For more info: evaluation@syntacore.com

(*) sufficient for evaluation and tapeout



Is fully open: <u>https://github.com/syntacore/scr1</u> and <u>https://github.com/syntacore/scr1-sdk</u>







Summary

- Syntacore offers high-quality RISC-V compatible CPU IP Founding member, fully focused on RISC-V since 2015 Silicon-proven and shipping in full-wafer production Turnkey IP customization services
 - with full tools/compiler support







