RISC V

- RISC-V-это бесплатная и открытая ISA, открывающая новую эру инноваций процессоров благодаря открытому стандартному. RISC-V ISA обеспечивает новый уровень свободной, расширяемой программной и аппаратной свободы в архитектуре, прокладывая путь для следующих 50 лет вычислительного дизайна и инноваций.

- **Спецификация доступна для свободного и бесплатного использования**, включая коммерческие реализации непосредственно в кремнии или конфигурировании ПЛИС. Имеет встроенные возможности для расширения списка команд и подходит для широкого круга применений.
- Создана в 2010 году исследователями из отделения информатики Калифорнийского университета в Беркли при непосредственном участии Дэвида Паттерсона[2][3]. Для развития и продвижения RISC-V в 2015 году создан международный фонд RISC-V[4] и ассоциация со штаб-квартирой в Цюрихе[5]; с 2018 года RISC-V Foundation работает в тесном партнёрстве с The Linux Foundation.
- **Стандарт RISC-V определяет сравнительно небольшое число стандартных инструкций, около 50 штук, многие из которых были типичны ещё для ранних RISC-I 1980 года. Стандартные расширения (M, A, F и D) расширяют набор на 53 инструкции, сжатый формат C определяет 34 команды. Используется 6 типов кодирования инструкций (форматов).**


RISC-V foundation now > 230 members.
RISC-V
Free, open, extensible ISA for all computing devices

Community Member Benefits

- Accelerated development, reduced risk through open source, ratified ISA.
- Eligible to participate in workgroups, influence strategy and adoption
- 6 support programs in Technical Deliverables, Compliance, Visibility, Learning, Advocacy, and Marketplace
- 1 voting Academic Board rep
- 1 non-voting Community Board rep
- Member logo / name listing on RISC-V website, by member level
- Event registration discount

This membership level is open to academic institutions, non-profits, and individuals not representing a legal entity and for organizations that are integrating RISC-V and want to have their voice considered in the RISC-V technical direction.

Community Member Requirements

- Membership open to
  - academic institutions
  - registered non-profits
  - individuals not representing a legal entity
- No annual membership fee

Strategic Member Benefits

- Community level benefits plus...
- Use of RISC-V Trademark for commercialization
- 3 Board reps elected for Strategic tier, includes Premier members that do not otherwise have a board seat.
- Eligible to lead workgroup and/or committee
- Solution / Product listing highlighted on the RISC-V Exchange
- 1 case study a year
- 1 blog per month
- 1 social media spotlight per month

This level is perfect for those organizations that are integrating RISC-V and want to have their voice considered in the RISC-V technical direction.

Strategic Member Requirements

- Membership open to any type of legal entity
- Annual membership fee based on employee size
  - 5,000+ employees $35k
  - 500-5,000 employees $15k
  - <500 employees $5k
  - <10 employees & organization <2 yrs old: $2k*

Premier Member Benefits

- Community level benefits plus...
- Use of RISC-V Trademark for commercialization
- Board seat and Technical Steering Committee seat included for $250k level
- Technical Steering Committee seat included for $100k level
- Solution / Product listing highlighted on RISC-V Exchange
- 4 case studies a year
- 2 blogs per month
- 2 social media spotlights per month
- Spotlight member profile
- Inclusion in event promotions

This level is perfect for those organizations that want a seat at the table on the Board of Directors, the Technical Steering Committee, speaking slots at events, and enhanced communications coverage via RISC-V content and social.

Premier Member Requirements

- Membership open to any type of legal entity
- $250k Annual membership fee that includes Board seat and TSC seat
- $100k Annual membership fee that includes TSC seat

Российские компании

Не только ARM: новые чипы Samsung будут использовать ядра RISC-V

Да, это про микроконтроллер на **RISCV**.

В Зеленограде создали первый в России чип микроконтроллера с открытым ядром

24.07.20 3

Зеленоградский «Миландр» заявил об успешной разработке и запуске в производство первого в России чипа микроконтроллера с так называемым открытым ядром, использование которого позволяет компании не делать лишних «авторских» отчислений.

ALIBABA ПРЕДСТАВИЛА RISC-V ПРОЦЕССОРЫ С ОТКРЫТЫМ ИСХОДНЫМ КОДОМ

Микрон и НИИМА «Прогресс» выпустят первый российский 32-битный микроконтроллер с ядром RISC-V и встроенной ГОСТ-криптозащитой для IoT

MobiDevices | гаджеты, технологии, обзоры @... · 12 июн. 2019 г.

#Qualcomm инвестировала в производителя процессоров RISC-V.

#RISCV

mobidevices.ru/qualcomm-risc-v

I want to join RISC-V now!

Questions? Email info@riscv.org

Please review the RISC-V International Association Documents and Next Steps below, and click the button to sign up!

Free Community level members (Individual and organizational) may not use the RISC-V trademark or tradename for any commercial purpose. Commercialization requires a paid Strategic or Premier membership.

RISC-V International Association Documents

The following documents are referenced in the RISC-V Membership application:

- RISC-V International Regulations
- Articles of RISC-V International Association
- RISC-V Membership Agreement – NOT FOR SIGNING – please sign up at the link below

The Standard Extensions

- **Extensions define instructions**
  - "I" for Integer is the only required extension in a RISC-V implementation and defines 40 instructions
- **The RISC-V Specification defines a number of "Standard Extensions"**
  - Standard Extensions are defined by the RISC-V Foundation and are optional
- **RISC-V allows for custom, "Non-Standard", extensions in an implementation**
- **Putting it all together (examples)**
  - RV32I – The most basic RISC-V implementation
  - RV32IMAC – Integer + Multiply + Atomic + Compressed
  - RV64GC – 64bit IMAFDC
  - RV64GCXext – IMAFDC + a non-standard extension

Extension	Description
I	Integer
M	Integer Multiplication and Division
A	Atomics
F	Single-Precision Floating Point
D	Double-Precision Floating Point
G	General Purpose = IMAFD
C	16-bit Compressed Instructions
Non-Standard User-Level Extensions	
Xext	Non-standard extension "ext"

Common RISC-V Standard Extensions
***Not a complete list**

Register File

- **RV32I/64I have 32 Integer Registers**
  - Optional 32 FP registers with the F and D extensions
  - RV32E reduces the register file to 16 integer registers for area constrained embedded devices
- **Width of Registers is determined by ISA**
- **RISC-V Application Binary Interface (ABI) defines standard functions for registers**
  - Allows for software interoperability
- **Development tools usually use ABI names for simplicity**

Register	ABI Name	Description	Saver
x0	zero	Hard-wired zero	-
x1	ra	Return address	Caller
x2	sp	Stack pointer	Callee
x3	gp	Global pointer	-
x4	tp	Thread pointer	-
x5-7	t0-2	Temporaries	Caller
x8	s0/fp	Saved register/Frame pointer	Callee
x9	s1	Saved register	Callee
x10-11	a0-1	Function Arguments/return values	Caller
x12-17	a2-7	Function arguments	Caller
x18-27	s2-11	Saved registers	Callee
x28-31	t3-6	Temporaries	Caller

- Микроконтроллеры, выпущенные в 2020 году:
- **ONiO: ONiO.zero** (16/32 бита, 1 кБ ПЗУ, 2 кБ ОЗУ, 8/16/32 кБ ППЗУ, 1-24 МГц, 0,36-1,44 Вт, встроенный радиоэлектро генератор на 800/900/1800/1900/2400 МГц) BLE, 802.15.4 UWB[44][45]
- **WCH: CH32V103** (32 бита, 10/20КБ ОЗУ, 32/64 КБ ППЗУ, до 80 МГц, корпуса LQFP48, QFN48 или LQFP64)[46] универсальный контроллер с USB 2.0, SPI, I2C, GPIO, USART, TouchKey, RTC, TIM, ADC
- ***Миландр: К1986ВК025*** *(32-битное ядро ВМ-310S CloudBEAR, ОЗУ 112 Кбайт, ППЗУ 256+8 Кбайт, ПЗУ 16 Кбайт, 60 МГц, сопроцессоров для шифров «Кузнечик», «Магма» и AES, корпус QFN88 10 x 10 мм)*
- **Espressif: ESP32-C3** (32-битное ядро RV32IMC, 400 Кбайт SRAM, 384 Кбайт ПЗУ, 160 МГц, Wi-Fi, Bluetooth LE 5.0, по контактам совместим с ESP8266)[47]
- **Bouffalo Lab: BL602 и BL604** (32-битный, динамическая частота от 1 МГц до 192 МГц, 276 КБ SRAM, 128 КБ ПЗУ, Wi-Fi, Bluetooth LE)[48]
- **Cmsemicon: ANT32RV56xx** (ядро RV32EC, 48 МГц, 32+8 Кбайт SRAM, 64 Кбайт)[49]

- **IP-ядра**[править | править код]
- Ряд компаний предлагают готовые блоки IP-ядер на базе архитектуры RISC-V, среди них:
  - ECHX1 — компания Western Digital (США),
  - Rocket — Калифорнийский университет в Беркли и компания SiFive (США),
  - ORCA — компания Vectorblox (Канада),
  - PULPino — Высшая техническая школа Цюриха (Швейцария) и Болонский университет (Италия),
  - Hummingbird E200 — компания Nuclei System Technology (Китай),
  - AndeStar V5 — компания Andes Technology (Тайвань)[15]),
  - Shakti — Индийский технологический институт в Мадрасе (Индия),
  - BM-310, BI-350, BI-651, BI-671 — компания Клаудбеар (Россия),
  - Семейство SCR компании Синтакор (Россия)[16].

BR-351

Компактное 32-битное высокопроизводительное процессорное RISC-V ядро. Идеально подходит для управляющих и вычислительных задач, требующих высокой производительности при ограниченном энергопотреблении.

Подробнее »

BR-651

64-битное высокопроизводительное процессорное RISC-V ядро. Идеально подходит для управляющих и вычислительных задач, требующих высокой производительности и поддержки 64-ных возможностей.

Подробнее »

BM-310

32-битное процессорное RISC-V ядро. Оптимизировано для минимизации энергопотребления и занимаемой площади при сохранении наилучшей производительности в своем классе. Идеально подходит для задач управления и IoT устройств.

Подробнее »

Процессорные ядра с поддержкой Linux

BI-350

Процессорный комплекс на базе 32-битного RISC-V ядра с последовательным исполнением команд. Облегченная конфигурация, оптимизированная для минимизации энергопотребления и занимаемой площади, идеально подходит для применения в IoT устройствах под управлением ОС Linux.

BI-651

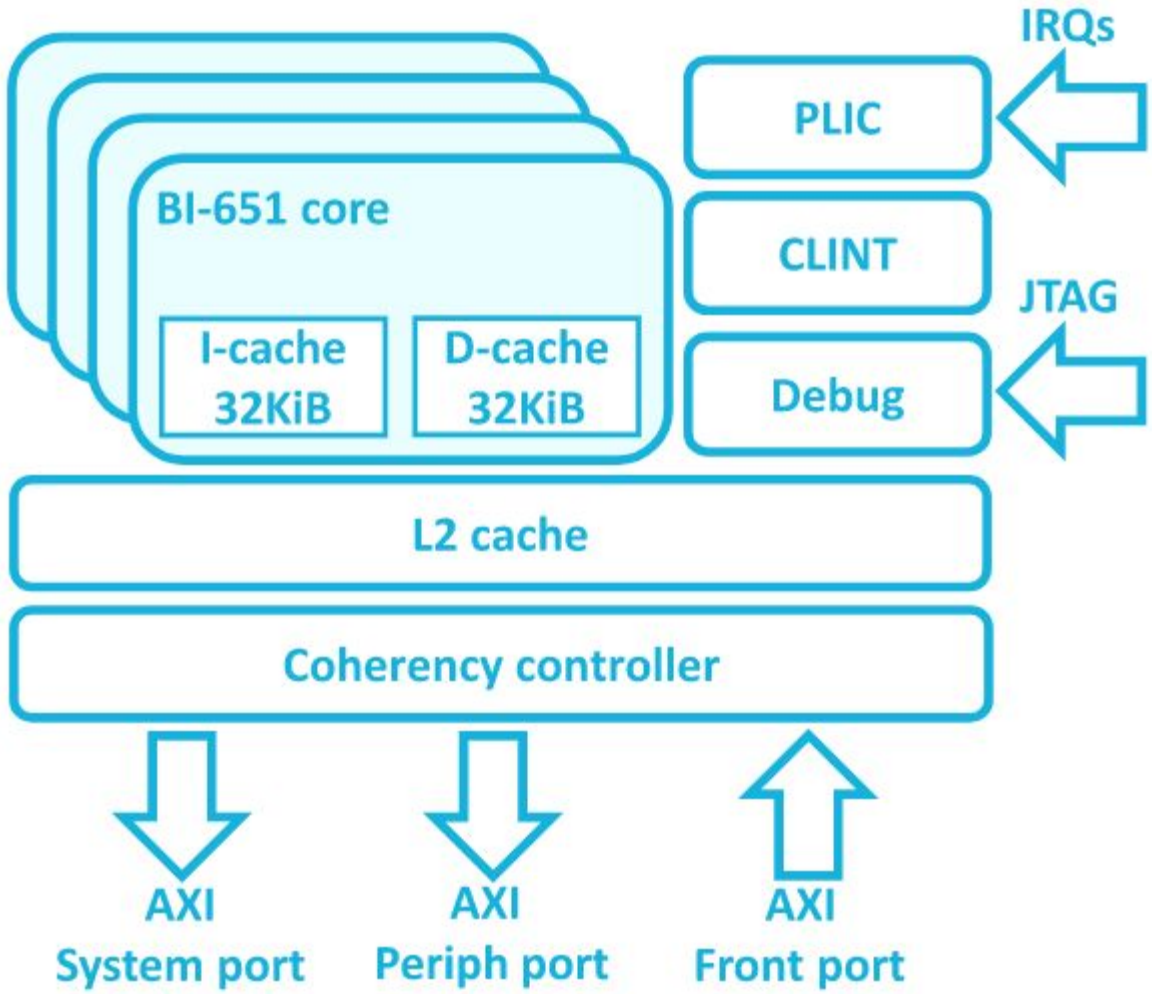
Процессорный комплекс на базе 64-битного RISC-V ядра с последовательным исполнением команд. Разработан для применения в устройствах под управлением ОС Linux, требующих высокой производительности при ограниченном энергопотреблении.

Подробнее »

BI-671

Процессорный комплекс на базе 64-битного RISC-V ядра с внеочередным исполнением команд. Разработан для применения в устройствах под управлением ОС Linux, требующих максимальной однопоточной производительности.

Подробнее »


IRQs
PLIC
BI-651 core
CLINT
JTAG
I-cache
32KiB
D-cache
32KiB
Debug
L2 cache
Coherency controller
AXI
System port
AXI
Periph port
AXI
Front port

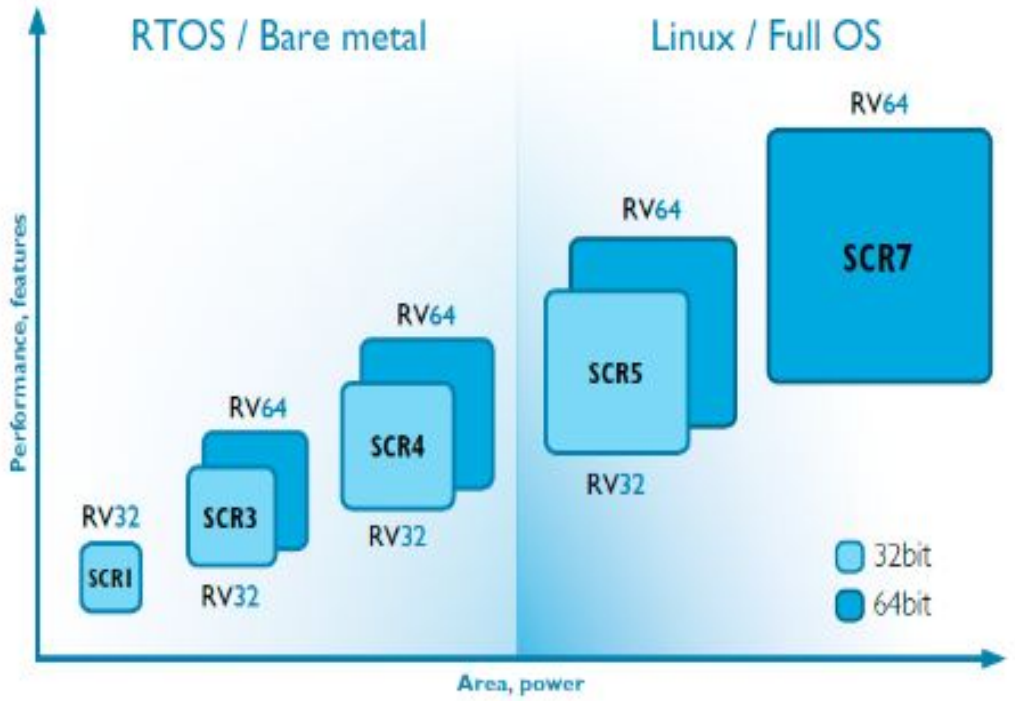
- 64-битное ядро RISC-V с 32-мя целочисленными регистрами (I расширение)
- Целочисленное умножение и деление (M расширение). Атомарные операции (A расширение)
- 16-битные инструкции для увеличения плотности кода (C расширение)
- IEEE 754-2008 совместимые вычисления с плавающей запятой (F+D расширения)

- Исполнение до 2-х инструкций в такт. До 4-х ядер в комплексе
- Machine, Supervisor и User уровни привилегированности
- 9-ти стадийный конвейер с последовательным исполнением команд
- Предсказание переходов: BTB, BHT, RAS. Sv39 режим виртуальной памяти
- 32 КБ 8-канальный L1 кэш инструкций. 32 КБ 8-канальный L1 кэш данных. Интегрированный 1-2 МБ L2 кэш
- Прерывания:
  - Platform Level Interrupt Controller (PLIC): 127 прерываний с 8-ю уровнями приоритета
  - Multi-Core Local Interruptor (CLINT): таймер и программные прерывания
- Контроль доступа к физической памяти
- Интегрированный контроллер отладки с поддержкой аппаратных точек останова
- AXI интерфейс к системной шине. AXI интерфейс к периферийным блокам.AXI порт для когерентного доступа ускорителей
- Производительность: 2.5 DMIPS/МГц, 5.0 CoreMark/МГц, 3.18 SPEC2006 INT/ГГц
- Частота: 1 ГГц (TSMC, 40нм G, при наихудших условиях),1.2 ГГц (TSMC, 28нм HPC+, при наихудших условиях)

Syntacore

Syntacore является одним из сооснователей головной организации RISC-V Foundation. Компания занимается разработкой 32-битных ядер на базе этой архитектуры, их кастомизацией под нужды конкретных заказчиков (DSP, акселераторы, оптимизация) и подготовкой ПО для разработки софта. Её ядра одними из первых были реализованы «в кремнии». Она же одной из первых продемонстрировала работу Linux на многоядерном CPU RISC-V. В портфолио компании есть три версии ядра для микроконтроллеров: SCR1, SCR3 и SCR4. Причём SCR1 является полностью открытым решением. Старшая модификация SCR5 рассчитана уже на работу с обычными ОС: 32/64-бит, 1-4 ядра, частота от 1 ГГц, 28 нм TSMC.

Stable, silicon-proven, competitive, available for evaluation



The SCR family includes **eight** cores, targeted at wide range of embedded applications: SCR1, SCR3, SCR4, SCR5 and SCR7 with more designs on the company roadmap .

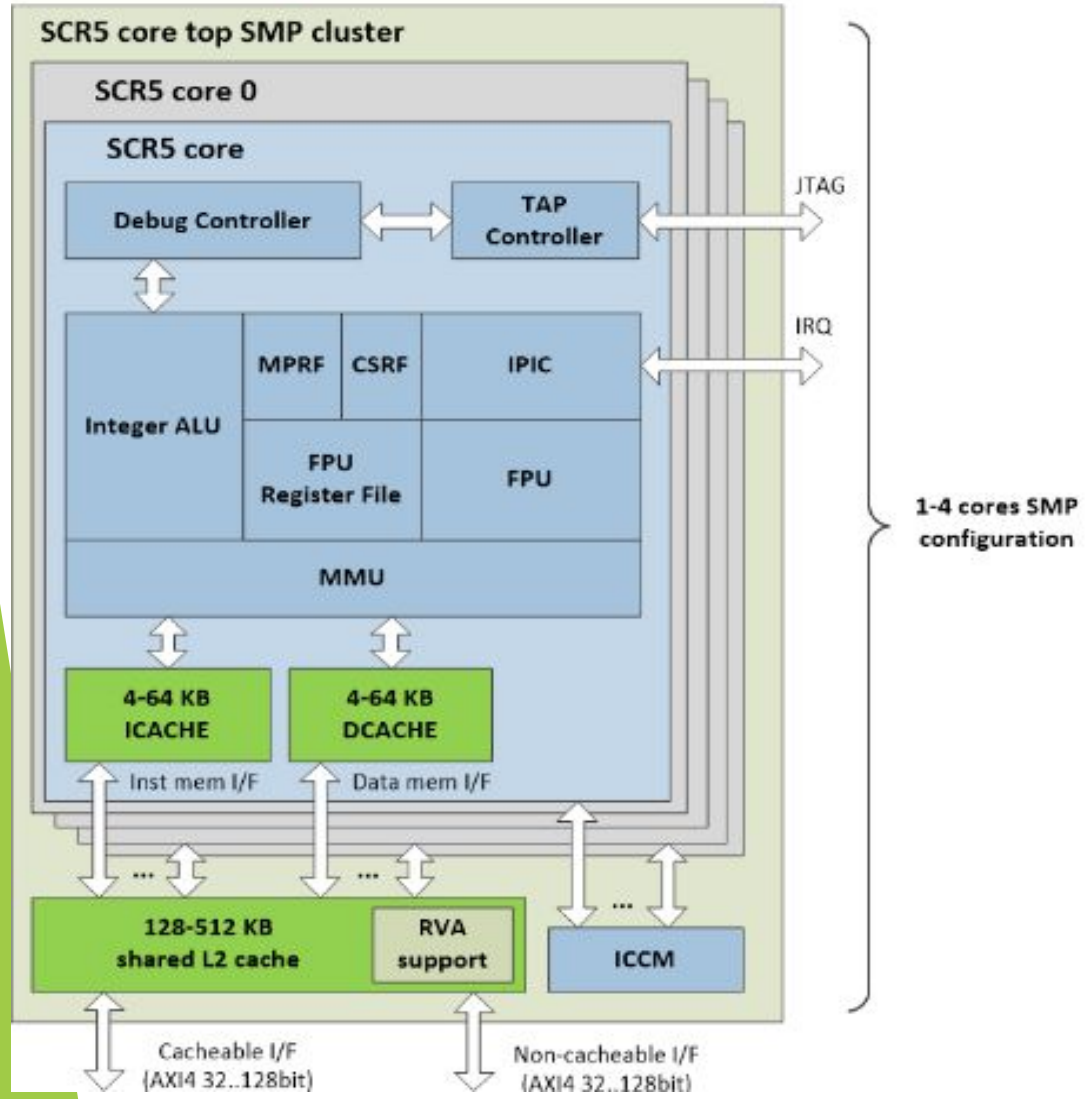
- **SCR1** Compact 32-bit MCU core for deeply embedded applications and accelerator control. Open-sourced under the permissive SHL license, which allows commercial use
- **SCR3** High-performance MCU core with privilege modes and MPU (32 or 64 bit)
- **SCR4** MCU core with high-performance FPU (32 or 64 bit)
- **SCR5** Entry-level APU/embedded core with Linux and SMP configurations support
- **SCR7** Efficient high-performance 64bit APU core with SMP up to 8-16 cores

	Open	AXI4	FPU	MMU	SMP	Linux	Cache
SCR1	+	-	-	MCU	-	-	TCM
SCR3	-	-	-	MCU	-	-	L1 caches
SCR4	-	+	+	MCU	-	-	L1 caches
SCR5	-	+	+	+	+	+	L1/L2 caches
SCR7	-	+	+	+	+	+	L1/L2 caches

SCR5 efficient application core (RV32 or RV64)

Efficient 32- or 64bit Linux-capable application core with virtual memory, MMU, L1/L2 caches, coherency and SMP support

Block diagram



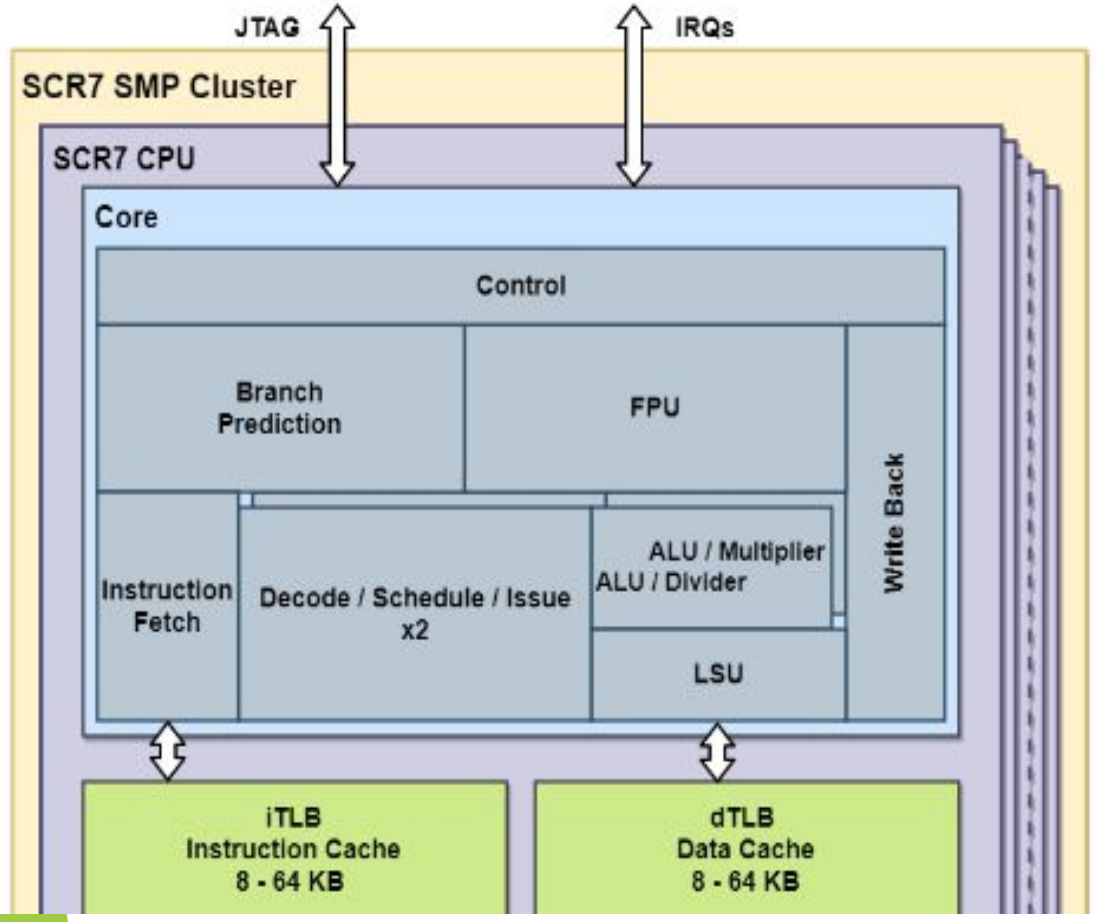
Key features

- Efficient entry-level 32- or 64bit RISC-V application core
- **RV32IMC[AFD]** or **RV64IMC[AFD]** ISA
- Harvard architecture, separate Instruction and Data memories
- Multicore configs support up to 4 SCRx cores
  - SMP and heterogeneous
- 7 to 9 stages pipeline, **1GHz**+ @tsmc28
- User-, Supervisor- and Machine-mode privilege levels
- Fully-featured memory subsystem with **Linux** support
-
  - Memory Managements Unit (MMU)
  - Page-based virtual memory
  - L1 and L2 caches with coherency, HW atomics, ECC
- High-performance IEEE 754-2008 compliant floating-point unit
  - Configurable single or double precision FP unit
  - 32 floating-point data registers
- AXI4- or AHB- compliant external interface
- Configurable Integrated Programmable Interrupt Controller (IPIC) and PLIC
  - up to 1024 IRQs
  - Low interrupt latency
- Advanced Integrated Debug Controller
  - JTAG compliant interface
  - HW/SW breakpoints support
  - ROM breakpoints support

SCR7 high-performance 64bit application core

64bit Linux-capable application core with virtual memory, MMU, L1/L2 caches, coherency and SMP support up to 8 cores per cluster

Block diagram



Key features

- High-performance 64bit RISC-V application core
- **RV64GC** ISA
- Flexible uarch template, 10-12 stage pipeline
- User-, Supervisor- and Machine-mode privilege levels
- Fully-featured memory subsystem with **Linux** support
- - Memory Managements Unit (MMU)
  - Page-based virtual memory
  - L1 and L2 caches with coherency, HW atomics, ECC
- High-performance IEEE 754-2008 compliant floating-point unit
- AXI4- or ACE- compliant external interface
- Configurable Integrated Programmable Interrupt Controller (IPIC) and PLIC
  - up to 1024 IRQs
- Advanced Integrated Debug Controller
  - JTAG compliant interface
  - HW/SW breakpoints support
  - ROM breakpoints support

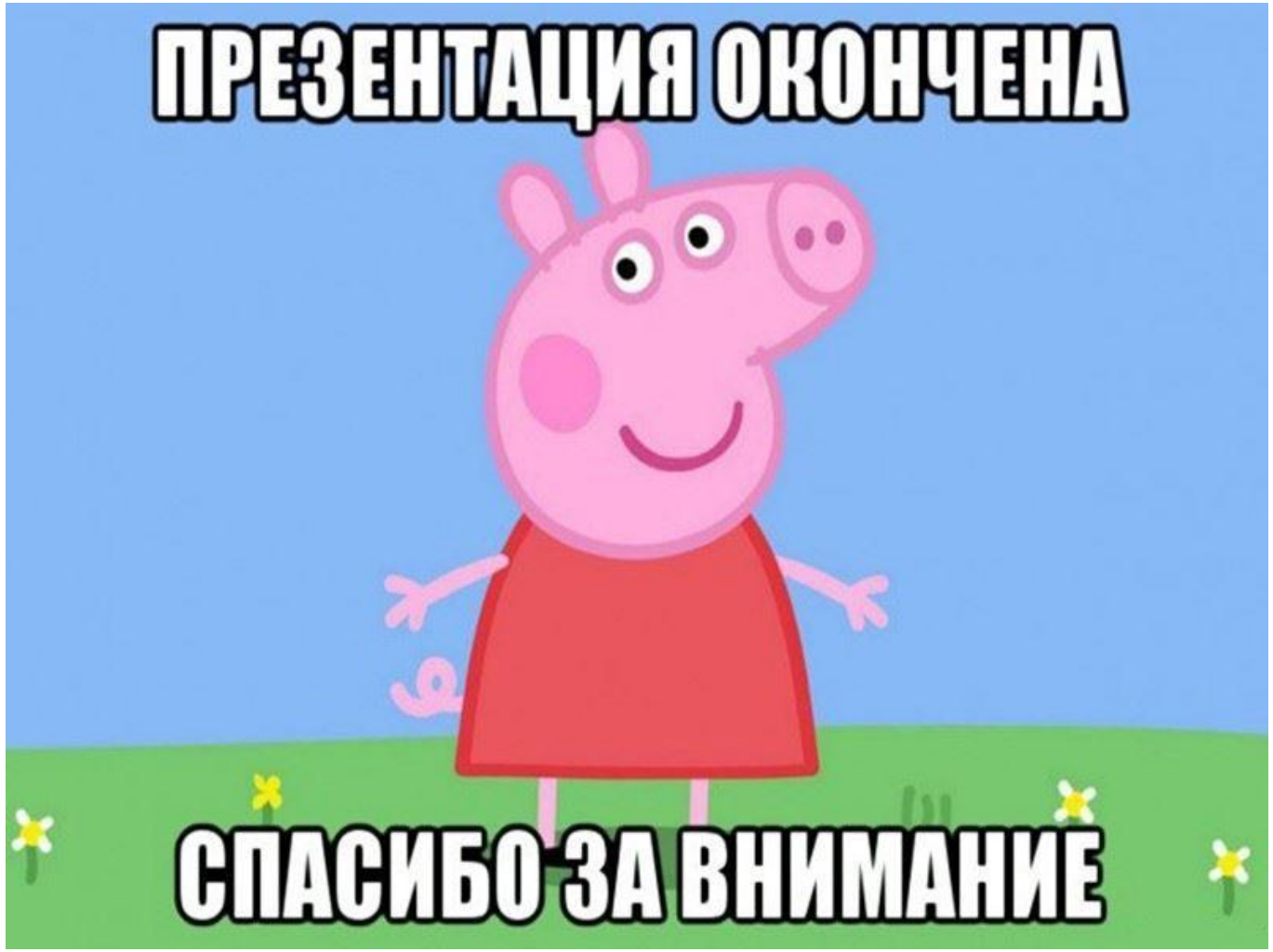
For all processor IP, Syntacore provides fully-featured, powered by the open-source SW development tools with technical support:

- Stable GCC-based C/C++ compiler toolchain
- Assembler, linker, binutils
- Standard C/C++ libraries for Linux and Bare Metal environments
    - glibc and newlib
- GDB debugger
- Low-cost openOCD-based JTAG connectivity support
- Eclipse-based IDE
- FPGA-based SDKs

The RISC-V SW ecosystem is diverse and rapidly growing, with stable Linux, FreeBSD, QEMU, GCC, binutils, number of RTOS/kernel ports and other SW packages available.

Pre-built tools for the SCR1 core and SDK can be downloaded at the following links:

- Linux
- Windows


ПРЕЗЕНТАЦИЯ ОКОНЧЕНА
СПАСИБО ЗА ВНИМАНИЕ

Example instruction	Instruction name	Meaning
`ld x1,80(x2)`	Load doubleword	Regs[x1] ← Mem[80+Regs[x2]]
`lw x1,60(x2)`	Load word	Regs[x1] ←$_{64}$ Mem[60+Regs[x2]]$_0$)^{32} ## Mem[60+Regs[x2]]
`lwu x1,60(x2)`	Load word unsigned	Regs[x1] ←$_{64}$ 0^{32} ## Mem[60+Regs[x2]]
`lb x1,40(x3)`	Load byte	Regs[x1] ←$_{64}$ (Mem[40+Regs[x3]]$_0$)^{56} ## Mem[40+Regs[x3]]
`lbu x1,40(x3)`	Load byte unsigned	Regs[x1] ←$_{64}$ 0^{56} ## Mem[40+Regs[x3]]
`lh x1,40(x3)`	Load half word	Regs[x1] ←$_{64}$ (Mem[40+Regs[x3]]$_0$)^{48} ## Mem[40+Regs[x3]]
`flw f0,50(x3)`	Load FP single	Regs[f0] ←$_{64}$ Mem[50+Regs[x3]] ## 0^{32}
`fld f0,50(x2)`	Load FP double	Regs[f0] ←$_{64}$ Mem[50+Regs[x2]]
`sd x2,400(x3)`	Store double	Mem[400+Regs[x3]] ←$_{64}$ Regs[x2]
`sw x3,500(x4)`	Store word	Mem[500+Regs[x4]] ←$_{32}$ Regs[x3]$_{32..63}$
`fsw f0,40(x3)`	Store FP single	Mem[40+Regs[x3]] ←$_{32}$ Regs[f0]$_{0..31}$
`fsd f0,40(x3)`	Store FP double	Mem[40+Regs[x3]] ←$_{64}$ Regs[f0]
`sh x3,502(x2)`	Store half	Mem[502+Regs[x2]] ←$_{16}$ Regs[x3]$_{48..63}$
`sb x2,41(x3)`	Store byte	Mem[41+Regs[x3]] ←$_{8}$ Regs[x2]$_{56..63}$

Example instrucmtion	Instruction name	Meaning
`add x1,x2,x3`	Add	`Regs[x1]←Regs[x2]+Regs[x3]`
`addi x1,x2,3`	Add immediate unsigned	`Regs[x1]←Regs[x2]+3`
`lui x1,42`	Load upper immediate	`Regs[x1]←`0^{32}`##42##`0^{12}
`sll x1,x2,5`	Shift left logical	`Regs[x1]←Regs[x2]<<5`
`slt x1,x2,x3`	Set less than	`if (Regs[x2]<Regs[x3]) Regs[x1]←1 else Regs[x1]←0`