SHAKTI Processors Overview

RISE Lab

IITM

Agenda

- Introduction to Shakti Processors.
- Variants of Shakti
- Shakti SDK
- Shakti Peripherals

Genesis of SHAKTI

- **SHAKTI** is an **IIT-Madras** coordinated initiative to develop a family of open source processors ranging from microcontrollers to server-grade multi-core processors.

- Today the **SHAKTI** initiative is building a comprehensive silicon ecosystem that will provide industry-grade open source processors, AI/ML accelerators, communication fabrics, interconnects and storage controllers.

- 5 startups spun off from this initiative.

Shakti - Journey

SHAKTI was born

2013

- Exploring ISA
- Approached by UCB to adopt RISC-V.
- Defined 3 major classes of cores : E-Class, C-Class & I Class.

Tape outs

2017

- **RISECREEK:**
  - Tapeout of C-class on Intel's 22nm FFL
- **RIMO:**
  - Tapeout of C-class on SCL's 180nm

Funding from MeiTY

2018

SHAKTI gets official funding, and India Microprocessor Development Program (IMDP) begins.

Incore Startup spinoff from Shakti

Industry adoption

2019

- IGCAR adopts SHAKTI
- ISRO adopts SHAKTI
- Thales funds SafeRV
- **Moushik**:
  - (an E-class based SoC) is taped-out at SCL 180nm

SHAKTI

Shakti - Timeline

2020 — Swadeshi µP Challenge

National level challenge was launched to design innovative solutions based on SoCs designed by SHAKTI .

2021 — Startups

Three more startups spin off from Shakti

- Mindgrove
- Securweave
- Vyoma

2022 — Secure

- Secure Shakti
- Tape in for IISU at SCL
- Fifth startup, Shakra

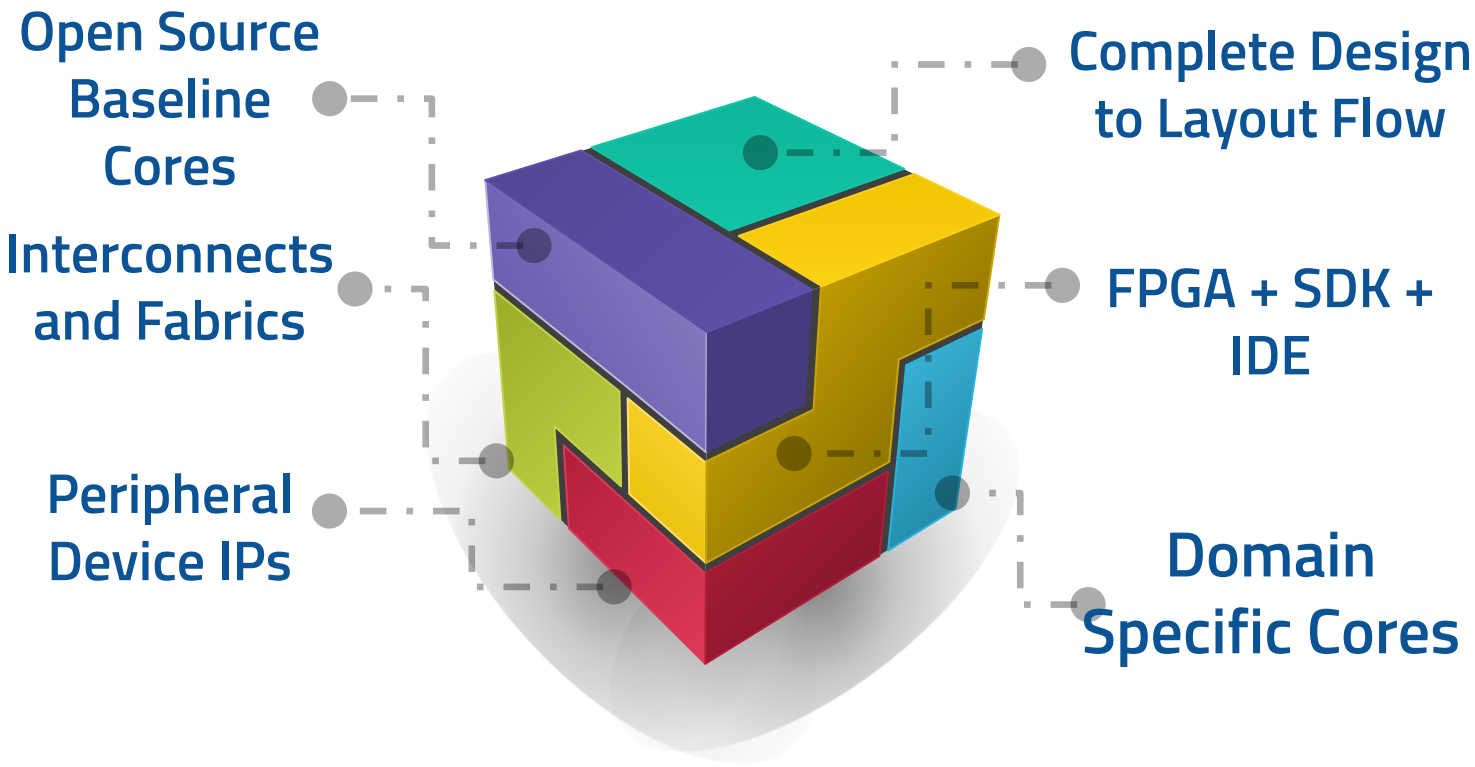
2023 — Accelerate

- Shaktimaan - AI accelerator
- I-class 1.0

2024 — Enhancements

- IP enhancements
- I-class enhancements
- C-Class enhancements
- Tape out of C - Class SoC
- P & V extension for C-Class

Shakti Ecosystem



About RISC-V

- RISC-V was originated by researchers from UC Berkeley in 2010. 2014 is when the standard was made open.
- RISC-V (pronounced “risk-five”) is an **open source implementation** of a RISC (Reduced Instruction Set Computing) based Instruction Set Architecture.
- RISC-V is open under **BSD license**, permitting any person or group to build RISC-V based hardware and software at **no licensing cost**.
- Learnt from the predecessor RISC ISAs.
- A Real ISA, capable of enabling hardware implementations and not only for simulation or binary translation.
- Provides “no” micro-architecture mandate and avoids over architecting for a particular technology (ASIC or FPGA).

Why RISC-V

- Openness (Open Standard, freely licensed)
- Modular architecture - 32 bit, 64 bit instruction set & variety of extensions for features like floating point, vector operations, PSIMD, Bit manipulation, etc.
- Flexibility - easy customization and optimization both software and hardware for specific use cases.
- Designed for pipelined efficiency - pipeline stages can be optimized based on the application.
- Visibility - More visibility into the codebase
- Future-proof - AI will generate future software and hardware.

Why Shakti

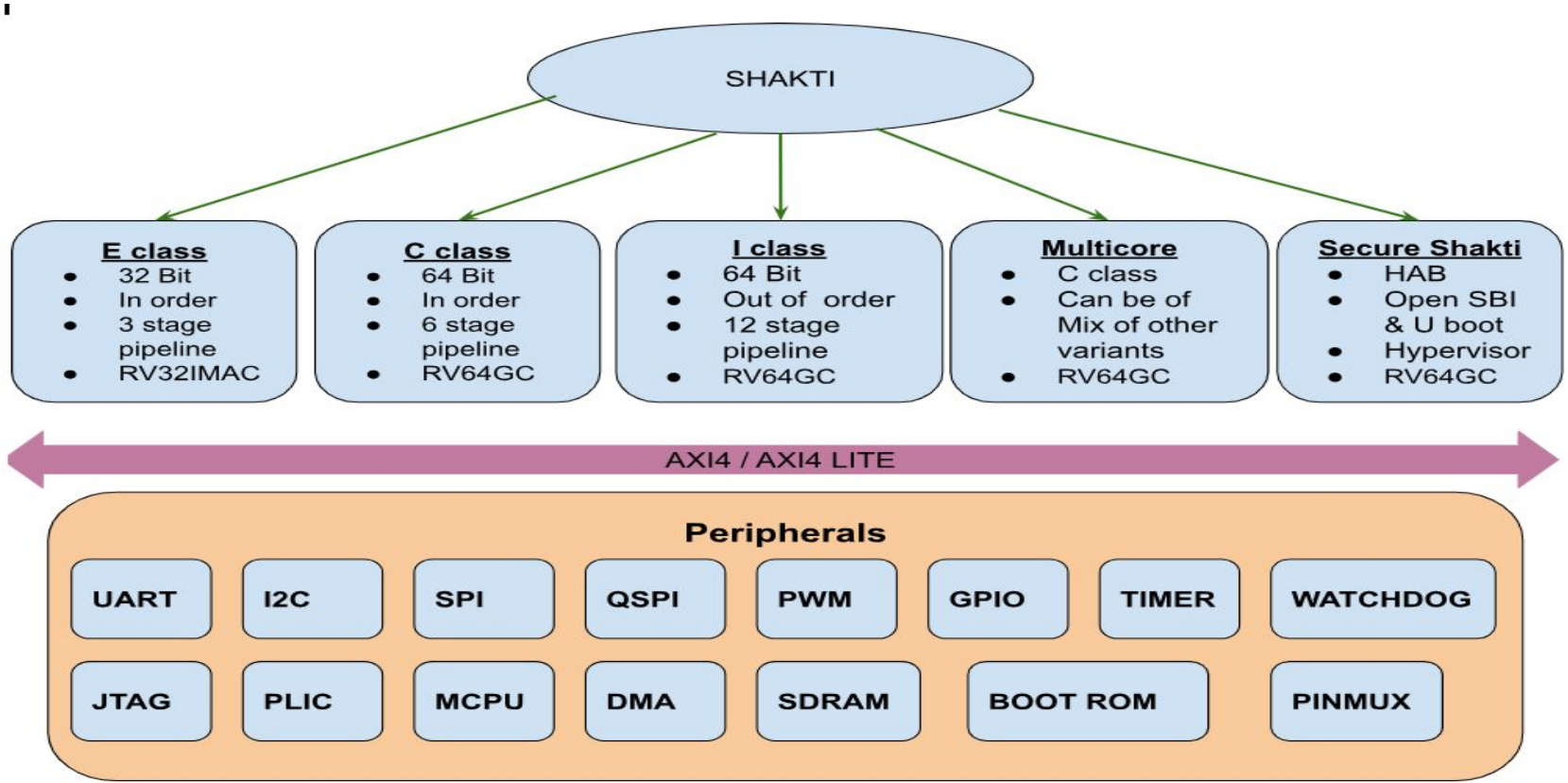
- Customizable Shakti Core variants.
- Multicore Shakti for high performance applications.
- Secure Shakti Core to protect the system from booting to runtime.
- Availability of peripheral IP cores(UART, SPI, QSPI, I2C, GPIO, TIMER, PWM, WATCHDOG TIMER, RTC, etc.)
- Cryptographic Accelerators for SHA, AES, ECC and RSA are available.
- ShaktiMAAN to accelerate the performance of Shakti (optimized for executing AI/ML workloads).
- Variety of Shakti cores are tested with different FPGA boards.
- Four successful Tapeouts.

SHAKTI Overview

- **Base Processors**
  - **E-Class** : 3 stage micro-controller
  - **C-Class** : 6 stage micro-processor
  - **I-Class** : 12 stage OoO processor
  - **Core IP blocks**
    - Single and Double IEEE 754 FPU
    - Parameterized Branch Predictors
    - Parameterized Caches.
- **Interconnects and Fabrics**
  - **Protocols**
    - AXI-4, AXI4 Lite
    - TileLink U/H/C
  - **Fabrics -** Custom, Crossbar,Mesh**,** AI accelerator optimized fabrics
- **Dev/Verification platforms**
  - **AAPG** - Random Assembly Generator
  - Core & SoC **Verification**
  - **RISC-V ISA Compliance**
- **Design Flow**
  - FPGA - Xilinx, Altera
  - ASIC: Cadence, Synopsys, Mentor
- **Peripheral IPs**
  - I2C, xSPI, qSPI, JTAG, UART, PWM, GPIO controller, PLIC, Watchdog, GP Timer.
  - Memory - SDRAM, BootROM, TCM, DMA.
- **SW Stack**
  - OS : Linux, FreeRTOS, Zephyr, Nuttx.
  - SDK and IDE (Arduino & Platform IO)

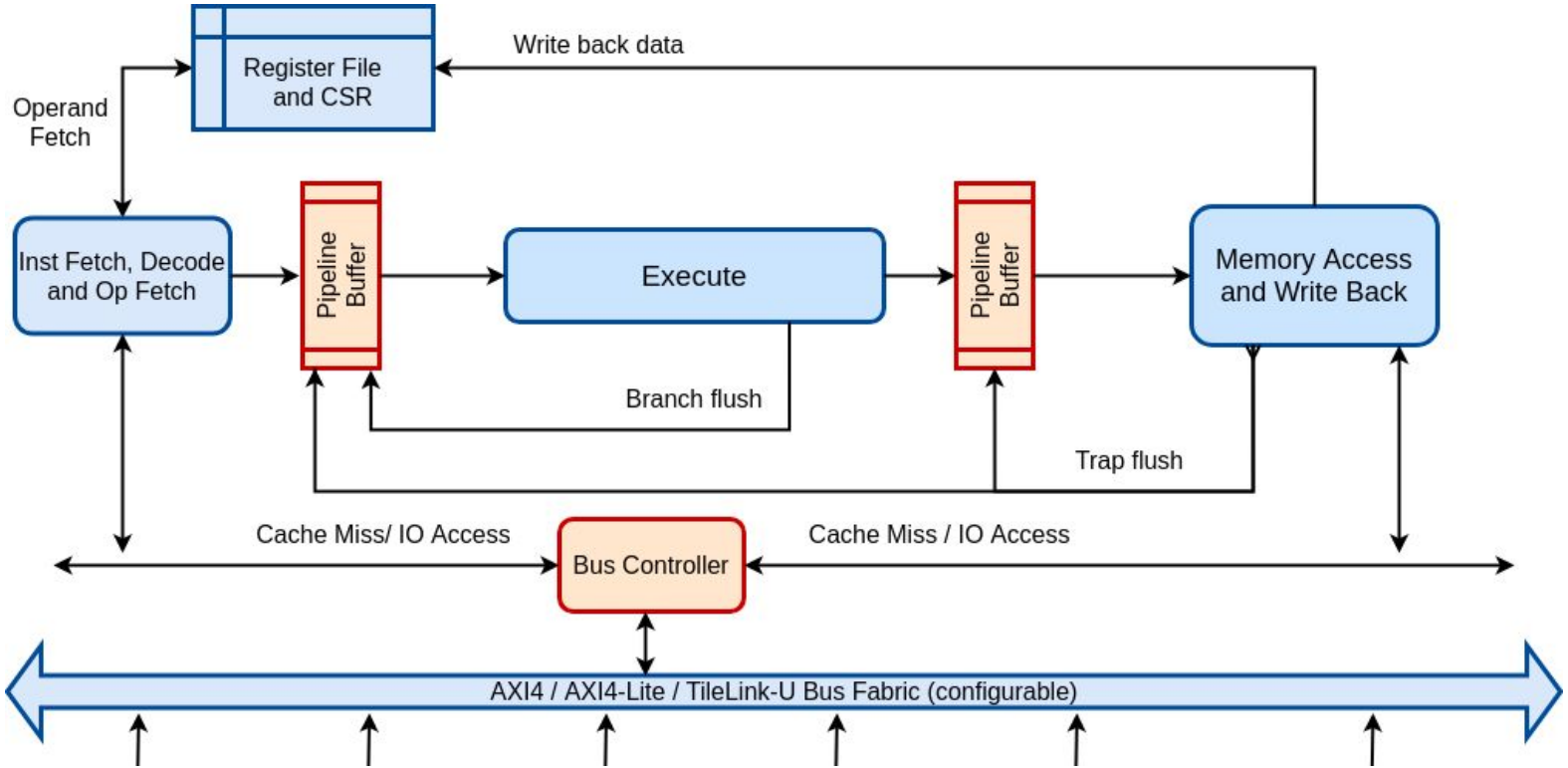
SHAKTI

SHAKTI Design Portfolio



The E-Class Core

E-Class Micro-Arch



E-Class Micro-Controller Core

Overview

- In-order 3 stage 32/64 bit microcontroller supporting a subset of RISC-V ISA.
- Low area and power consumption - operational freq. of < 400MHz.
- Optimized variants for FPGA based soft-cores.
- AXI4/AXI4-Lite/TileLink peripherals supported
- Positioned against ARM's M class cores (Cortex Mx Series)

Specifications

- Open source IP supporting RV*IMAC.
- Supports Machine and User-modes only. User-mode trap handling is optional.
- Push button flow to generate variants and subsets of ISA.
- Optimized sequential Multiplier and Divider for ASICs and FPGAs
- OpenOCD based SoC debug support through JTAG.
- OS Ports: FreeRTOS, Zephyr.

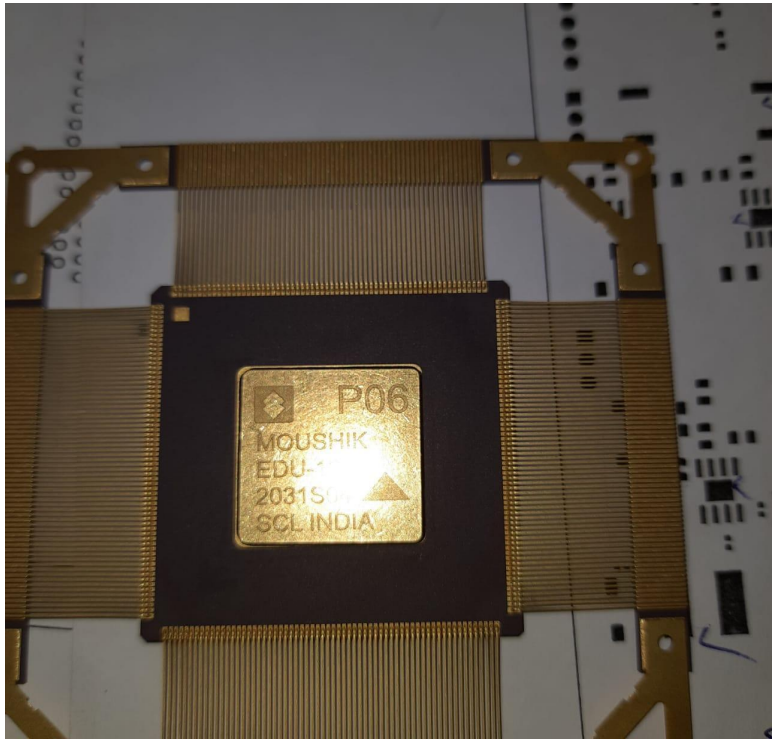
PPA

- Uses <3K LUT on a 7-series Xilinx FPGA.
- DMIPS/MHz: <1

Target Domains: IoT devices, Edge Devices, Robotic Control, Smart cards

Moushik SoC & EVK

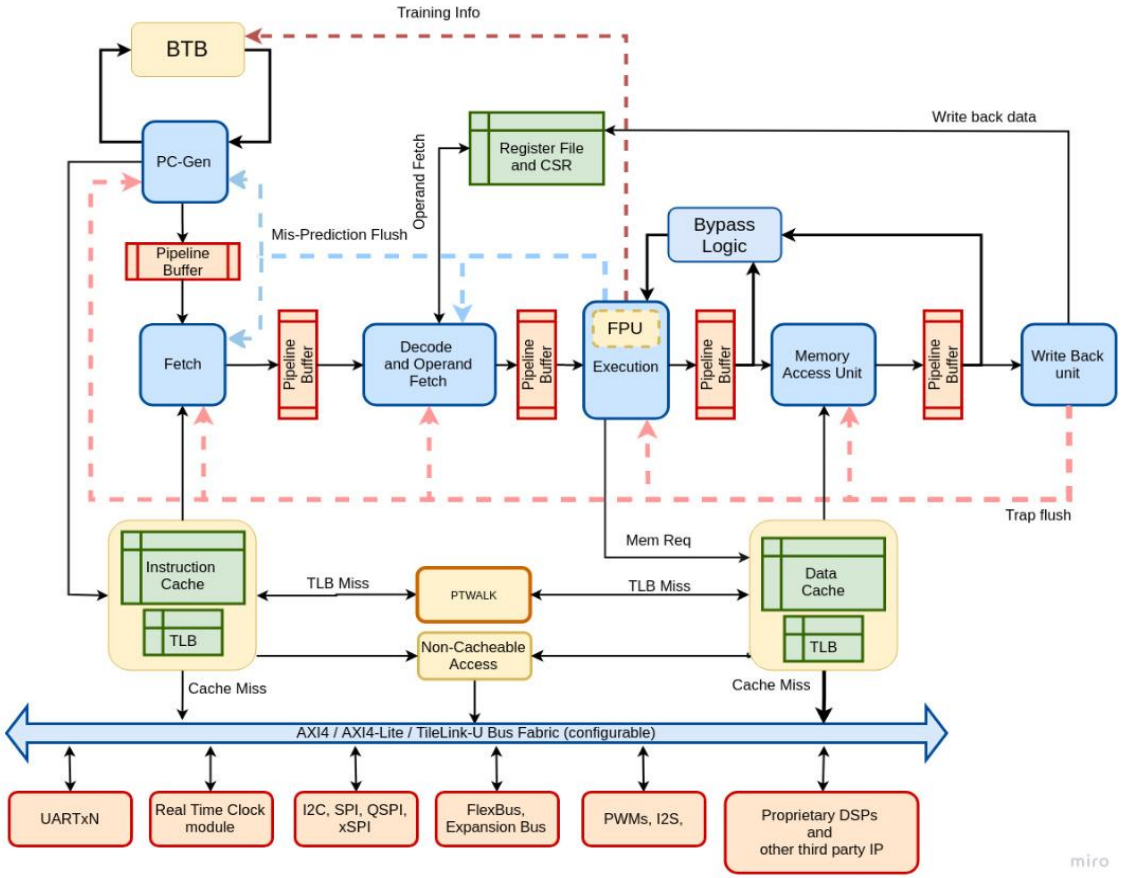
Moushik SoC



The C-Class Core

SHAKTI

C-Class Micro Architecture



Optional Modules:

- Branch Predictor
- Return Address Stack
- Instruction Cache
- Data Cache
- Floating Point Unit
- PTWalk +TLBs (only when Supervisor enabled)

SHAKTI

C-Class Micro-Processor

Overview

- An in-order 6-stage 64-bit core supporting the entire stable RISC-V ISA.
- Targets mid-range compute systems: 500MHz-1+GHz.
- Supports RISC-V Linux, secure L4, Nuttx.
- Variants for low-power and high-performance.
- Positioned against ARM's Cortex A35/A55

Specifications

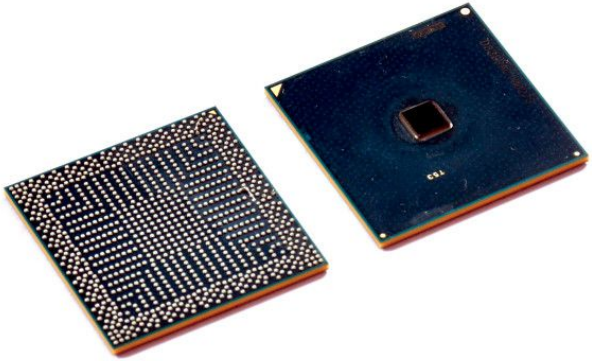
- Supports RISC-V ISA: RV64IMAFDC.
- Compatible with latest privilege spec of RISC-V ISA and supports the sv39/48 virtualization scheme.
- Single and Double Precision Floating point units compliant with IEEE-754.
- Supports the OpenOCD based debug environment through JTAG.
- Includes a High performance branch predictor with a Return-Address-Stack.
- Caches: upto 16KB Instruction and Data caches.
- Includes operand bypass for better performance.
- Boots risc-v linux os.

PPA

- Uses <20K LUT on a 7-series Xilinx FPGA.
- DMIPS/MHz: 1.72
- Coremarks/MHz : 2.91

Target Domains: Reliable Computing, Secure Computing, IoT & Edge Computing hubs, Auto/Aerospace/Industrial Controls

Other Tapeouts demonstrated



- ASIC : RISECREEK
- Manufacturer: Intel
- Date of Shipment: July 2018
- Technology Node: 22nm FinFet
- Die Size: 4 x 4 mm2
- Functional IOs: 324
- Packaging: BGA
- Core Voltage: 0.75V
- IO Voltage: 1.8V

- ASIC : RIMO
- Manufacturer: SCL
- Date of Shipment: October 2018
- Technology Node: 180nm CMOS
- Die Size: 12 x 12 mm2
- Functional IOs: 138
- Packaging: CQFP
- Core Voltage: 1.8v
- IO Voltage: 1.8V / 3.3V

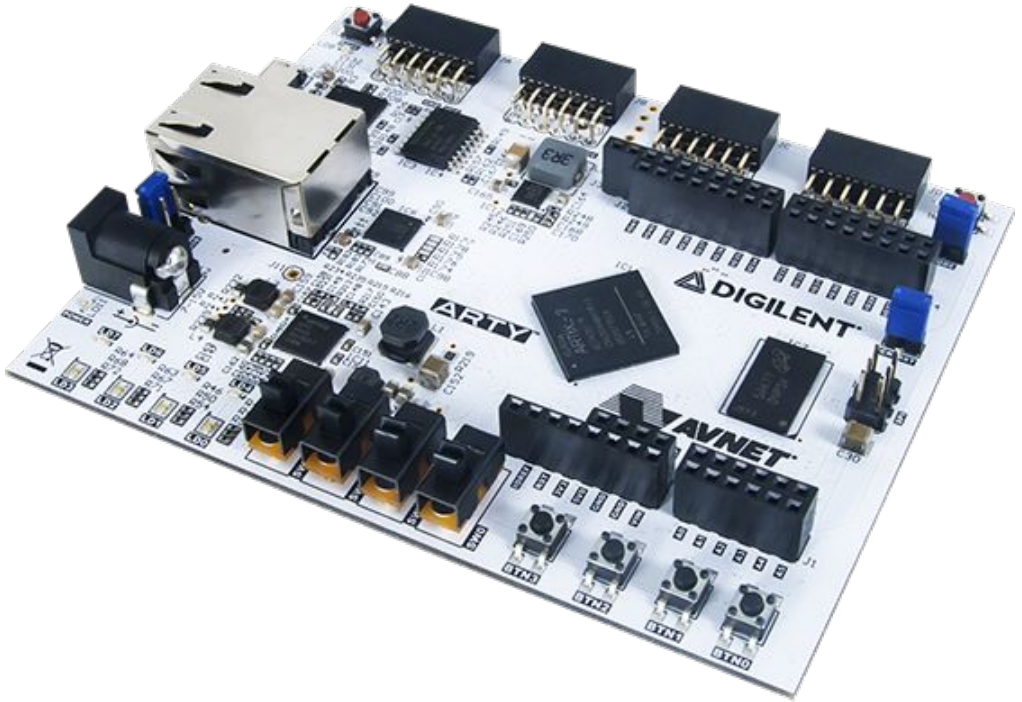
FPGA Based SoC Templates

Board Support

Shakti based SoCs are supported on Multiple Boards

- Nexys Video
- AC701
- VCU118
- Aldec
- proFPGA
- ZC702.
- Arty 100T & Arty 35T
- Genesys
- VC707

For SP2020 Hackathon, SoCs (Pinaka, Parashu & Vajra) built for Arty 100T & Arty 35T were used by 70+ teams.

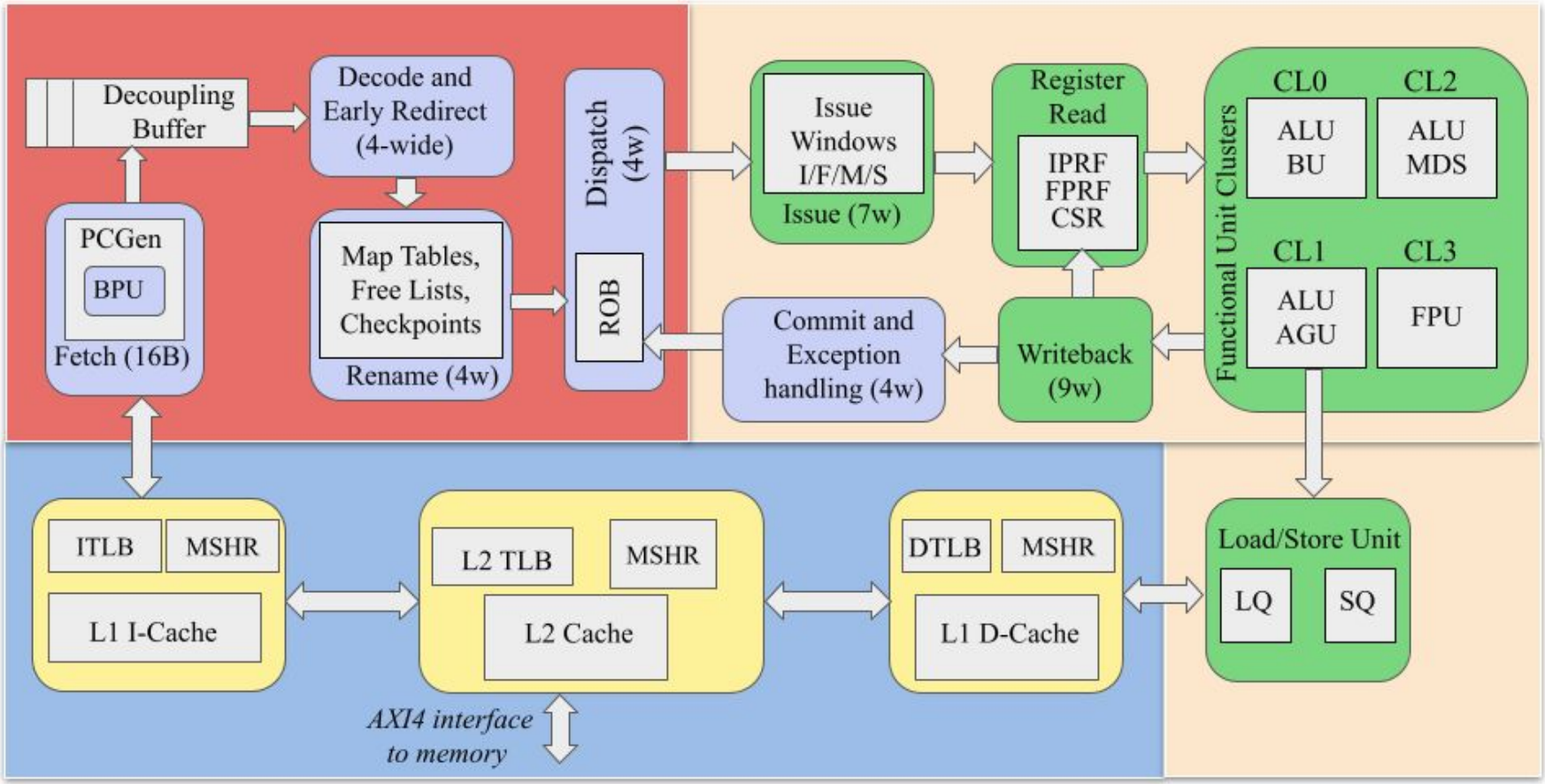


I Class

Shakti I-Class 2.0 : Overview

- 64-bit superscalar, out-of-order RISC-V core for general purpose applications
  - One of the first OoO processors from India (soon to be open-source)
- RV64GC: integer, mul/div, atomics, single/double precision FP (IEEE 754), compressed, ZiFencei, ZiCSR
- 4-wide out-of-order processor, 12-stage integer pipeline
- Supervisor mode sv39, supports RISC-V Linux
- AXI4 interface
- OpenOCD based debug through JTAG (debug 1.0) and FPGA log support
- Hardware Performance Monitors
- Configurable core, implementation in Bluespec System Verilog

I-Class Architecture

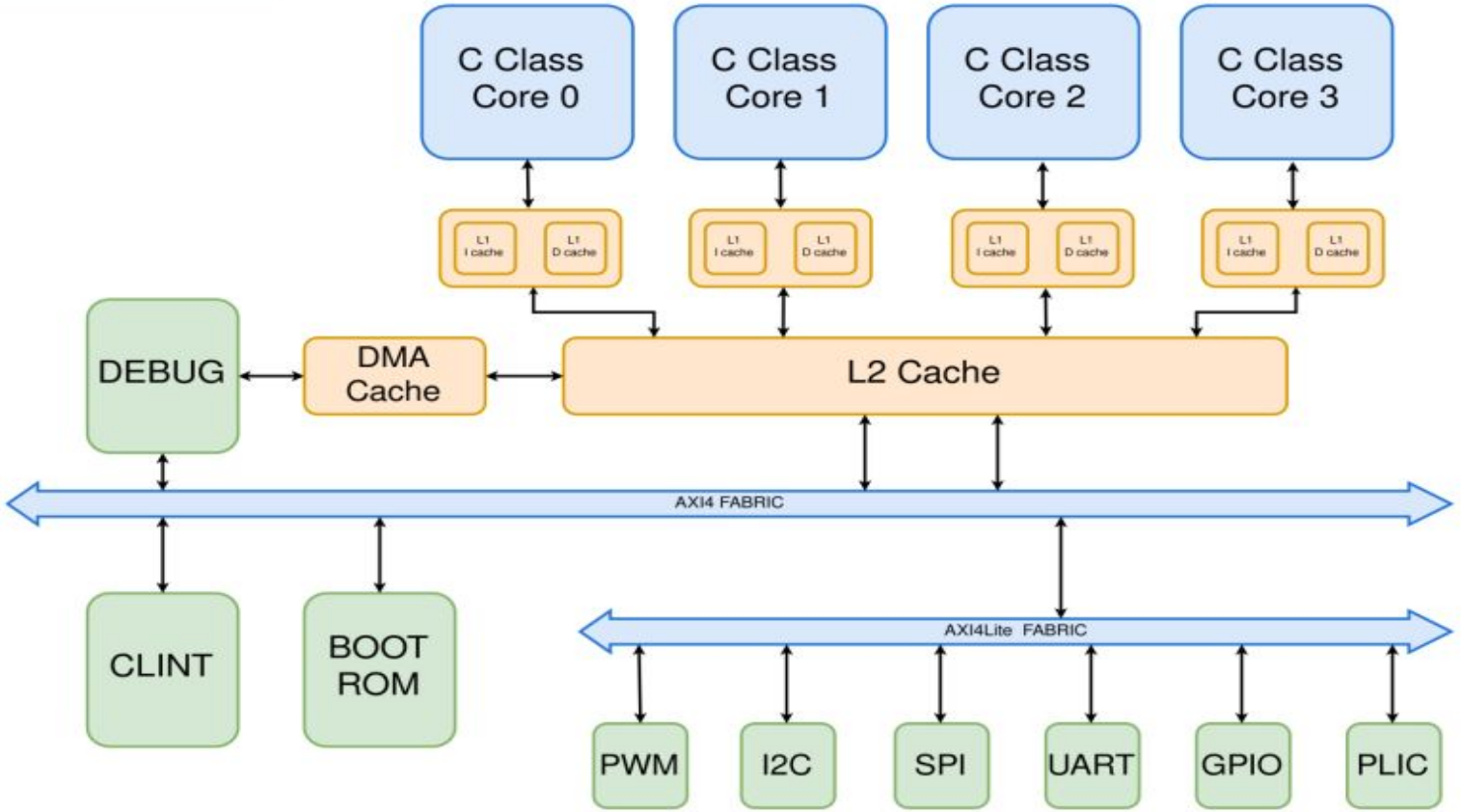


Multicore Shakti

Multicore Shakti

- Connects N number of Shakti C-class cores.
- Fully supports Atomic spec (A extension) of RISCV - used to implement locks in multicore systems.
- Cache coherence: MESI protocol, implementation from flute
- 16kB L1-I and 16kB L1-D Caches.
- 256kB L2 cache.
- CLINT Peripheral /hart timer.
- Inter processor Interrupts.
- AXI4 interface
- OpenOCD based debug through JTAG (RISCV Debug1.0 with JTAG based per hart debugging support)
- Configurable core, implementation in BSV
- SoC tried out on Aldec and profpga
- Simulation support
- Target Applications:
    - Desktop, Server, Mobiles, etc.

Multicore Shakti

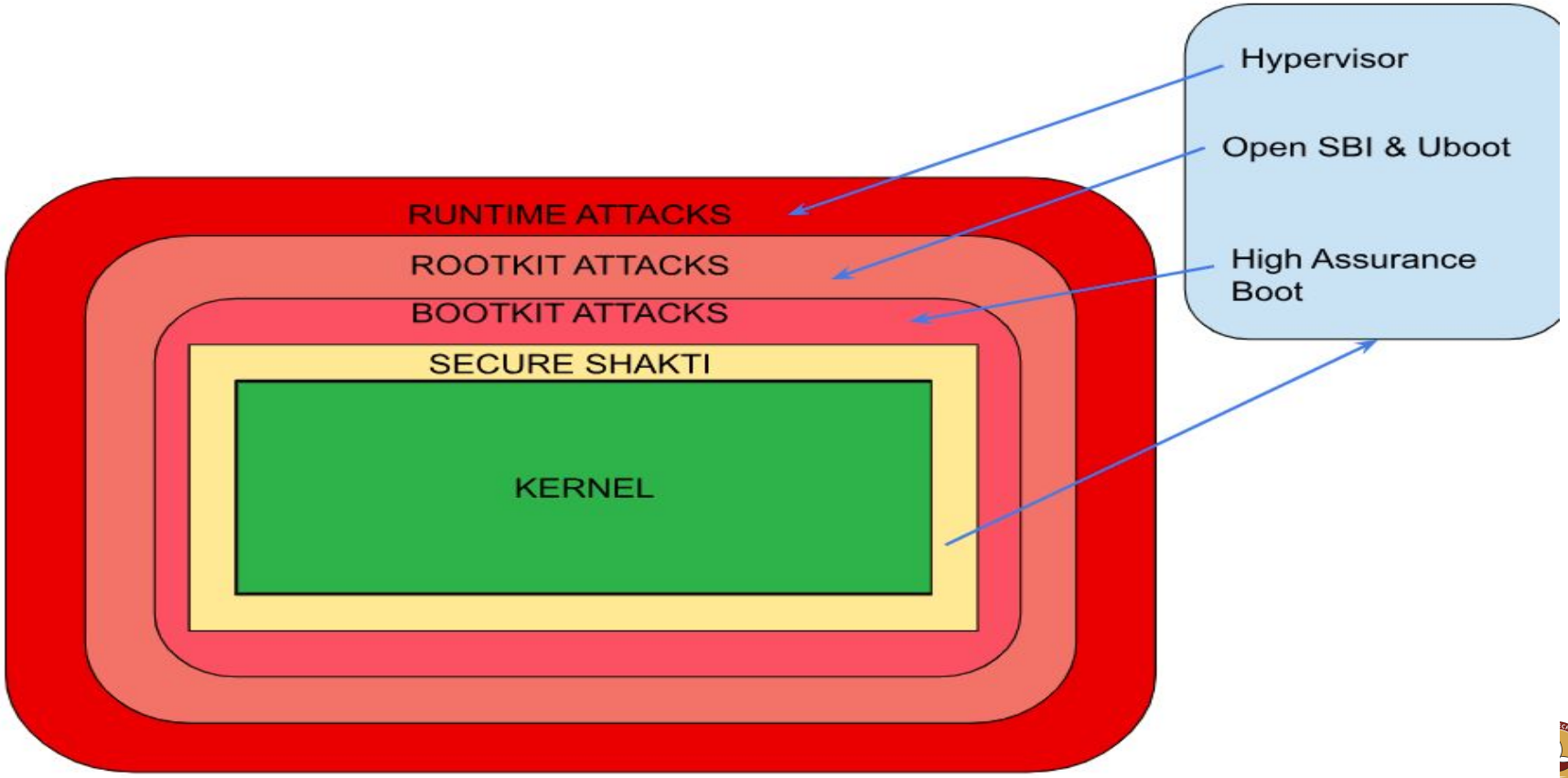


Secure Shakti

Outline

- Security is a major concern
- Advancement in Technology also resulted in increase in vulnerabilities.
- Defence and Aerospace – results in massacre effects.
- Threats from Device power on to any run time malware attacks needs to be handled.
- Shakti, having many variants to cater to all categories starting from simple embedded application to complex server applications.

Secure SHAKTI



Secure Shakti

- C class with Cryptographic accelerators.
- Secures the system from bootkit, rootkit & malware vulnerabilities.
- Three levels of protection.
  - Hardware high assurance boot
  - OpenSBI and Secure u-boot
  - Hypervisor
-

Hardware High Assurance Boot

- Complete implementation in hardware as it is the first level of trust and to have better performance.
- Protects system during boot time.
- Detects and prevents execution of modified binaries.
- Mitigates unauthorised firmware from running on the device.
- Requires all the intended executables to be signed using cryptographic hash function & public key algorithm.
- Host system has private key being to sign the binaries.
- Processor uses the public key to verify the signature.

Hardware High Assurance Boot

- Signed binary is created by computing cryptographic hash and the same is encrypted using private key.
- When Executable is being prepared on the host system, the hash value is also computed.
- When loading the executable, processor reads the binary image with signature.
- Signature is verified by computing the hash of the program along with decryption using the public key.
- If both the signatures are matching, the executable code doesn't have any vulnerability and taken for the next stage.

Open SBI and Secure u boot

- Second level of verification in the chain of trust.
- Software based verification.
- open SBI, opensource RISCV Supervisor Binary Interface, runs at higher privilege level.
- Initialise hardware & allow lower privilege levels like OS & calls.

Open SBI and Secure u boot

- U-Boot, open source primary boot loader, package instructions to boot OS kernel.
- Performs software based hash verification.
- Kernel images which boots the kernel is signed.
- The verification is similar to first level of chain of trust except the all the computations are done in software.
- After successful verification, the untampered kernel is used to boot the kernel.

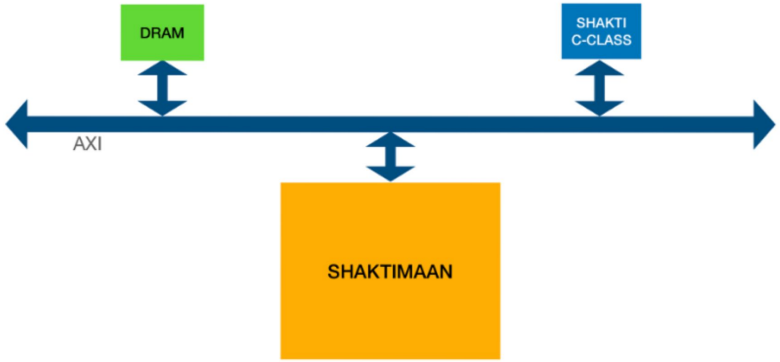
Hypervisor

- First two levels Security features makes sure there is not major security attacks to OS.
- Protects the system against runtime vulnerabilities.
- Protects the sensitive memory regions (critical code and data areas) of the kernel.
- Prevents unauthorized privilege escalations.
- Makes sure that security critical hardware settings are immutable.
- When hacker tries to access protected areas, hypervisor detects and blocks the access and report the same to UI.

ShaktiMAAN

ShaktiMAAN

- Accelerator for AI/ML + other matrix based workloads.
- Connected on AXI bus.
- Design time configurable TOPS, buffer sizes and datapath width.
- SoC has C-Class as the controller.
- Custom CISC ISA
- C based Software Framework for developing applications.
- Implementation proven on FPGA. Demonstrated simple applications written with software framework. Speedup of 50x and above depending on design configurations.



Application Development

Outline

- Shakti SDK Features
- Shakti SDK Architecture
- Shakti Tools
- Drivers for Peripherals
- Pinmux
- OS
- Manuals

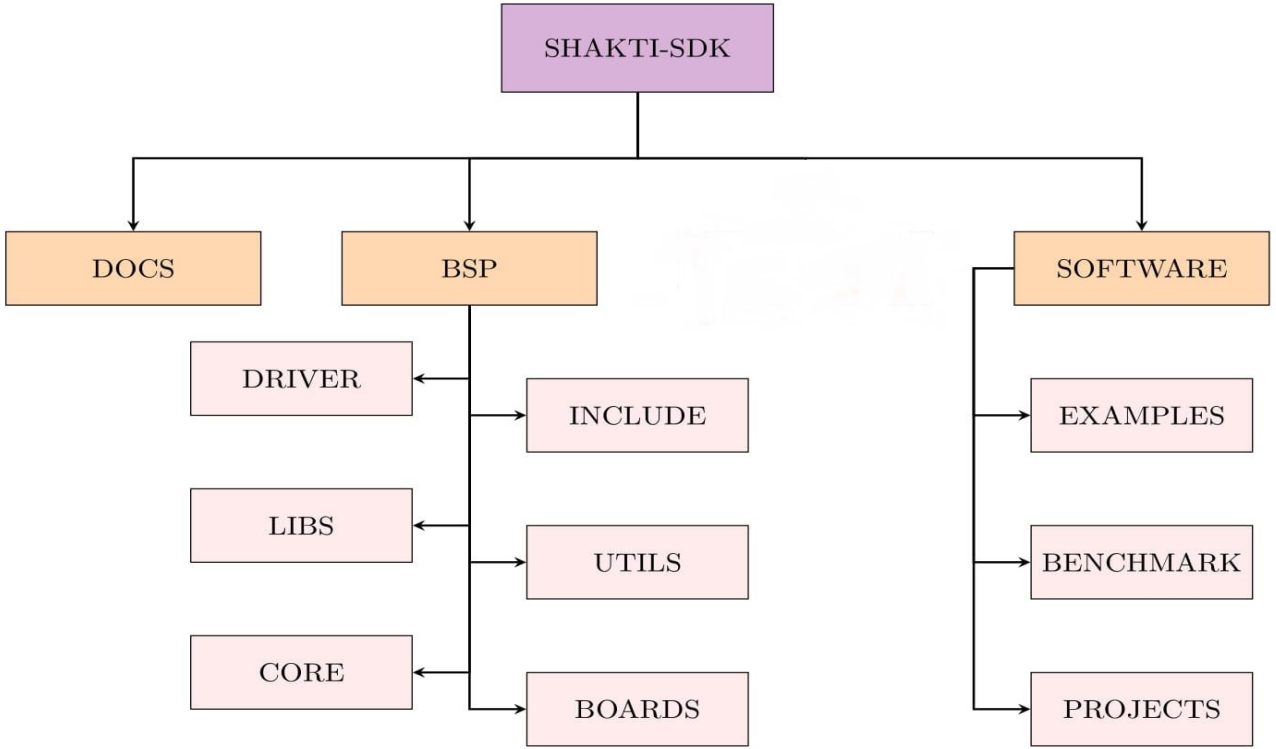
Shakti SDK Features

- Open-source software development platform for SHAKTI.
- Clean separation between boot, drivers, core and application layers.
- Driver support for PWM, QSPI, SPI, PLIC, CLINT, UART, I2C, GPIO, RTC, Watchdog, GPTimer and XADC.
- Multiple sensors connected and proven with SHAKTI-SDK.
- Standalone and Debug mode supported.

Shakti SDK Features

- Multilevel logging, Flash programming & Dynamic memory management supported.
- Single place for bare-metal application development, projects and benchmarks.
- SDK extendable for any FPGA board.
- SDK can be adapted to RTOS.
- Visual Studio Code IDE
  - Using Platform IO Extension
- Arduino IDE support.
- ESP8266 & ESP 32, FTDI, External Flashes and many sensors integrated.

Shakti SDK Architecture



https://gitlab.com/shaktiproject/software/shakti-sdk

Shakti Peripherals IPs

Drivers for Peripherals

UART

- Configurable Baud rate
  - Upto 115200
- Programmable character size.
- Programmable stop bits, parity bits.
- Read / Write options
  - Polling based
  - Interrupt based
- Integrated with ESP chip, 4G Modem, Lora & Zigbee.

PWM

- FPGA board Clock speed - 50MHz
- Configurable PWM
  - Clock divisor
  - Period
  - Duty Cycle
- Tested upto 7.5 MHz PWM frequency
- Timer mode support.

Peripheral Drivers

I2C

- Supports Standard (100kHz) and Fast mode I2C (400kHz).
- Master Mode support.
- Configurable clock speed.
- 7 bit addressing.
- Supports
  - Polling based
  - Interrupt based

SPI

- Standard SPI.
- Configurable clock speed (Tested upto 12.5MHz).
- Supports both 3 wire and 4 wire interface.
- Programmable clock polarity(CPOL) and phase (CPHA)
- Programmable data size from 1 - 32 bits.

Peripheral Drivers

PLIC

- RISC-V compatible interrupt controller.
- Registers editable at bit level.
- Default and vector interrupts supported.
- Priority can be configured.
- Complete driver support with examples

CLINT

- Configure timer at tick level.
- Interrupt mode supported.
- Complete driver support with examples

Peripheral Drivers

GPIO

- 32 GPIO's.
- 16 dedicated GPIOs, routed through PMOD.
- Remaining 16 GPIOs can be pin muxed.
- Few GPIO pins can be configured for external interrupt.
- Manys sensors verified and examples present in SDK.

XADC

- Xilinx On Chip ADC completely supported.
- Examples for On Chip temperature and electrical parameters.
- Examples for Dedicated and Auxiliary ADC pins

Peripheral Drivers

GP Timer

- 32 bit General Purpose Timer.
- Supports Up/Down/Up-down Counters.
- Also supports PWM.
- Captures the input.
- Continuous Counter mode.

Watchdog Timer

- 32 bit Watchdog timer.
- Prevents the core from Stalling during code execution.
- Supports hard reset as well as interrupt mode.
- When the active bit is not triggered, the SoC will get reset on timeout.

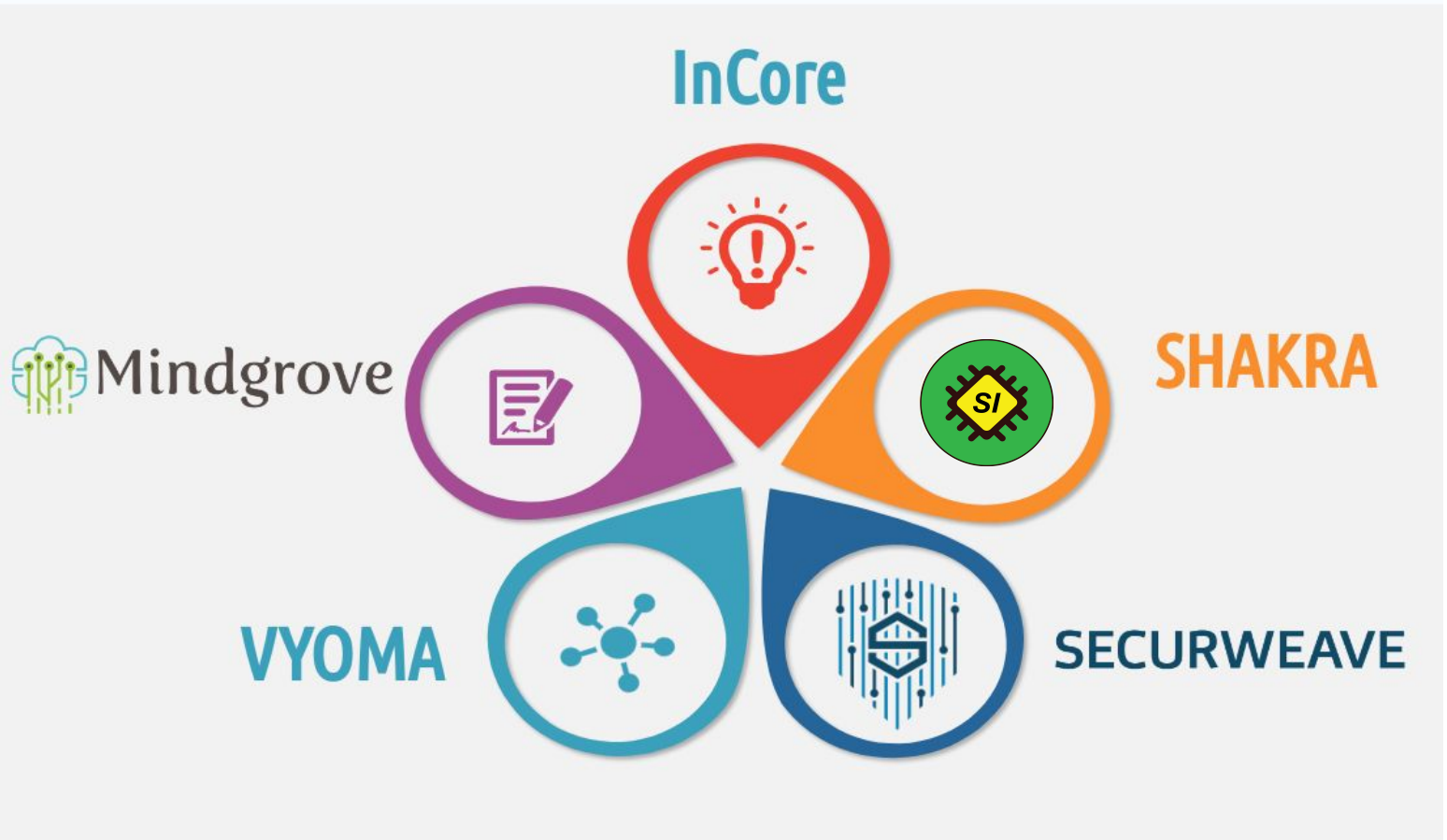
PinMux

- By default all pins are configured as GPIO
- GPIO pins can be assigned to different peripheral devices
- Pin mux register has to be configured to enable a peripheral function on a pin.
- Pin mux register is 32 bits.
- Python based script is available to easily configure the pin mux functionality and generate bsv code.

OS

- RTOS
  - Free RTOS
  - Zephyr
  - Nuttx.
- Linux OS

Startups



References:

Shakti Documentation: https://shakti.org.in/documentation.html

Shakti Blogs: https://blogshakti.org.in/

Shakti FPGA files: https://gitlab.com/shaktiproject/sp2020

Shakti SDK: https://gitlab.com/shaktiproject/software/shakti-sdk

Shakti on Arduino:
https://blogshakti.org.in/how-to-print-hello-world-on-shakti-using-arduino-ide/

Linux on Shakti: https://gitlab.com/shaktiproject/software/linux-on-shakti

Platform IO: https://registry.platformio.org/platforms/platformio/shakti

Shakti cores: https://gitlab.com/shaktiproject/cores

Shakti peripherals: https://gitlab.com/shaktiproject/uncore/devices

Queries

Thank You !