krv-m0 Introduction

Introduction

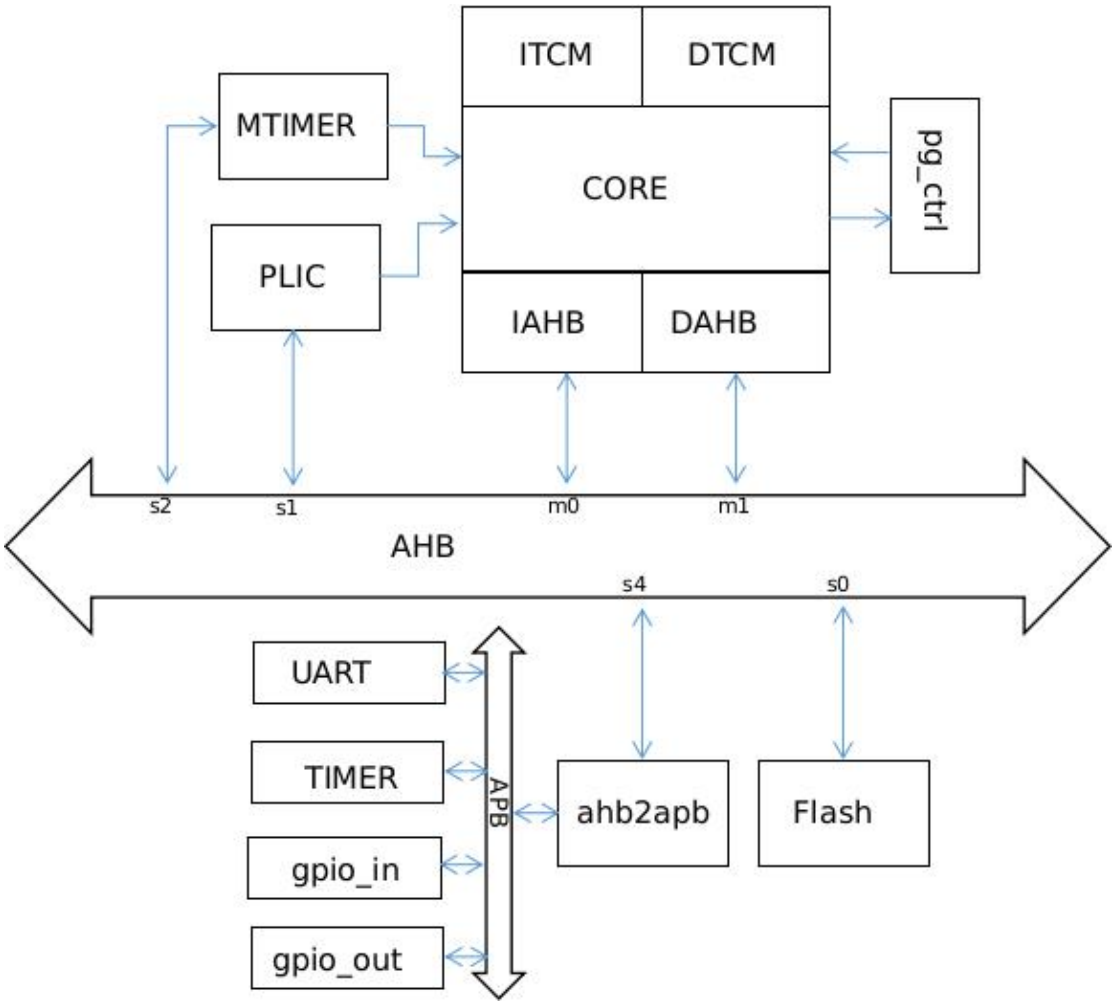
KRV-m0 is a RISCV-based 32-bit micro-controller subsystem. It consists of a RISCV processor Core with two tightly coupled memories for instruction and data acceleration, a machine timer and a platform level interrupt controller (KPLIC) for timer and external interrupts respectively and lite AHB interconnection with some APB peripherals. krv_m0 uses clock-gating, operands gating and power gating to control power consumption.

Features

- RISC-V 32-b integer ISA standard
- Instruction/data TCM for acceleration
- precise exception and external interrupt support
- Machine Timer support
- integrated platform-level interrupt controller supporting 32 external interrupts
- lite AHB interconnection
- APB peripherals of UART/TIMER/GPIO
- low power design with clock-gating, operands gating and power gating

Design Details

KRV_M Block Diagram



Main Functional Details

KRV-m Core

KRV-m Pipeline

Diagram

KRV-m core has a 5-stage pipeline as shown is Figure below

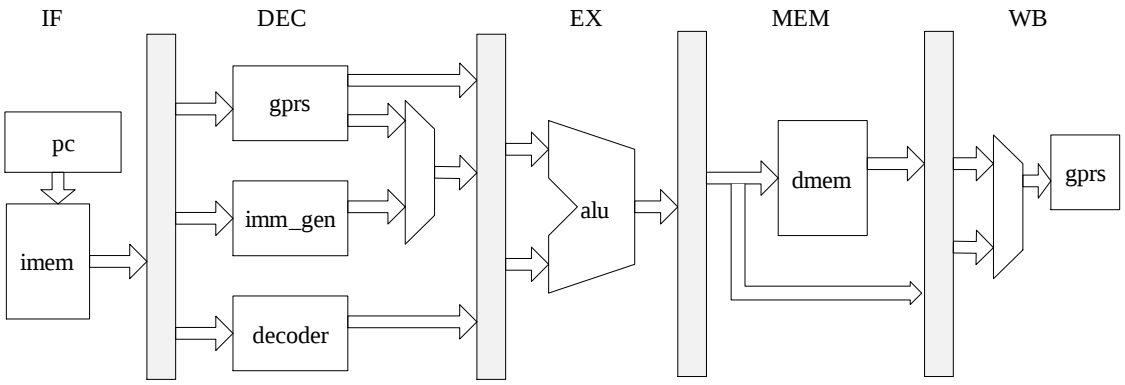


Figure KRV-m Pipeline

General Description

- IF Stage – Instruction Fetch Stage
  The program counter is maintained in this stage. While in normal sequence, the program counter increased by 4 after each instruction read from the instruction memory (imem).
  The boot start address is connected to the core top and need to be specified by SoC.

- DEC Stage – Decoder Stage
  In Decode stage. Instruction fetched from the instruction memory are decoded to get the alu operation and source data from general purpose registers(GPRS) or immediate generation

block (imm_gen) and forwards to the ALU in next stage.

- EX – Execute Stage
  ALU locates in this stage and is responsible for the instruction execution including the arithmetic/logic calculate for arithmetic and conditional branch instruction and address generation for load/store instruction.

- MEM – Memory Stage
  In MEM stage, the DMEM is accessed for load/store instructions. the ALU result and the load data are forwarded to WB stage for GPRS write back if needed.

- WB – Write Back Stage
  In this stage, the ALU result or the load data depending on whether the instruction is a load or computational is written back to the GPRS selected by the RD.

Exceptions and Interrupts

In RISC-V, the term exception is used to refer to an unusual condition occurring at run time associated with an instruction in the current RISC-V thread, and the term interrupt is used to refer to an external event that occurs asynchronously to the current RISC-V thread.

The exception and interrupt flow are illustrated as below:

KRV-m supports only M-mode and the Machine level CSRs listed below are implemented to support exceptions and interrupts.

Table N-N Machine Level CSRs in KRV-m

Name	Address	default value	R/W	description
mvendorid	0xF11	0x0000_0000	RO	Machine Vendor ID
marchid	0xF12	0x0000_0000	RO	Machine Architecture ID
mimpid	0xF13	0x1000_0000	RO	Machine Implementation ID

mhartid	0xF14	0x0000_0000	RO	Hart ID
mstatus	0x300	0x0000_0008	RW	Machine Status, only MIE is implemented
misa	0x301	0x4000_0100	RW	Machine ISA
mie	0x304	0x0000_0800	RW	Machine Interrupt enable, only MEIE/MTIE implemented
mtvec	0x305	0x0000_0000	RW	Machine Trap-Vector Base-Address
mepc	0x341	0x0000_0000	RW	Machine Exception Program Counter
mcause	0x342	0x0000_0000		Machine Cause Register
mtval	0x343	0x0000_0000	RW	Machine Trap Value
mip	0x344	0x0000_0000	RO	Machine Interrupt pending, only MEIP/MTIP implemented

Exceptions

There are below kinds of exceptions in KRV-m:

- **Instruction address misaligned exception**

When the instruction address is not aligned to a four-byte boundary, an exception is raised.

- **Load X0 exception**

Loads with a destination of x0 must still raise any exceptions and action any other side e_ects even though the load value is discarded.

Note: In KRV-m, the memory access error (like non-exist memory address access or read/write privilege violation) will not introduce exception. So the programmer needs take care of the memory access is valid or the transaction may not take effect depending on the device behavior (write maybe ignored and read may return all 0). Also the device connected to AHB should assert its HREADY in a finite time after a transaction targeted received not matter whether it's valid or not.

- **Illegal Instruction exception**

Attempts to access a non-existent CSR raise an illegal instruction exception. Attempts to

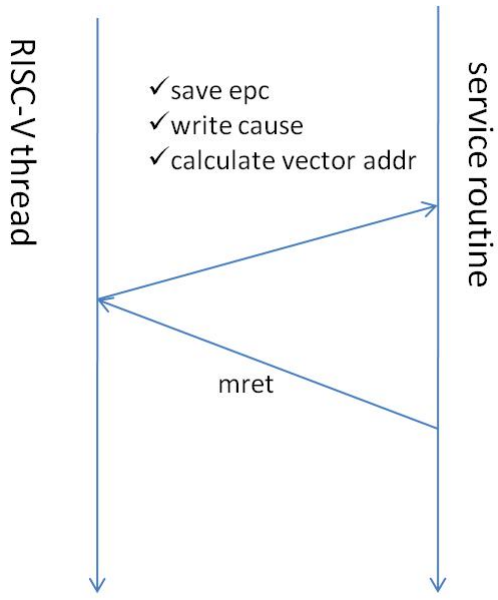
write a read-only register also raise illegal instruction exceptions.

KRV-m support precise exception. As the three types of exceptions happen, the PC pointed to the instruction which causes the exception is saved in the mepc and the corresponding bit in mcause is set to indicate the exact exception type as in Table N-N. The value in mtvec is used to calculate the vector address in the way defined by mode bit in the register as below:

- mode = 0: Direct All exceptions set pc to BASE.
- mode = 1: Vectored Asynchronous interrupts set pc to BASE+4*cause.

Table N-N mcause value

Interrupt	Exception	Description
1	8	external interrupt
0	0	Instruction address misaligned
0	2	Illegal instruction
0	5	Load access fault



FigureN-N Trap process

For exception caused by jump address misaligned, load X0 or illegal instruction, the detection happens in the DEC stage and in the next

cycle when the pipeline is not stalled, the pc will take the vector address and flush the instructions already in FETCH stage and for exception caused by branch address misaligned, the branch is determined to taken or not at EX stage, the pc also take the vector address in the next cycle but the instructions both in FETCH and DEC stage will be flushed.
For the external interrupt, when the core receives interrupt request from the KPLIC or MTIMER, it will flush the fetch stage and finish all the other instructions which are already in the pipeline and turn to the vector address to service the interrupt.
After the exception or interrupt service routine finishes, a MRET instruction will be needed for return and the value kept in mepc will be loaded to pc to continue with the instruction flow.

Tightly-Coupled-Memory

There are two 16KB tightly coupled memories in KRV-m which are the ITCM and DTCM. They are implemented using on-chip SRAM to store specified instruction and data respectively for the performance purpose.

ITCM

The ITCM is auto-loaded after the chip power up reset. The code in the ROM (EEPROM or FLASH) located from address 0x0000_0000 to 0x0000_3FFF is read from IAHB and stored into the ITCM.
During the auto-load process, the pipeline is stalled and the IAHB is the only master using the bus. After the auto-load finished, the instruction memory remapped and the address 0x0000_0000 to 0x0000_3FFF will always points to ITCM.

DTCM

The DTCM is mapped to from 0x0004_0000 to 0x0007_FFFF, the software is responsible for the load if there is any data needed to be accelerated, or this space can be used to store intermediate data.
The DTCM can be accessed manually using the LOAD/STORE instruction. Also there is a DMA interface for DMA access.

KRV-m platform-level Interrupt Controller (KPLIC)

KPLIC is an implementation of RISC-v PLIC. The features of KPLIC are listed as below:

- Support 32 external interrupts sources
- programmable level sensitive or edge triggered
- 256 priority levels supported
- up to 15 edge trigger interrupt pulses are recorded while the previous one is waiting for completion

Figure KPLIC Diagram

Programmable Registers

Name	Offset	Default Value	Description
KPLIC_INT_TYPE	0x000	32'h0	individual interrupt type control: 0: level sensitive 1: edge triggered
KPLIC_INT_ENAB LE	0x008	32'h0	individual interrupt enable control: 0: disabled 1: enabled
KPLIC_INT_PENDING _STATUS	0x010	32'h0	interrupt pending status, read-only
KPLIC_TARGET_PRI ORITY	0x018	32'h0	target priority mask 31:24 – reserved 23:16 – reserved 15:8 – reserved 7:0 – machine hart priority
KPLIC_MPPI	0x020	32'h0	maximum priority pending interrupt (read-only)
KPLIC_COMPLETION	0x028	32'h0	Interrupt completion register. A valid write with a valid INTID will be treated as completion signal. (write only)
KPLIC_INT_PRIORITY	0x128~0x 144	{32{8' h0}}	individual interrupt priority

KRV-m AHB interconnection

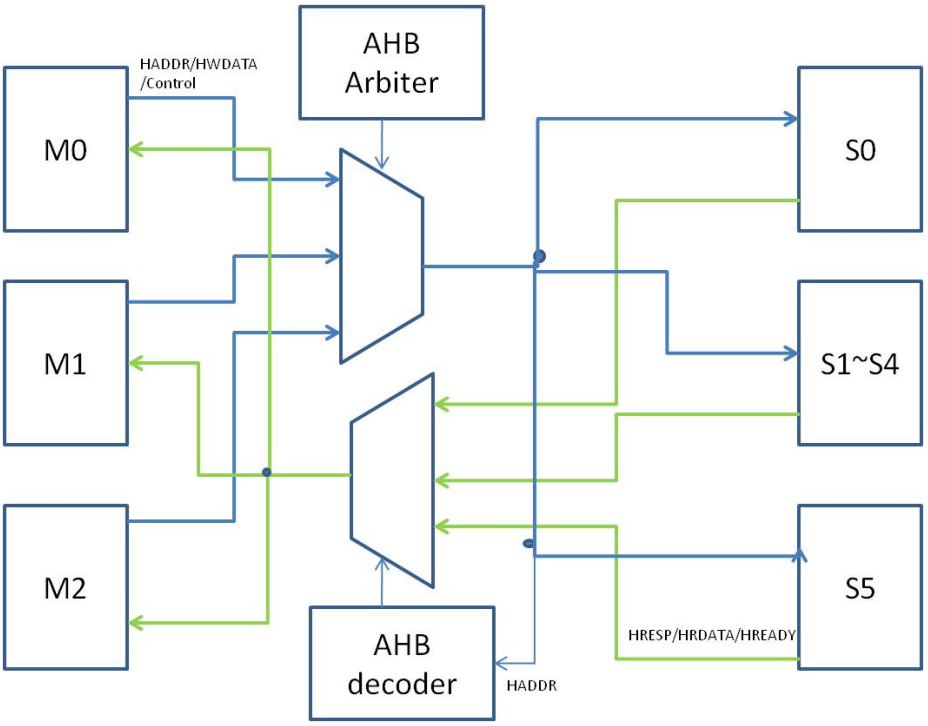


Figure N-N AHB in KRV-m

There is one AHB in KRV-m with 3 master-port and 6 slave-port. The connection and address map is listed in the tables below:

Table AHB interconnection

Port-Name	Block Name	Description
M0	IAHB	Instruction AHB
M1	DAHB	Data AHB
M2	Reserved	For future use
S0	Reserved for FLASH	0x0000_0000 ~ 0x3FFF_FFFF
S1	KPLIC	0x4000_0000 ~ 0x43FF_FFFF
S2	MTIMER	0x4400_0000 ~ 0x4FFF_FFFF
S3	Reserved	0x5000_0000 ~ 0x6FFF_FFFF
S4	APB	0x7000_0000 ~ 0x7FFF_FFFF
S5	Reserved	0x8000_0000 ~ 0x9FFF_FFFF
S6	Reserved	0xA000_0000 ~ 0xFFFF_FFFF

Low Power Design

KRV-m Power Modes

There are three power modes in KRV-m as below:

Power Mode	Description
Run	Core runs
Light Sleep	Core clock-gated
Deep Sleep	Core power-gated

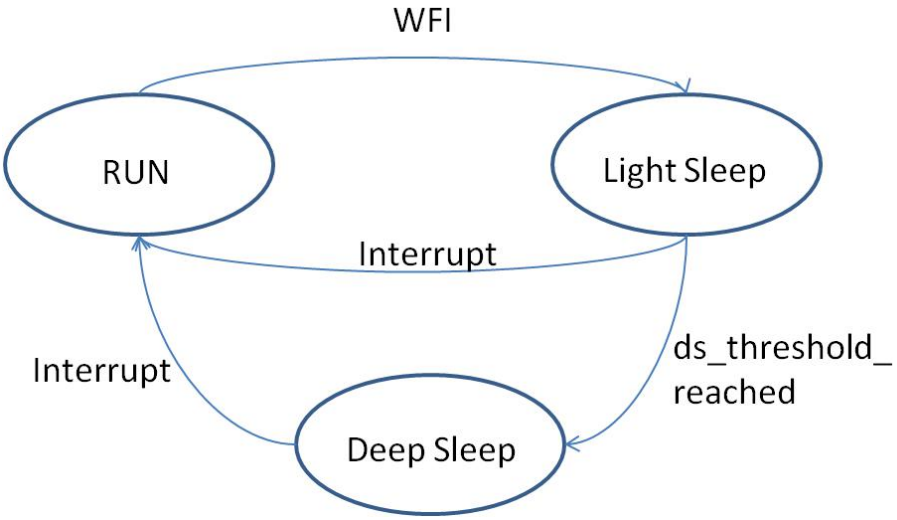
The Run-Mode is the reset mode of the KRV-m and the program runs normally in this mode. As currently the KRV-m supports only the Machine-level and the core will enter the Light-Sleep-Mode when a WFI (Wait-For-Interrupt) instruction encountered. Under the Light-Sleep-Mode, the clock of the core will be gated to save the dynamic power and an independent counter will be started. If after a certain period, there is no external interrupt, the core will enter the Deep-Sleep-Mode and after a certain sequence which is done by the pg_ctrl block, the power of the core will be gated to save the static power as well. As the pg_ctrl block resides in the AON (Always-On) power domain, it will be responsible to detect the external interrupt request and wake up the core after an external interrupt comes from KPLIC. The core will jump to the next instruction after the WFI and continue to execute.

KRV-m power control unit

The power control unit is used to manage the power mode and control and clock gating and power gating. Its responsibility includes:

- detect the WFI from core
- power mode control
- detect the interrupt request from KPLIC during LS and DS mode
- gating clock in LS mode
- manage the power down and power up sequence

Power Mode Control



The processor will enter the Light Sleep mode after all the instructions before WFI finished. In the Light-Sleep mode, a 40-bit counter will be started and when it reaches the deep_sleep threshold before an external interrupt, the processor will start the power down sequence and turn off the power and enter the Deep-Sleep mode. When under the Light-Sleep mode and Deep-Sleep mode, the power control unit will continue to detect external interrupt from KPLIC and is responsible to wake up the processor when there is one coming.

Clock-Gating

In Light-Sleep mode, the cpu clock will be gated to save dynamic power. All the instructions before WFI are guaranteed to completed before the clock off and the instructions after WFI are all stalled before the processor wake up.

Power-Gating

In Deep-Sleep mode, the cpu power will be turned off to save leakage power. The power down and power up operations need right sequence to guarantee the right operation of the processor and its

connected sub-system.

Table N-N Power Down Save/Restore Register List

Register Name	Description
X1 ~ X31	GPRS X1 to X31
pc	program counter
mcsr	machine level control and status registers

Verification

Verification Plan:
1: the risc-v test from https://github.com/riscv/riscv-tests with some local modification as below:
2: Add some csr tests to test the csr instructions as well as the machinel level csrs implemented in krv-m
3: tests of KPLIC
4: tests for clock gating and power gating
5: Boot of Zephyr sample hello world/synchronization/philosopher

FPGA verification with Micro-semi SmartFusion2