Syntacore™
Custom cores and tools

RISC-V EXTENSIONS IN LINUX KERNEL

October, 2024

Sergey Matyukevich, info@syntacore.com

RISC-V extensions

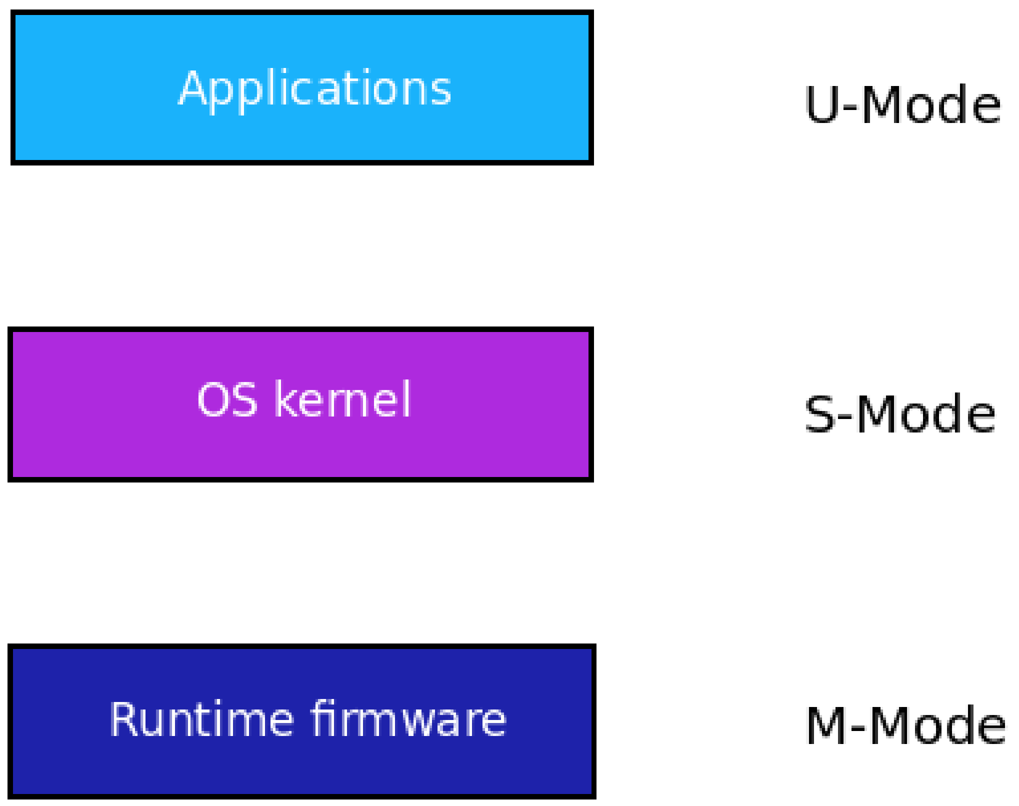
RISC-V Linux timeline

Kernel recipes: discovery

Userspace recipes: discovery

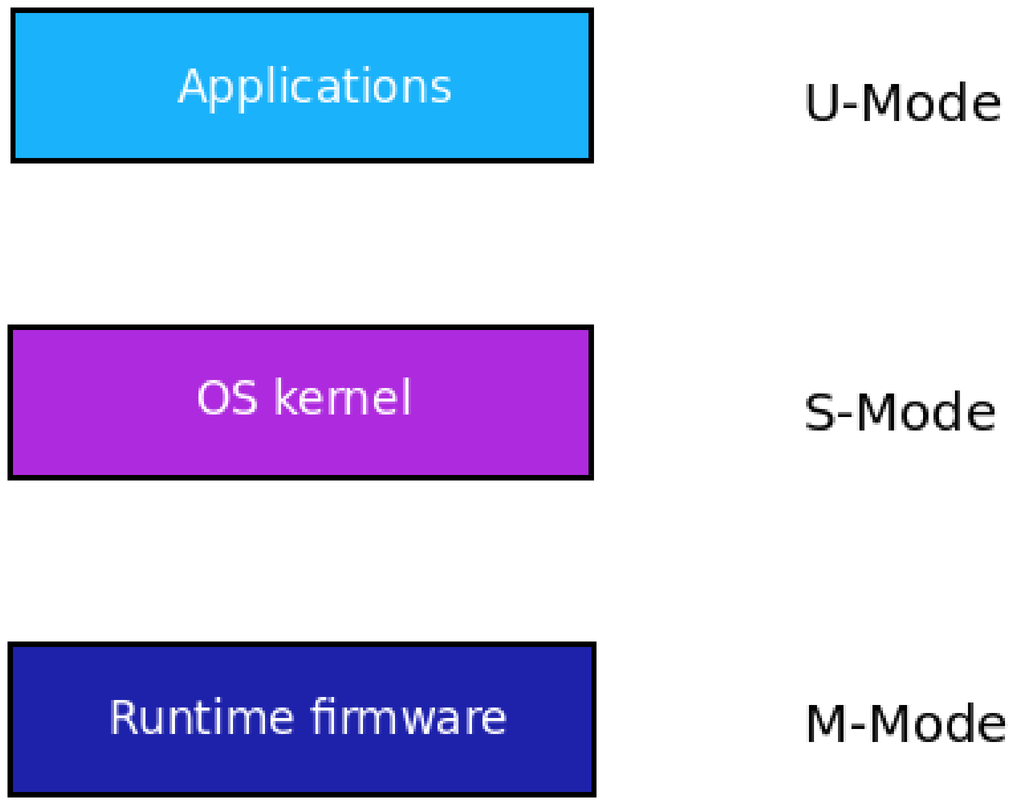
RISC-V privilege modes

- M-mode
  - full access to hardware
  - exceptions/interrupts



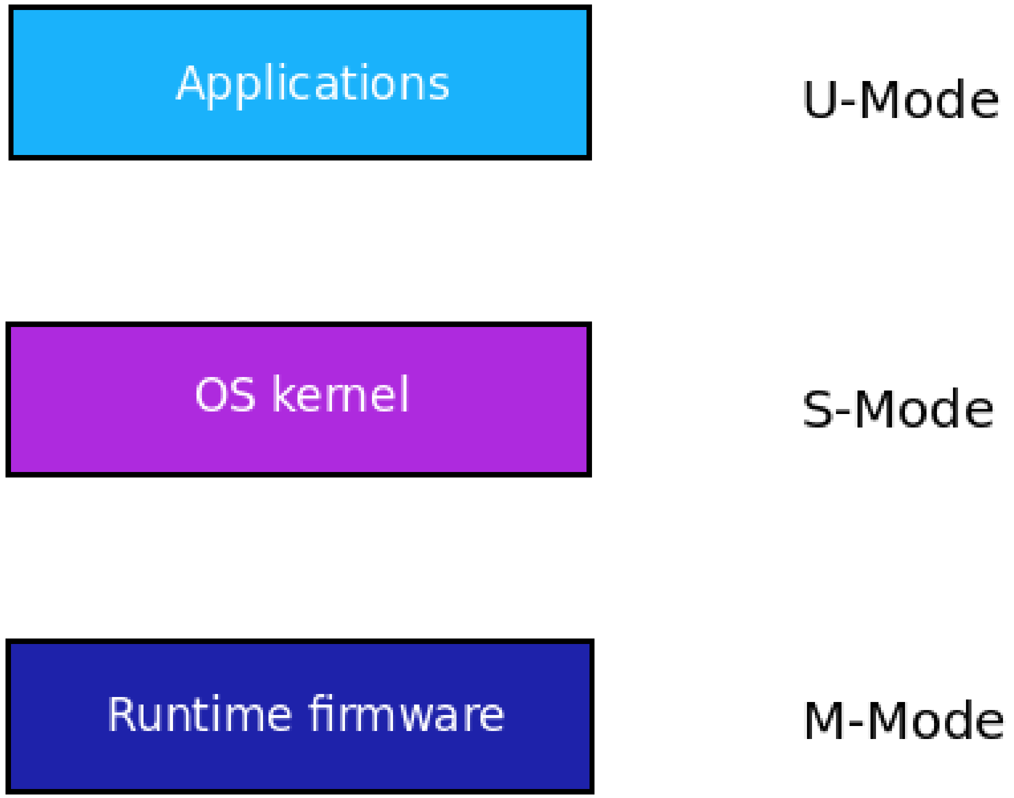
RISC-V privilege modes

- M-mode
  - full access to hardware
  - exceptions/interrupts
- S-mode
  - page-based virtual memory
  - delegation (exceptions/interrupts)



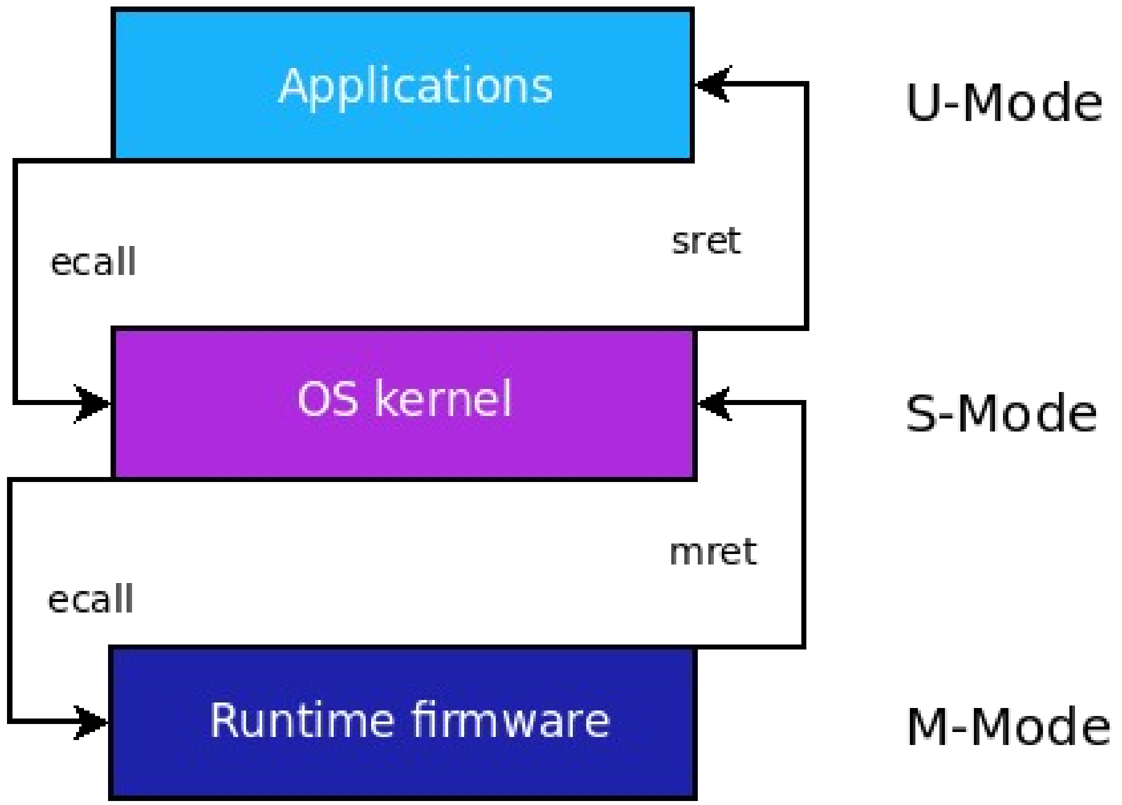
RISC-V privilege modes

- M-mode
  - full access to hardware
  - exceptions/interrupts
- S-mode
  - page-based virtual memory
  - delegation (exceptions/interrupts)
- U-mode



RISC-V privilege modes

- M-mode
  - full access to hardware
  - exceptions/interrupts
- S-mode
  - page-based virtual memory
  - delegation (exceptions/interrupts)
- U-mode



RISC-V extensions

Classic extensions

- IMAFDQCBVH
- RV64IMAFDC (RV64GC)
- MISA register: ABC...XYZ

RISC-V extensions

Classic extensions

- IMAFDQCBVH
- RV64IMAFDC (RV64GC)
- MISA register: ABC…XYZ

Z-ext – standard unprivileged extensions

- Zicsr, Zbb, Zifencei
- 2nd letter as relationship with classic extensions
    - Zam, Zfh, Zk*, Zb*

RISC-V extensions

Classic extensions

- IMAFDQCBVH
- RV64IMAFDC (RV64GC)
- MISA register: ABC…XYZ

Z-ext – standard unprivileged extensions

- Zicsr, Zbb, Zifencei
- 2nd letter as relationship with classic extensions
  - Zam, Zfh, Zk*, Zb*

S-ext — standard privileged extensions

- Sv — virtual memory (Sv39)
- Sm — machine level (Smrnmi)
- Ss — supervisor level (Sstc)
- Sh — hypervisor level (Shtvala)

RISC-V extensions

Classic extensions

- IMAFDQCBVH
- RV64IMAFDC (RV64GC)
- MISA register: ABC…XYZ

Z-ext – standard unprivileged extensions

- Zicsr, Zbb, Zifencei
- 2nd letter as relationship with classic extensions
    - Zam, Zfh, Zk*, Zb*

S-ext — standard privileged extensions

- Sv — virtual memory (Sv39)
- Sm — machine level (Smrnmi)
- Ss — supervisor level (Sstc)
- Sh — hypervisor level (Shtvala)

X-ext – non-standard extensions

- XSyntacore*
- XTheadVector

RISC-V extensions: profiles

RVA22S64

rv64imafdc_zicbom_zicbop_zicboz_zicntr_zicsr_zifencei_zihintpause
_zihpm_zfhmin_zca_zcd_zba_zbb_zbs_zkt_svinval_svpbmt

RISC-V extensions: 2023-2024

smmpm

sscsrind zvfbfwma svvptc

smcntrpmf zcmp zvbc

smdbltrp zcd zvfhmin zsto zcf zvksh zaamo

zalrsc svadu zca zcmt

zvksc zicfiss ssnpm

smrnmi zvknha zvknc zicntr zvknhb ssqosid

zabha zvkng zcmop ssaia

zihpm zvkned

zfbfmin zvks ssccfg smnpm

zvfh zce zfa

zicfilp smaia zcb zvbb zvkg zihintntl

sspm zvkt zvksed zvkb zvksg zvkn zicond

smcdeleg smcsrind zacas ssdbltrp

supm zimop zvfbfmin

source: https://wiki.riscv.org/display/HOME/Ratified+Extensions

RISC-V extensions

RISC-V Linux timeline

Kernel recipes: discovery

Userspace recipes: discovery

Linux: extensions timeline

2018-2019

v5.4

- RV32IMAFDC
- RV64IMAFDC
- MMU Sv39

2020-2021

v5.15

- MMU ASID
- NOMMU

2022

v6.1

- Zicbom (cache)
- Zihintpause (idle)
- Sstc (timer)
- Sscofpmf (perf)
- H (KVM)
- MMU Sv48, Sv57

2023

v6.6

- RVV
- Zicboz
- Zbb (strings)
- Svnapot

2024

v6.11*

- Zbb (bitops)
- Zkr (entropy)
- Zvk* (crypto)
- Zbc (crc32)
- Zawrs (spinlocks)

Linux: maintenance guidelines

Kernel patches are accepted for extensions that:

- have been **officially frozen or ratified** by the RISC-V Foundation
- have been implemented in hardware that is widely available
  - see Documentation/riscv/patch-acceptance.rst

Not yet ratified functionality:

- send RFC patches for discussion and early feedback
- maintain custom Linux kernel trees

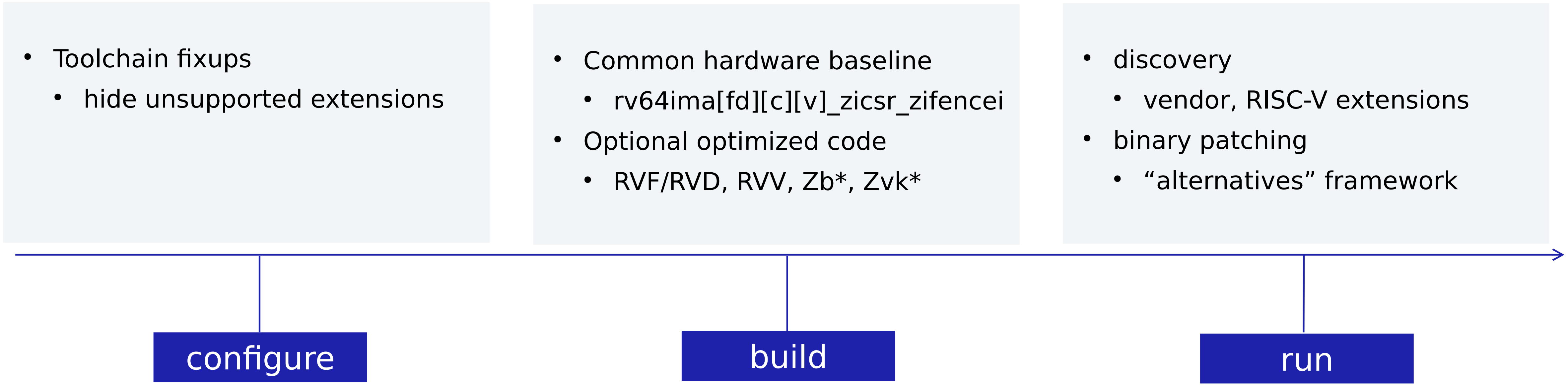
Linux: RISC-V extensions

RISC-V extension	Kernel use	Application use	OS support
Sstc, Sscofpmf	🟩	🟥	- kernel feature - discovery
RVF/RVD	🟥	🟩	- context/debug - discovery
RVV	🟩	🟩	- context/debug - discovery
Zba/Zbb/Zbs/Zbc	🟩	🟩	- discovery
Zbk/Zkn	🟥	🟩	- discovery

Linux: RISC-V extensions

- Sscofpmf
  - perf record
- Sstc
  - accessing timer from S-mode
- RVV (vector)
  - vectorize copy_to_user/copy_from_user
- Zb* (bitmanip)
  - string operations, bit operations, checksums
- Zvk* (vector crypto)
  - crypto algorithms

RISC-V Linux recipes



- Toolchain fixups
  - hide unsupported extensions

- Common hardware baseline
  - rv64ima[fd][c][v]_zicsr_zifencei
- Optional optimized code
  - RVF/RVD, RVV, Zb*, Zvk*

- discovery
  - vendor, RISC-V extensions
- binary patching
  - “alternatives” framework

Linux: extensions discovery

- Runtime discovery

Linux: extensions discovery

- Runtime discovery: not yet …
  - static Device Tree description

```
# allwinner D1s

riscv,isa-base = "rv64i";
riscv,isa-extensions = "i", "m", "a", "f", "d", "c",
        "zicntr", "zicsr", "zifencei", "zihpm";

# StarFive JH7110

riscv,isa-base = "rv64i";
riscv,isa-extensions = "i", "m", "a", "c",
        "zba", "zbb",
        "zicntr", "zicsr", "zifencei", "zihpm";

# RVA22

riscv,isa-base = "rv64i";
riscv,isa-extensions = "i", "m", "a", "c", "f", "d", "h", "v",
        "zba", "zbb", "zbc", "zbs",
        "zicntr", "zicsr", "zifencei", "zihpm", "zihintpause",
        "zicbom", "zicboz", "svnapot",
        "svpbmt", "svinval", "sscofpmf", "sstc";
```

Linux: extensions discovery

- Runtime discovery: not yet …
  - static Device Tree description
- MISA register
  - 26 bits for extensions (ABC…XYZ)
  - not extensible

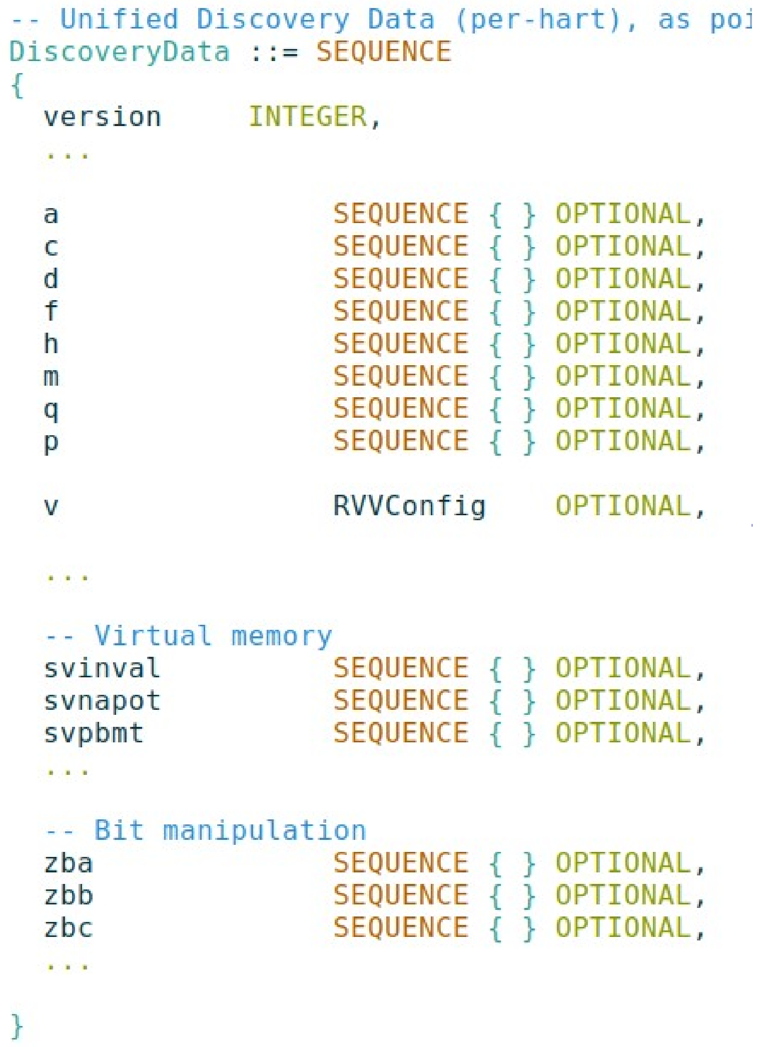
```
# allwinner D1s

riscv,isa-base = "rv64i";
riscv,isa-extensions = "i", "m", "a", "f", "d", "c",
        "zicntr", "zicsr", "zifencei", "zihpm";

# StarFive JH7110

riscv,isa-base = "rv64i";
riscv,isa-extensions = "i", "m", "a", "c",
        "zba", "zbb",
        "zicntr", "zicsr", "zifencei", "zihpm";

# RVA22

riscv,isa-base = "rv64i";
riscv,isa-extensions = "i", "m", "a", "c", "f", "d", "h", "v",
        "zba", "zbb", "zbc", "zbs",
        "zicntr", "zicsr", "zifencei", "zihpm", "zihintpause",
        "zicbom", "zicboz", "svnapot",
        "svpbmt", "svinval", "sscofpmf", "sstc";
```

Linux: extensions discovery

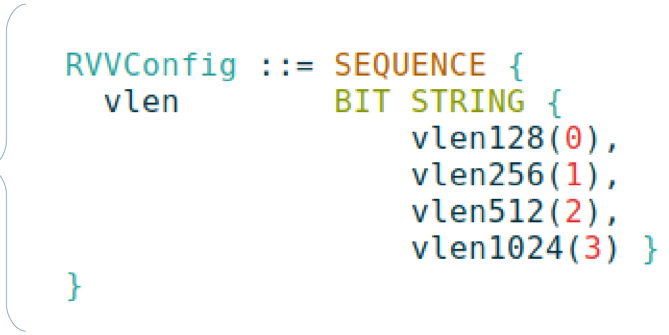
- Runtime discovery: not yet …
  - static Device Tree description
- MISA register
  - 26 bits for extensions (ABC…XYZ)
  - not extensible
- RISC-V Unified Discovery TG
  - v0.1.1-pre (development)
    - MCONFIGPTR register (base address)
    - ASN.1 schema

```
# allwinner D1s

riscv,isa-base = "rv64i";
riscv,isa-extensions = "i", "m", "a", "f", "d", "c",
        "zicntr", "zicsr", "zifencei", "zihpm";

# StarFive JH7110

riscv,isa-base = "rv64i";
riscv,isa-extensions = "i", "m", "a", "c",
        "zba", "zbb",
        "zicntr", "zicsr", "zifencei", "zihpm";

# RVA22

riscv,isa-base = "rv64i";
riscv,isa-extensions = "i", "m", "a", "c", "f", "d", "h", "v",
        "zba", "zbb", "zbc", "zbs",
        "zicntr", "zicsr", "zifencei", "zihpm", "zihintpause",
        "zicbom", "zicboz", "svnapot",
        "svpbmt", "svinval", "sscofpmf", "sstc";
```

RISC-V unified discovery

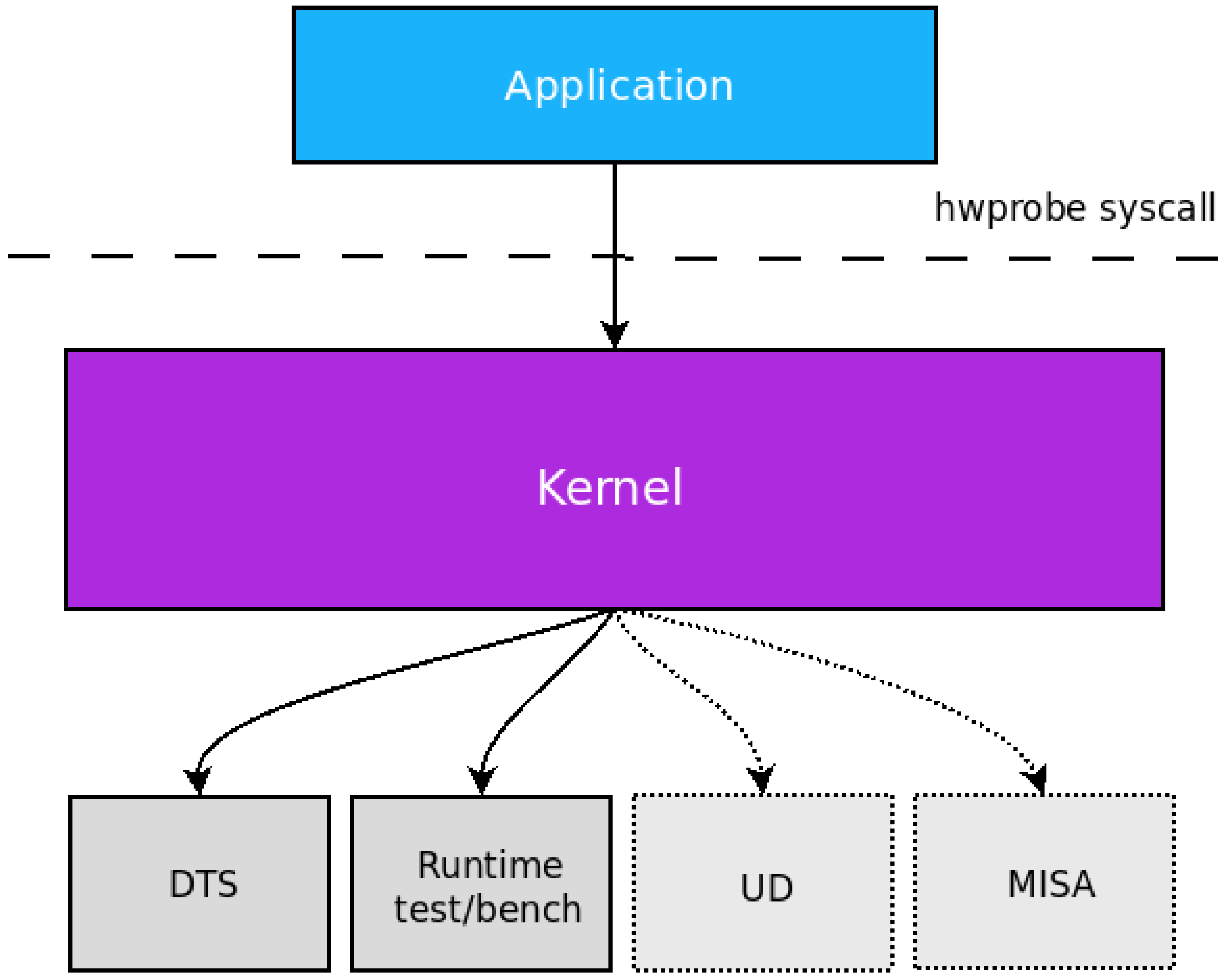


```
-- Unified Discovery Data (per-hart), as poi
DiscoveryData ::= SEQUENCE
{
  version     INTEGER,
  ...

  a                  SEQUENCE { } OPTIONAL,
  c                  SEQUENCE { } OPTIONAL,
  d                  SEQUENCE { } OPTIONAL,
  f                  SEQUENCE { } OPTIONAL,
  h                  SEQUENCE { } OPTIONAL,
  m                  SEQUENCE { } OPTIONAL,
  q                  SEQUENCE { } OPTIONAL,
  p                  SEQUENCE { } OPTIONAL,

  v                  RVVConfig    OPTIONAL,

  ...

  -- Virtual memory
  svinval            SEQUENCE { } OPTIONAL,
  svnapot            SEQUENCE { } OPTIONAL,
  svpbmt             SEQUENCE { } OPTIONAL,
  ...

  -- Bit manipulation
  zba                SEQUENCE { } OPTIONAL,
  zbb                SEQUENCE { } OPTIONAL,
  zbc                SEQUENCE { } OPTIONAL,
  ...

}
```



```
RVVConfig ::= SEQUENCE {
  vlen          BIT STRING {
                    vlen128(0),
                    vlen256(1),
                    vlen512(2),
                    vlen1024(3) }
}
```

Linux: extensions discovery



Linux: extensions discovery

- Basic hardware info
  - mvendorid/mimpid/marchid
- ISA extensions
  - F/D/C/V
  - Bitmanip (Zba/Zbb/Zbc/Zbs)
  - Scalar crypto (Zbkb/Zbkc/Zbkx/Zknd/Zkne/Zknh/Zksed/Zksh/Zkt)
  - Vector ctypto (Zvbb/Zvbc/Zvkb/Zvkg/Zvkned/Zvknha/Zvknhb/Zvksed/Zvksh/Zvkt)
  - Zimop/Zcmop
  - Zicboz
  - Zicond
  - and more…

Linux: hwprobe

```
struct riscv_hwprobe {
        int64_t key;
        uint64_t value;
};

int sys_riscv_hwprobe(struct riscv_hwprobe *pairs, size_t pairc,
                                    size_t cpuc, cpu_set_t *cpus, unsigned int flags) {
        return syscall(NR_riscv_hwprobe, pairs, pairc, cpuc, cpus, 0);
}

static struct riscv_hwprobe query[] = {
        {RISCV_HWPROBE_KEY_MVENDORID, 0},
        {RISCV_HWPROBE_KEY_IMA_EXT_0, 0},
        {RISCV_HWPROBE_KEY_ZICBOZ_BLOCK_SIZE, 0},
};

bool probe_features(cpu_set_t *cpus)
{
        int ret;

        ret = sys_riscv_hwprobe(&query[0], sizeof(query) / sizeof(query[0]), sizeof(*cpus), cpus, 0);
        if (ret != 0)
                return false;
        ...

        return true;
}
```

Syntacore™

Custom cores and tools

Q&A