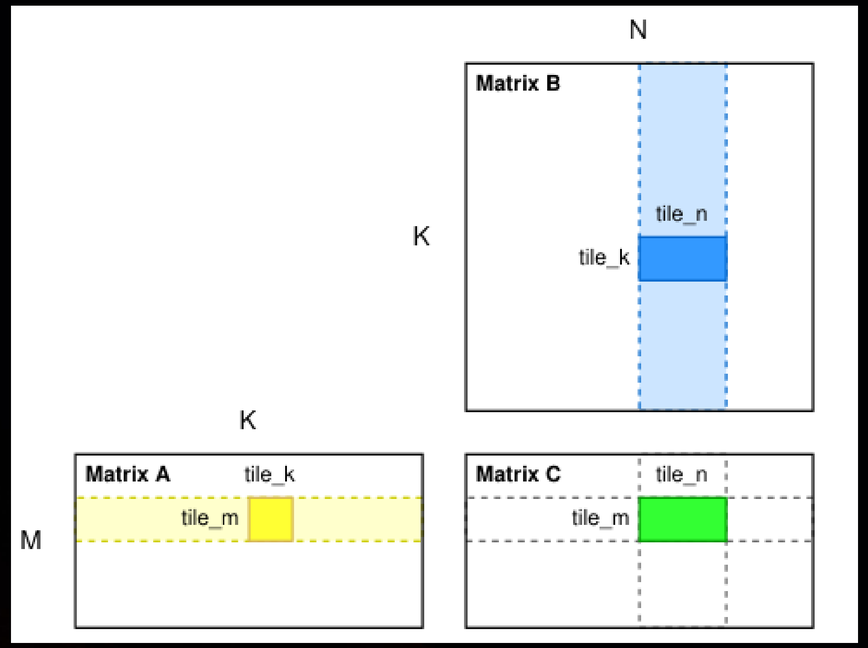
# Experiences with Extending The RISC-V ISA for Matrix/AI

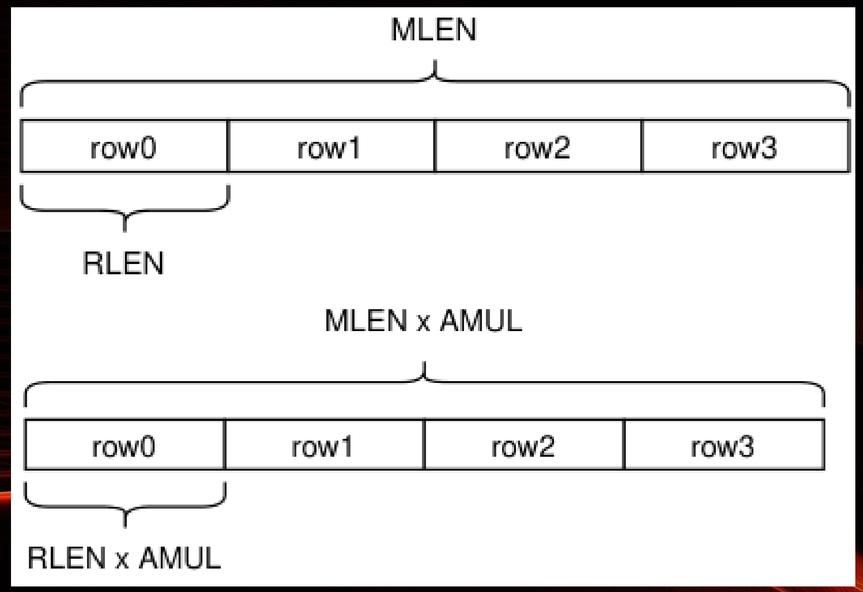
Fan Fujie (范福杰)
R&D Director, Stream Computing

# CONTENTS

# Architecture

- Tile-based Matrix Multiplication.
- RISC-style Instructions & GPR Architecture.
- Configurable Parameters for Implementations.
- Separate Tile Registers & Accumulation Registers.
  - 8 architectural tile registers, tr0 ~ tr7
  - 8 architectural accumulation registers, acc0 ~ acc7
- Unprivileged CSRs for Type & Size Settings.




| tr0 |
|---|
| tr1 |
| tr2 |
| tr3 |
| tr4 |
| tr5 |
| tr6 |
| tr7 |

Tile Register File.

| acc0 |
|---|
| acc1 |
| acc2 |
| acc3 |
| acc4 |
| acc5 |
| acc6 |
| acc7 |

Accumulation Register File.

# Architecture

Type system with optional standard extensions.

| Extension | Type Support | Typical Applications |
|---|---|---|
| Zmi4 | 4-bit integer | Edge-side AI inference; LLM Inference |
| Zmi8 | 8-bit integer | Edge-side AI inference; LLM Inference |
| Zmi16 | 16-bit integer | Scientific computing |
| Zmi32 | 32-bit integer | Scientific computing |
| Zmi64 | 64-bit integer | Scientific computing |
| Zmf8e4m3 | FP8 (E4M3) | AI inference & training (forward) |
| Zmf8e5m2 | FP8 (E5M2) | AI inference & training (backward) |
| Zmf16e5m10 | FP16 | Cloud-side AI inference & training |
| Zmf16e8m7 | BF16 | Cloud-side AI training (or inference) |
| Zmf32e8m23 | FP32 | Scientific computing |
| Zmf19e8m10 | TF32 | Cloud-side AI inference & training |
| Zmf64e11m52 | FP64 | Scientific computing |

# Architecture

希姆计算 STREAM COMPUTING

**Config-setting Instructions**

| | |
|---|---|
| msettype | msettypei |
| msettilem | msettilemi |
| msettilen | msettileni |
| msettilek | msettileki |
| … | ... |

**Load/Store Instructions**

| | |
|---|---|
| mlae*.m | msae*.m |
| mlbe*.m | msbe*.m |
| mlce*.m | msce*.m |
| mltre*.m | mstre*.m |
| … | ... |

**Data Move Instructions**

| | |
|---|---|
| mmve*.a.t | mfmve*.f.t |
| mmve*.t.a | mbcar.m |
| mmve*.x.a | mbcaee*.m |
| mmve*.x.t | mtae*.m |
| … | ... |

**Matrix Multiply Instructions**

| | |
|---|---|
| mma.mm | moma.mm |
| msma.mm | mfma.mm |
| mwma.mm | mfwma.mm |
| mqma.mm | mfqma.mm |
| … | ... |

**Element-wise Instructions**

| | |
|---|---|
| madd.mm | mfadd.mm |
| msub.mm | mfsub.mm |
| mmin.mm | mfmin.mm |
| mmax.mm | mfmax.mm |
| … | ... |

**Type-convert Instructions**

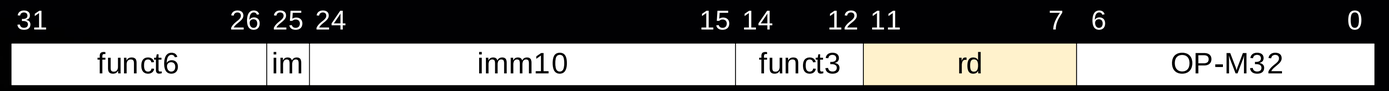
| | |
|---|---|
| mcvt.x.xu.m | mfwcvt.fw.f.m |
| mcvt.xu.x.m | mfncvt.f.fw.m |
| mwcvt.xw.x.m | mfcvt.x.f.m |
| mncvt.x.xw.m | mfcvt.f.x.m |
| … | ... |

https://github.com/riscv-stc/riscv-matrix-spec

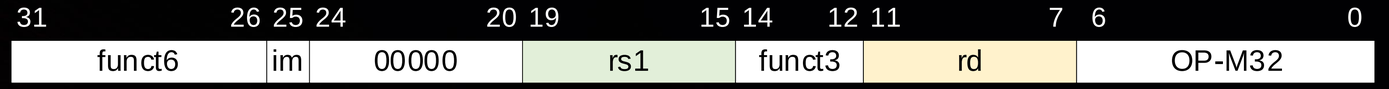
# Architecture

## Instruction Encoding (32-bit)

| Field | Desc. |
|---|---|
| OP-M32 | The major opcode (1110111) |
| funct6 | Function |
| imm | Immediate |
| ls | Load or store |
| tr | Transposed |
| di | Moving directions |
| fp | Float-point operations |
| sa | Destination saturated |
| sn | Source signed (for integer) |
| ma | Matmul or element-wise |
| nw | Narrowing or widening (for cvt) |
| eew | Element width |

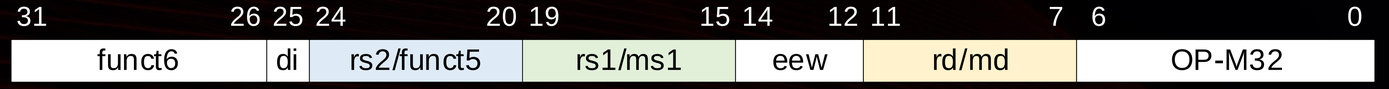



Configuration Immediate Instructions.




Configuration Instructions.


Load/Store Instructions.




Data Move Instructions.


Arithmetic & Logic Instructions.


Type-convert Instructions.

# Architecture

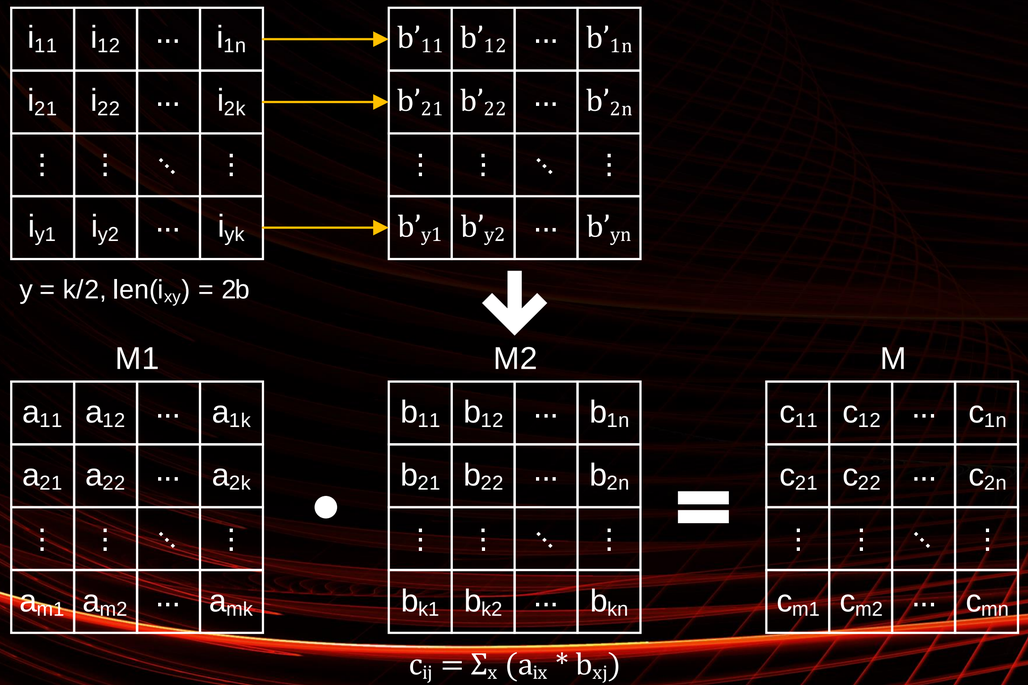
Intrinsic Example for Matrix Multiplication.

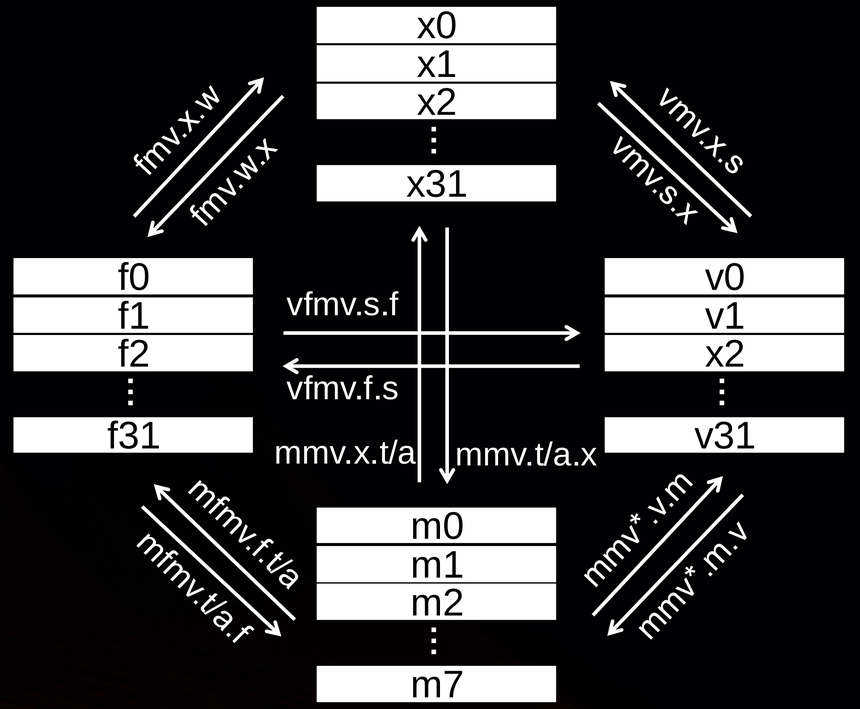
```
void matmul_float16(c, a, b, m, k, n) {
    msettype(e16);                              // use 16bit input matrix element
    for (i = 0; i < m; i += mtilem) {           // loop at dim m with tiling
        mtilem = msettilem(m-i);
        for (j = 0; j < n; j += mtilen) {       // loop at dim n with tiling
            mtilen = msettilen(n-j);

            out = mwsub_mm(out, out)            // clear output reg
            for (s = 0; s < k; s += mtilek) {   // loop at dim k with tiling
                mtilek = msettilek(k-s);

                tr1 = mlae16_m(&a[i][s], k*2);  // load left matrix a
                tr2 = mlbe16_m(&b[s][j], n*2);  // load right matrix b
                out = mfwma_mm(tr1, tr2);       // tiled matrix multiply,
                                                // double widen output
            }

            out = mfncvt_f_fw_m(out);           // convert widen result
            msce16_m(out, &c[i][j], n*2);       // store to matrix c
        }
    }
}
```

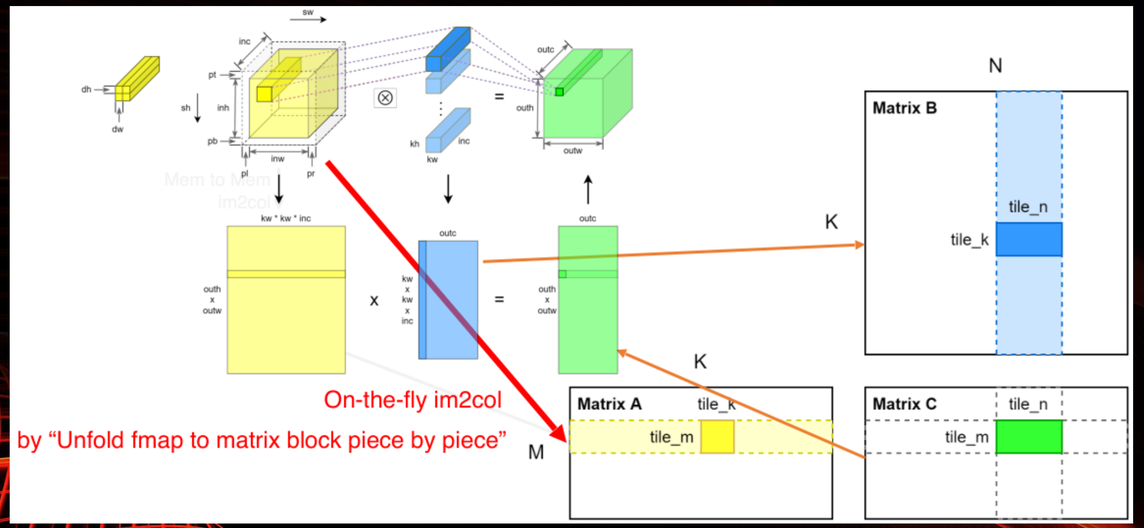
# Architecture

**Standard Extensions：**

- Type-support extensions: Zmi4, Zmf8e4m3, etc.
- Matrix mode extensions: Zmab, Zmabt, Zmatb
- Matrix-vector extension: Zmv
- Im2col/col2im extensions: Zmi2c, Zmc2i
- Matrix sparsity extensions: Zmspa, Zmspb



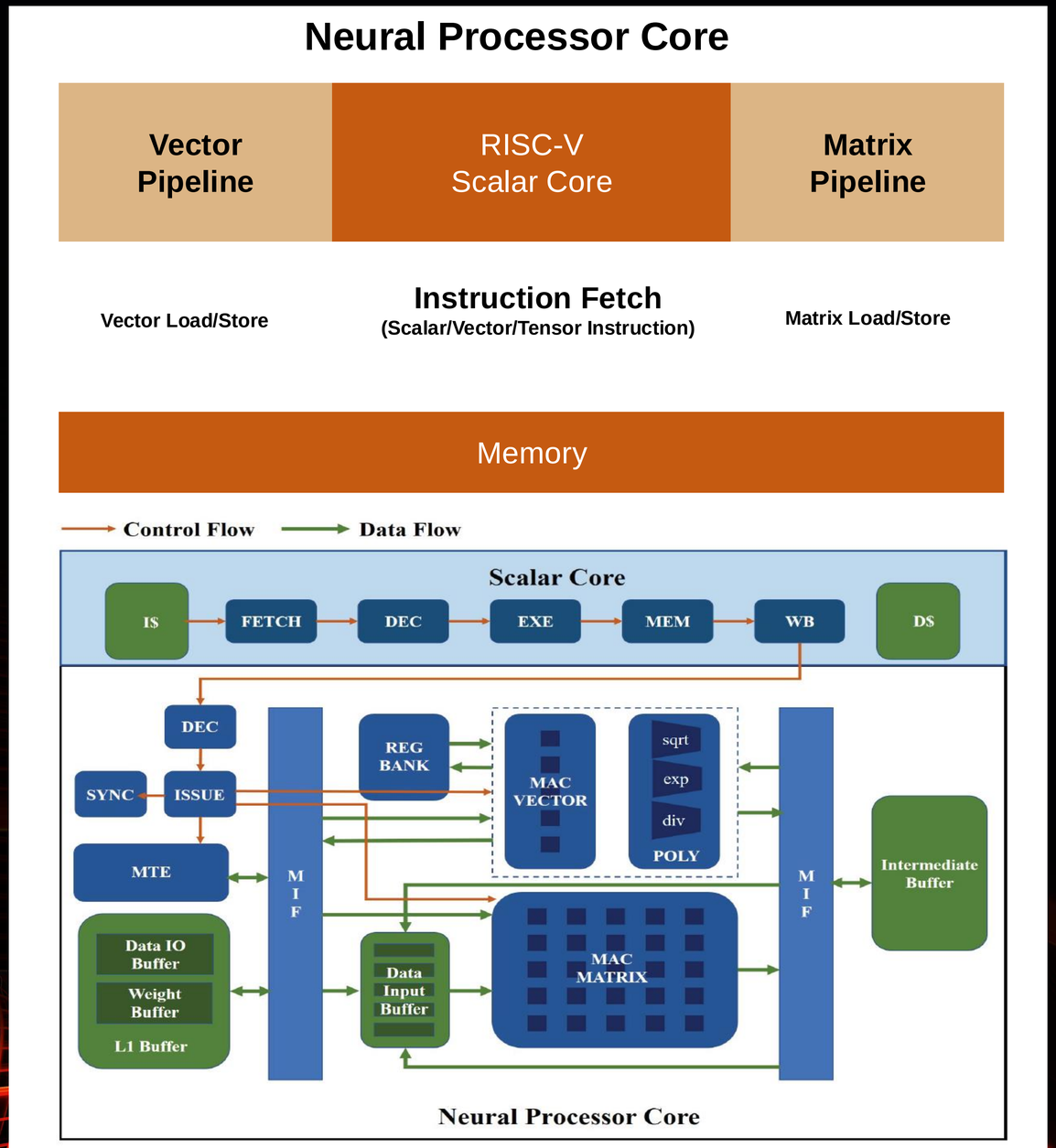



# Implementation

## STCP920 Artificial Intelligent Computing Chip

| | |
|---|---|
| **Architecture** | RISC-V Extended DSA (Vector+Matrix Inside) |
| **Data Type** | FP16/INT8 |
| **Integer** | 256 TOPS @ INT8 |
| **Float-point** | 128 TFLOPS @ FP16 |
| **Memory** | 16 GB LPDDR4X |
| **Process** | 12nm |
| **Application** | Inference |
| **Power** | 160W |
| **Virtualization** | Support 1/2/4 |
| **SDK** | Stream Computing Software Development Kit |

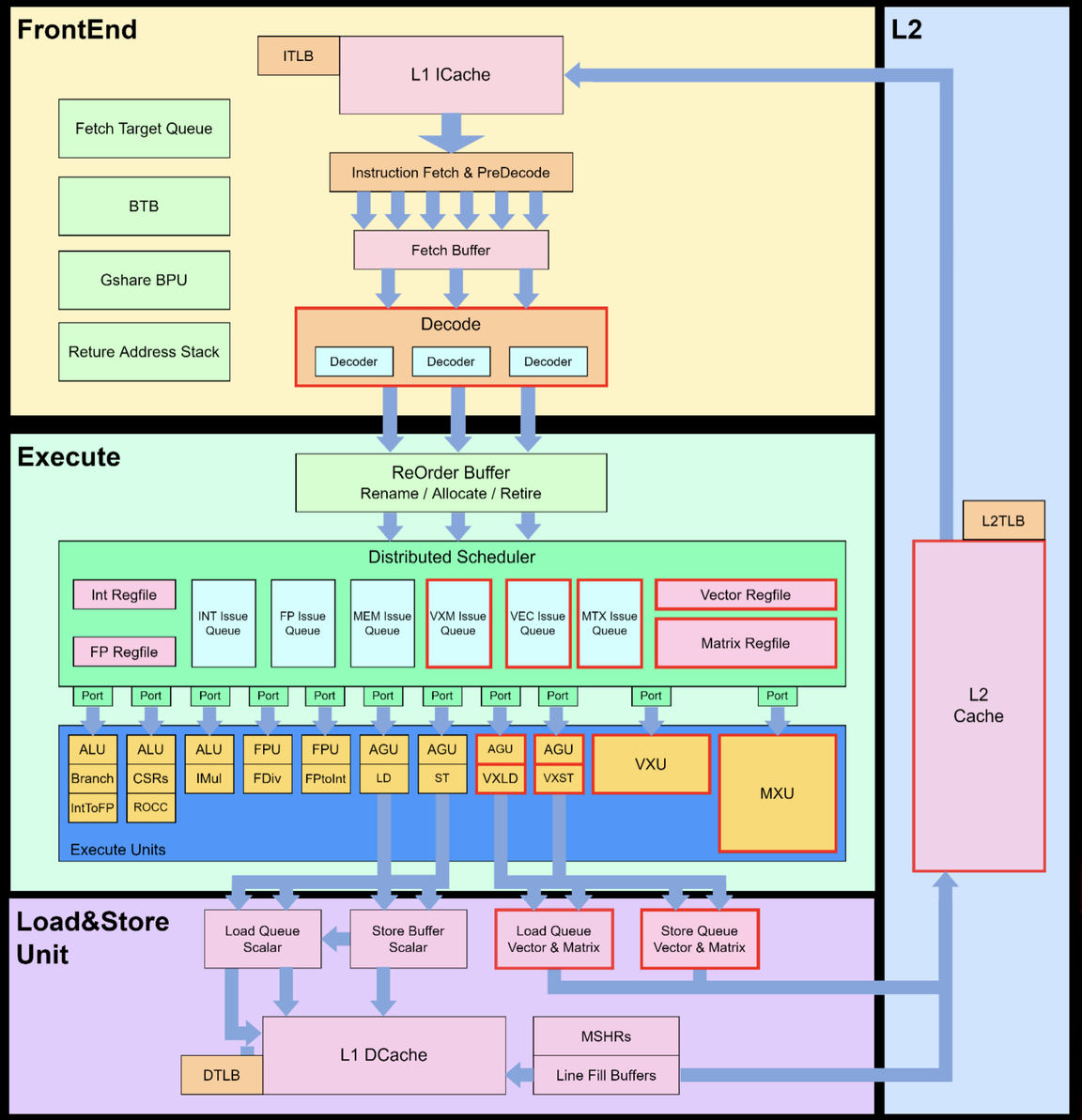
# Implementation

## Stream Computing Out-of-Order Processor

### SCOOP = BOOM + Vector + Matrix

- 10-stage pipeline
- Out-of-order issue & execute
- RVV 1.0: VLEN from 128 to 1024
- RVMatrix 0.2: MLEN from 1024 to 65536
- Register renaming for RVV & RVMatrix
- Speculative execution for CSR configs
- Outer production matrix MAC
- Inner production matrix MAC

https://github.com/riscv-stc/chipyard



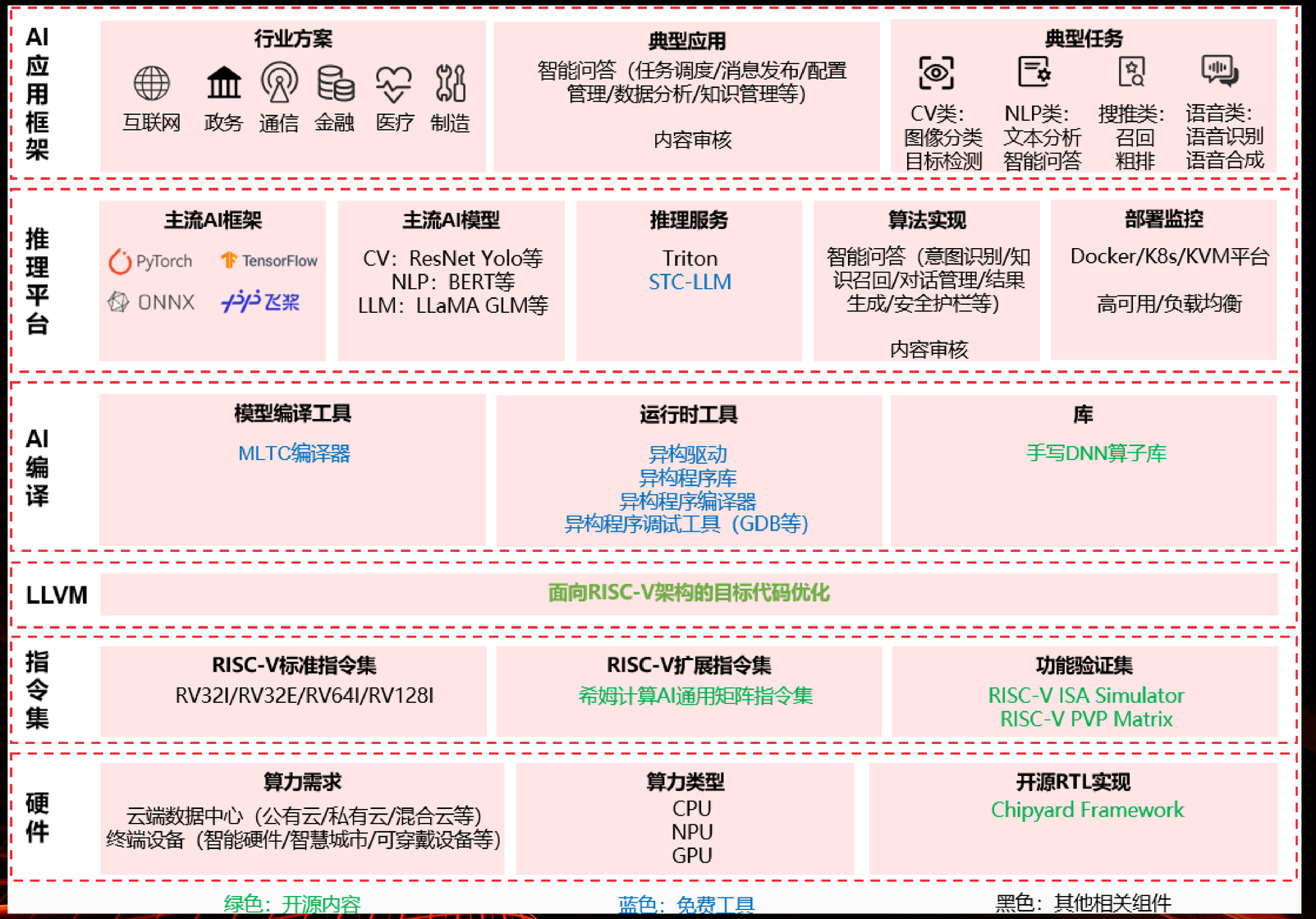
# Eco-system

**General Components：**

- Specification & Intrinsic Lib
- Compiler & Assembler
- Debug Tools（GDB）
- Simulators (Spike/QEMU）
- Verification Tools (RVPVP)
- Operator/Model Lib
- Open-source RTL Platform

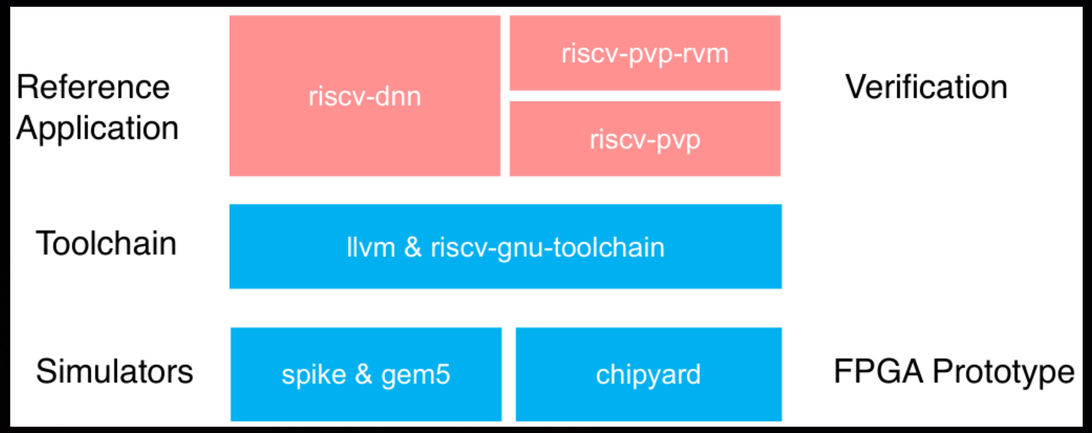
**Dedicated Components：**

- Heterogenous Programming Environments (HPE)
- AI Compilers（MLIR/TVM）
- Binary Translation Tools



# Open-source Projects

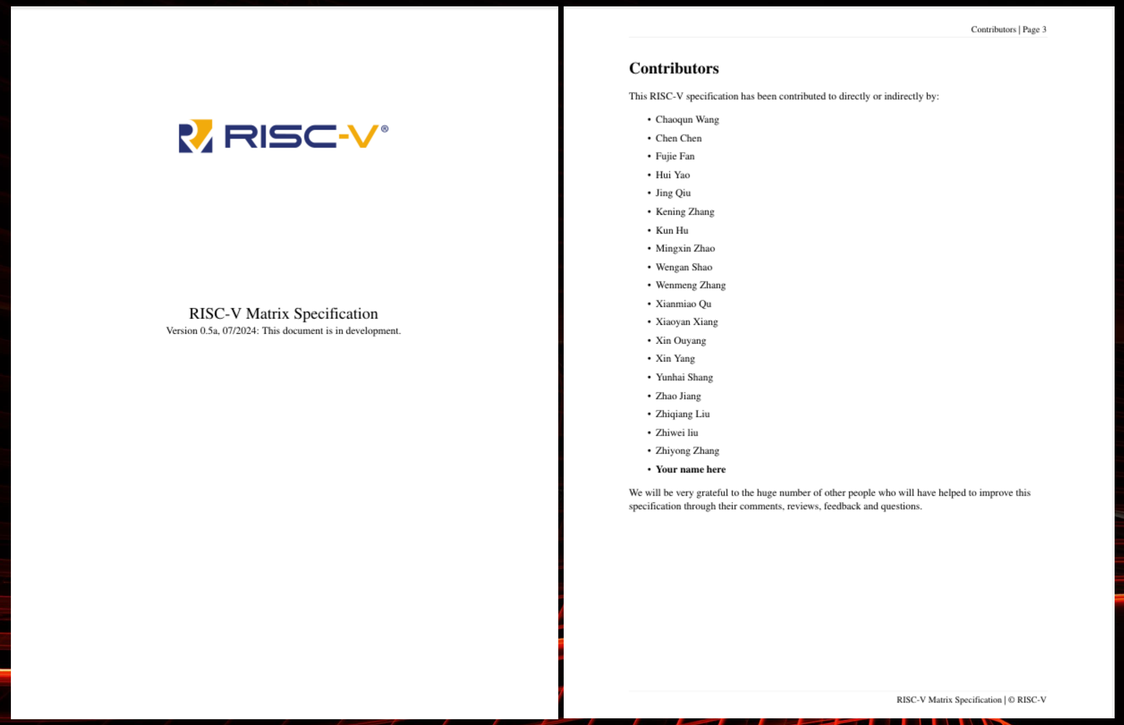
- **riscv-matrix-spec**
  - The matrix extension proposal.
- **llvm-project**
  - LLVM toolchain to support matrix extension proposal.
- **riscv-openocd-matrix**
  - GDB debug tool for matrix programs.
- **riscv-isa-sim**
  - Spike ISS to support matrix extension proposal.
- **riscv-pvp-matrix**
  - RISC-V matrix extension ISA verification using RISC-V PVP.
- **riscv-dnn**
  - A small DNN library for RISC-V, using RISC-V vector and matrix extensions.
- **chipyard**
  - Chipyard project to support matrix extension proposal with BOOM core.



# Acknowledgement

## Contributors:

- Chaoqun Wang、Chen Chen、Fujie Fan、Hui Yao、Jing Qiu、Kening Zhang、Kun Hu、Mingxin Zhao、Wengan Shao、Wenmeng Zhang、Xianmiao Qu、Xiaoyan Xiang、Xin Ouyang、Xin Yang、Yunhai Shang、Zhao Jiang、Zhiqiang Liu、Zhiwei liu、Zhiyong Zhang
- **Your name here**



RISC-V Matrix Specification

Version 0.5a, 07/2024: This document is in development.



### Contributors

This RISC-V specification has been contributed to directly or indirectly by:

- Chaoqun Wang
- Chen Chen
- Fujie Fan
- Hui Yao
- Jing Qiu
- Kening Zhang
- Kun Hu
- Mingxin Zhao
- Wengan Shao
- Wenmeng Zhang
- Xianmiao Qu
- Xiaoyan Xiang
- Xin Ouyang
- Xin Yang
- Yunhai Shang
- Zhao Jiang
- Zhiqiang Liu
- Zhiwei liu
- Zhiyong Zhang
- **Your name here**

We will be very grateful to the huge number of other people who will have helped to improve this specification through their comments, reviews, feedback and questions.

希姆计算
STREAM COMPUTING

# THANKS

希姆计算公众号

希姆计算开源主页