SiFive

Creating Agile AI Systems with Open RISC-V Solutions

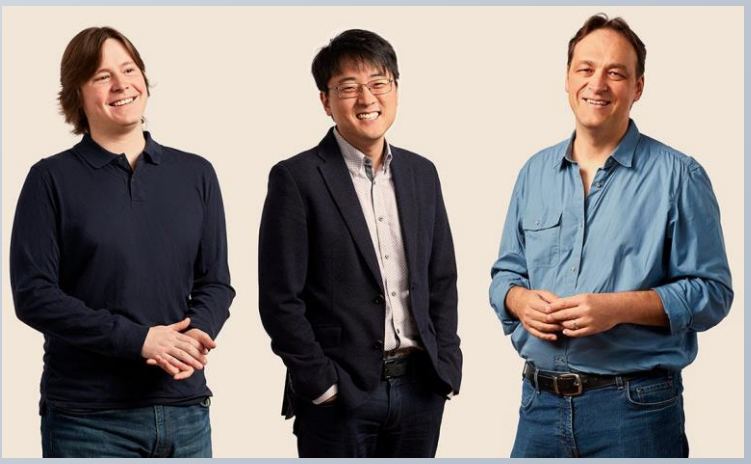
Hong-Rong Hsu 許宏榮
Senior Staff Engineer, SiFive

April 25, 2024

RISC-V: The Global Open-Standard ISA

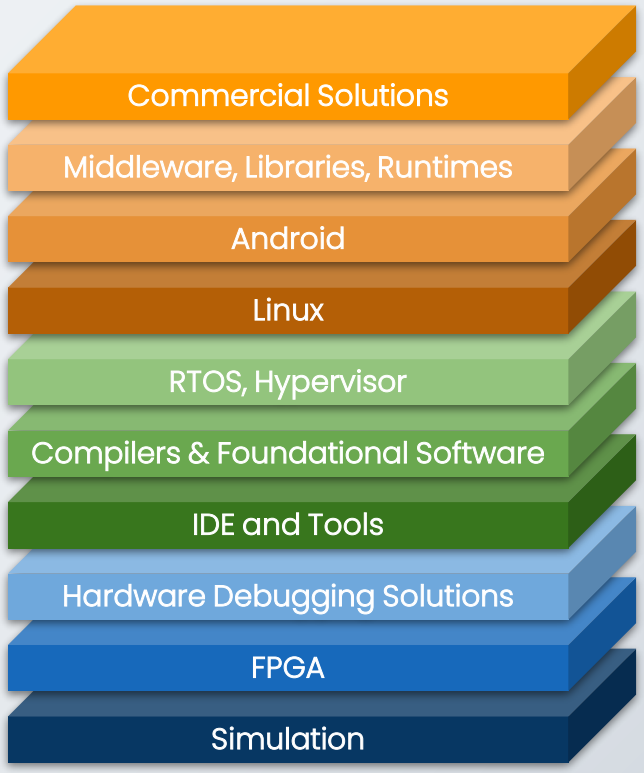
- A single freely available open-standard managed by RISC-V International
  - Many new extensions in flight, quickly achieving parity with other ISAs
- Implementations being developed all around the world
- Over 10 billion RISC-V SoCs shipped worldwide
- Already a commercial reality
- Established architecture in embedded space and AI accelerators
- Major semiconductor makers announce support of RISC-V application processors
- Ecosystem maturing rapidly for consumer and infrastructure markets

The Inventors of RISC-V



SiFive's founders are the same UC Berkeley professor and PhDs who invented the RISC-V Instruction Set Architecture (ISA) in 2010

RISC-V Software Ecosystem



The success of RISC-V is built on open standards

Open architecture driving exponential growth

Android on RISC-V is a first-class citizen

SiFive is the biggest contributor of RISC-V software tools and OS

RISE consortium accelerates software optimizations on RISC-V

RISE

Google SAMSUNG intel Qualcomm NVIDIA MEDIATEK Red Hat

SiFive Imagination VENTANA Rivos ANDES TECHNOLOGY T-HEAD

SiFive
Company Update

What does SiFive do?

350

design wins

SiFive

intel. Qualcomm SAMSUNG Google

amazon.com MARVELL RENESAS MICROCHIP

ALPHAWAVE SEMI FURIOSA WD Western Digital Synaptics

Coherent Logix™ tenstorrent FADU ARCHITEK

SOPHGO huami ESWIN Bouffalo Lab

+

Multiple Top 10 Semi manufacturers
Multiple Top 5 Robotaxi manufacturers
Multiple Top 5 US Datacenter & Storage suppliers
Multiple Top Datacenter vendors in Asia
Publicly Listed Chinese Car OEM
Multiple A&D Prime Contractors

A lot more than Embedded now

The Undisputed Leader in RISC-V Computing

Broadest portfolio of processors from embedded to high-performance computing

CPU Cores

SiFive Essential™

32 and 64-bit Processors

- Microcontrollers, IoT devices, real-time control, control-plane processing
- Highly customizable to application-specific requirements
- Mature, industry-proven designs

SiFive Performance™

64-bit Application Processors

Consumer

- High-performance RISC-V processor with best compute density and power efficiency
- Android ready

Infrastructure

- Highest performance, most advanced RISC-V processor
- Scale-out, high-performance, processing capabilities with vector compute, NoC and D2D

AI Cores

SiFive Intelligence™

Scalable 64-bit AI Processors

- Edge AI, Cloud, Training, Inference
- Very high performance and efficiency for AI workloads with vector processing
- Built on top of RISC-V Vectors, SiFive Intelligence Extensions and AI hardware accelerators

Functional Safety

SiFive Automotive™

32/64-bit Safety Processors

- Broadest range of RISC-V safety processors, from MCU to high-performance SoC, with ASIL-B and ASIL-D options
- Multi-core/cluster, vectors, virtualization, and security features
- Strong automotive RISC-V ecosystem

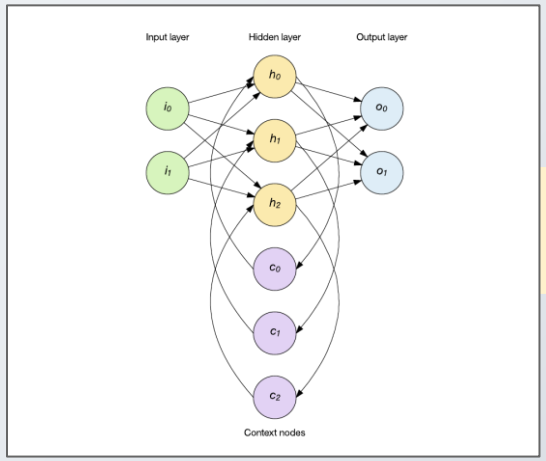
SiFive

RISC-V enables high performance compute for AI everywhere

AI's constant evolution requires agile hardware

SiFive

Recurrent Neural Network (RNN)



Motion Processing
Speech Recognition
Trading / Financials

Convolutional Neural Network (CNN)

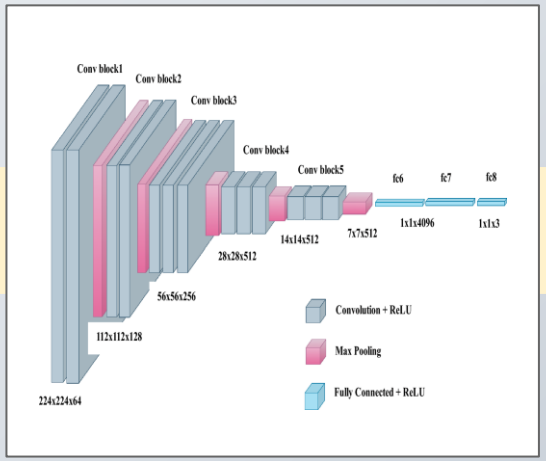
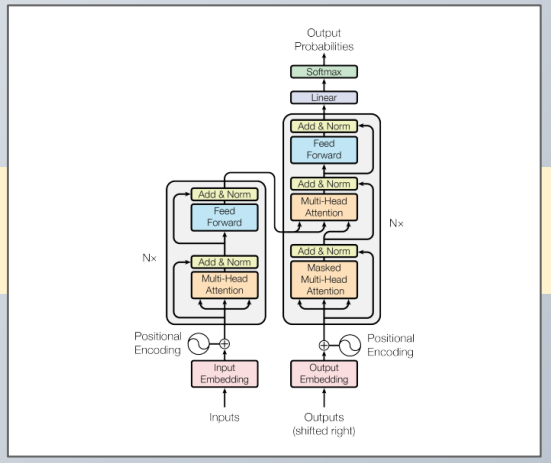


Image processing
Object Identification
Handwriting

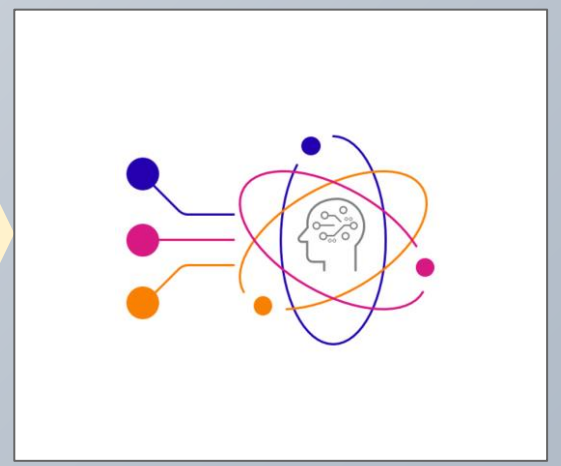
Transformer Network

Attention Mechanism



Natural Language Processing
Generative AI
Image Processing

AI Next Network What will it be?



Need for programmable AI hardware
Need for open standards for AI (RISC-V Vectors + LLVM + MLIR)

Future-proof NPUs require programmable AI hardware and flexible software based on open standards

RISC-V Vector Extension

RISC-V designed to support a capable scalable vector extension

- Vectors one of the original reasons we built RISC-V
- Now a RISC-V International ratified open standard

RISC-V vectors optimized for AI space:

- Efficient support of mixed-precision datatypes (narrowing/widening)
- Efficient support of sparse compute (compress, gather)
- Scalable to large vector lengths while maintaining binary compatibility
- Designed as compiler target
- Same ISA scales down to efficient microcontrollers, up to high-performance applications processors, and beyond to AI-supercomputer-class systems

Vectorize once, deploy everywhere

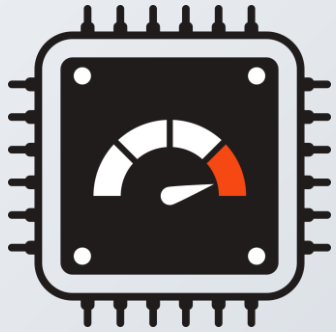
1D Vectors versus 2+D Matrix for AI Acceleration

SiFive

- While most compute is in easily accelerated, easily coded 2+D matrix operations, most complexity and diversity is in 1D vector operations
- Newer more sophisticated models also tend to shift workload back to 1D vector operations
- Tight integration of **scalar+vector+matrix** compute, vector efficiency, and vector software support are key for effective AI experimentation and deployment

SiFive Intelligence

SiFive is empowering the new computing era

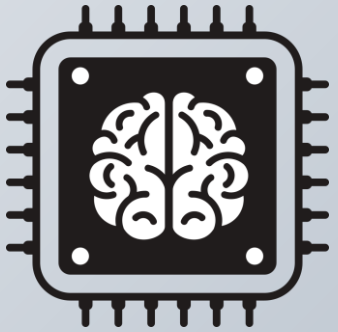


Large-scale high-performance general-purpose CPU

SiFive Performance Family

P470 P670

P870



High-performance NPU

SiFive Intelligence Family
Vector CPU

Built-in AI hardware engine

or

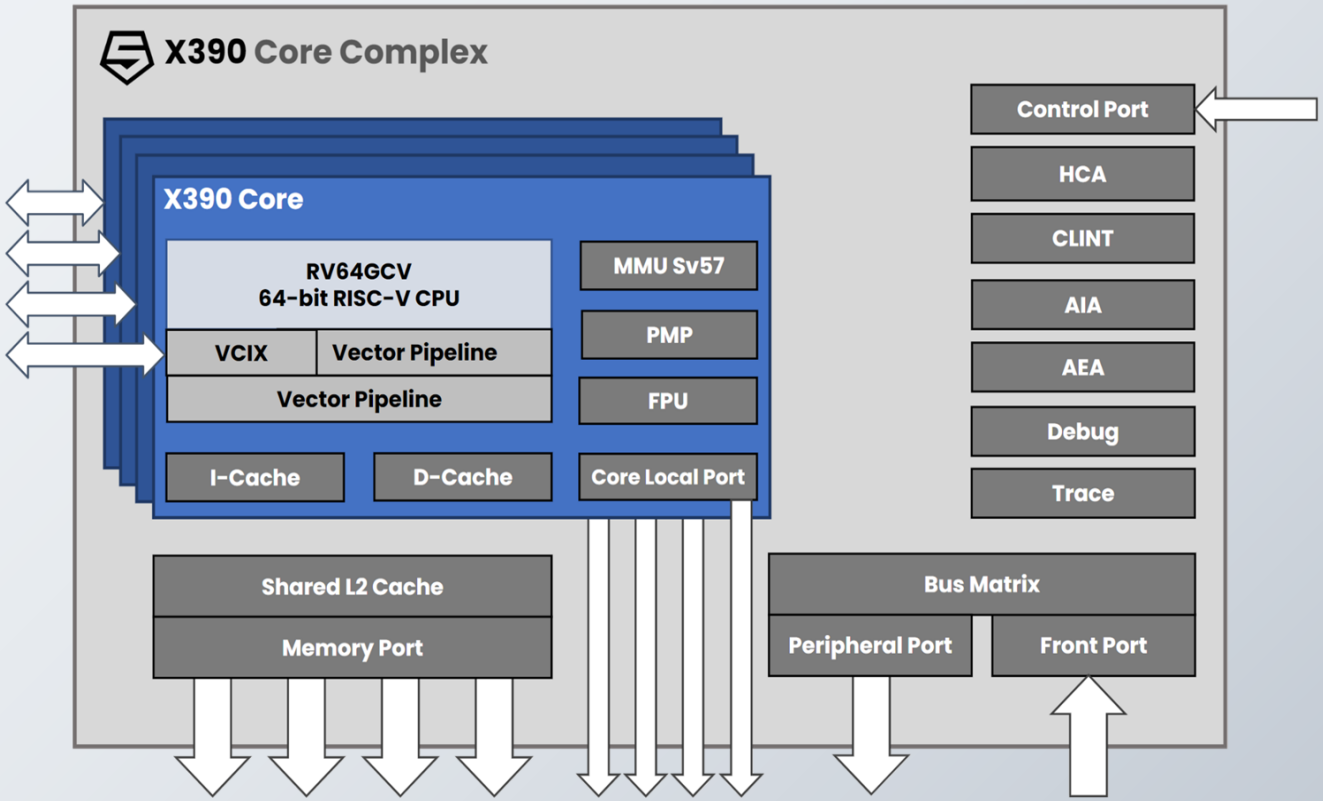
SiFive Intelligence Family
Vector CPU

Customer AI hardware engine

X280

X390

SiFive Intelligence X380/X390 cores



Vector Processing for AI/ML workloads

- SiFive Intelligence Extensions, custom instructions that accelerate AI/ML performance critical operations
- VCIX interface for direct connectivity of vector accelerators
- VLEN:DLEN at 2:1 ratio
- Separate vector load/store units (full-duplex operation)
- Dual vector ALU implementation available
- 1024-bit VLEN (X390), 512-bit VLEN (X380)

Scalar processing architecture

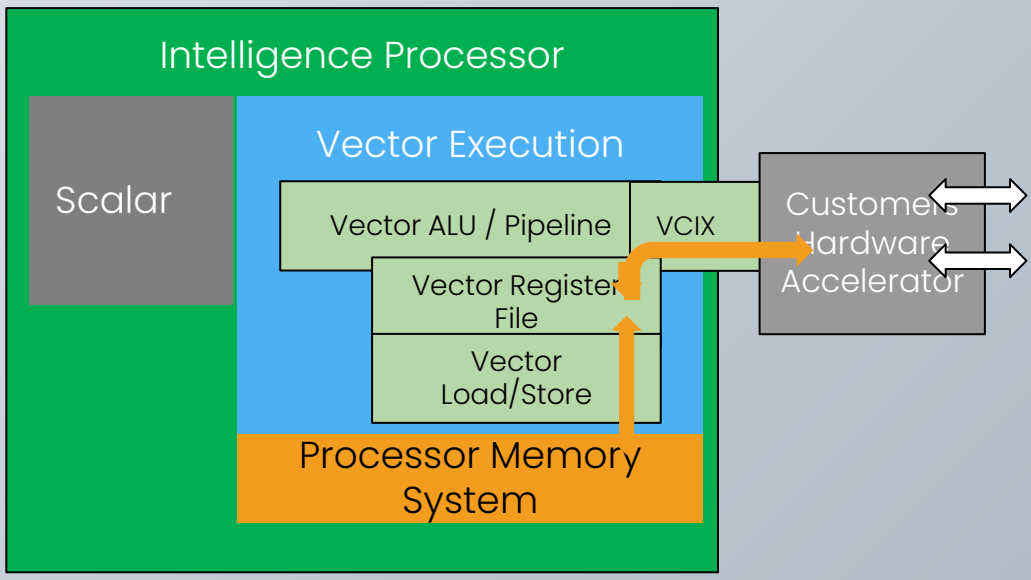
- 64-bit RISC-V ISA. 8-stage dual-issue in-order pipeline
- Linux-capable Applications processor. MMU, Caching architecture

Scalar Performance

- 5.7 CoreMarks/MHz 3.3 Dhrystone/MHz
- 4.5 SpecINT2006/GHz 3.4 SpecFP2006/GHz (HPC)

Vector Coprocessor Interface eXtension (VCIX)

- Low-latency interface to vector pipeline and register file
- Support of broad range of external hardware accelerators
- Handshaking for variable latency with Request/Response protocol.
- Two functional modes:
  - Initiate / Complete (data reads, write acknowledge)
  - Command or Data push (no acknowledge needed)
- LMUL support (⅛ to 8)
- C-intrinsic programming model, with Freedom tools, Auto-vectorization compiler
- RTL, SystemC and SystemC-TLM simulation support packages

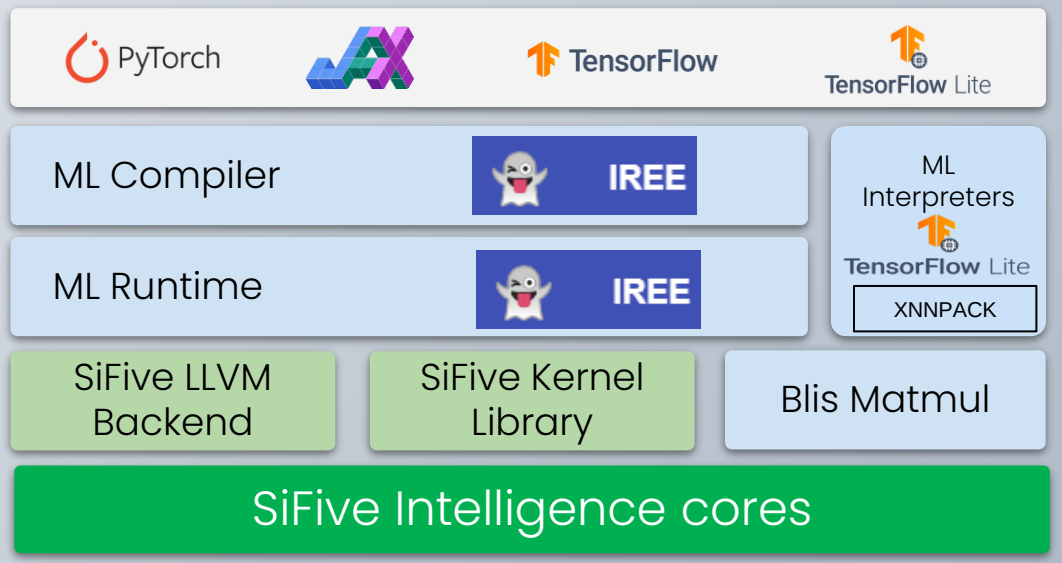


SiFive Intelligence software support

SiFive-optimized algorithms integrated into MLIR-based IREE framework

IREE: Full-featured AI framework

- MLIR based AI/ML Compiler + Runtime
- 80% AOT code-gen + 20% hand-tuned library
- Compatible with TF/TFLite, PyTorch, etc.
- Targets all system scales
  - embedded to data center
  - bare-metal to Linux
- SiFive enhancements at all levels of the stack
- 3+ year SiFive/Google/AMD collaboration
  - See C4ML 2023 talk

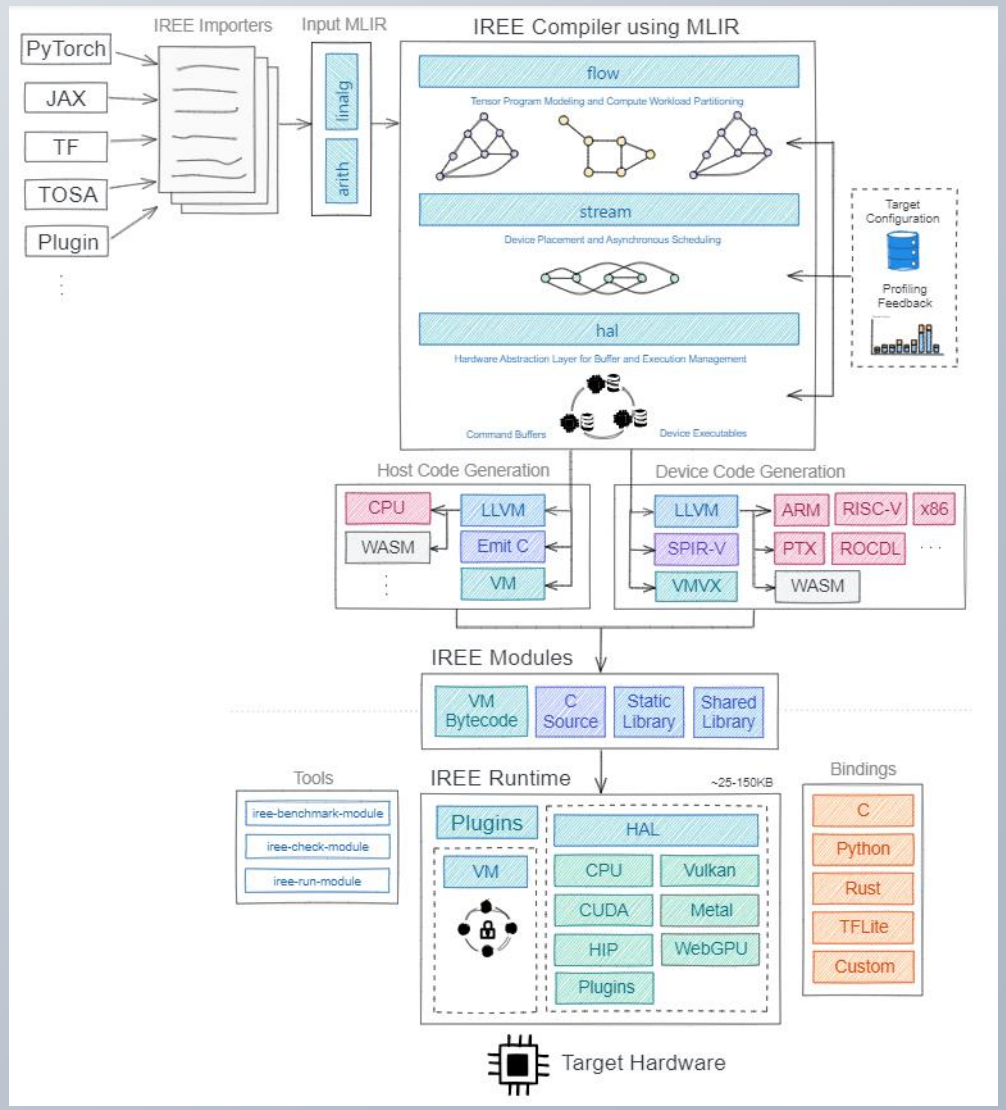


Note: IREE is moved out of Google to become an independent project

Open Source SiFive contribution | SiFive Software | SiFive Hardware IP

Software Stack Integration

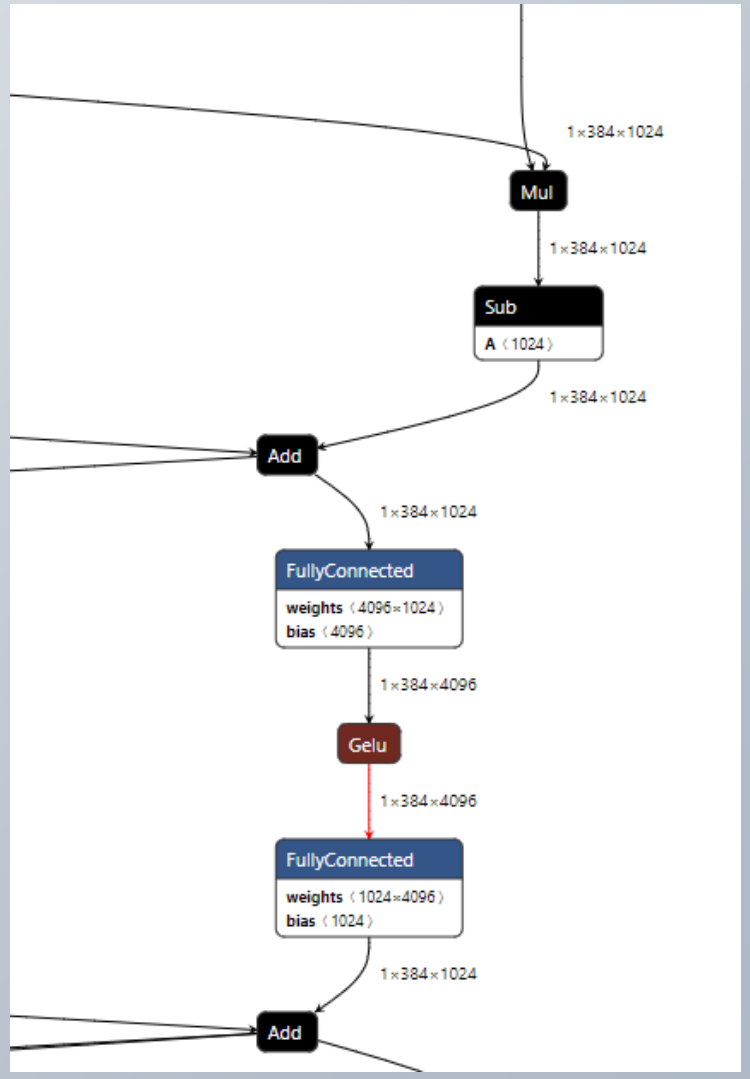
- Total solution
  - Take all SiFive Software stack
- SiFive CPU + Custom TPU based on **SiFive Software Stack**
  - Custom ukernel plug-in
  - Custom MLIR code-gen pass
  - HAL(Hardware Abstraction Layer) driver plug-in
- SiFive CPU + Custom TPU based on **Customer's Software Stack**
  - SiFive Kernel Library for CPU performance
  - VCIX MLIR/Library/Intrinsics for TPU delegation



VCIX MLIR Dialect

- Why VCIX?
  - Tight coupling between CPU and TPU/co-processor
  - CPU and co-processor runs in parallel
  - One program with scalar, vector, and co-processor instructions interleaved
- VCIX dialect allows people to
  - manage the control & optimization flow in MLIR's world

```
scf.for %arg0 = %c0 to %c384 step %c1 {
  scf.for %arg1 = %c0 to %c4096 step %vl {
    // RVV process GELU
    // Push %vl data from vector registers to co-processor via VCIX
  }
  scf.if (%arg0 == %matmul_tile_m_size) {
    // Send command to trigger co-processor for matmul via VCIX
  }
}
// Pop the matmul results back via VCIX
```



BERT-large usage example

Model deployment through IREE to SiFive Intelligence

SiFive

PyTorch

BERT-large-uncased, FP32
BERT-large-uncased, FP16
EfficientNet_b0, FP32
EfficientNet_b7, FP32
Deit-small-distilled, FP32
MobileBERT, FP32
VisionTransformer base, FP32
MobileNet v3 small, FP32
Albert-V2, FP32
AlexNet, FP32
BERT-base, FP32
ViT, FP32
MiniLM, FP32
MobileNetV3-Small, FP32
SqueezeNet, FP32
ResNet18, FP32
ResNet50 V1/V2, FP32
ResNet101, FP32
WideResNet50, FP32
…etc

https://github.com/nod-ai/SHARK/tree/main/tank
https://github.com/nod-ai/SHARK-TestSuite/tree/main/e2eshark

Tensorflow

MiniLM, INT32
ResNet50, FP32
BERT for Masked LM, FP32
BERT Large, FP32
EffcientNet, FP32
Albert-V2
BERT-Base ucased, INT32
Distillbert ucased, INT32
EffcientNet v2, FP32
Electra_small-discriminator, FP32
VisionTransformer base, FP32
LayoutLM-base-uncased, INT32
StableDiffusion, FP32
CamemBERT
ConvBERT-Base
DistilBERT
ConvNeXT-Tiny
MobileBERT
ViT
LayoutLM
RoBERTa
GPT2, INT8
…etc

TFLite

MobileBert, FP32 & INT8
MobileBert-EdgeTPU-S, FP32 & INT8
Deeplab V3, FP32
Posenet, FP32
Visual Wake Words, INT8
SqueezeNet, FP32
DeeplabV3, FP32
Resnet50, INT8
DenseNet, FP32
EfficientNet-Lite, INT8
Inception-V4, UINT8
Bird Classifier, FP32
Inception-V4, FP32
MobileNet V1, FP32
MobileNet V2, FP32 & INT8
MobileNet v3 small, FP32
SSD_MobileNetV1, UINT8
SSD_MobileNetV2, FP32 & NT8 & UINT8
SSD-SpaghettiNet-Large, FP32 & UINT8
Rosetta/AlBert/Midas, FP32
Lightning, INT8
Magenta/MNAsNet, FP32
…etc

More Open Source Projects Engagement

SiFive

Llama.cpp

- A pure C/C++ inference framework
- SiFive optimized GGML library
- Achieve TinyLlama-1B real-time performance on CPU-only SiFive platform:

```
llama_new_context_with_model: n_ctx      = 2048
llama_new_context_with_model: freq_base  = 10000.0
llama_new_context_with_model: freq_scale = 1
llama_kv_cache_init:        CPU KV buffer size =    44.00 MiB
llama_new_context_with_model: KV self size  =   44.00 MiB, K (f16):   22.00 MiB, V (f16):   22.00 MiB
llama_new_context_with_model:        CPU input buffer size   =     9.02 MiB
llama_new_context_with_model:        CPU compute buffer size =   144.00 MiB
llama_new_context_with_model: graph splits (measure): 1

system_info: n_threads = 5 / 5 | AVX = 0 | AVX_VNNI = 0 | AVX2 = 0 | AVX512 = 0 | AVX512_VBMI = 0 | AVX512_VNNI = 0 | FMA = 0 | NEON = 0 | ARM_FMA = 0 | F16C = 0 | FP16_VA = 0 | W
 MATMUL_INT8 = 0 |
sampling:
        repeat_last_n = 64, repeat_penalty = 1.100, frequency_penalty = 0.000, presence_penalty = 0.000
        top_k = 40, tfs_z = 1.000, top_p = 0.950, min_p = 0.050, typical_p = 1.000, temp = 0.700
        mirostat = 0, mirostat_lr = 0.100, mirostat_ent = 5.000
sampling order:
CFG -> Penalties -> top_k -> tfs_z -> typical_p -> top_p -> min_p -> temperature
generate: n_ctx = 2048, n_batch = 512, n_predict = -1, n_keep = 0


 Building a website can be done in 10 simple steps:\nStep 1: You need to have a domain name and an email address. GoDaddy, Bluehost, and Namecheap are some popular domain registra
al options available such as Bluehost, Hostgator, and Dreamhost. Step 3: Install WordPress by visiting the WordPress official website. Once you have installed WordPress, you will
. Step 4: Choose a theme for your blog. There are thousands of themes available online at ThemeForest or Genesis Framework. Step 5:

llama_print_timings:        load time =     910.33 ms
llama_print_timings:      sample time =      92.11 ms /   129 runs   (    0.71 ms per token,  1400.58 tokens per second)
llama_print_timings: prompt eval time =    4887.39 ms /    20 tokens (  244.37 ms per token,     4.09 tokens per second)
llama_print_timings:        eval time =   39539.60 ms /   128 runs   (  308.90 ms per token,     3.24 tokens per second)
llama_print_timings:       total time =   44893.71 ms /   148 tokens
```

Summary

- SiFive ready to deliver on all future AI computing demands
  - From edge to cloud
  - HW: CPU only, CPU + built-in AI engine, CPU + custom NPU
  - SW: SiFive AI/ML Software Stack
- SiFive AI/ML Software Stack is Open
  - MLIR based IREE compiler/runtime
  - LLVM RVV code-gen, VCIX MLIR dialect
  - XNNPACK, Libraires
- Contact us if you are interested in RISC-V AI solutions!
  - hongrong.hsu@sifive.com
  - https://www.sifive.com/contact-sales
  - We are hiring!

SiFive

WE ARE HIRING

Bring your innovations to life faster

Apply Now!
https://www.sifive.com/careers/

Intern

Design Verification

- BS (4th grade)/MS (1st or 2nd grade) Degree in CS/EE/CE or related fields
- C/C++/Python/Java
- Period: 7/1/2024 to 9/30/2024

New Grad

Architecture Performance Analysis Engineer

- BS/MS Degree in CS/EE/CE or related fields
- Familiar with Computer Architecture
- Assembly, C/C++, Python or Shell

Empowering innovators

www.sifive.com