StarFive
赛昉科技

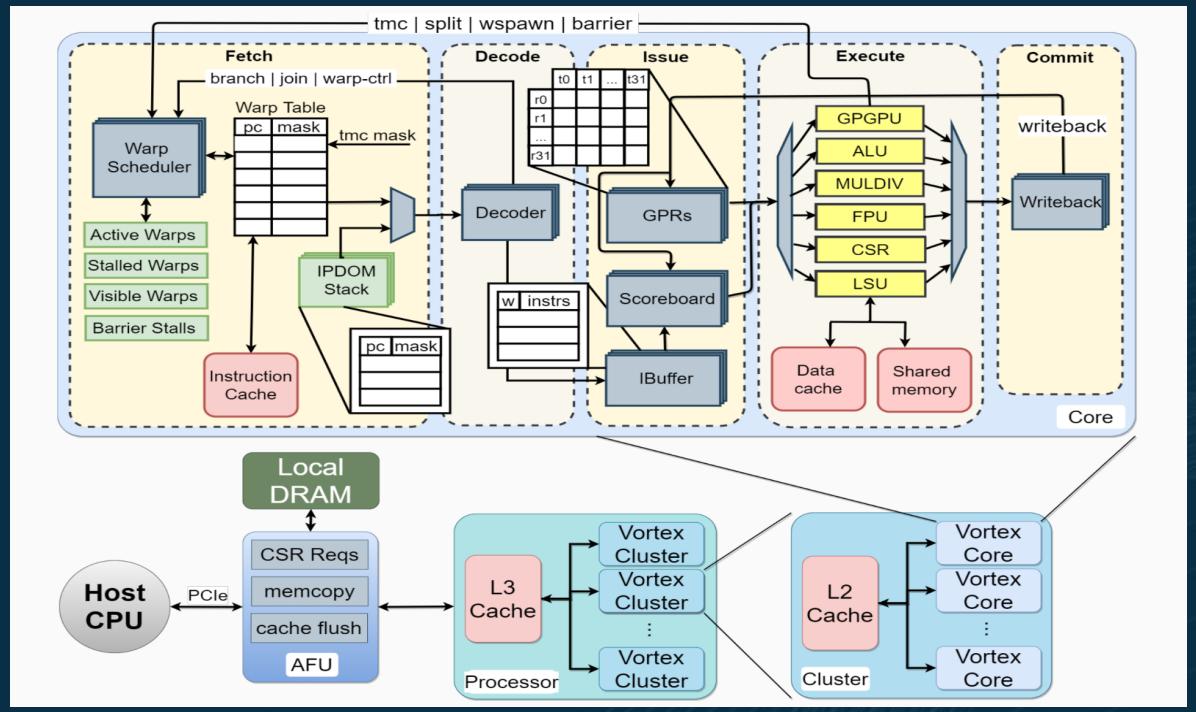
CUDA On RISC-V GPGPU

夏品正

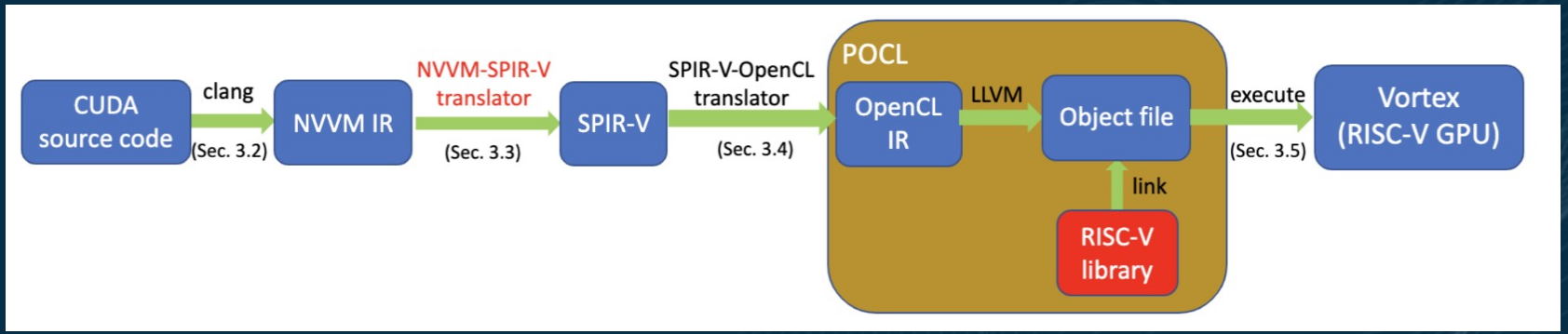
赛昉科技

RISC-V GPGPU

Vortex (open-sourced by GATech)
- RV64IMFD
- Miniaml **SIMT** extension
- Texture extension
- Crypto extension
- Performance Model + RTL



CUDA support

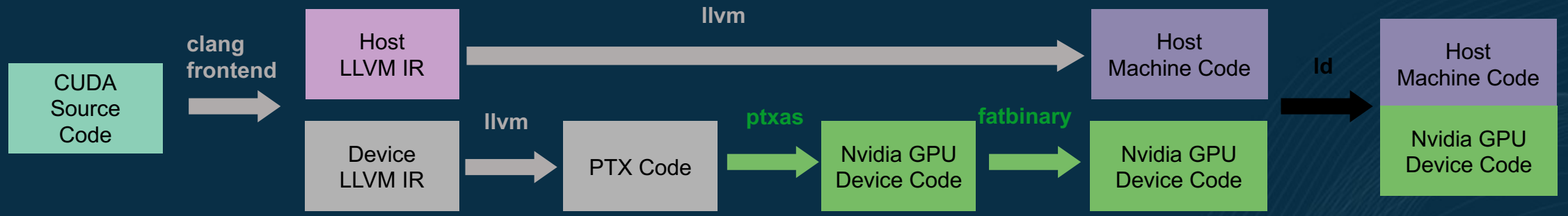


Keep it Simple !

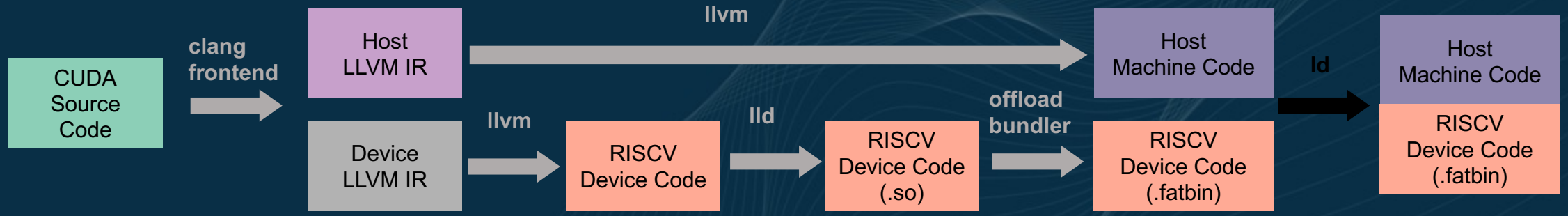
RISC-V GPGPU CUDA Compiler

CUDA clang driver

Compile flow for Nvidia GPU



Compile flow for RISC-V GPGPU



RISC-V GPGPU CUDA Compiler

CUDA Clang driver

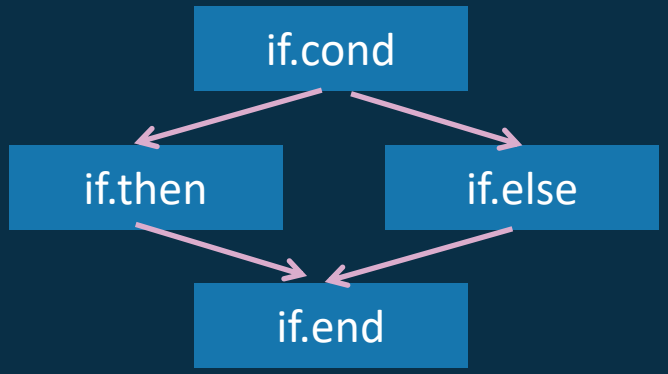
```
             +- 0: input, "test.cu", cuda, (host-cuda)
          +- 1: preprocessor, {0}, cuda-cpp-output, (host-cuda)
       +- 2: compiler, {1}, ir, (host-cuda)
       |              +- 3: input, "test.cu", cuda, (device-cuda, sm_35)
       |           +- 4: preprocessor, {3}, cuda-cpp-output, (device-cuda, sm_35)
       |        +- 5: compiler, {4}, ir, (device-cuda, sm_35)
       |     +- 6: backend, {5}, assembler, (device-cuda, sm_35)
       |  +- 7: assembler, {6}, object, (device-cuda, sm_35)
       |  +- 8: offload, "device-cuda (nvptx64-nvidia-cuda:sm_35)" {7}, object
       |  |- 9: offload, "device-cuda (nvptx64-nvidia-cuda:sm_35)" {6}, assembler
       |- 10: linker, {8, 9}, cuda-fatbin, (device-cuda)
    +- 11: offload, "host-cuda (x86_64-unknown-linux-gnu)" {2}, "device-cuda (nvptx64-nvidia-cuda)" {10}, ir
 +- 12: backend, {11}, assembler, (host-cuda)
+- 13: assembler, {12}, object, (host-cuda)
14: linker, {13}, image, (host-cuda)
```

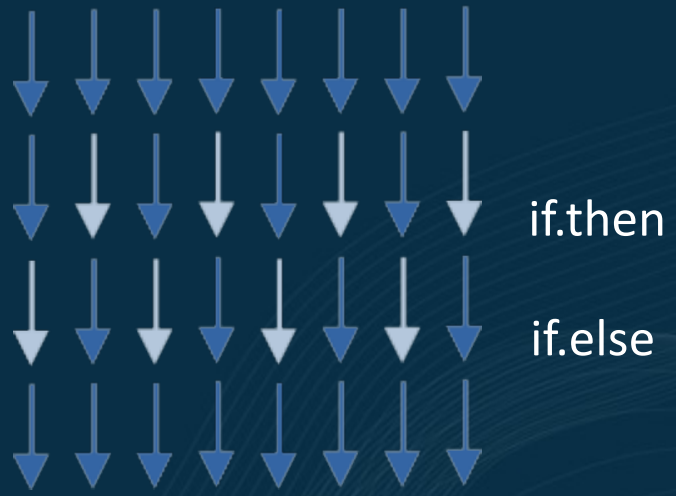
```
             +- 0: input, "test.cu", cuda, (host-cuda)
          +- 1: preprocessor, {0}, cuda-cpp-output, (host-cuda)
       +- 2: compiler, {1}, ir, (host-cuda)
       |              +- 3: input, "test.cu", cuda, (device-cuda, rv64)
       |           +- 4: preprocessor, {3}, cuda-cpp-output, (device-cuda, rv64)
       |        +- 5: compiler, {4}, ir, (device-cuda, rv64)
       |     +- 6: backend, {5}, assembler, (device-cuda, rv64)
       |  +- 7: assembler, {6}, object, (device-cuda, rv64)
       |  +- 8: linker, {7}, image, (device-cuda, rv64)
       |  +- 9: offload, "device-cuda (rv64-starfive-cuda:rv64)" {8}, image
       |- 10: linker, {9}, cuda-fatbin, (device-cuda)
    +- 11: offload, "host-cuda (x86_64-unknown-linux-gnu)" {2}, "device-cuda (rv64-starfive-cuda)" {10}, ir
 +- 12: backend, {11}, assembler, (host-cuda)
+- 13: assembler, {12}, object, (host-cuda)
14: linker, {13}, image, (host-cuda)
```

RISC-V GPGPU CUDA Compiler

CUDA LLVM Codegen

```
if (X[i] < 0) {
        X[i] += Y[i];
} else {
        X[i] -= Y[i];
}
```





Stack-based SIMT

```
if.cond:
        ...
        split %cond
        bgez %cond if.else
        jump if.then
if.then:
        ...
        jump if.end
if.else:
        ...
        jump if.else
if.end:
        join
```

Predicated-based SIMT

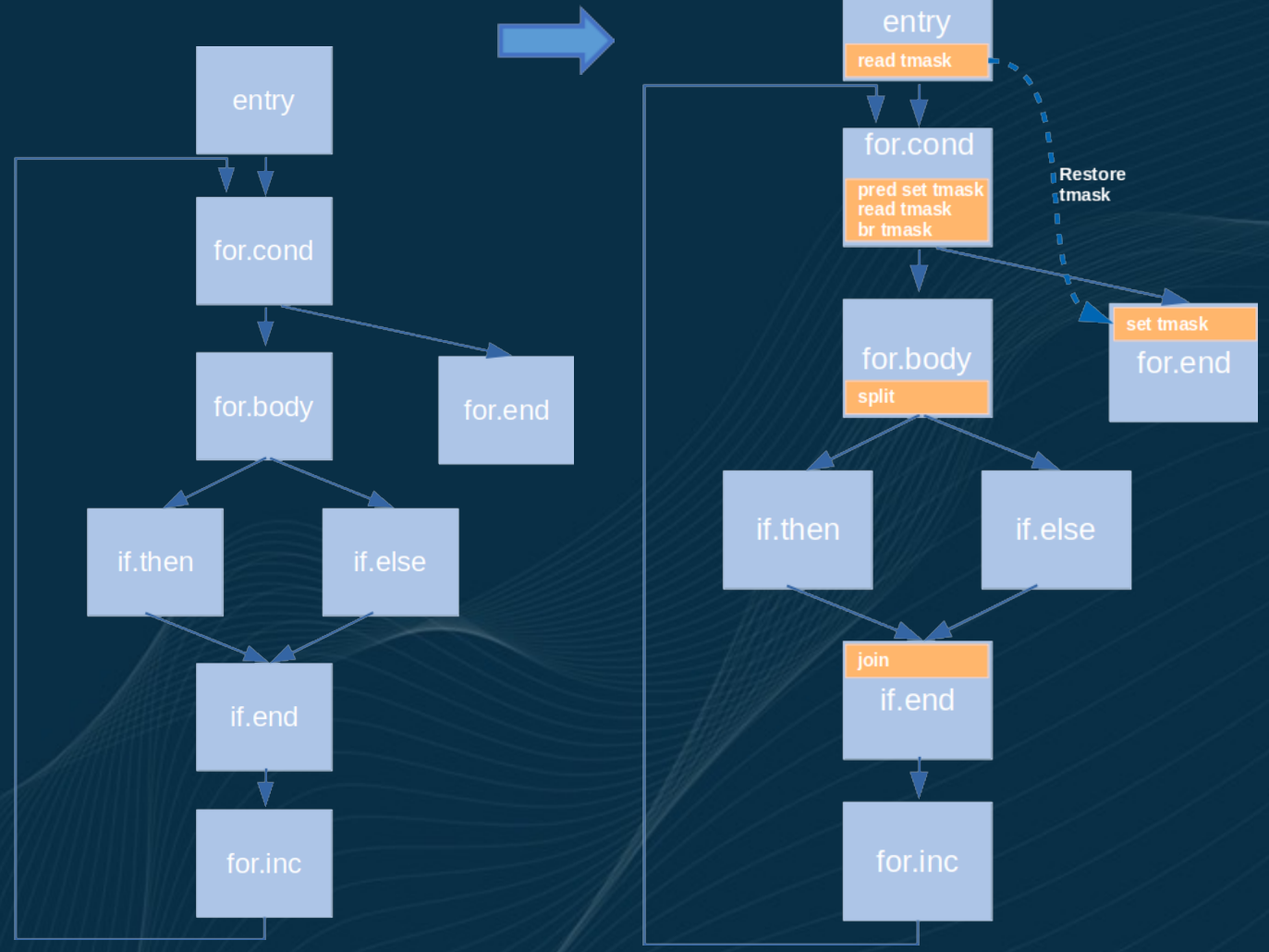
```
if.cond:
        ...
if.then:
        %cond?  ...
if.else:
        !%cond? ...
if.end:
        ...
```

RISC-V GPGPU CUDA Compiler

CUDA LLVM Codegen

```
__global__ void
vectorAddSub(const float *A, const float *B, float *C, int numElements)
{
    int idx = blockDim.x * blockIdx.x + threadIdx.x;

    for (int i = idx * numElements; i < (idx + 1) * numElements; i++) {
        if (i % 2 == 0) {
            C[i] = A[i] + B[i];
        }
        else {
            C[i] = A[i] - B[i];
        }
    }
}
```



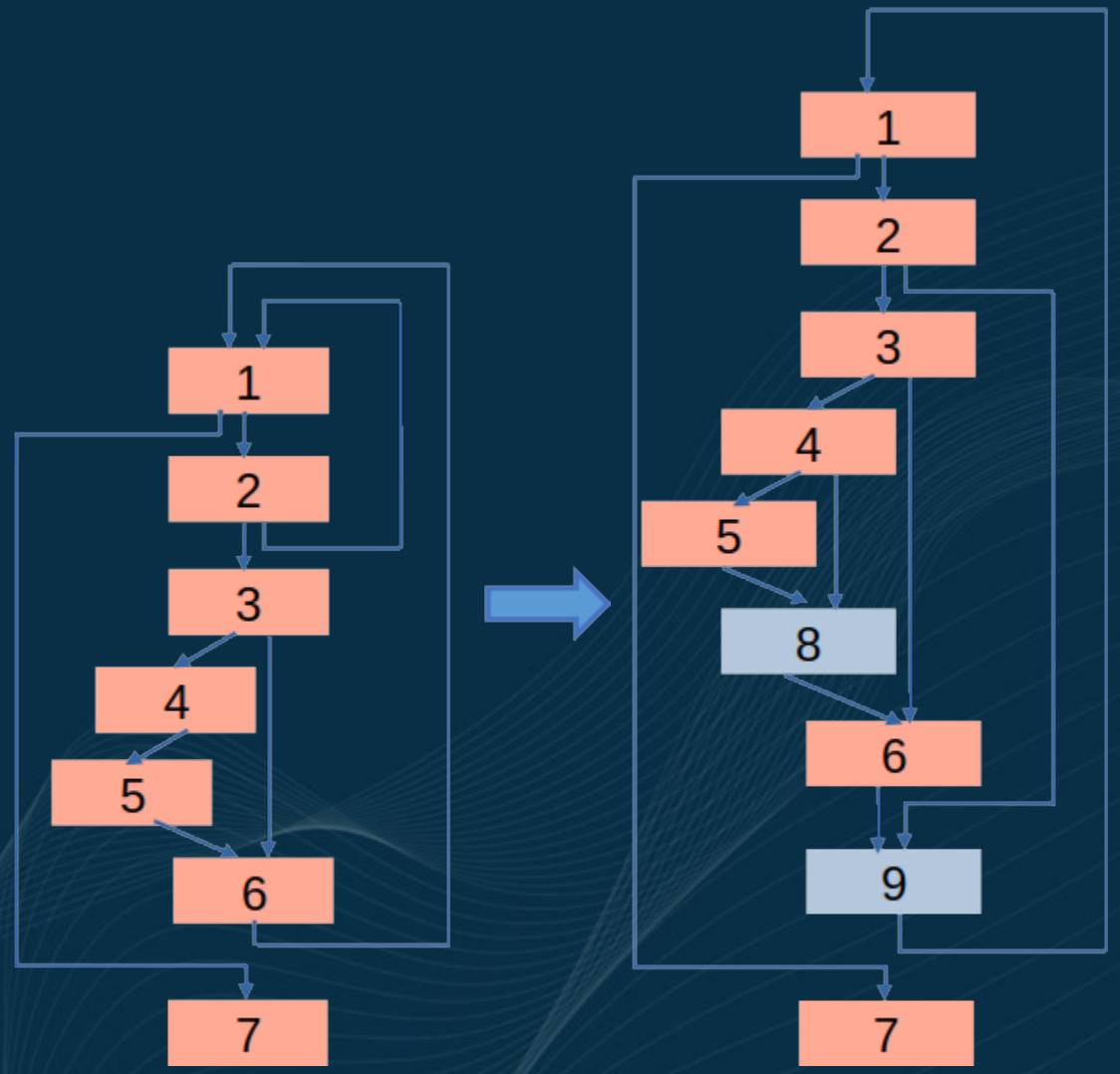
RISC-V GPGPU CUDA Compiler

CUDA LLVM Codegen

Transform and Codegen

- **While** statement
- **For** statement
- **If** statement
- **Switch** statement
- **Select** statement
- [* **without** goto/break/return]

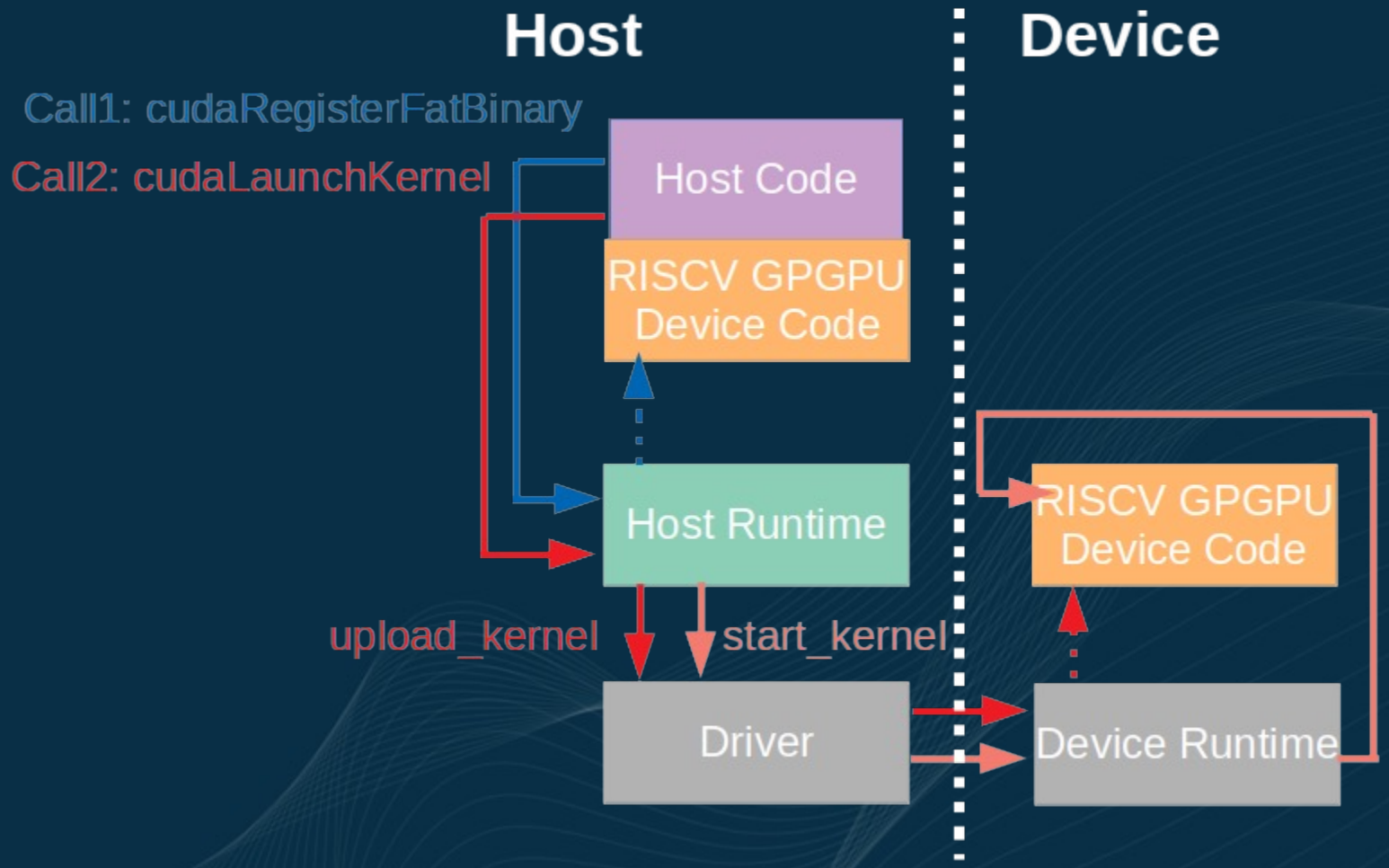
Hard
Future work



RISC-V GPGPU CUDA Runtime

CUDA Runtime

- Host Runtime
  - Device management
  - Memory management
  - Data transfer
  - Execution control
- Device Runtime
  - Command process
  - Kernel execution



Some Demo

VectorAdd

```
[Vector addition of 4096 elements]
[host runtime] Call Trace: {cudaMalloc} function has been called.
[host runtime] Call Trace: {cudaMalloc} function has been called.
[host runtime] Call Trace: {cudaMalloc} function has been called.
Copy input data from the host memory to the CUDA device
[host runtime] Call Trace: {cudaMemcpy} function has been called.
[host runtime] Call Trace: {cudaMemcpy} function has been called.
CUDA kernel launch with 16 blocks of 256 threads
[host runtime] Call Trace: {cudaLaunchKernel} function has been called.
        [driver] start kernel
        [driver] kernel info: kernel_fn_addr = 0x810016e0
        [driver] kernel info: args_addr = 0x9000c000
        [driver] kernel info: grid_x = 16
        [driver] kernel info: grid_y = 1
        [driver] kernel info: grid_z = 1
        [driver] kernel info: block_x = 256
        [driver] kernel info: block_y = 1
        [driver] kernel info: block_z = 1
[core0 warp0 thread0]:          [device runtime] CTA [0 ~ 3]
[core1 warp0 thread0]:          [device runtime] CTA [4 ~ 7]
[core2 warp0 thread0]:          [device runtime] CTA [8 ~ 11]
[core3 warp0 thread0]:          [device runtime] CTA [12 ~ 15]
[core0 warp0 thread0]:          [device runtime] finish execution
[core1 warp0 thread0]:          [device runtime] finish execution
[core2 warp0 thread0]:          [device runtime] finish execution
[core3 warp0 thread0]:          [device runtime] finish execution
Copy output data from the CUDA device to the host memory
[host runtime] Call Trace: {cudaMemcpy} function has been called.
Test PASSED
[host runtime] Call Trace: {cudaFree} function has been called.
[host runtime] Call Trace: {cudaFree} function has been called.
[host runtime] Call Trace: {cudaFree} function has been called.
Done
```

MatrixMul

```
[Matrix Multiply Using CUDA] - Starting...
MatrixA(64,64), MatrixB(64,64)
Computing result using CUDA Kernel...
[host runtime] Call Trace: {cudaLaunchKernel} function has been called.
        [driver] start kernel
        [driver] kernel info: kernel_fn_addr = 0x81001900
        [driver] kernel info: args_addr = 0x9000c000
        [driver] kernel info: grid_x = 2
        [driver] kernel info: grid_y = 2
        [driver] kernel info: grid_z = 1
        [driver] kernel info: block_x = 32
        [driver] kernel info: block_y = 32
        [driver] kernel info: block_z = 1
[core0 warp0 thread0]:          [device runtime] CTA [0 ~ 0]
[core1 warp0 thread0]:          [device runtime] CTA [1 ~ 1]
[core2 warp0 thread0]:          [device runtime] CTA [2 ~ 2]
[core3 warp0 thread0]:          [device runtime] CTA [3 ~ 3]
done
[host runtime] Call Trace: {cudaLaunchKernel} function has been called.
        [driver] start kernel
        [driver] kernel info: kernel_fn_addr = 0x81001900
        [driver] kernel info: args_addr = 0x9000c000
        [driver] kernel info: grid_x = 2
        [driver] kernel info: grid_y = 2
        [driver] kernel info: grid_z = 1
        [driver] kernel info: block_x = 32
        [driver] kernel info: block_y = 32
        [driver] kernel info: block_z = 1
[core0 warp0 thread0]:          [device runtime] CTA [0 ~ 0]
[core1 warp0 thread0]:          [device runtime] CTA [1 ~ 1]
[core2 warp0 thread0]:          [device runtime] CTA [2 ~ 2]
[core3 warp0 thread0]:          [device runtime] CTA [3 ~ 3]
Performance= inf GFlop/s, Time= 0.000 msec, Size= 524288 Ops, WorkgroupSize= 1024 threads/block
Checking computed result for correctness: Result = PASS

NOTE: The CUDA Samples are not meant for performancemeasurements. Results may vary when GPU Boost is enabled.
```

Resnet

Cooperate with TVM CUDA backend, most layer supported.

Future Work

Compiler and Toolchain

More robust compiler, new instruction extension, debugger, profiling tools

Library Development

Runtime library, device library

Application Software

Demo, benchmark, application-related library, application porting guide

微信公众号

Github

开源社区

联系我们

销售：

sales@starfivetech.com

商务：

marketing@starfivetech.com

招聘：

hr.starfivetech@starfivetech.com

技术支持：

support@starfivetech.com

THANK YOU !