

# CANN开发者说

## -TileLang-Ascend是如何诞生的?

解文浩 TileAI 社区

邢静远/杨犇 CANN社区

# 目录

**Part 1 CANN+TileAI**

Part 2 TileLang-Ascend

Part 3 生态合作

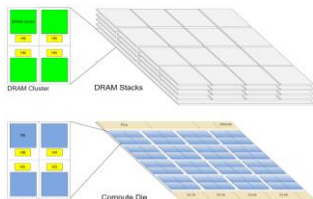
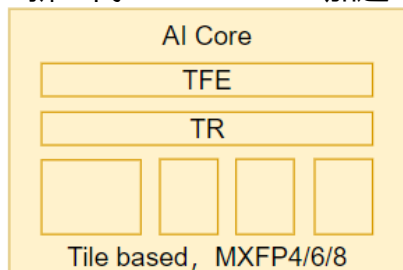
# 趋势与机遇：基于Tile的软硬件协同

AI加速器发展的新趋势使得基于Tile编程语言变得越来越重要。

## Tile-AI加速硬件

- 更高的数据传输和计算效率
- 本地大容量SRAM，再结合3D DRAM，片上互联，提供超高计算带宽
- 要求以更大的Tile粒度来进行指令集设计和体系结构设计

新一代Tile-based AI加速



3D先进封装存储



片上网络

推动



## Tile-编程语言

- 在更大的Tile粒度上进行编程和编译，可以更方便和高效地优化模型算子
- 实现Tile编程计算抽象和Tile硬件抽象一致，高效协同

### Tile Programming

Block-wide cooperative execution on regular Tiles of data

Data & operation granularity is array / tensor



Tile tensor addition  
(array granularity)

# TileLang: Tile编程语言

- Tile-level的类Python的AI编程语言 (DSL)
  - 开发更简单
  - 提供了更好的性能
  - 可以更加细粒度地控制计算调度
- 面对 Tile 硬件趋势的兴起, 推出Tile编程语言TileLang  
主要核心技术团队来自北京大学tileAI项目
- 2025-01-20开源, 现已获得超过3.9K stars  
<https://github.com/tile-ai/tilelang>
- 支持多硬件后端  
GPU、NPU、TPU、CPUs

# TileLang

TILE LANGUAGE

## Tile Language

Tile Language (tile-lang) is a concise domain-specific language designed to streamline the development of high-performance GPU/CPU kernels (e.g., GEMM, Dequant GEMM, FlashAttention, LinearAttention). By employing a Pythonic syntax with an underlying compiler infrastructure on top of [TVM](#), tile-lang allows developers to focus on productivity without sacrificing the low-level optimizations necessary for state-of-the-art performance.



Global Memory

Shared Memory

Register Files



(a) Efficient GEMM with Multi-Level Tiling on GPUs

```
import tilelang.language as T

def matmul(A: T.Buffer, B: T.Buffer, C: T.Buffer):
    with T.Kernel(
        ceildiv(M, block_M), ceildiv(M, block_M), threads=128
    ):
        (bx, by):
            Buffer Allocation
            A_shared = T.alloc_shared(block_M, block_M)
            B_shared = T.alloc_shared(block_M, block_M)
            C_local = T.alloc_fragment(block_M, block_M, device=device)
            T.clear(C_local)

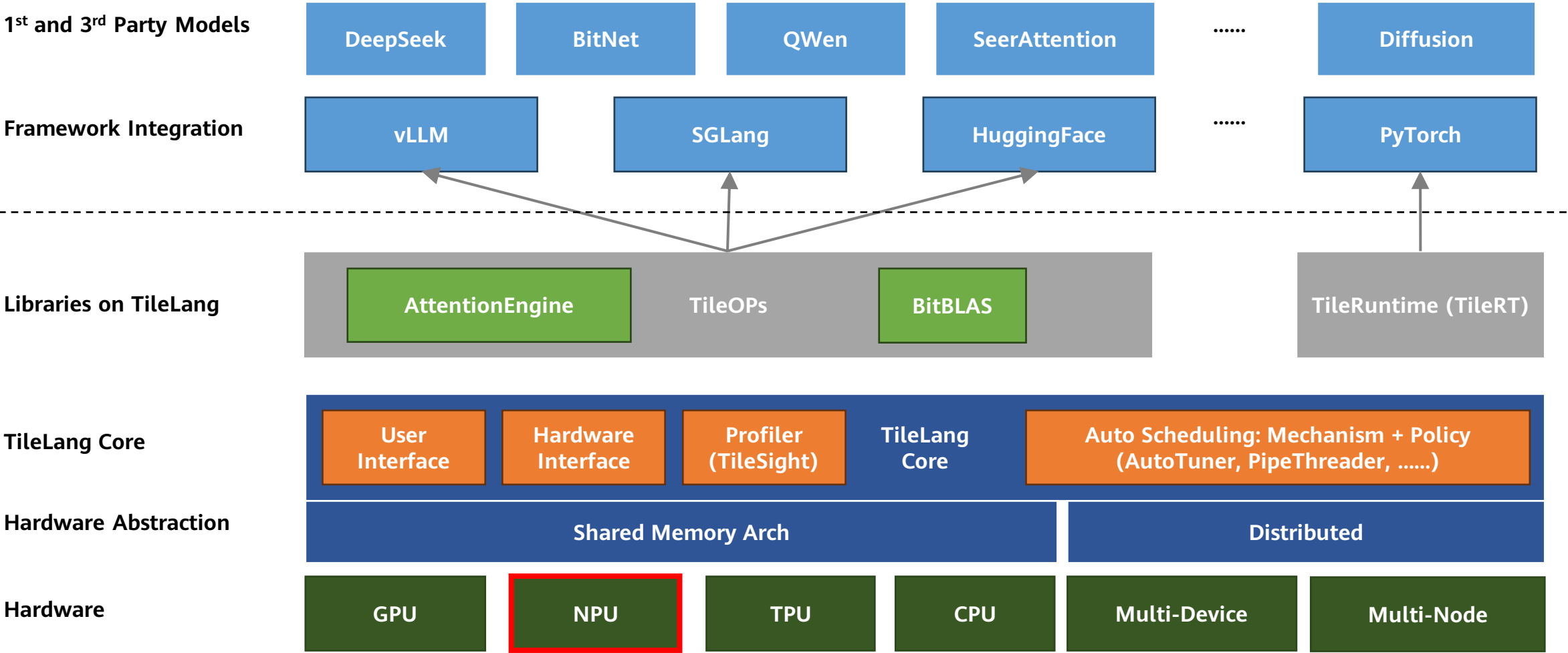
            Main Loop with Pipeline Annotation
            for in T.Pipelined(ceildiv(block_M, block_M), threads=3):
                Copy Data from Global to Shared Memory
                T.copy(A_shared, A, block_M, block_M)
                T.copy(B_shared, B, block_M, block_M)

                GEMM
                T.gemm(A_shared, B_shared, C_local)

            Write Back to Global Memory
            T.copy(C_local, C, block_M, block_M)
```

(b) Describing Tiled GPU GEMM with TileLang

# TileAI系统和生态



# 目录

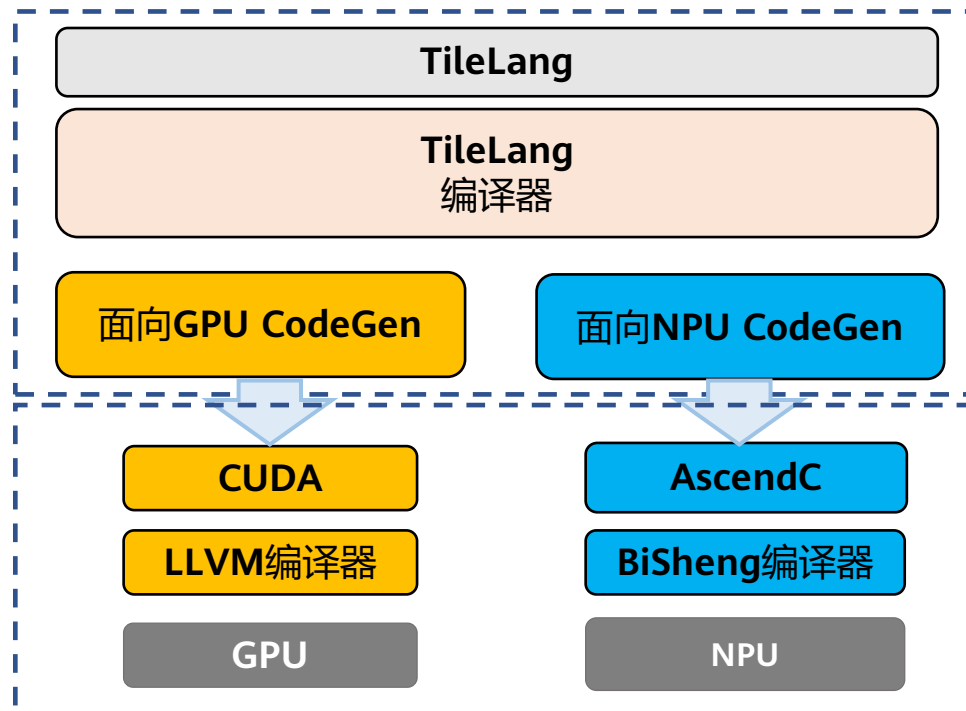
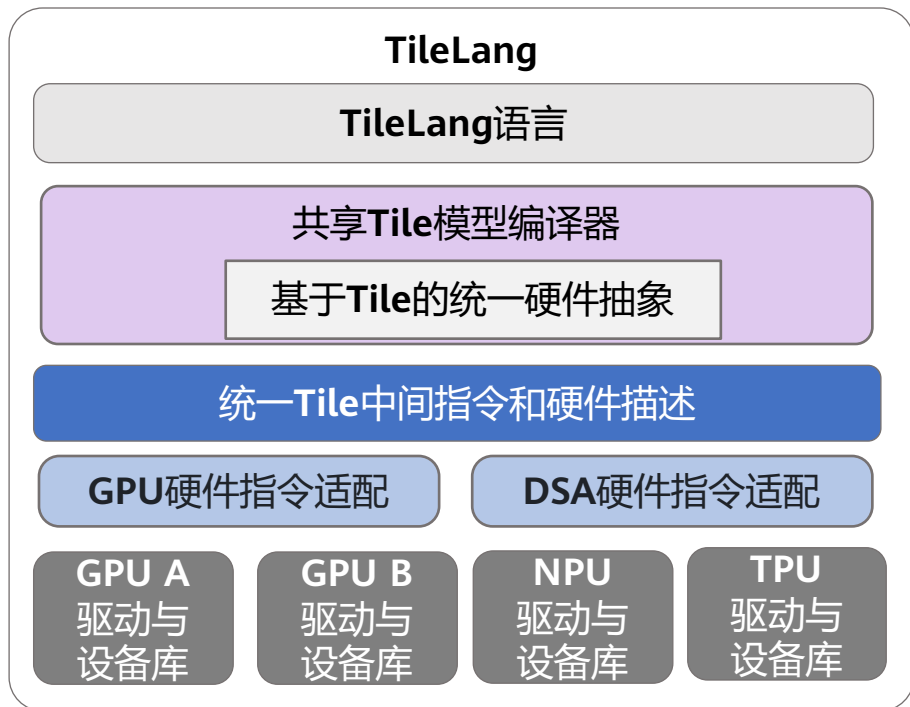
Part 1 TileAI + CANN

**Part 2 TileLang-Ascend**

Part 3 生态合作

# TileLang-Ascend合作——适配NPU

**Ascend适配TileLang：**基于Tile的统一硬件表达的抽象，提供轻量级的底层装换，并提供扩展接口提供Ascend专属能力。



# TileLang-Ascend合作——原语


原语类型	原语名称	用法示例	功能介绍
内核原语	kernel	T.kernel(block_num, is_npu=True) as (cid, vid)	该原语对应到Ascend C的kernel调用处，其中block_num对应于<<< >>>中的第一个参数，表示这个kernel开启多少个子任务数量。cid的范围为 [0,block_num)[0, block\\_num)[0,block_num), vid的范围为0或1。因为A2的cv核默认配比为1:2, 可以通过vid指定当前vector的索引。
内存分配原语	alloc_L1/L0A/L0B/L0C/UB	T.alloc_L1/L0A/L0B/L0C/UB(shape, dtype)	用于分配位于片上内存的buffer；通过指定shape和数据类型标记buffer的信息。
数据搬运原语	copy	T.copy(src, dst)	将src上的buffer拷贝到dst上，注意buffer可以通过BufferLoad或者BufferRegion指定一小块区域。
计算原语	gemm, add, mul, reduce_max...	T.reduce_max(dst, src, tmp, dim)	其中dim指定为对应规约的维度，目前只支持二维的规约。
同步原语	set_flag, wait_flag, set_cross_flag, wait_cross_flag, pipe_barrier	T.set_cross_flag(pipe: str, eventId: int)	其中pipe为需要同步的流水线，eventId为同步事件编号。






# TileLang-Ascend合作——RoadMap

[RoadMap] Development Plan of tilelang-ascend #3

 Open

 xwhzz opened on Sep 28 · edited by xwhzz Edits Contributor ...

### Framework Enhancement

- ☐ Automatic separation of cube and vector code, and elimination of explicit allocation of workspace on global memory
- ☐ Automatic pipeline injection and insertion of necessary synchronization between different hardware resources
- ☐ Automatic buffer reuse
- ☒ Dynamic Shape Support, including automatic TilingData extraction
- ☒ More complete tilelang primitives support
- ☐ PTO instruction support

### Kernel Support




- ☐ Implementation of TileOPs using tilelang-ascend

### Performance Optimization

- ☐ Optimization of instruction templates
- ☐ Complete integration into the existing network infrastructure
- ☐ Target comparable performance of corresponding AscendC programs

### Tool development

- ☐ Profiling Tools for performance analysis

 2  2  2

# 目录

Part 1 TileAI + CANN

Part 2 TileLang-Ascend

**Part 3 生态合作**

## 欢迎大家和我们交流、合作!

群聊: CANN社区交流2群



该二维码7天内(11月21日前)有效, 重新进入将更新

扫码联系项目经理



扫一扫上面的二维码图案, 加我为朋友。

扫码关注项目-github



<https://github.com/tile-ai/tilelang-ascend>

扫码关注项目-gitcode



<https://gitcode.com/cann/cann-recipes-infer/tree/master/ops/tilelang>

<https://gitcode.com/cann>

**CANN**