PR #34015 完整报告

sgl-project/sglang

[diffusion] Sana: bit-exact fused aten LayerNorm+modulate under BCG (H200 denoise -4.8%)

合并时间：2026-08-08 16:05

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/34015

执行摘要

- 一句话：Sana 融合 adaLN 内核，BCG 下去噪提速 4.8%
- 推荐动作：值得精读，尤其是三点：
  1. _sana_ln_modulate 的 " 按执行上下文条件融合 " 设计——把 Python 发射开销与 GPU kernel 时间分开度量，只在回放零成本时启用融合，是 CPU-launch-bound 场景下很实用的取舍；
  2. 位精验证 + 白名单 + 永久禁用三层兜底，让加速路径在无法保证数值一致的环境里自动让位，可作为高风险优化的默认策略；
  3. tail chunk 掩码对 aten 串行 Welford 顺序的复刻细节，连同 #34008 一起构成现成的 " 任意 hidden 都能位精融合 LN+modulate" 的基础设施。 建议后续 FLUX.1（#34004）接线时复用本 PR 的 raw 变体与验证框架。

功能与动机

Sana 的 transformer 在每次 DiT forward 中于 41 处执行 *eager* adaLN 链 LN(x) * (1 + scale) + shift（20 个 block 的 norm1/norm2 各 40 处 + 末尾 SanaModulatedNorm 1 处），每处是 4 个 aten 内核，且因为 DiT 传播 patch-embed/GLUMBConv 的 permute 布局，LayerNorm 每次调用还要付一次隐藏的 layout copy。元素运算约占 Sana 单步 GPU 时间的 51%，而现有融合都不适用：CuTe-DSL 的 norm+scale+shift 系列要求 hidden % 256 == 0，Sana 的 hidden 是 2240（70 头 x 32）不满足。#34008 的 Triton 内核能位精复刻 aten 的 vectorized_layer_norm_kernel 数值，但假设 hidden % 512 == 0。本 PR 的目标是把该内核泛化并接入 Sana，且不引入质量门控或数值差异。

实现拆解

1. 内核泛化：支持任意 hidden % 4 == 0（python/sglang/kernels/ops/diffusion/triton/layernorm_modulate.py） - _layernorm_modulate_kernel 的 pass 1 串行 Welford 循环由 D // 512 改为 (D + 511) // 512：最后一个不完整块按 i * 128 + lanes < D // 4 生成 vec_valid 掩码，向量索引到达 N/4 的线程跳过迭代、以更小的 Welford 计数进入 warp fold，从而在元素顺序上原样复刻 aten 的归约序列；pass 2 的加载与存储同样补上 cols < D 掩码。 - can_use_fused_layernorm_modulate 的约束从 hidden % 512 == 0 放宽为 hidden % 4 == 0（即 aten 自身的向量化要求）；hidden % 512 == 0 的路径编译结果与之前完全一致，保证 GLM-Image 回归测试保持绿色。

2. 新增直接调用变体与 warp 启发式调整：新增 fused_layernorm_modulate_raw，去掉 torch.ops custom op 分发（每次调用约 40 us CPU 开销），注册入口改为 fused_layernorm_modulate = register_custom_op(fused_layernorm_modulate_raw, ...)，torch.compile 场景仍走注册 op；num_warps 阈值从 hidden >= 4096 降到 hidden >= 2048（H200 上 Sana (2, 1024, 2240) 由 43.1 us 降至 14.3 us）。
3. Sana按执行上下文条件接线（python/sglang/multimodal_gen/runtime/models/dits/sana.py）：
   - 新增 _eager_ln_modulate 保留原 norm(x) * (1 + scale) + shift eager 链；_sana_ln_modulate 作为条件分发入口，替换 SanaModulatedNorm.forward 与 SanaTransformerBlock.forward 的 norm1/norm2 共 3 处调用点。
   - 融合仅在 CUDA-graph 上下文（正在 stream capture 或非默认流）启用；默认流 eager 服务刻意保持 eager 链，因为 Sana eager 属 CPU 发射瓶颈，一个 Triton launch 的 Python 开销比整个 5 层 aten 链还大（实测 +14% forward wall）。
   - 每个新输入签名 (shape, stride, dtype, scale.shape/stride, shift.stride, norm.eps) 在 BCG warmup 的非捕获阶段与 eager 链做 torch.equal 逐位校验，通过后加入 _SANA_FUSED_LN_MOD_OK_SIGS 白名单；任何异常或位不一致都会置 _SANA_FUSED_LN_MOD_DISABLED 永久回退 eager；捕获期间遇到未验证签名直接走 eager（无法同步验证）。融合路径显式执行 x.contiguous()，复刻 aten LayerNorm 对 Sana transposed 激活的内部 copy。
4. 测试配套：新增 test/registered/kernels/ops/diffusion/test_sana_ln_modulate.py，4 组参数覆盖真实 (., 1024, 2240) 形状（含 transposed 布局、nmod=6/2 两种 adaLN 切分），断言默认流不触发融合、非默认流触发验证且位精一致；既有 test_glm_image_ln_modulate.py 保持绿色未改动。

关键文件：

- python/sglang/multimodal_gen/runtime/models/dits/sana.py（模块 Sana 模型；类别 source；类型 data-contract；符号 _eager_ln_modulate, _sana_ln_modulate, _SANA_FUSED_LN_MOD_OK_SIGS, _SANA_FUSED_LN_MOD_DISABLED）：Sana 模型接线主体：新增条件融合分发 _sana_ln_modulate，替换 norm1/norm2 与 SanaModulatedNorm 共 3 处 adaLN 站点，并内置签名白名单与永久禁用兜底。
- python/sglang/kernels/ops/diffusion/triton/layernorm_modulate.py（模块 融合内核；类别 infra；类型 infrastructure；符号 fused_layernorm_modulate, fused_layernorm_modulate_raw, _layernorm_modulate_kernel, can_use_fused_layernorm_modulate）：融合内核本体：将支持范围从 hidden % 512 == 0 泛化到 hidden % 4 == 0，新增 raw 直接调用变体并调整 num_warps 启发式，是位精加速得以成立的核心。
- test/registered/kernels/ops/diffusion/test_sana_ln_modulate.py（模块 单元测试；类别 test；类型 test-coverage；符号 test_sana_fused_ln_modulate_is_bit_exact）：新增单元测试，覆盖真实 serving 形状（含 transposed 布局），断言默认流不触发融合、非默认流触发验证且位精一致，是融合启停策略的直接守护。

关键符号：_sana_ln_modulate, _eager_ln_modulate, fused_layernorm_modulate_raw, fused_layernorm_modulate, _layernorm_modulate_kernel,

can_use_fused_layernorm_modulate, test_sana_fused_ln_modulate_is_bit_exact

关键源码片段

python/sglang/multimodal_gen/runtime/models/dits/sana.py

Sana 模型接线主体：新增条件融合分发 _sana_ln_modulate，替换 norm1/norm2 与 SanaModulatedNorm 共 3 处 adaLN 站点，并内置签名白名单与永久禁用兜底。

```
# python/sglang/multimodal_gen/runtime/models/dits/sana.py
# 全局开关与白名单：一旦发现当前平台位精失败就永久回退 eager
_SANA_FUSED_LN_MOD_DISABLED = False
_SANA_FUSED_LN_MOD_OK_SIGS: set = set()


def _sana_ln_modulate(
    norm: nn.LayerNorm,
    x: torch.Tensor,
    scale: torch.Tensor,
    shift: torch.Tensor,
) -> torch.Tensor:
    """单内核完成 ``LN(x) * (1 + scale) + shift``，与 eager 逐位一致。

    scale / shift 是 Sana 的 ``(batch, 1, dim)`` adaLN 行。每个新输入
    签名先用 ``torch.equal`` 与 eager 链对比（位精确性取决于 aten 实际
    分派的 LayerNorm 内核）；任何不一致都会永久禁用快速路径。
    """
    global _SANA_FUSED_LN_MOD_DISABLED

    # 已被禁用、torch.compile 场景或非 CUDA 输入直接走 eager
    if _SANA_FUSED_LN_MOD_DISABLED or torch.compiler.is_compiling() or not x.is_cuda:
        return _eager_ln_modulate(norm, x, scale, shift)

    # 融合只在 CUDA-graph 上下文启用：Sana 默认流 eager 是 CPU 发射瓶颈，
    # 一个 Triton launch 的 Python 开销比整个 5 层 aten 链还大（+14%），
    # 而 BCG 回放时 Python 开销为零，GPU 收益才能全部兑现
    capturing = torch.cuda.is_current_stream_capturing()
    if not capturing and torch.cuda.current_stream() == torch.cuda.default_stream():
        return _eager_ln_modulate(norm, x, scale, shift)

    # 用 shape/stride/dtype/eps 组成签名，命中白名单则直接融合
    sig = (
        x.shape,
        x.stride(),
        x.dtype,
        scale.shape,
        scale.stride(),
        shift.stride(),
        norm.eps,
    )
```

```
    if sig in _SANA_FUSED_LN_MOD_OK_SIGS:
        return fused_layernorm_modulate_raw(
            x.contiguous(), scale[:, 0], shift[:, 0], norm.eps
        )
    if capturing:
        # 捕获期间无法同步验证，未验证签名回退 eager，避免录错图
        return _eager_ln_modulate(norm, x, scale, shift)

    # 非捕获路径（BCG warmup 阶段）：逐位校验通过后把签名记入白名单
    if (
        x.dtype is torch.bfloat16
        and x.dim() == 3
        and scale.dim() == 3
        and scale.shape[1] == 1
        and shift.shape == scale.shape
        and is_plain_layer_norm(norm, x.shape[-1])
    ):
        x_c = x.contiguous()
        if not can_use_fused_layernorm_modulate(x_c, scale[:, 0], shift[:, 0]):
            return _eager_ln_modulate(norm, x, scale, shift)
        try:
            out = fused_layernorm_modulate_raw(
                x_c, scale[:, 0], shift[:, 0], norm.eps
            )
        except Exception as exc:
            logger.warning_once(f"Disabling Sana fused LN+modulate fast path: {exc}")
            _SANA_FUSED_LN_MOD_DISABLED = True
        else:
            ref = _eager_ln_modulate(norm, x, scale, shift)
            if torch.equal(out, ref):
                _SANA_FUSED_LN_MOD_OK_SIGS.add(sig)
                return out
            # 位精失败：该平台 aten dispatch 与内核复刻不一致，永久禁用
            logger.warning_once(
                "Sana fused LN+modulate fast path is not bit-exact against "
                "this platform's LayerNorm dispatch; falling back to eager"
            )
            _SANA_FUSED_LN_MOD_DISABLED = True
            return ref

    return _eager_ln_modulate(norm, x, scale, shift)
```

python/sglang/kernels/ops/diffusion/triton/layernorm_modulate.py

融合内核本体：将支持范围从 hidden % 512 == 0 泛化到 hidden % 4 == 0，新增 raw 直接调用变体并调整 num_warps 启发式，是位精加速得以成立的核心。

```
# python/sglang/kernels/ops/diffusion/triton/layernorm_modulate.py
# pass 1：按 aten 的元素顺序做 per-thread 串行 Welford。
# 原实现只支持 hidden % 512 == 0，且 128 个线程跑相同向量数；
```

```
# 泛化后循环上限改为 `(D + 511) // 512`，末块用掩码对齐 aten 的归约顺序。
for i in tl.static_range((D + 511) // 512):
    cols = i * 512 + lanes[:, None] * 4 + tl.arange(0, 4)[None, :]
    if (i + 1) * 512 <= D:
        # 完整块：所有 128 个线程都参与，与原有快速路径编译结果一致
        x4 = tl.load(
            x_ptr + row_base[:, None, None] + cols[None, :, :],
            mask=row_mask[:, None, None],
            other=0.0,
        ).to(tl.float32)
        mean, m2, cnt = _push_vec4(x4, mean, m2, cnt, row_mask, ROWS, 128, MASKED=False)
    else:
        # 部分尾块：向量索引 i*128 + t 达到 N/4 的线程跳过本迭代，
        # 以更小的 Welford 计数进入 warp fold，复现 aten 串行顺序
        vec_valid = (i * 128 + lanes < D // 4)[None, :]
        x4 = tl.load(
            x_ptr + row_base[:, None, None] + cols[None, :, :],
            mask=row_mask[:, None, None] & vec_valid[:, :, None],
            other=0.0,
        ).to(tl.float32)
        mean, m2, cnt = _push_vec4(x4, mean, m2, cnt, vec_valid, ROWS, 128, MASKED=True)

# pass 2：归一化 + 调制，末块补 cols < D 掩码，避免越界读写
for i in tl.static_range((D + 511) // 512):
    cols = i * 512 + tl.arange(0, 512)
    mask = row_mask[:, None]
    if (i + 1) * 512 > D:
        mask = mask & (cols < D)[None, :]
    x = tl.load(
        x_ptr + row_base[:, None] + cols[None, :], mask=mask, other=0.0
    ).to(tl.float32)
    y = _round_bf16_to_fp32(rstd * (x - mean))
    sc = tl.load(
        scale_ptr + batch[:, None] * scale_row_stride + cols[None, :],
        mask=mask, other=0.0,
    ).to(tl.float32)
    sh = tl.load(
        shift_ptr + batch[:, None] * scale_row_stride + cols[None, :],
        mask=mask, other=0.0,
    ).to(tl.float32)
    one_plus = _round_bf16_to_fp32(1.0 + sc)
    y = _round_bf16_to_fp32(y * one_plus) + sh
    tl.store(y_ptr + row_base[:, None] + cols[None, :], y, mask=mask)
```

test/registered/kernels/ops/diffusion/test_sana_ln_modulate.py

新增单元测试，覆盖真实 serving 形状（含 transposed 布局），断言默认流不触发融合、非默认流触发验证且位精一致，是融合启停策略的直接守护。

```
# test/registered/kernels/ops/diffusion/test_sana_ln_modulate.py
```

```
@pytest.mark.parametrize(
    "shape,nmod,transposed",
    [
        ((2, 1024, 2240), 6, False), # 真实 Sana 1024px 形状，hidden 2240 覆盖 partial tail chunk 分支
        ((2, 1024, 2240), 2, False), # nmod=2 对应 SanaModulatedNorm 的切分方式
        ((1, 1024, 2240), 6, True), # transposed 模拟 patch-embed / GLUMBConv 的 permute 布局
        ((1, 37, 2240), 6, False),
    ],
)
def test_sana_fused_ln_modulate_is_bit_exact(shape, nmod, transposed):
    torch.manual_seed(0)
    batch, seq, hidden = shape
    norm = torch.nn.LayerNorm(hidden, eps=1e-6, elementwise_affine=False).cuda()
    x = (torch.randn(batch, seq, hidden, device="cuda") * 4).bfloat16()
    if transposed:
        x = x.permute(0, 2, 1).contiguous().permute(0, 2, 1)
    emb = torch.randn(batch, nmod, hidden, device="cuda").bfloat16()
    shift, scale = emb.chunk(nmod, dim=1)[0], emb.chunk(nmod, dim=1)[-1]

    # 默认流 eager 服务必须保持原样，不得触发融合验证
    n_sigs = len(sana._SANA_FUSED_LN_MOD_OK_SIGS)
    _sana_ln_modulate(norm, x, scale, shift)
    assert len(sana._SANA_FUSED_LN_MOD_OK_SIGS) == n_sigs

    # 非默认流（BCG warmup/capture 路径）应触发融合并完成逐位校验
    with torch.cuda.stream(torch.cuda.Stream()):
        out = _sana_ln_modulate(norm, x, scale, shift)
        assert len(sana._SANA_FUSED_LN_MOD_OK_SIGS) == n_sigs + 1 # 已验证
        out2 = _sana_ln_modulate(norm, x, scale, shift) # 命中白名单的快速通道
    torch.cuda.synchronize()
    assert torch.equal(out, _eager_ln_modulate(norm, x, scale, shift))
    assert torch.equal(out2, out) and not sana._SANA_FUSED_LN_MOD_DISABLED
```

评论区精华

该 PR 没有任何 reviewer 评论（review_comments_count 为 0，唯一一条 issue 评论是作者 BBuf 贴出的 CI 运行链接），由作者自审自合并（merged_by 亦为 BBuf），因此没有可提炼的讨论交锋。核心设计论证全部沉淀在 PR body 中，可视为作者的自我审查记录，关键决策包括：

- 为什么 eager 模式不融合：

Sana's eager mode is CPU-launch bound, and one Triton launch costs more Python time than the whole 5-deep aten enqueue pipeline it replaces — an in-process alternating A/B on the real model measured +14% forward wall when fusing eagerly, despite -4.8% GPU kernel time.

- 为什么可以不带质量门控：内核逐位复刻 aten 的 vectorized_layer_norm_kernel（串行 Welford + guarded-rcp 快速路径 + shfl.down 归并树 + MUFU.RSQ + 逐 op bf16 舍入），因此融合 needs no quality gate。
- 失败兜底策略：

any mismatch permanently falls back to eager (#34008 recipe)。

- FLUX.1 的接线前置条件：body 明确提示 Sana 的教训——Triton launch 的 Python 开销可能超过被替换的 aten 链，FLUX eager 服务必须先做同样的 CPU-launch-bound 检查再接线。
  - 暂无高价值评论线程

风险与影响

- 风险：
  - 位精依赖平台 aten dispatch：torch.equal 校验结果依赖 aten 在具体平台 / 版本上分派的 LayerNorm 内核。若某平台数值与 Triton 复刻不一致，_sana_ln_modulate 会永久禁用并回退 eager，正确性有兜底，但该平台拿不到性能收益。
  - 模块级全局状态：_SANA_FUSED_LN_MOD_DISABLED 与 _SANA_FUSED_LN_MOD_OK_SIGS 是 sana.py 的模块级全局。若同一进程加载多个 Sana 实例或未来其他模型复用同一模块，全局开关会互相影响；当前 Sana 是唯一使用方，风险可控但需留意。
  - 捕获期未验证签名走 eager：BCG capture 过程中若遇到白名单外的新签名，会回退 eager 并录制进 CUDA graph，融合收益在该签名上永久丢失（不会出错）。warmup 阶段未覆盖的输入来源会成为优化盲区。
  - 同步验证的开销与时机：非默认流且非捕获的未验证签名会触发 torch.equal 对比（含隐式同步）。该路径设计上只在 BCG warmup 迭代出现，但若外部在非默认流上做常驻 eager 推理，会引入同步开销。
  - H200 调优启发式：num_warps = 4 if hidden >= 2048 else 2 是在 H200 上调的；其他 GPU（A100、B200 等）未必最优，属于性能风险而非正确性风险。
  - 覆盖范围：测试仅覆盖 CUDA + bf16，非 CUDA 场景直接走 eager（x.is_cuda 检查），无正确性回归面，但融合路径缺少多平台验证。
- 影响：
  - 用户侧：Sana 用户在 --enable-breakable-cuda-graph（BCG）配置下，denoise 从 405.2 ms 降至 385.9 ms（-4.8%），e2e 约 -2.6%；默认 eager 配置 trace 级不变（launch 序列与 main 一致）。
  - 系统侧：BCG 配置下每 5-step 窗口的 492 次 aten LN + 4 核链接入点变为 492 次融合内核，GPU kernel 启动数减少 13.5%（10932 -> 9456）。
  - 团队侧：确立了 " 仅在 CUDA-graph 上下文启用融合 + 位精验证白名单 + 永久禁用兜底 " 的扩散模型优化模式，后续 FLUX.1（#34004）、其他 DiT 可复制；内核库能力从 hidden % 512 == 0 扩到 hidden % 4 == 0，潜在受益模型面扩大。

- 风险标记：位精依赖平台 aten dispatch，模块级全局禁用标志，H200 调优启发式，捕获期未验证签名走 eager

关联脉络

- PR #34008 [diffusion] GLM-Image bit-exact fused aten LayerNorm+modulate / qk-LN (H200 30-step denoise -8.1%): 本 PR 直接依赖该 PR：内核文件与 GLM-Image 接线来自 #34008，本 PR 只新增 tail-chunk 泛化、raw 变体与 Sana 接线，且 GLM-Image 回归测试保持绿色。
- PR #34004 [diffusion] FLUX.1 fused adaLN modulate (bit-exact) + RoPE cache hoist, LN-affine folding behind quality=high (H200 e2e -3.5% lossless / -6.9% high): PR body 作为 follow-up 实测：泛化内核在 FLUX.1 的 hidden=3072 上同样位精且 23.8 us，严格优于 quality=high 折叠路径的 27.2 us，为后续 FLUX 接线铺路。