# PR #32046 完整报告

sgl-project/sglang

[AMD]Qwen3.5 integration gfx950 fmha fp8 hd256

合并时间：2026-08-03 15:25

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/32046

## 执行摘要

- 一句话：Qwen3.5 在 gfx950 新增 opt-in fp8 hd256 prefill 快速路径
- 推荐动作：建议精读。这是 " 在共享注意力后端安全接入硬件专属 kernel 快速路径 " 的典型范例：严格的正确性门控、opt-in 环境变量、默认路径字节级不变、以及用大规模 A/B 数据支撑内核选择决策。值得关注的设计点： 1) 所有正确性风险条件（sinks/softcap/ 滑动窗口 / 缓存前缀）都显式出现在门控里而非依赖调用方约定； 2) 回退内核的取舍基于同容器同版本实测而非直觉； 3) 未附带测试导致的 CI 盲区，应作为反面教材与 #33399 一起跟踪。

## 功能与动机

PR body 明确说明动机：Qwen3.5 在 gfx950 (MI350x) 上使用 attention head_dim 256，无法命中 aiter 快速 attention 路径，prefill 回退到较慢的 mha_batch_prefill；ROCm/aiter PR #3732（flash_attn_varlen_fp8_pertensor_func）新增了 fp8 head_dim-256 FMHA asm 内核，可加速该形状的 compute-bound prefill attention。本 PR 将该内核作为 opt-in 快速路径接入，针对性提升长上下文 prefill 的吞吐与 TTFT，重点覆盖 MoE 变体 amd/Qwen3.5-397B-A17B-MoE-MXFP4。作者还解释了 chunked-prefill 的约束：快速路径只对无前缀 chunk 生效，因此长 prompt 需要 --chunked-prefill-size -1 或 65536 与 --max-prefill-tokens 65536 保持单块无前缀。

## 实现拆解

1. 变更入口与符号：在 python/sglang/srt/layers/attention/aiter_backend.py 的 aiter 导入块中新增 flash_attn_varlen_fp8_pertensor_func 导入，这是 fp8 per-tensor varlen FMHA 内核的唯一新增外部依赖；该符号只存在于 aiter main 分支构建（含 ROCm/aiter PR #3732）。
2. 快速路径门控：在 AiterAttnBackend.forward_extend 中、window_size 确定之后、原有 NHD paged 路径之前插入一个复合条件分支。条件包括环境变量 SGLANG_AITER_FMHA_FP8_ASM=1、is_gfx95_supported()、forward_mode.is_extend()、extend_prefix_lens_cpu 非空且全零（无缓存前缀纯 prefill）、window_size == (-1, -1)、sinks is None、logits_soft_cap == 0.0、layer.qk_head_dim 与 v_head_dim 均为 256、kv_cache_dtype == fp8_dtype。这些门控与姊妹 PR #28482 保持一致，并吸收了 review 中关于 target_verify 模式与 sink/softcap 安全性的意见。

3. fp8 量化与内核调用：命中快速路径时，Q/K/V 先 contiguous 再按 (num_tokens, num_heads, head_dim) 重排，然后 .to(fp8_dtype)；反量化使用 per-tensor 尺度——q 的 scale 取 layer.k_scale 或 self.k_scale，k/v 沿用现有变量 k_descale/v_descale（对应 review 中 one_scale 修正）；调用 flash_attn_varlen_fp8_pertensor_func，cu_seqlens 使用 self.qo_indptr[:bs0]，max_q_len 使用 forward_metadata.max_q_len，softmax_scale=layer.scaling、causal=True；输出若 dtype 与 input_dtype 不一致则转回，最后按 (-1, tp_q_head_num * head_dim) 返回。
4. 回退路径不变：任何条件不满足（例如 chunked prefill 续块带缓存前缀、bf16 KV、非 gfx950、head_dim 非 256）都会原样落入 vectorized-5d 或原有 mha_batch_prefill_func 路径，保证默认行为与未打补丁版本一致。
5. 测试与配置配套：本 PR 没有新增任何单元测试或 CI 用例（合计 +38/-0，仅 1 个文件）；正确性依赖作者在 MI350x 上手动执行的 gsm8k 评测（0.977-0.98，与基线噪声相当），性能依托 Databricks 上的大规模 A/B 基准。部署需要 aiter main 分支，且长文本场景要求大 chunked-prefill 配置。合并前 CI 状态（amd-bot 评论）确认新代码路径在 PR CI 中一次都没有执行。后续 #33399 将移除全局变量、修复 shape 问题并在 fp8 KV cache 下默认启用。

关键文件：

- python/sglang/srt/layers/attention/aiter_backend.py（模块 注意力后端；类别 source；类型 core-logic；符号 AiterAttnBackend.forward_extend, flash_attn_varlen_fp8_pertensor_func）：唯一变更文件：在 AiterAttnBackend.forward_extend 中新增 fp8 hd256 prefill 快速路径，涵盖 aiter 导入、复合门控、fp8 量化与内核调用、dtype 回退，是全部功能与风险的载体。

关键符号：AiterAttnBackend.forward_extend, flash_attn_varlen_fp8_pertensor_func

## 关键源码片段

### python/sglang/srt/layers/attention/aiter_backend.py

唯一变更文件：在 AiterAttnBackend.forward_extend 中新增 fp8 hd256 prefill 快速路径，涵盖 aiter 导入、复合门控、fp8 量化与内核调用、dtype 回退，是全部功能与风险的载体。

```
# python/sglang/srt/layers/attention/aiter_backend.py（合并后结构，已整理）
# 新增的 aiter 导入（原有 try 导入块内追加一行）：
# from aiter import flash_attn_varlen_fp8_pertensor_func
# 该符号来自 ROCm/aiter PR #3732，只在 aiter main 分支构建中存在。

# AiterAttnBackend.forward_extend：window_size 计算之后、NHD paged 路径之前
if (
    get_bool_env_var("SGLANG_AITER_FMHA_FP8_ASM", "False") # 显式 opt-in，默认关闭
    and is_gfx95_supported() # 仅 gfx950 (MI350x)
    and forward_batch.forward_mode.is_extend() # 仅 extend 模式
    and forward_batch.extend_prefix_lens_cpu is not None
    and not any(forward_batch.extend_prefix_lens_cpu) # 无缓存前缀：纯 prefill
    and window_size == (-1, -1) # 内核不支持滑动窗口
    and sinks is None # 内核不支持 attention sinks
```

```
        and self.logits_soft_cap == 0.0 # 内核不支持 softcap
        and layer.qk_head_dim == 256
        and layer.v_head_dim == 256 # Qwen3.5 的 head_dim
        and self.kv_cache_dtype == fp8_dtype # 仅 fp8 KV cache 下启用
    ):
        # Q/K/V 先转连续并重排为 (num_tokens, num_heads, head_dim),
        # 再量化到 fp8；per-tensor 反量化尺度分别来自 layer 或后端缓存。
        q_c = q.contiguous().view(-1, layer.tp_q_head_num, layer.head_dim)
        k_c = k.contiguous().view(-1, layer.tp_k_head_num, layer.head_dim)
        v_c = v.contiguous().view(-1, layer.tp_v_head_num, layer.v_head_dim)
        fp8_q_descale = (
            layer.k_scale if layer.k_scale is not None else self.k_scale
        )
        o = flash_attn_varlen_fp8_pertensor_func(
            q_c.to(fp8_dtype),
            k_c.to(fp8_dtype),
            v_c.to(fp8_dtype),
            fp8_q_descale,
            k_descale,
            v_descale,
            self.qo_indptr[:bs0], # cu_seqlens_q
            self.qo_indptr[:bs0], # cu_seqlens_k
            self.forward_metadata.max_q_len,
            self.forward_metadata.max_q_len,
            softmax_scale=layer.scaling,
            causal=True,
        )
        # 内核输出为 fp8 时转回模型输入 dtype，保证下游接口一致。
        if o.dtype != self.input_dtype:
            o = o.to(self.input_dtype)
        return o.view(-1, layer.tp_q_head_num * layer.head_dim)
```

## 评论区精华

review 的 13 条内联评论集中在正确性与性能取舍，主要交锋：

1. 反量化尺度：yichiche 指出应使用 q/k/v 各自 scale 而非 one_scale，作者以 fixing commit 落实；
2. 全局变量：yichiche 建议减少全局变量、仅用 is_gfx95_supported 门控，作者采纳并内联环境变量；
3. 回退内核选择：yichiche 建议回退到 flash_attn_varlen_func，作者用同容器同版本 A/B 证明该函数在 head 256 下非 asm 且大 ISL 不占优，最终保留 mha_batch_prefill_func；
4. 模式门控：yichiche 指出 target_verify 后 forward_extend 可能从非 extend 模式到达，要求改用 is_extend() 并显式排除 sinks 与 softcap，合并版本全部纳入；
5. fp8 KV 前提：yichiche 指出非 fp8 KV 时额外量化伤性能，最终以 kv_cache_dtype == fp8_dtype 收口。 此外 amd-bot 的 CI 审查明确警告：新路径默认关闭且无测试设置，PR CI 从未执行该代码，绿色不能证明正确性。

- 反量化尺度应使用 q/k/v 各自 scale 而非 one_scale (correctness): 作者以 fixing commit "replacing one_scale with k,v,q scale by the reviewer remark for the new pertensor kernel" 落实，合并版本使用 fp8_q_descale/k_descale/v_descale。
- 减少全局变量，仅用 is_gfx95_supported 门控 (design): 作者采纳，最终在 forward_extend 内直接内联 get_bool_env_var("SGLANG_AITER_FMHA_FP8_ASM", "False") and is_gfx95_supported()。
- 回退内核选择：flash_attn_varlen_func vs mha_batch_prefill_func (performance): 保留 mha_batch_prefill_func 作为回退路径，并附上竞争基准图佐证。
- 门控条件收紧：is_extend、sinks、logits_soft_cap (correctness): 合并版本已包含 forward_mode.is_extend()、sinks is None、logits_soft_cap == 0.0 三个门控，与 #28482 保持一致。
- 非 fp8 KV cache 时不应走 fp8 内核 (performance): 最终以 kv_cache_dtype == fp8_dtype 作为门控之一，bf16 KV 场景完整走原始路径。
- PR CI 从未执行新代码路径 (testing): 合并时未解决，仅依赖作者手动验证；后续由 #33399 负责默认启用与测试补齐。

# 风险与影响

- 风险：
  - 正确性风险：快速路径完全依赖手动验证，PR CI 从未执行（amd-bot 确认：环境变量仅在本文件引用、runner 为 gfx942、需要 fp8 KV + hd256）。若后续有人在 gfx950 上开启该变量，错误将直接进入生产。门控虽严，但无自动化回归保护，属于低概率高风险。
  - 性能风险：A/B 数据显示标准 IL8192/OL1024 在低并发（conc 4）下 TTFT 反而 +91%，conc 256 下 +427% 并伴随吞吐 -12.3%；60k ISL 场景 conc 256 下 ITL 只提升 1.6% 而 TTFT 仍达 168s。混合负载或小 batch 开启快速路径可能得不偿失，需要按负载特征选择是否开启。
  - 兼容性风险：flash_attn_varlen_fp8_pertensor_func 仅存在于 aiter main 分支（PR #3732 之后）；旧版 aiter 构建会 ImportError，虽然 import 块有 try/except 兜底，但 aiter 后端整体本来就需要 aiter，升级 aiter 版本是硬前提。环境变量命名在 PR 演进中发生过不一致（body 中为 SGLANG_AITER_FMHA_FP8_HD256，最终代码为 SGLANG_AITER_FMHA_FP8_ASM），文档与代码存在轻微漂移风险。
  - 维护风险：与 #28482 的 SGLANG_AITER_FP8_CONTIG_PREFILL 并存，两条 opt-in 路径作用于同一函数、同一内核，互斥关系未在代码中显式声明；后续 #33399 又删变量改默认启用，说明该 PR 是过渡实现。
- 影响：
  - 用户影响：范围严格限定为 AMD gfx950 (MI350x) + aiter 后端 + head_dim 256 模型（如 Qwen3.5）+ fp8 KV cache 且显式设置 SGLANG_AITER_FMHA_FP8_ASM=1 的部署；默认用户零感知，无 API/ 模型行为变化。
  - 系统影响：目标场景为 60k 级长上下文 prefill，可获 TTFT -21~-40%、吞吐 +21~+39% 的显著收益；但短序列低并发与超高并发下存在 TTFT 回退观测，不建议无条件开启。

- 团队影响：为 AMD 平台 Qwen3.5/MI350x 性能路径铺路，与 #28482 形成互补（MoE 长上下文 vs 标准形状），并直接孕育 #33399 的默认启用演进；同时暴露了 AMD 专用路径缺乏 CI 覆盖的流程问题。
- 风险标记：新代码路径未被 CI 执行，默认关闭的 opt-in 分支，依赖 aiter main 分支内核，低并发短序列 TTFT 回退，环境变量命名前后不一致

## 关联脉络

- PR #28482 [AMD] Contiguous varlen FMHA for Qwen3.5 mxfp4: 同一 fp8 hd256 FMHA 快速路径的姊妹 PR，共享 flash_attn_varlen_fp8_pertensor_func 内核与无缓存前缀 / 无滑动窗口 / 无 sinks/ 无 softcap 等门控；本 PR body 明确说明两者关系与分工（MoE 长上下文 vs 标准 IL8192/OL1024 形状）。
- PR #3732 HD256 FMHA FP8 GFX950: 上游 ROCm/aiter PR，提供 flash_attn_varlen_fp8_pertensor_func asm 内核，是本 PR 全部性能收益的源头；PR body 与实现均依赖该内核的存在。
- PR #33399 Remove the global variable and fix a shape issue to enable fp8 prefill by default: yichiche 在 Issue 评论中披露的 follow-up PR：删除全局变量（SGLANG_AITER_FMHA_FP8_ASM）并修复 shape 问题，在 fp8 KV cache 下默认启用 fp8 prefill，代表该能力从 opt-in 到默认启用的正式化演进。