A Scalable Multi-Path Microarchitecture for Efficient GPU Control Flow

HPCA 2014
GPU for non-graphics computing

• Single Instruction Multiple Thread (SIMT) execution model
  • Use a single instruction sequencer to operate on a group of threads

• Improve efficiency by amortizing instruction fetch and decode cost.

• Reduce thread level parallelism (TLP) with divergent control flow
  • Current GPUs serialize the execution of divergent paths
Single-Path Stack Execution Model

- Per warp (4 threads) stack is used to manage divergence
- Serialize the execution of different control flow paths (B & C)
- Immediate postdominator (IPDOM) is set as the RPC
  - The earliest point where all divergent threads are guaranteed to execute
Stack-Based Reconvergence Limitations

- Assume two instructions per block
- Allow only a single control flow path to execute at a time -> reduce the number of running threads
- Potential: during the idle cycles due to long latency, alternative paths could make progress.
Contribution

• Goal
  • Allow concurrent scheduling of any number of warp splits while maintaining IPDOM reconvergence
  • Enable early reconvergence opportunistically at run-time

• Scheme
  • Warp split table (ST)
    – the state of warp splits executing in parallel basic blocks
  • Reconvergence table (RT)
    – reconvergence points for the splits
Operation of MP IPDOM

**Pending mask:** threads that have not yet reached the reconvergence point

- Upon each update to the pending active mask in the RT table, the Pending Mask is checked if it is all zeros.
Timing Diagram of MP IPDOM

- Single-Path Stack

- Multi-Path Stack

• Occupy the idle cycles with the alternative control path
Opportunistic Early Reconvergence

• IPDOM reconvergence point: the **earliest guaranteed** reconvergence point.

• In certain situations, there are **opportunities** to reconverge at **earlier points** than the IPDOM point.

• **Not guaranteed** for all executions.
A divergence at $BR^C_{A-D}$ results in two splits B_{1010} and C_{0101}, and split C_{0101} reaches $BR^C_{A-D}$ before split B_{1010} finishes executing basic block B.
Opportunistic Early Reconvergence

- An early reconvergence opportunity: two splits ($B_{0101}$ and $B_{1010}$) of the same warp executing the same basic block $B$

- The **early reconvergence point** is the program counter of the next instruction of the leading warp split ($B_{1010}$).
Opportunistic Early Reconvergence

- Warp split $B_{0101}$ reaches the early reconvergence point $B_R$
- $B_{0101}$'s entry in the ST is invalidated
- Pending mask of the reconvergence entry $B_{R1111}$ is updated
- The reconvergence entry $B_{R1111}$ moves from the RT to the ST
SIMD Unit Utilization

- Same SIMD unit utilization for SPS, DPS and basic MP: all reconverge at the IPDOM reconvergence points
- Opportunistic rec. opt: an average of 48% and up to 182%

Idle Cycles
Overall Performance

• The speedup comes mainly from:
  • Reduced idle cycles (i.e., more warp split instructions per cycle)
  • Improved SIMD units’ utilization (i.e., more throughput per warp split instruction).

• MP with opportunistic reconvergence has harmonic mean speedup of:
  • 32% over the SPS model
  • 18.6% over the basic MP
  • 12.5% over DPS models
Conclusions

• Propose a novel mechanism that enables multi-path execution in GPUs. Achieved by two tables:
  • One tracks the concurrent executable paths upon every branch
  • The other tracks the reconvergence points of these branches

• Modify the proposed model to enable opportunistic early reconvergence at run time

• Evaluations show 32% speedup over conventional single-path SIMT execution