Dynamic Warp Resizing: Analysis and Benefits in High-Performance SIMT

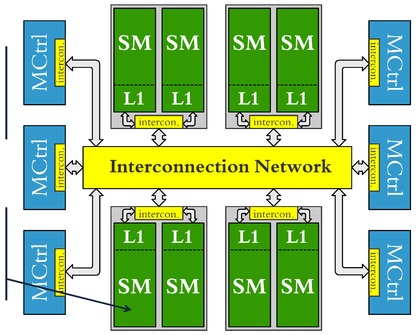
Ahmad Lashgar[1] , Amirali Baniasadi[3] , and Ahmad Khonsari[1 3]
[1]School of ECE, University of Tehran [2]ECE Department, University of Victoria
[3]School of Computer Science, Institute for Research in Fundamental Sciences

ICCD 2012 IEEE COMPUTER SOCIETY

1 Background

GPUs are NoC of SIMD cores (SM) and Memory Controllers

Each SM is a deep-multithreaded processor (1K thread/SM)

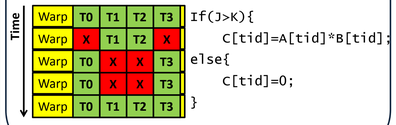


Threads are grouped into coarser schedulable elements (warps)

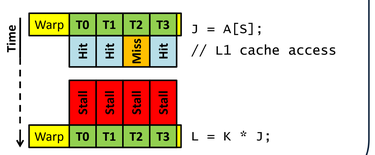
Benefits:
- Memory access Coalescing
- Improving SIMD efficiency

Disadvantages:
- Branch divergence
- Memory divergence

Branch Divergence:
Threads of a warp take different diverging paths

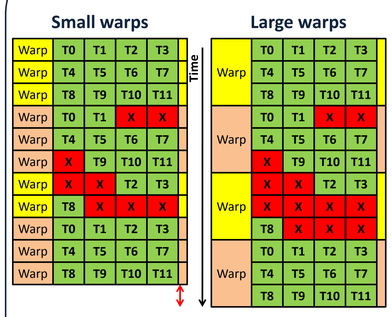


Memory Divergence:
Threads of a warp take different L1 outcome



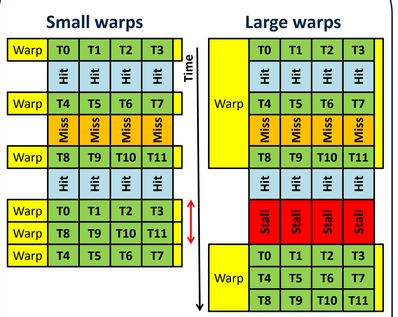
2 Warp Size: Small vs. Large

Warp Size Impact on Branch Divergence



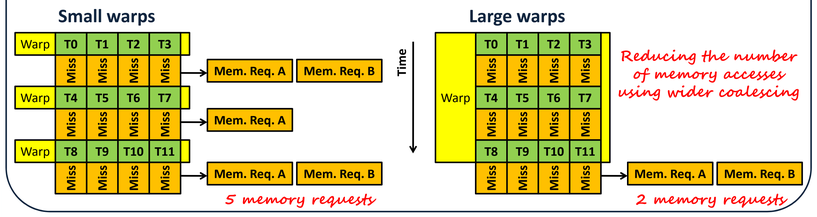
Saving some idle cycles

Warp Size Impact on Memory Divergence



Improving memory access latency hiding

Warp Size Impact on Memory Access Coalescing



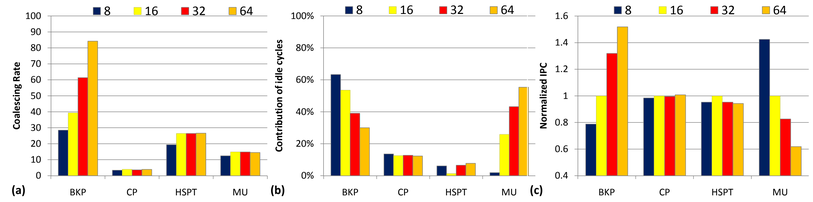
Reducing the number of memory accesses using wider coalescing

5 memory requests

2 memory requests

3 Motivation

- Large warps improve memory access coalescing
- Small warps reduce synchronization overhead, memory divergence, and branch divergence

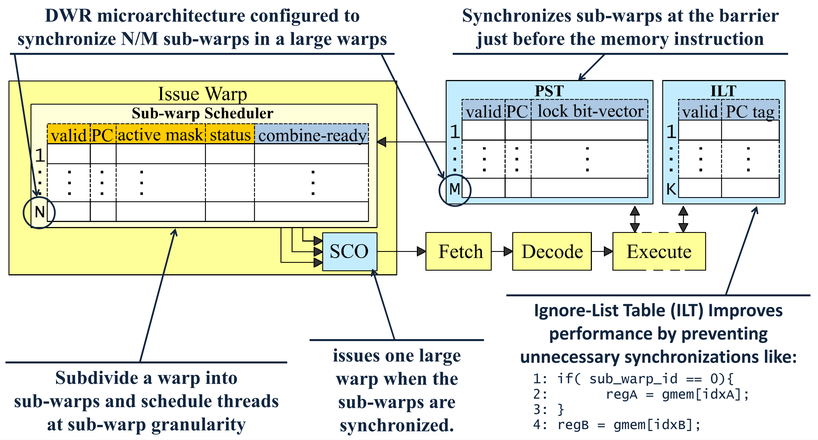


(a) Coalescing rate, (b) Idle cycle share, and (c) Performance under different warp sizes. IPC is normalized to a GPU using 16 threads per warp.

4 Dynamic Warp Resizing (DWR)

DWR microarchitecture configured to synchronize N/M sub-warps in a large warps

Synchronizes sub-warps at the barrier just before the memory instruction



Subdivide a warp into sub-warps and schedule threads at sub-warp granularity

issues one large warp when the sub-warps are synchronized.

Ignore-List Table (ILT) Improves performance by preventing unnecessary synchronizations like:

```
1: if( sub_warp_id == 0){
2:       regA = gmem[idxA];
3: }
4: regB = gmem[idxB];
```

5 ISA Modification

Baseline instruction sequence

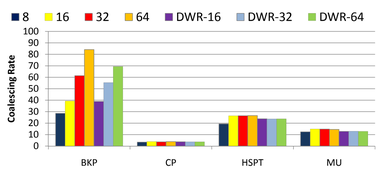
```
cvt.u64.s32     %rd1, %r3;
ld.param.u64    %rd2, [__parm1];
add.u64         %rd3, %rd2, %rd1;
ld.global.s8    %r5, [%rd3+0];
mov.u32         %r6, 0;
setp.eq.s32     %p2, %r5, %r6;
@%p2 bra        $Lt_0_5122;
mov.s16         %rh2, 0;
st.global.s8    [%rd3+0], %rh2;
```

DWR instruction sequence

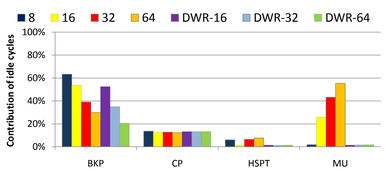
```
cvt.u64.s32     %rd1, %r3;
ld.param.u64    %rd2, [__parm1];
add.u64         %rd3, %rd2, %rd1;
bar.synch_partner 0;
ld.global.s8    %r5, [%rd3+0];
mov.u32         %r6, 0;
setp.eq.s32     %p2, %r5, %r6;
@%p2 bra        $Lt_0_5122;
mov.s16         %rh2, 0;
bar.synch_partner 0;
st.global.s8    [%rd3+0], %rh2;
```

6 Experimental Results

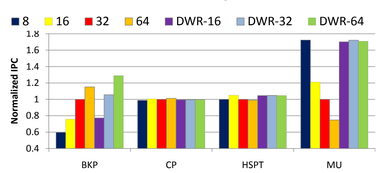
Each configuration of DWR is notated by DWR-x where x denotes the largest warp size.



Coalescing rate



Share of idle cycles



Performance

Benchmarks Characteristics. Large-warp intensive instructions (LAT) shows the number of LATs and the number of ignored LATs under DWR (with maximum warp size of 64).

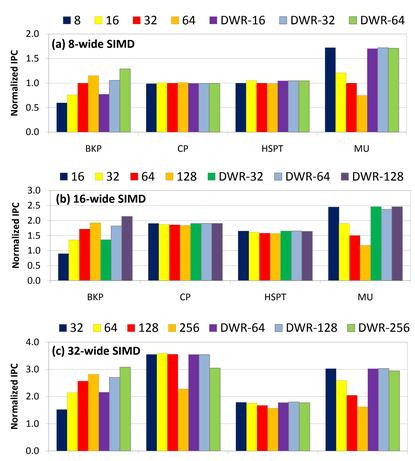
Abbr.	Grid Size	Block Size	#Insn	LAT
BKP	2x(1,64)	2x(16,16)	2.9M	0/17
CP		(16,8)	113M	0/5
HSPT	(43,43)	(16,16)	76M	0/20
MU	(196)	(256)	75M	3/11

Baseline configuration for GPGPU-sim

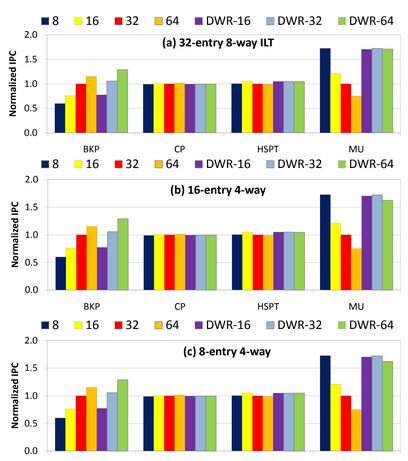
NoC	
#SMs : #Memory Ctrls	16 : 6
#SM Sharing a Network Interface	2
Clocking	
Core : Interconnect : DRAM	1300 : 650 : 800 MHz
Memory	
#Banks Per Memory Ctrls	8
DRAM Scheduling Policy	FCFS
GDDR3 memory timing	tRRD=12, tRCD=21, tRAS=13, tRP=34, tRC=9, tCL=10
SM	
Warp size : SIMD width	32 : 8
Max. allowed CTA/SM : Thread/SM	8 : 1024
Register file : Shared memory	64KB : 16KB
L1 Data $	48KB : 12-way : LRU : 64Byte blocks
L1 Texture $	16KB : 2-way : LRU : 64Byte blocks
L1 Constant $	16KB : 2-way : LRU : 64Byte blocks

7 Sensitivity

Sensitivity to SIMD width



Sensitivity to ILT size



Abstract

Modern GPUs synchronize threads grouped in a warp at every instruction. These results in improving SIMD efficiency and makes sharing fetch and decode resources possible. The number of threads included in each warp (or warp size) affects divergence, synchronization overhead and the efficiency of memory access coalescing. Small warps reduce the performance penalty associated with branch and memory divergence at the expense of a reduction in memory coalescing. Large warps enhance memory coalescing significantly but also increase branch and memory divergence. Dynamic workload behavior, including branch/memory divergence and coalescing, is an important factor in determining the warp size returning best performance. We propose Dynamic Warp Resizing (DWR) to adjust warp size during runtime and according to program characteristics.

Contact

Ahmad Lashgar
a.lashgar@ece.ut.ac.ir http://lashgar.hourb.com/

Amirali Baniasadi
amirali@ece.uvic.ca http://www.ece.uvic.ca/~amiralib/

Ahmad Khonsari
ak@ipm.ir http://ece.ut.ac.ir/faculty/Khonsari/