TLB and Pagewalk Performance in Multicore Architectures with Large Die-Stacked DRAM Cache

Adarsh Patil

Adviser: Prof. R Govindarajan

Perspective Seminar

6th Nov 2015

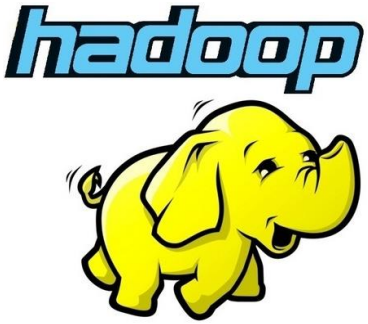
Computer Science and Automation

Outline

- Introduction
  - Address Translation - TLBs and Page Walks
  - Die stacked DRAM caches
- Objective
- Experimental Setup
  - Framework
  - Methodology
- Results
- Conclusion and Future Work

Computing Trends

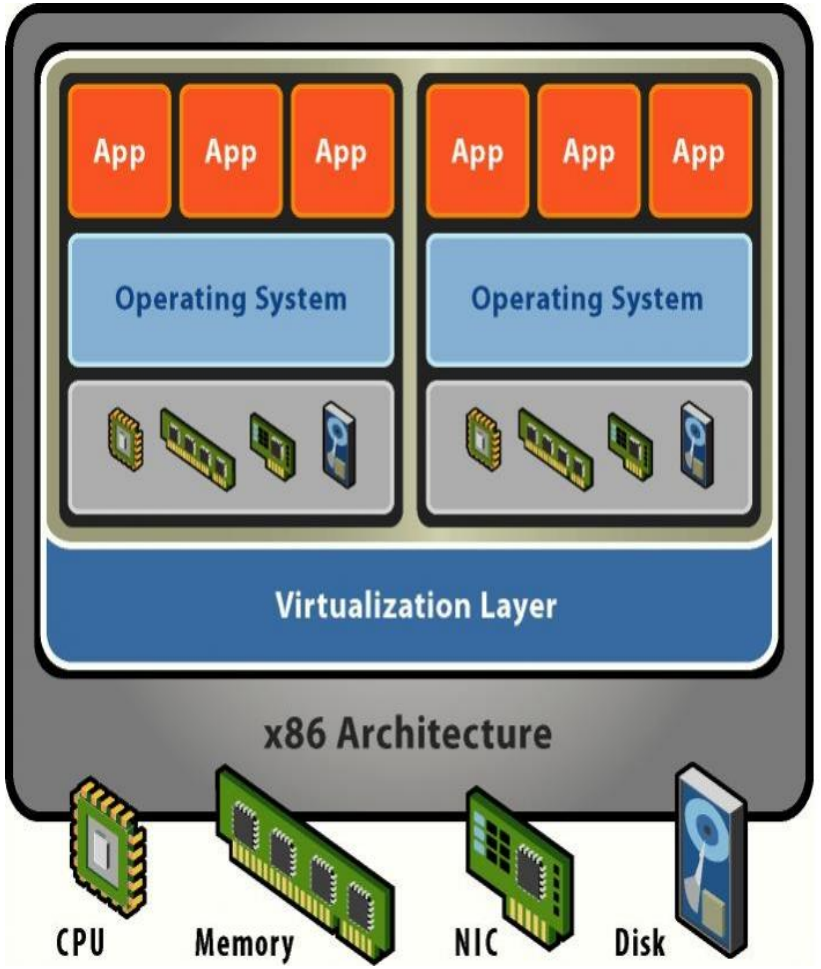
- Software
  - Large memory footprint



*Apps in Big Data Bench / Cloud Suite Benchmark

Computing Trends

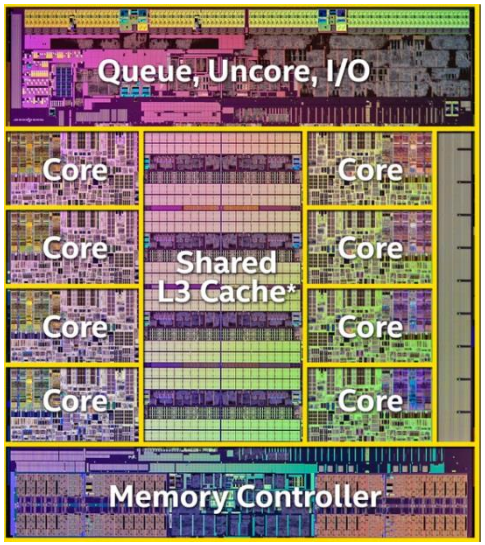
- Software
  - Large memory footprint
  - Virtualization and cloud computing



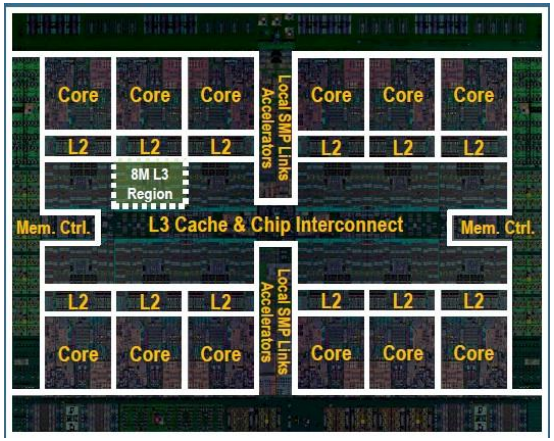
*Source : VMware

Computing Trends

- Software
  - Large memory footprint
  - Virtualization and cloud computing
- Architectural
  - Multicore / Manycore architectures

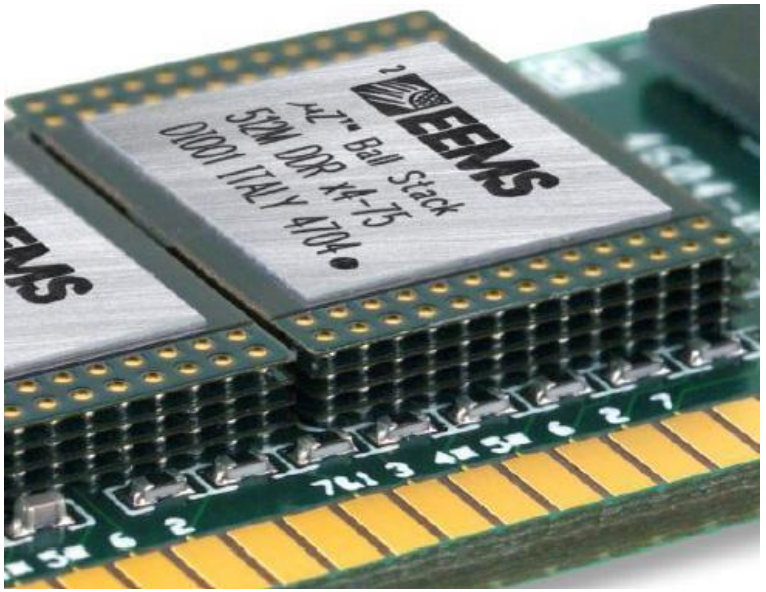


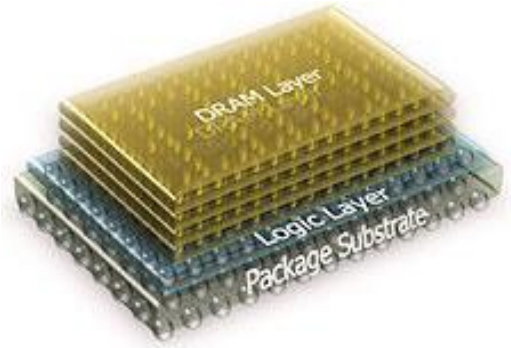
Intel Haswell-E & IBM Power 8



Computing Trends

- Software
  - Large memory footprint
  - Virtualization and cloud computing
- Architectural
  - Multicore / Manycore architectures
  - Large Die stacked DRAM cache



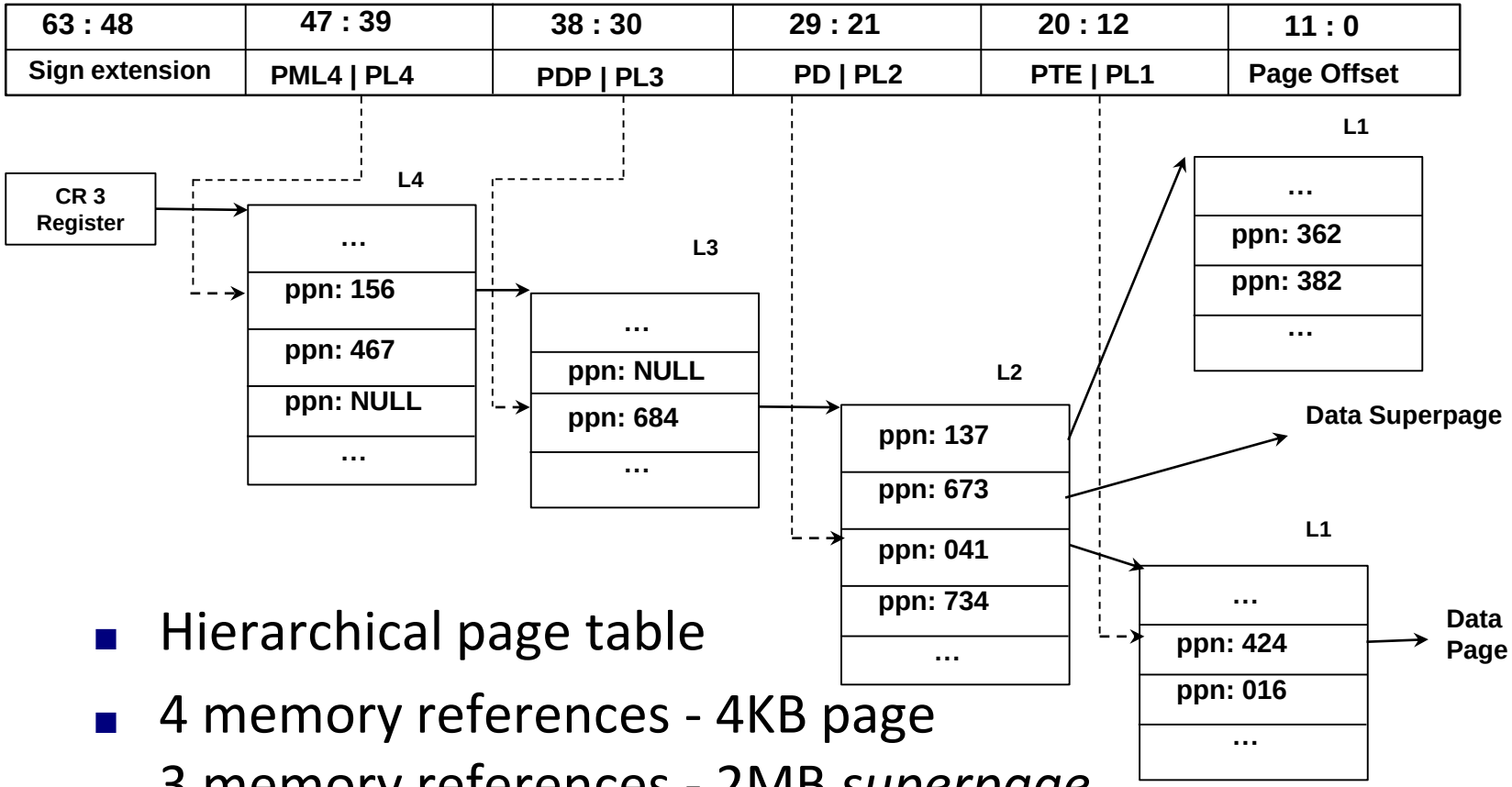


*Source : Invensas, Tessera

Paged Virtual Memory

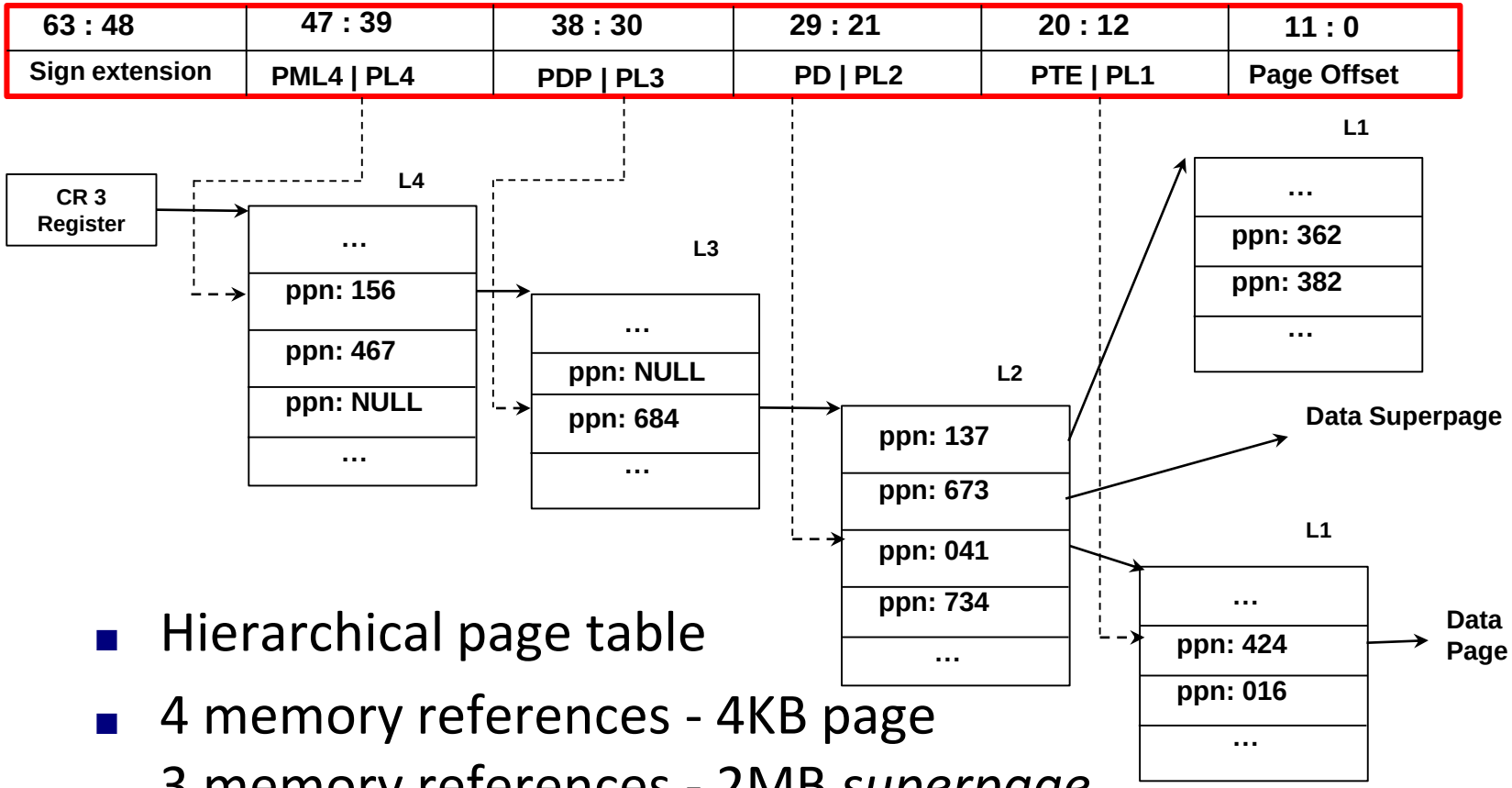
- Virtual address space divided into "pages"
- "Page Table" : In-memory table, organized as *radix tree*, to map virtual to physical address and store meta-information (replacement, access privilege, dirty bit etc.)
- Page table entries cached in fast lookup structures called "Translation Lookaside Buffers (TLBs)"
- Page Table has evolved to 4-level tree to accommodate 48-bit VA

Page Table Structure



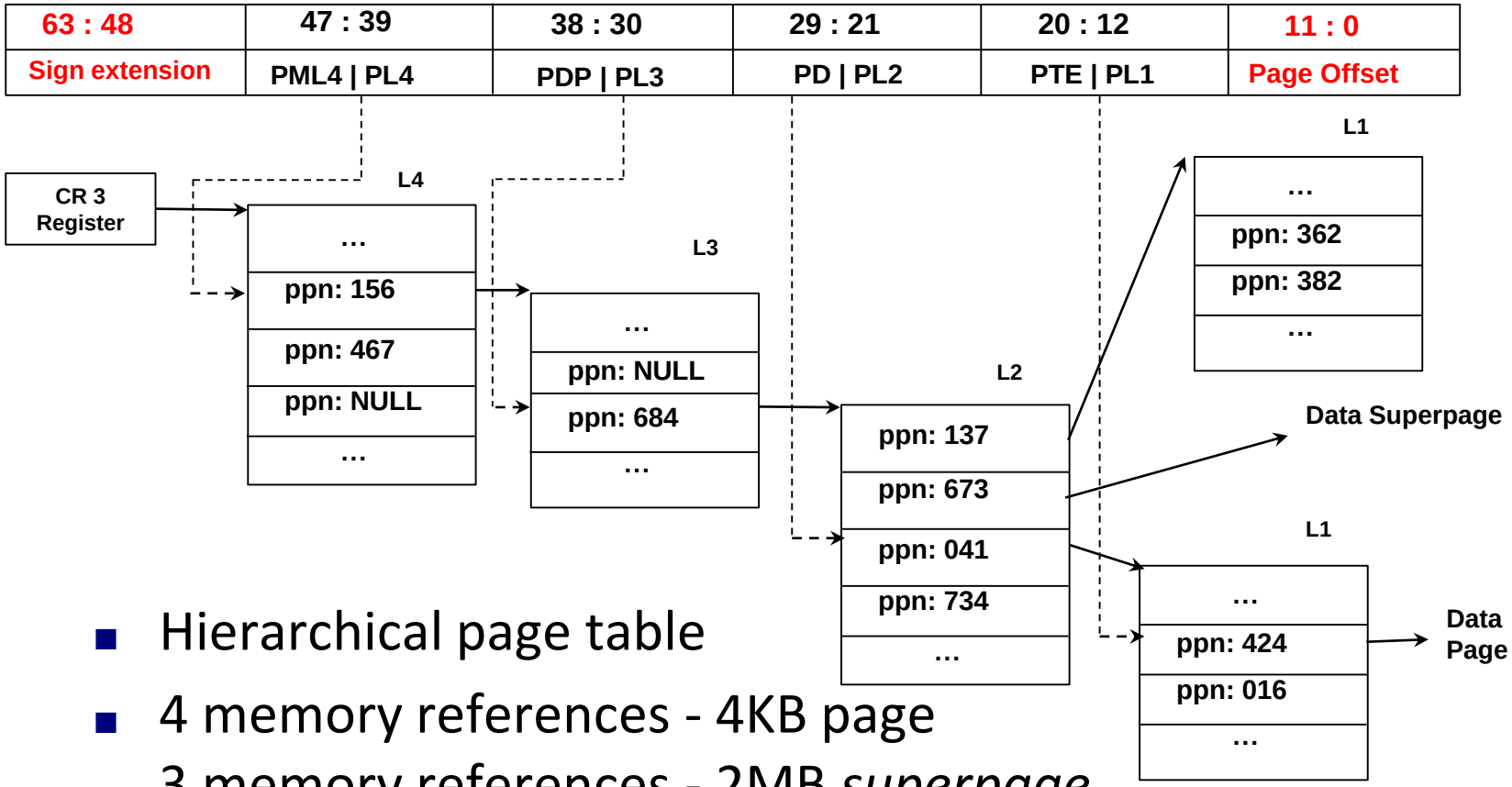
- Hierarchical page table
- 4 memory references - 4KB page
  3 memory references - 2MB *superpage*
- Each entry is 8 bytes
- TLB stores VA to PA

Page Table Structure



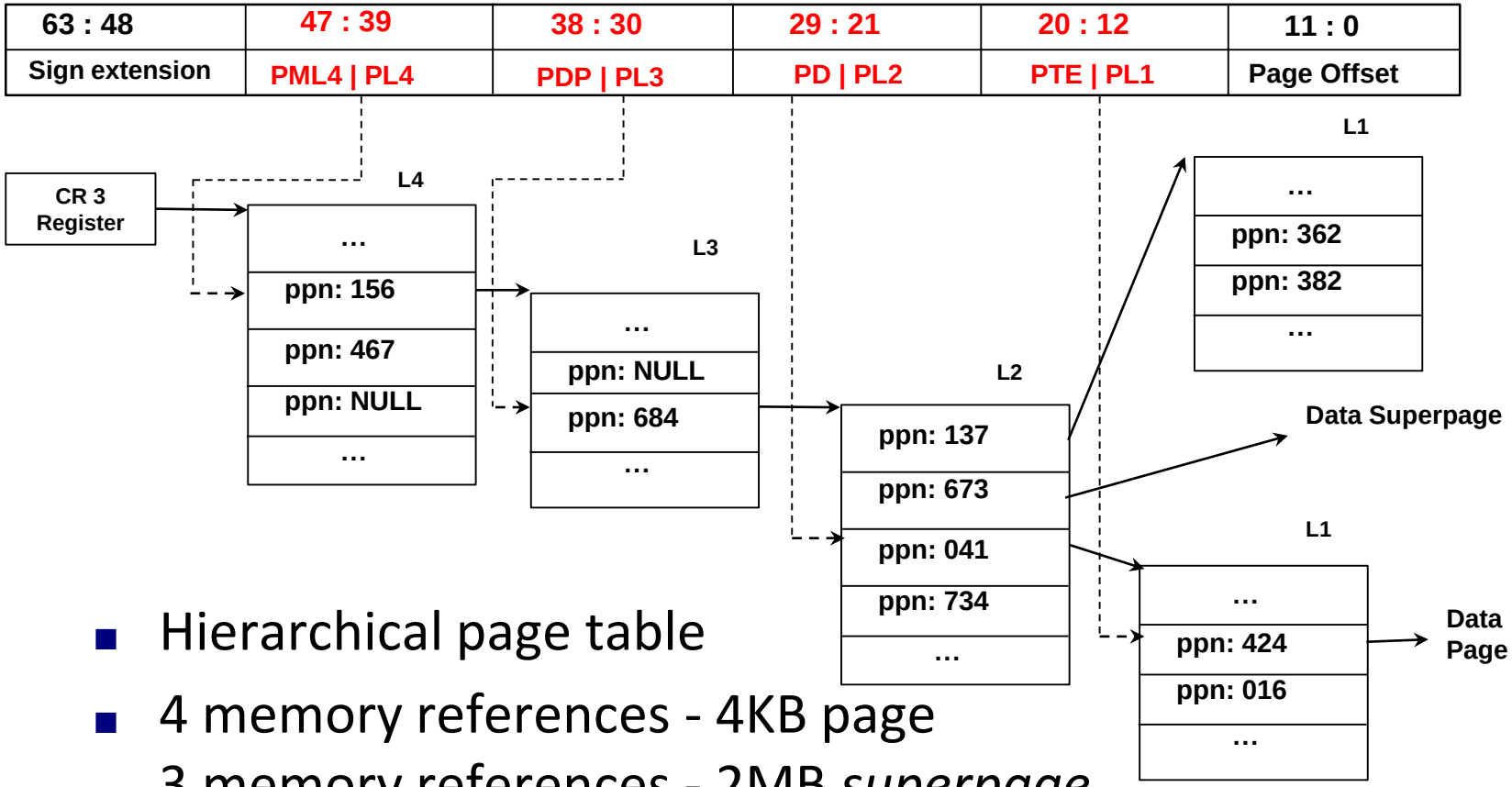
- Hierarchical page table
- 4 memory references - 4KB page
  3 memory references - 2MB *superpage*
- Each entry is 8 bytes
- TLB stores VA to PA

Page Table Structure



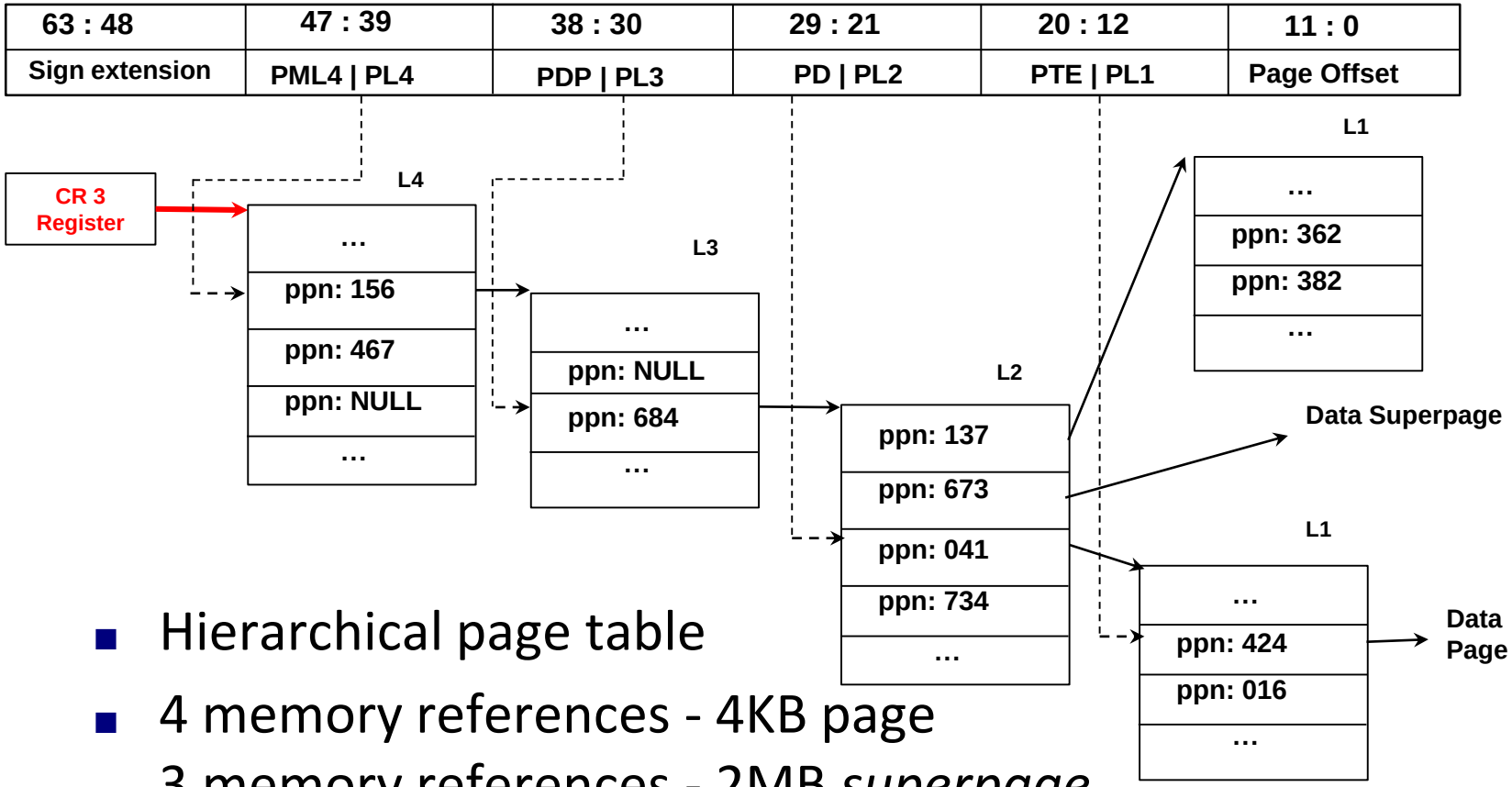
- Hierarchical page table
- 4 memory references - 4KB page
  3 memory references - 2MB *superpage*
- Each entry is 8 bytes
- TLB stores VA to PA

Page Table Structure



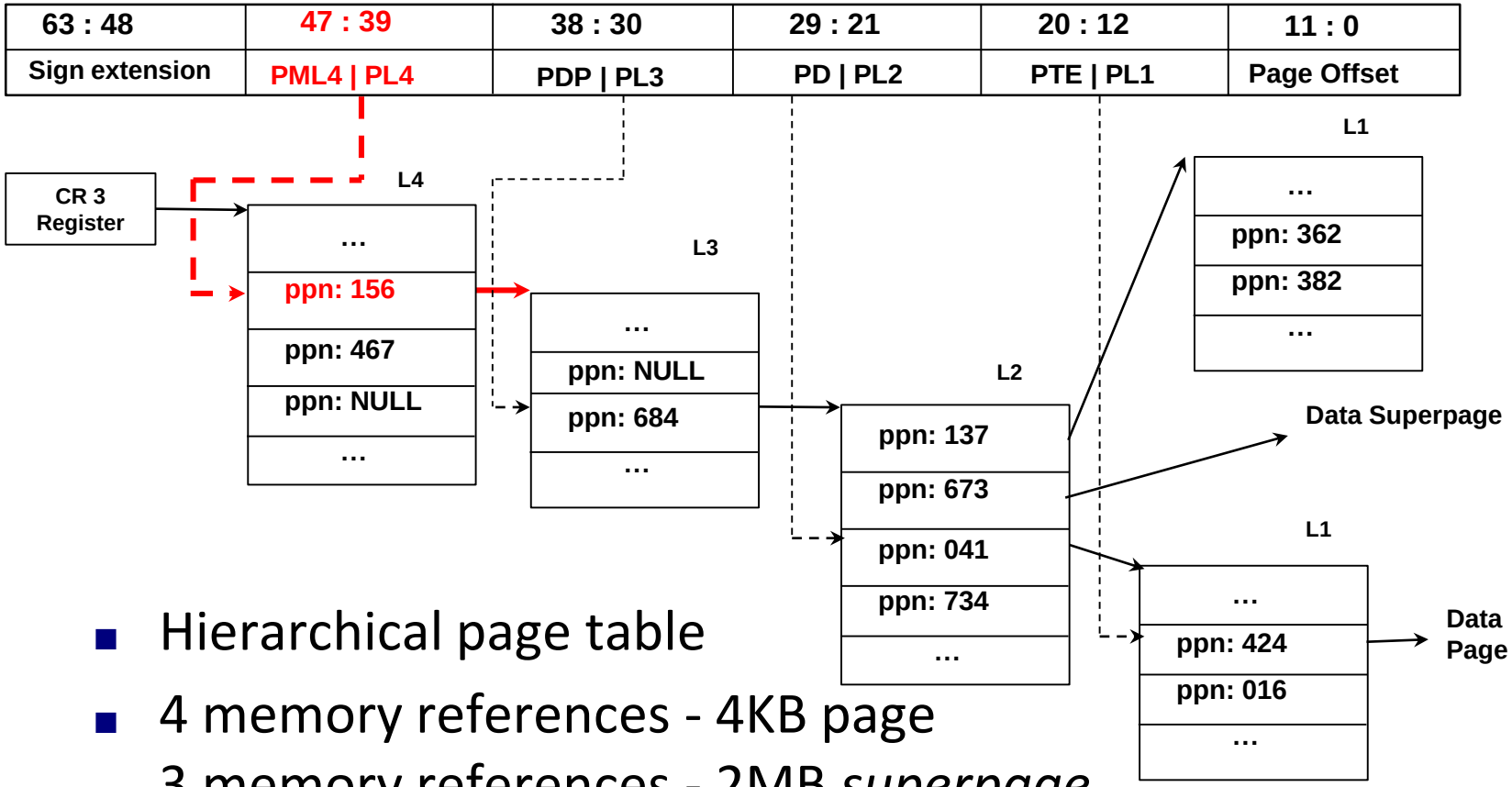
- Hierarchical page table
- 4 memory references - 4KB page
  3 memory references - 2MB *superpage*
- Each entry is 8 bytes
- TLB stores VA to PA

Page Table Structure



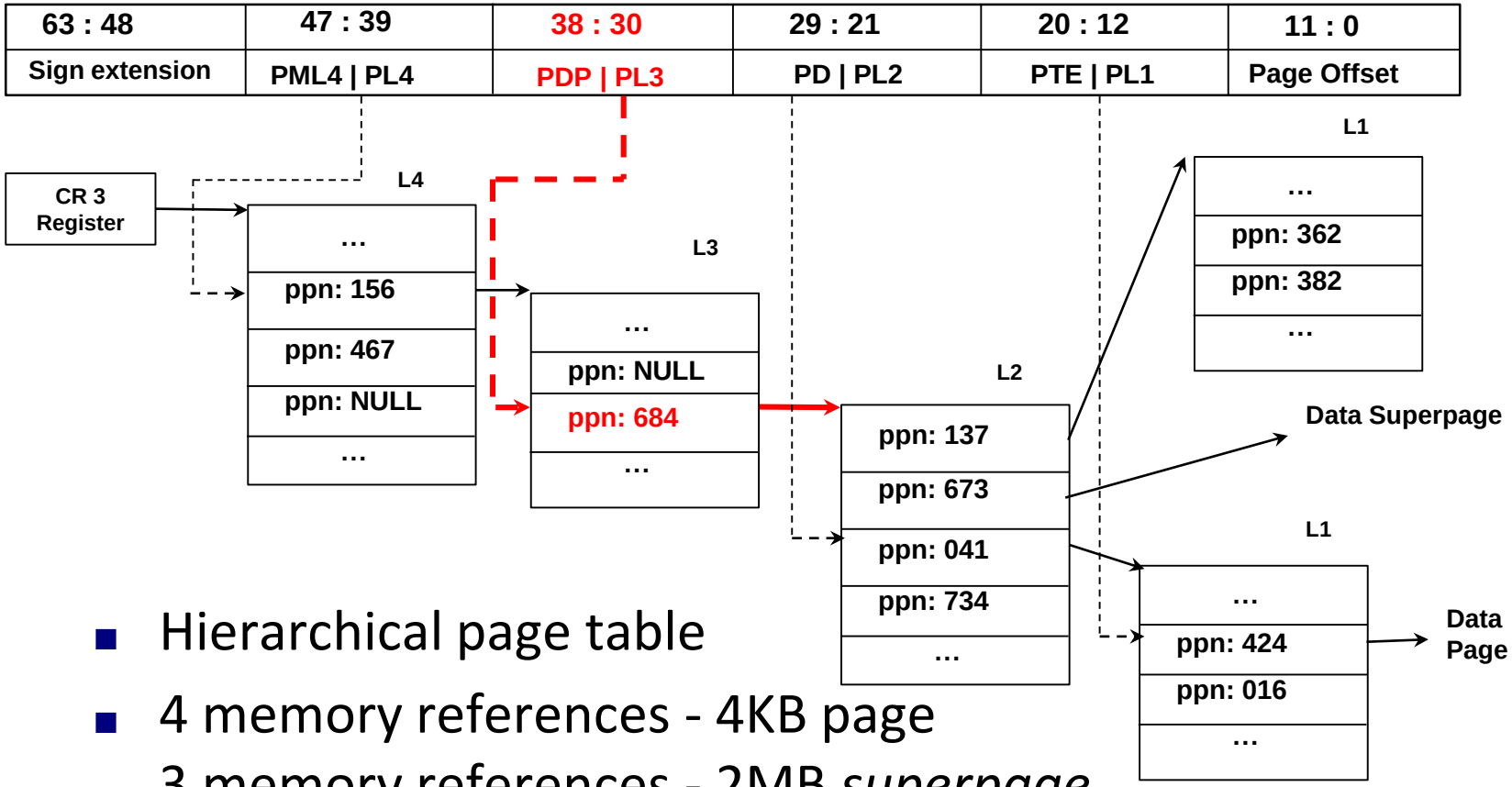
- Hierarchical page table
- 4 memory references - 4KB page
  3 memory references - 2MB *superpage*
- Each entry is 8 bytes
- TLB stores VA to PA

Page Table Structure



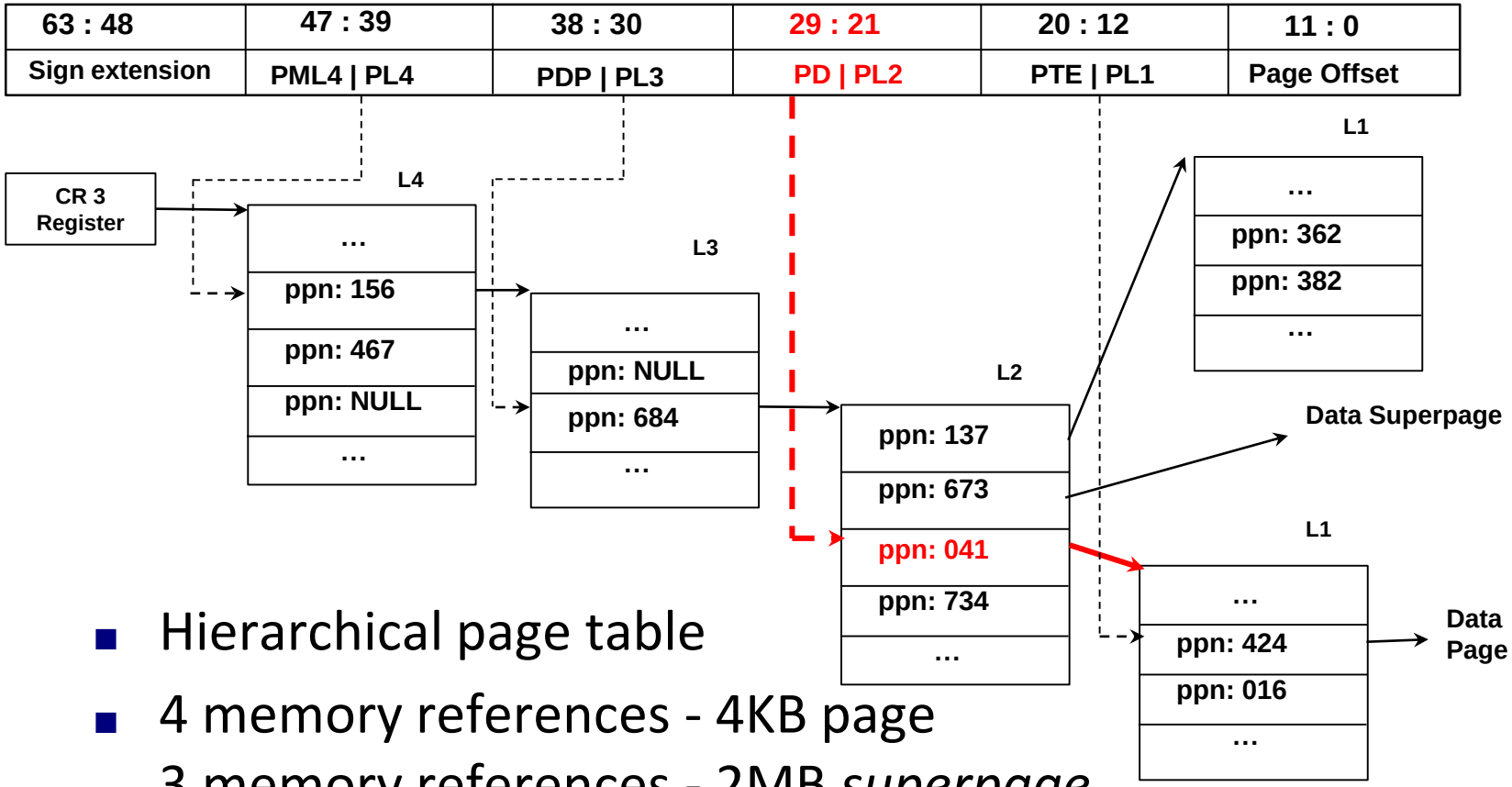
- Hierarchical page table
- 4 memory references - 4KB page
  3 memory references - 2MB *superpage*
- Each entry is 8 bytes
- TLB stores VA to PA

Page Table Structure



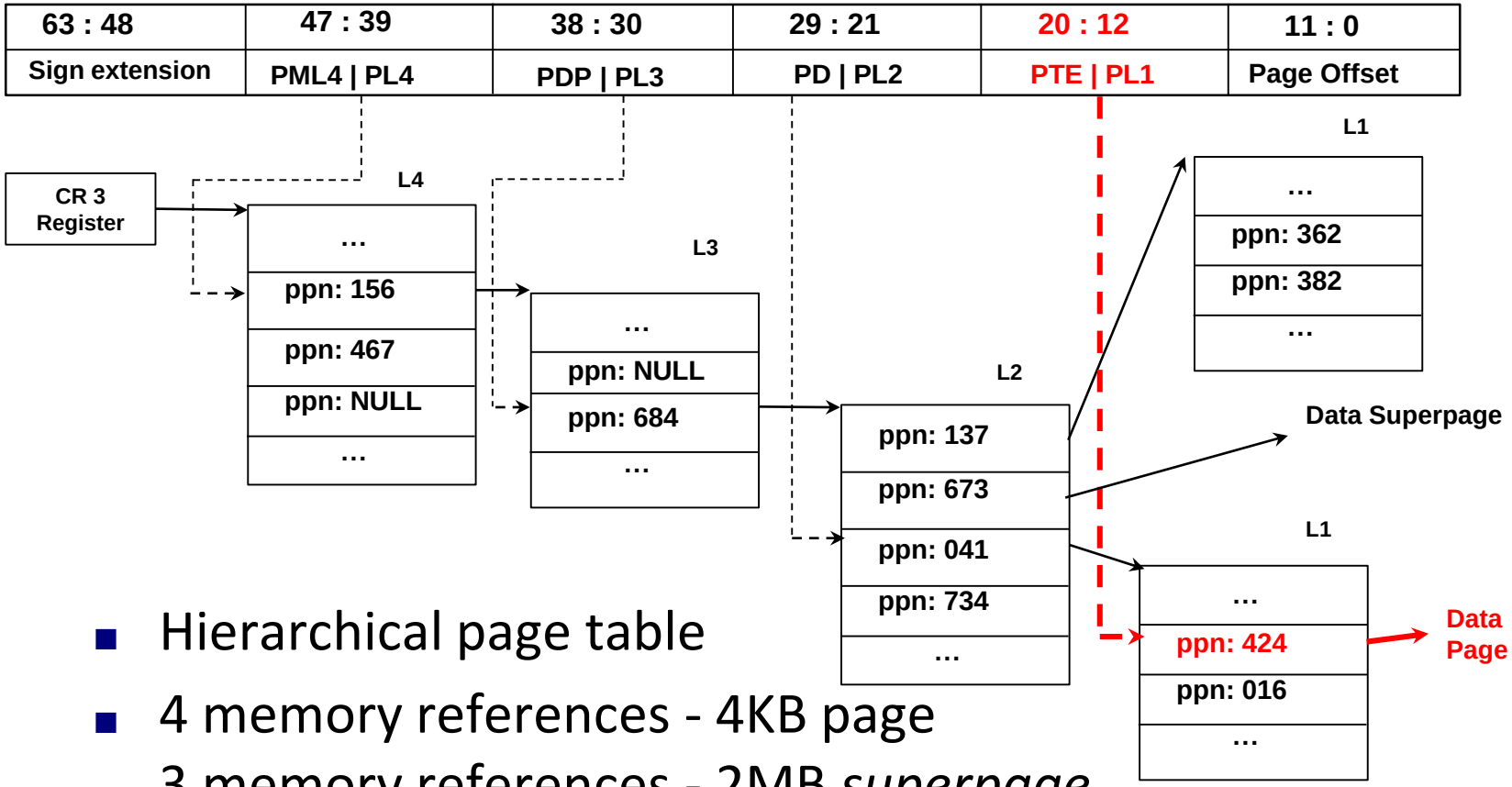
- Hierarchical page table
- 4 memory references - 4KB page
  3 memory references - 2MB *superpage*
- Each entry is 8 bytes
- TLB stores VA to PA

Page Table Structure



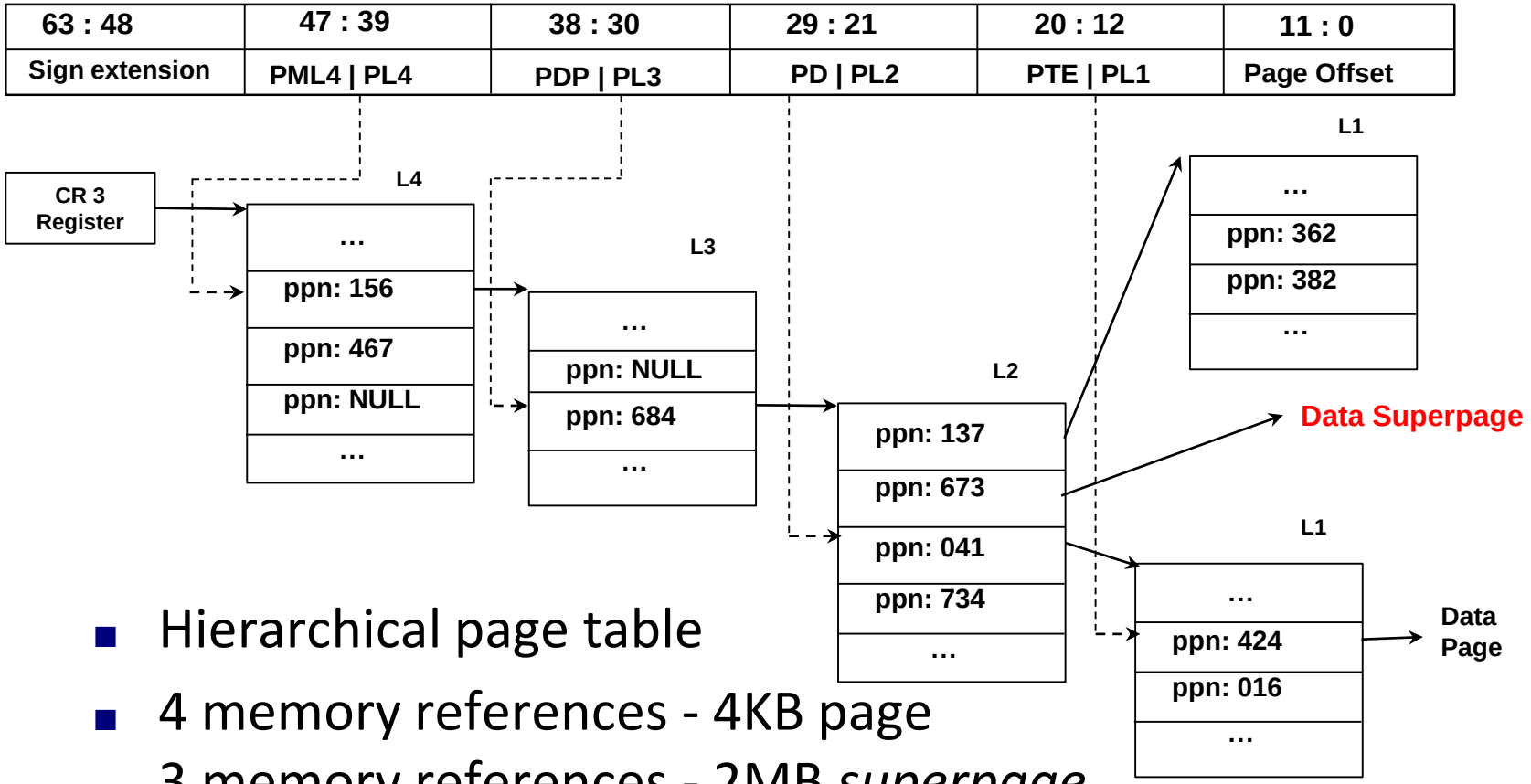
- Hierarchical page table
- 4 memory references - 4KB page
  3 memory references - 2MB *superpage*
- Each entry is 8 bytes
- TLB stores VA to PA

Page Table Structure



- Hierarchical page table
- 4 memory references - 4KB page
  3 memory references - 2MB *superpage*
- Each entry is 8 bytes
- TLB stores VA to PA

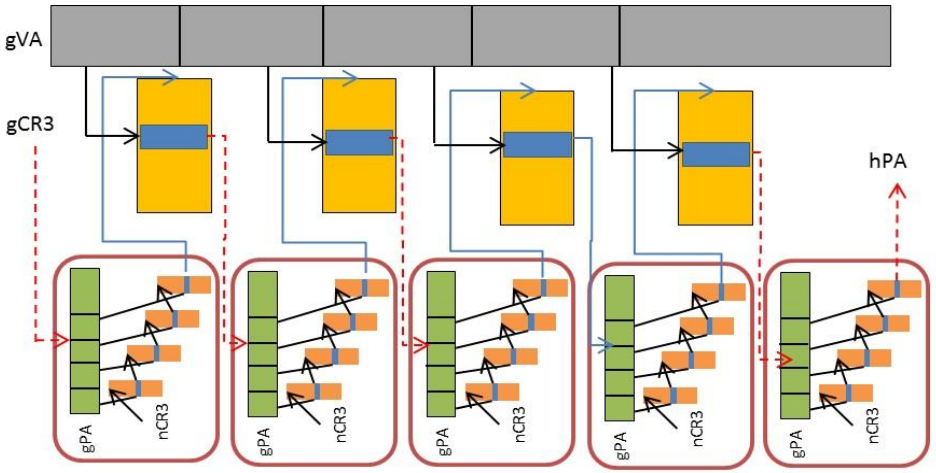
Page Table Structure



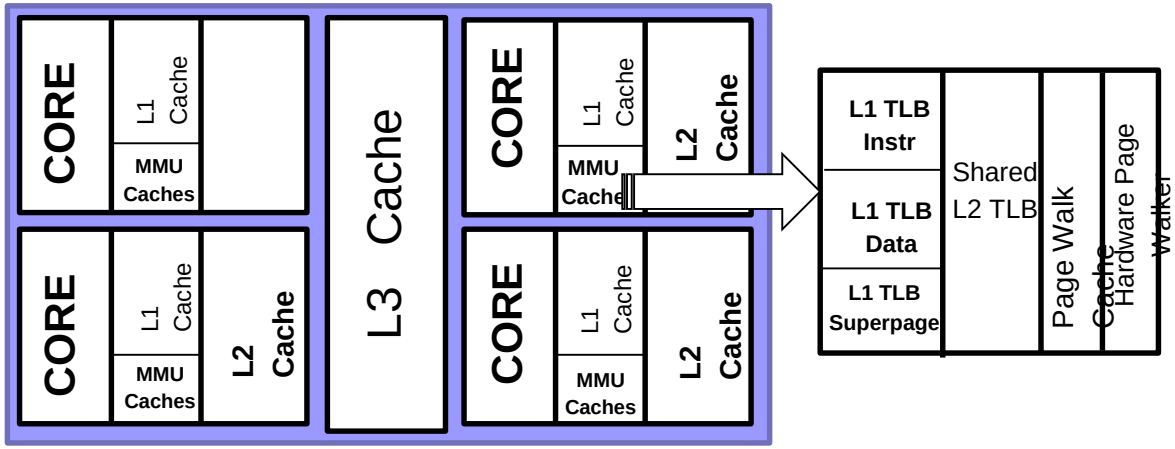
- Hierarchical page table
- 4 memory references - 4KB page
  3 memory references - 2MB *superpage*
- Each entry is 8 bytes
- TLB stores VA to PA

Page Table Structure-Virtualization

- Guest Page Table (gPT)
  - Translate guest virtual to guest physical
  - Setup and modified by guest independently
- Nested Page Table (nPT)
  - Translate host virtual to host physical
  - Controlled by host
- Upto 24 memory references on page walk
- TLB stores to end to end translation



Address Translation in Hardware



Page Walk cycles

Translation Page Walk

Miss

Multi-level TLB

< 4 cycles

CPU

VA

PA

Set

L1 Cache VIPT

4 cycles

?

Miss

Large PIPT Caches

Miss

Memory

Hit

Hit

6 / 10 cycles

180-200 cycles

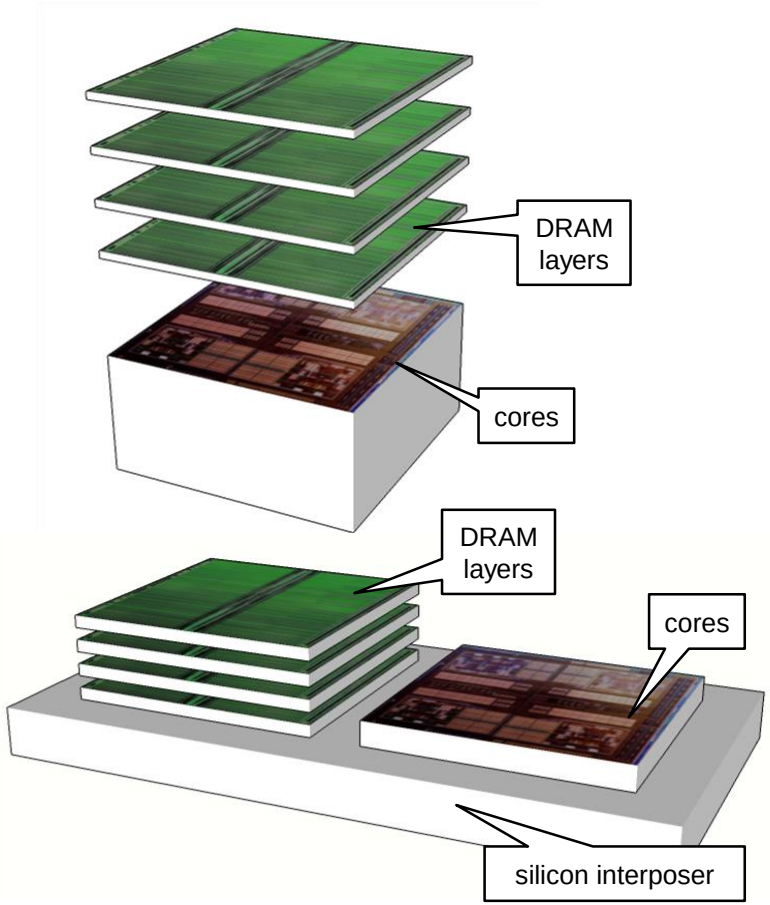
Data

TLB-reach & page walk latencies

- TLB-reach: amount of data that can be accessed without causing a miss
  - Clustered [HPCA ‘14] and Coalesced [MICRO ’12] TLBs
  - Superpage friendly TLBs [HPCA ‘15] using skewed TLBs
  - Shared last level TLBs [HPCA ‘11] evaluates shared TLBs for multi-cores
  - Direct segment [ISCA ‘13] - *primary region* abstraction to map part of the virtual address space using segment registers and avoid paging completely
  - Redundant memory mappings [ISCA ‘15] - allocation in units called *ranges (eager paging in OS)* and maintain ranges in a separate range-TLB, compatible with traditional paging.
- Speeding up miss handling
  - AMD proposed Accelerating 2D page walks [ASPLOS ‘08] by using page walk caches for virtualization
  - Characterize TLB behaviors and sensitivity of individual SPEC 2000 [SIGMETRICS ‘02] and PARSEC [PACT ‘09] applications

Die stacked DRAM Cache

50% Lower power consumption*	8X bandwidth improvement*
20% lesser latency than memory	Large capacities ~GBs
35% Smaller package size*	JEDEC W-IO / HBM standards



*Source: Applied Materials and HMC

Die-stacked DRAM caches

- Meta data overhead – Tag storage
  - Loh-Hill Cache [MICRO '11] propose to co-locate tags and data in same row along with a *missmap* and direct mapped Alloy cache [MICRO '12] uses TAD (tag and data) structure to reduce latency of access.
  - Bimodal cache [MICRO '14] - caches in 64B and 512B block sizes, uses the abundant bandwidth to store meta data a bank in another channel
  - ATCache [PACT '14] stores hot meta-data matches in on chip SRAM.
- Translation piggyback for tag match
  - Tagtables [HPCA '15] organize tags as a hierarchical but flipped page table organization
  - Tagless Fully associative DRAM cache [ISCA '15] translates VA to cache address and moves VA to PA translation off critical path

Objective

- Experimentally measure overheads of paged address translation in x86-64 architecture
  - Determine page walk latency on a TLB miss
  - Assess the effects of caching of the page walk levels in cache hierarchy
  - Impact of page walk on IPC compared to an ideal TLB / address translation.
- Correlate the TLB-reach problem with size and latency of last level die-stacked caches
  - Count percentage of occurrences where TLB miss occurs for the blocks that hit in large LLCs

Experimental Framework

- MARSSx86 – QEMU based full system simulator
  - Added shared L2 TLB and superpage TLB structures
  - Page walk handler - Reduced page walk levels for superpages
  - Flat cache hierarchy for stacked L4 cache
- Unmodified Linux 2.6.38 with THP
- Configuration
  - four 5way OoO core, 4GB memory, x86-64 ISA
  - L1 cache i/d private , 32KB, 2/4 cycles; L2 cache private, 256KB, 6 cycles
  - L3 cache, 4MB shared, 9 cycles
  - Die stacked L4 cache, shared, 64/128/256/512/1024MB, 16way SA, 40 cycles
  - L1 i/d TLB, 64 entries – 4KB pages
  - L2 TLB shared, 1024 entries – 4KB pages
  - L1 dTLB 4 entries – 2MB pages

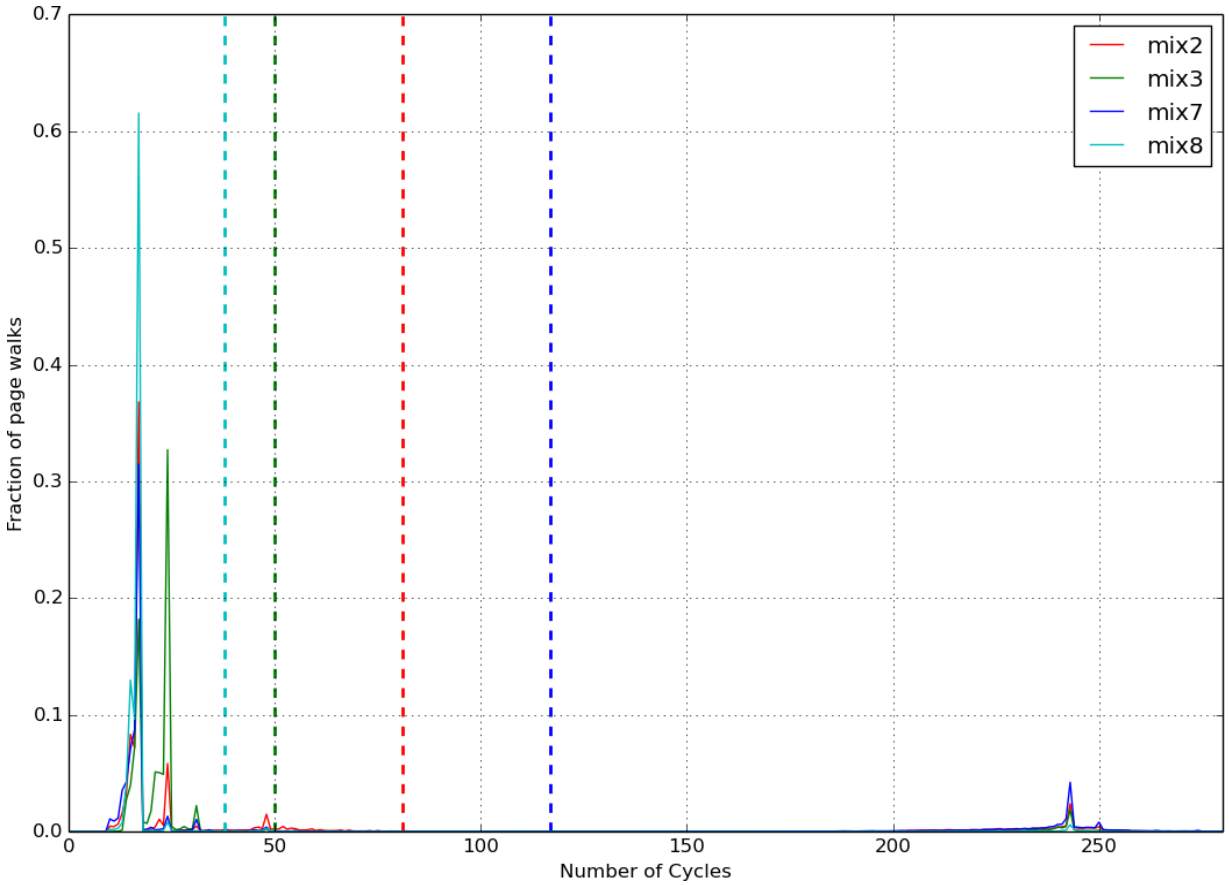
Experimental Methodology

- Multi-programmed
  - SPEC CPU 2006 programs
  - High CPI apps chosen
  - 8 Billion instr fast forward
  - 4 Billion cycles detailed
- Multi-threaded
  - PARSEC - 4 threads
  - ‘simlarge’ input data set
  - Entire ROI detailed mode
  - Apps with unbounded and large working set apps

SPEC CPU 2006 mix	
mix1	milc, mcf, omnetpp, gcc
mix2	GemsFDTD, leslie3d, dealII, soplex
mix3	cactusADM, libquantum, tonto, shinpx3
mix4	lbm, bwaves, zuesmp, sjeng
mix5	milc, GemsFDTD, cactusADM, lbm
mix6	mcf, omnetpp, soplex, leslie3d
mix7	bwaves, astar, zeusmp, gcc
mix8	gobmk, bzip2, h264ref, hmmer

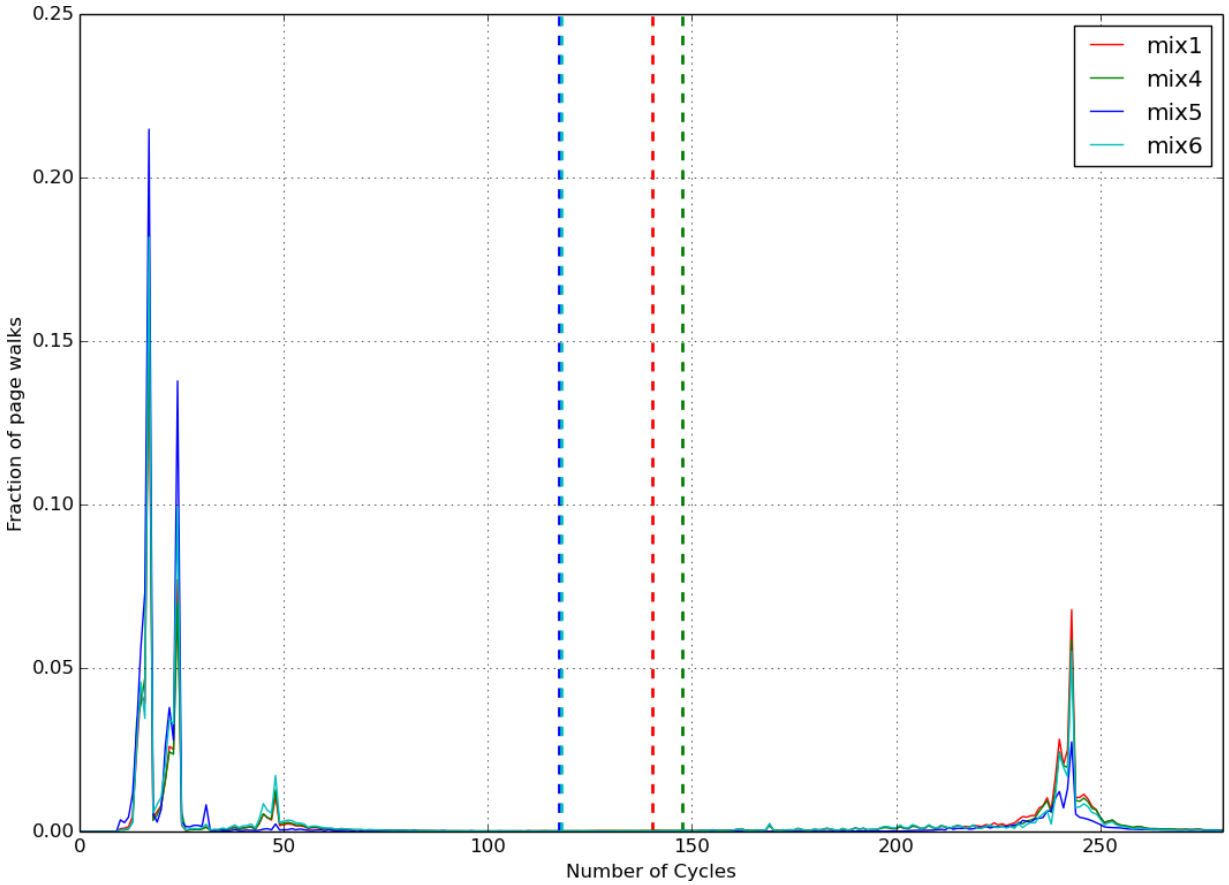
PARSEC 4-threads		
canneal	dedup	ferret
fluidanimate	freqmine	raytrace

Page Walk Latency



Multi-programmed workloads
Average page walk latency ranges from 37 to 147 cycles

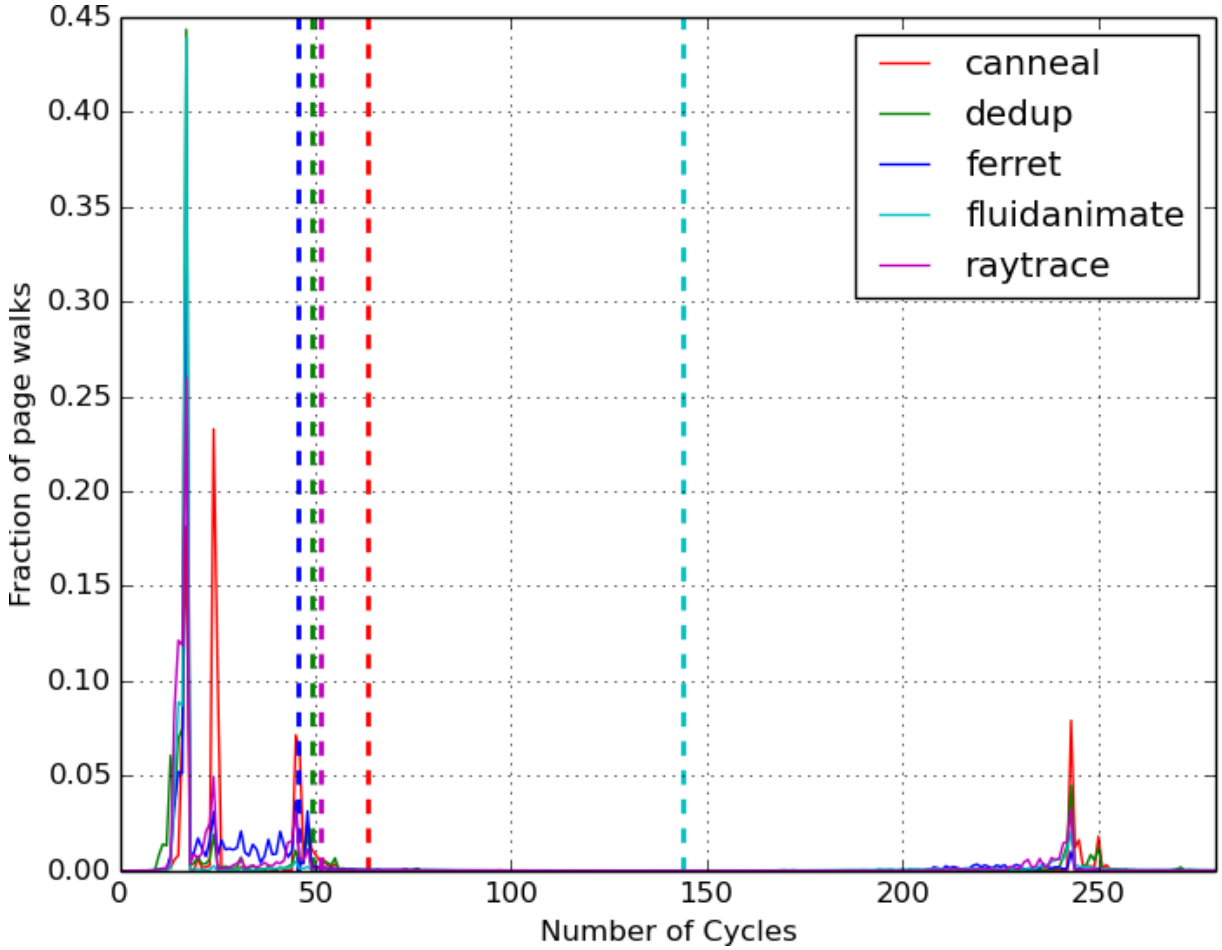
Page Walk Latency



Multi-programmed workloads
Average page walk latency ranges from 37 to 147 cycles

Page Walk Latency range	Count
37-74	2
74-111	1
111-148	5

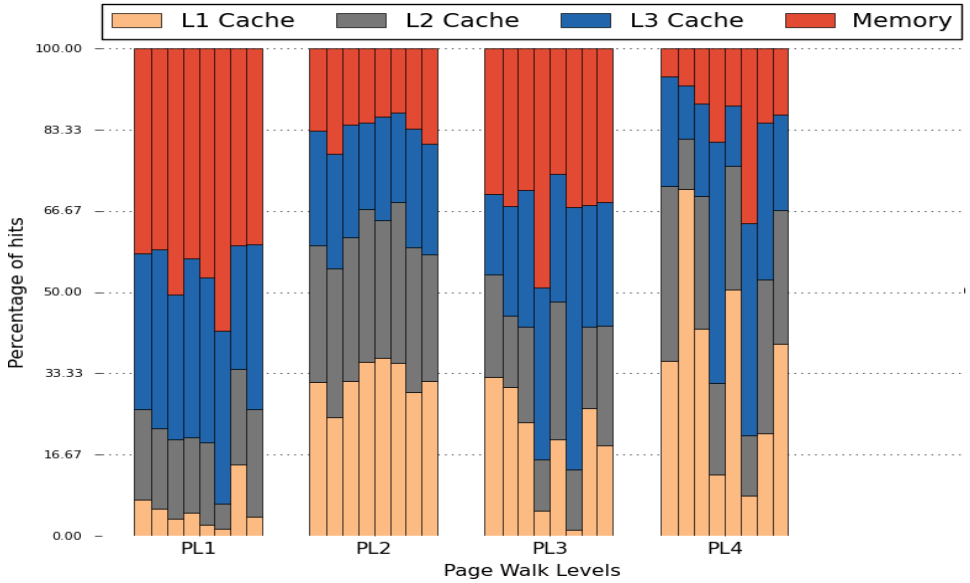
Page Walk Latency



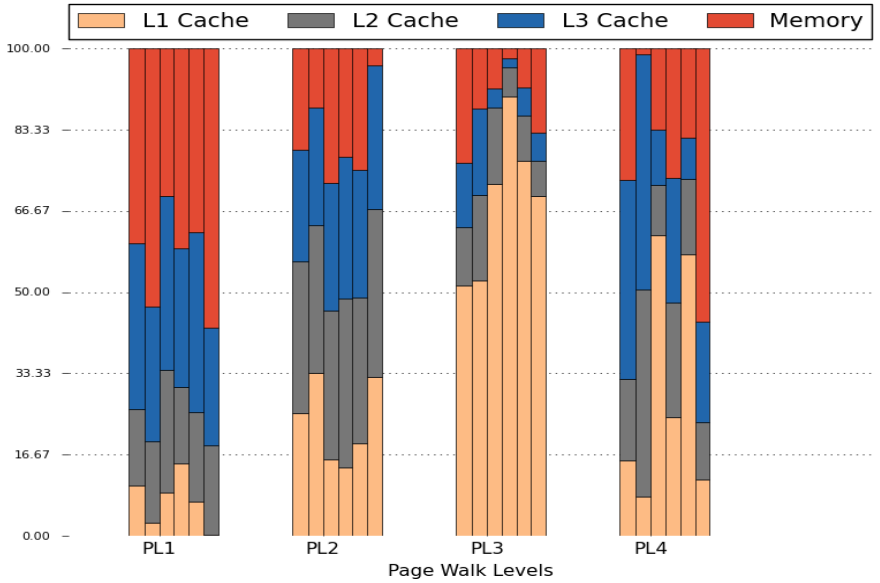
Multi-threaded workloads
Average page walk latency ranges from 18 to 144 cycles

Page Walk Latency range	Count
18-50	3
50-82	2
114-146	1

Page Walk Levels locality



Multi-programmed
mix1 to mix8, left to right

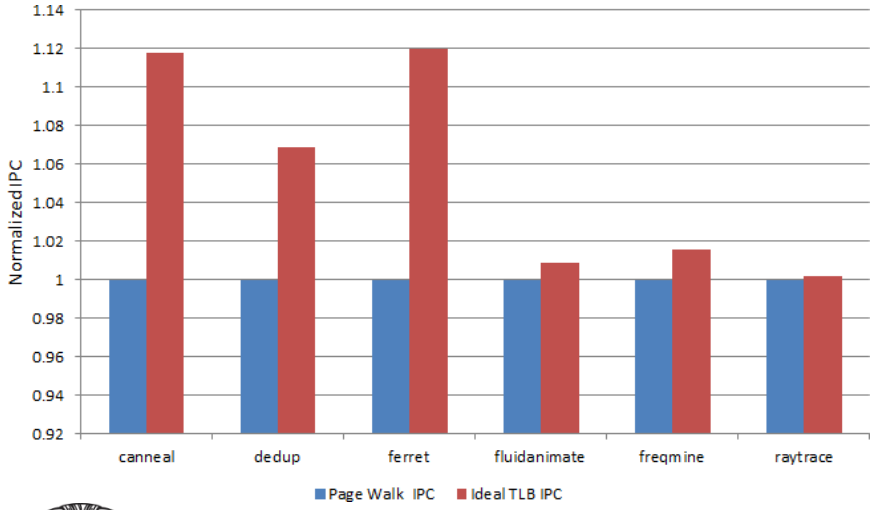


Multi-threaded
canneal, dedup, ferret, fluidanimate, freqmine, raytrace

- PL1 has highest memory access
- PL2 has a uniform hit percentage in almost all cache levels and we observe that most of the 8 translations in a cache line are used
- PL3 level sees around 50 % of hits in either L1 or L2 cache
- Cache pollution - AMD procs perform page walk in L2 cache

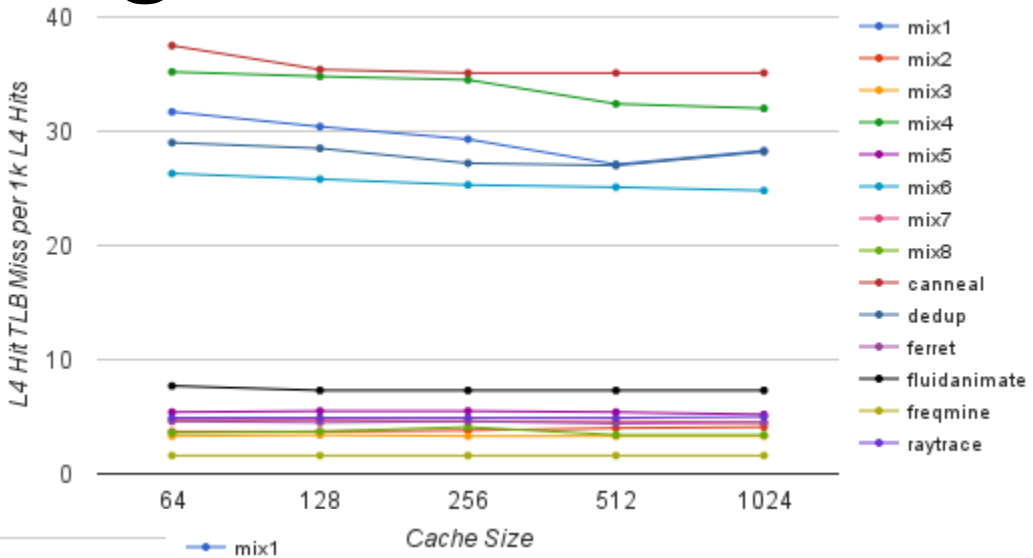
IPC impact due to Page Walks

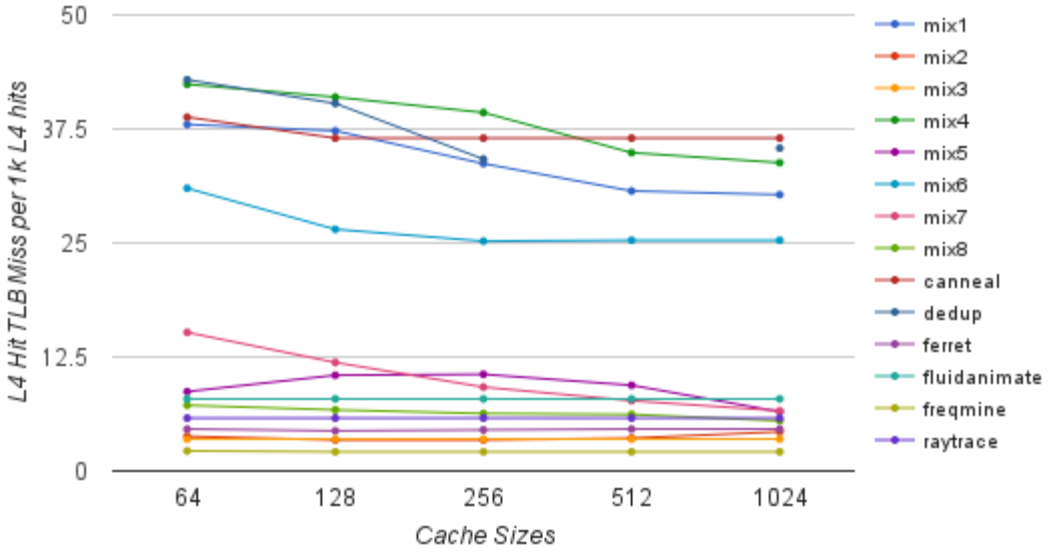
- Modern arch hide memory access latency using ROB and LSQ
- 128 entry ROB and 80 entry LSQ
- Impact on IPC due to page walk latency as compared to an ideal TLB
  - Zero page walk overhead
  - No cache pollution
  - Returns translation in a single cycle



TLB-reach and large LLCs

64B block size L4 cache





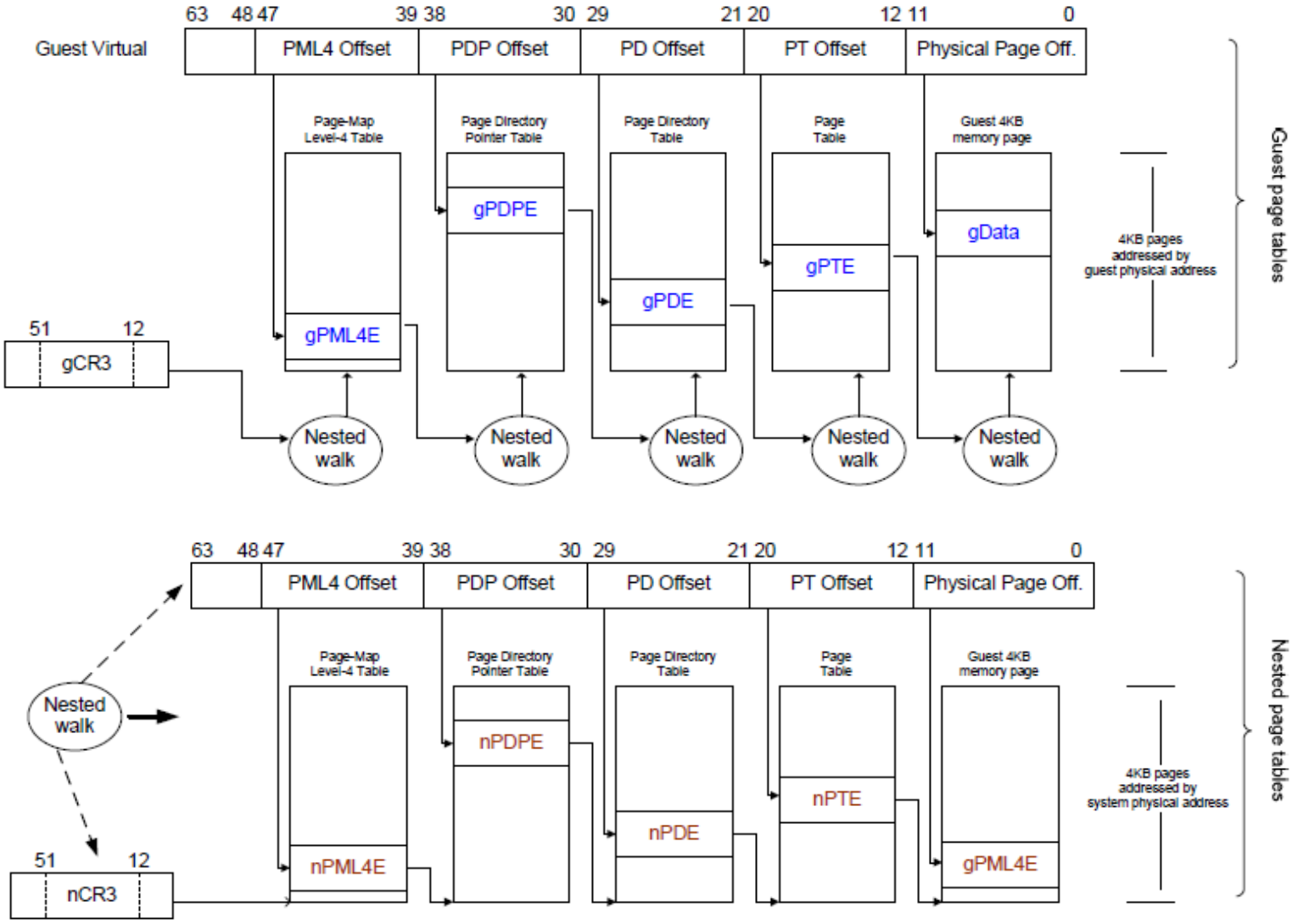
512B block size L4 cache

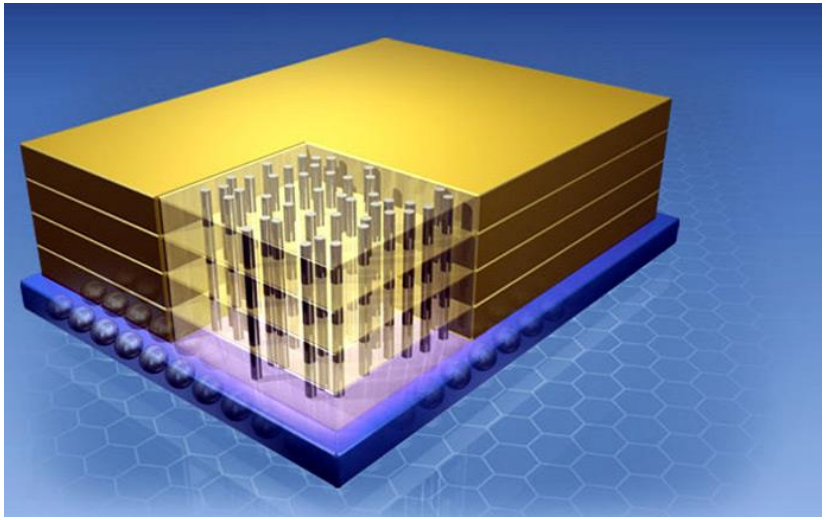
Conclusion

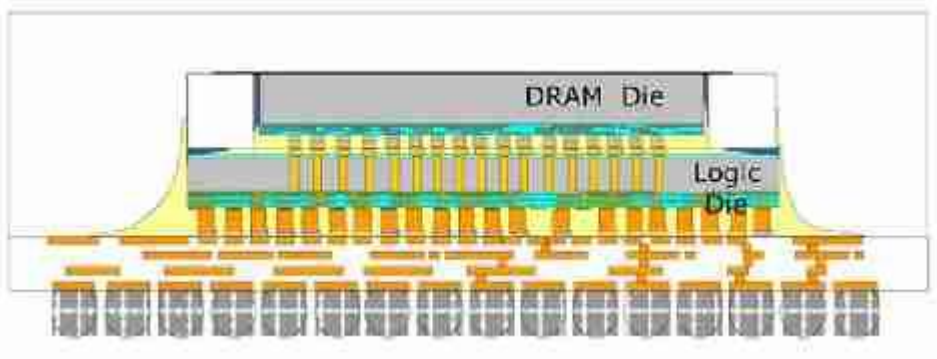
- Conclusion
  - Measure effect of Page walk latency on IPC
  - Framework to study effects of page walk latency and TLB-reach
  - Quantify TLB-reach problem in context of large die stacked caches
- Future Work
  - Using large footprint application in Cloud Suite & Big data bench
  - Detailed timing simulation of die-stacked DRAM
  - Superpage TLBs

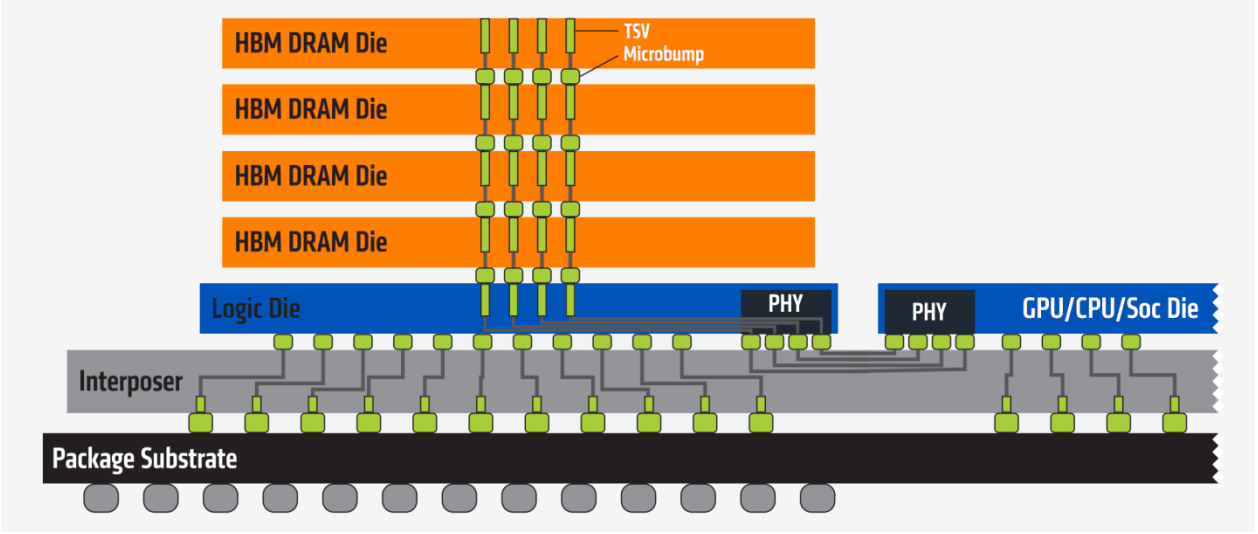

QEMU
page
latency
HBM
DRAMs
caches
superpages
paged
system
translation
TLB
bandwidth
memory
organization
efficiency
performance
die-stacking
x86-64
address
power
full
walk
cache
architecture
multi-threaded
multi-programmed
based
MARSx86
virtualization
simulator
HMC
large
MMU
IPC

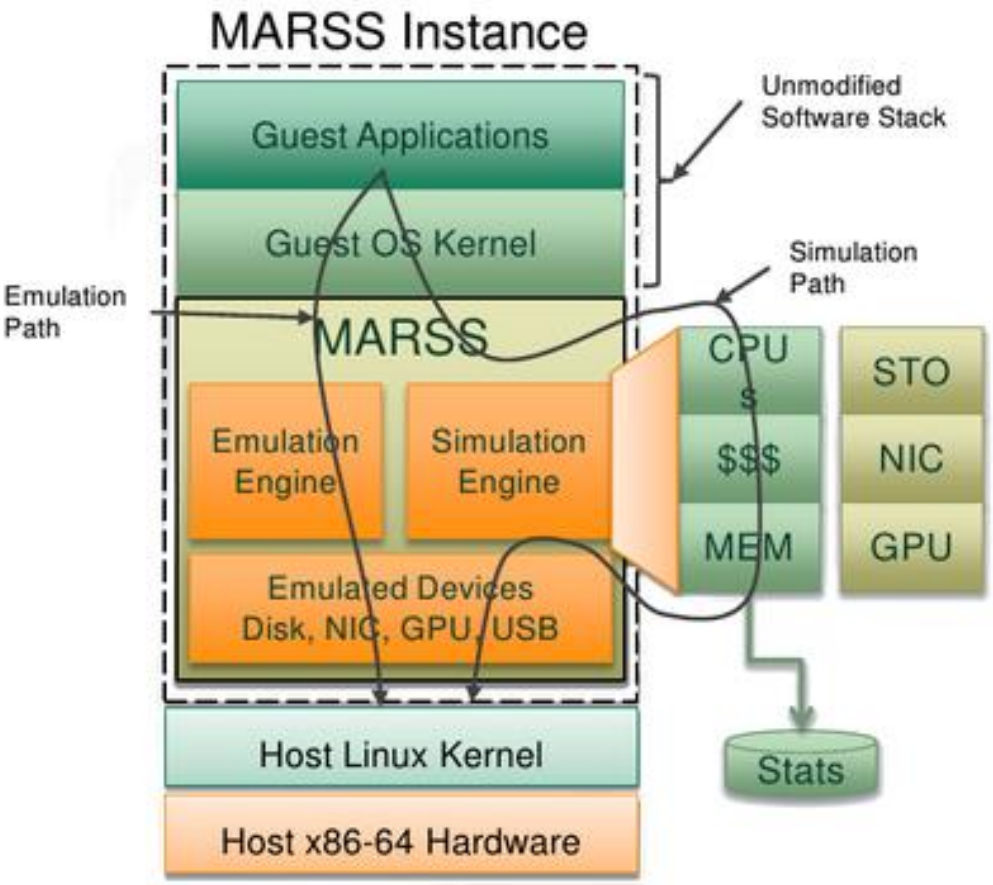
BACKUP SLIDES


63 48 47 39 38 30 29 21 20 12 11 0
Guest Virtual
PML4 Offset
PDP Offset
PD Offset
PT Offset
Physical Page Off.
Page-Map Level-4 Table
Page Directory Pointer Table
Page Directory Table
Page Table
Guest 4KB memory page
gPDPE
gPDE
gPTE
gData
gPML4E
51 12
gCR3
Nested walk
Nested walk
Nested walk
Nested walk
Nested walk
4KB pages addressed by guest physical address
Guest page tables
63 48 47 39 38 30 29 21 20 12 11 0
PML4 Offset
PDP Offset
PD Offset
PT Offset
Physical Page Off.
Page-Map Level-4 Table
Page Directory Pointer Table
Page Directory Table
Page Table
Guest 4KB memory page
Nested walk
nPDPE
nPDE
nPTE
gPML4E
nPML4E
51 12
nCR3
4KB pages addressed by system physical address
Nested page tables




DRAM Die
Logic Die


HBM DRAM Die
HBM DRAM Die
HBM DRAM Die
HBM DRAM Die
TSV
Microbump
Logic Die
PHY
PHY
GPU/CPU/Soc Die
Interposer
Package Substrate


MARSS Instance
Guest Applications
Guest OS Kernel
MARSS
Emulation Engine
Simulation Engine
Emulated Devices
Disk, NIC, GPU, USB
Host Linux Kernel
Host x86-64 Hardware
Unmodified Software Stack
Simulation Path
Emulation Path
CPU s
$$$
MEM
STO
NIC
GPU
Stats