Design, Implementation and Evaluation of the SVNAPOT Extension on a RISC-V Processor

Nikolaos-Charalampos Papadopoulos[1], Stratos Psomadakis[1],
Vasileios Karakostas[2], Nectarios Koziris[1], Dionisios Pnevmatikatos[1]

ncpapad@cslab.ece.ntua.gr

1. National Technical University of Athens – 2. National and Kapodistrian University of Athens

Motivation

The RISC-V SVNAPOT Extension aims to remedy the performance overhead of the Memory Management Unit (MMU) under heavy memory loads, by introducing intermediate Naturally-Power-Of-Two (NAPOT) multiples of the 4K, 2M and 1G page sizes.

i	*pte.ppn*[*i*]	Description	*pte.napot_bits*
0	x xxxx xxx1	8 KiB contiguous region	1
0	x xxxx xx10	16 KiB contiguous region	2
0	x xxxx x100	32 KiB contiguous region	3
0	x xxxx 1000	64 KiB contiguous region	4
0	x xxx1 0000	128 KiB contiguous region	5
...	...	...	...
1	x xxxx xxx1	4 MiB contiguous region	1
1	x xxxx xx10	8 MiB contiguous region	2
...	...	...	...

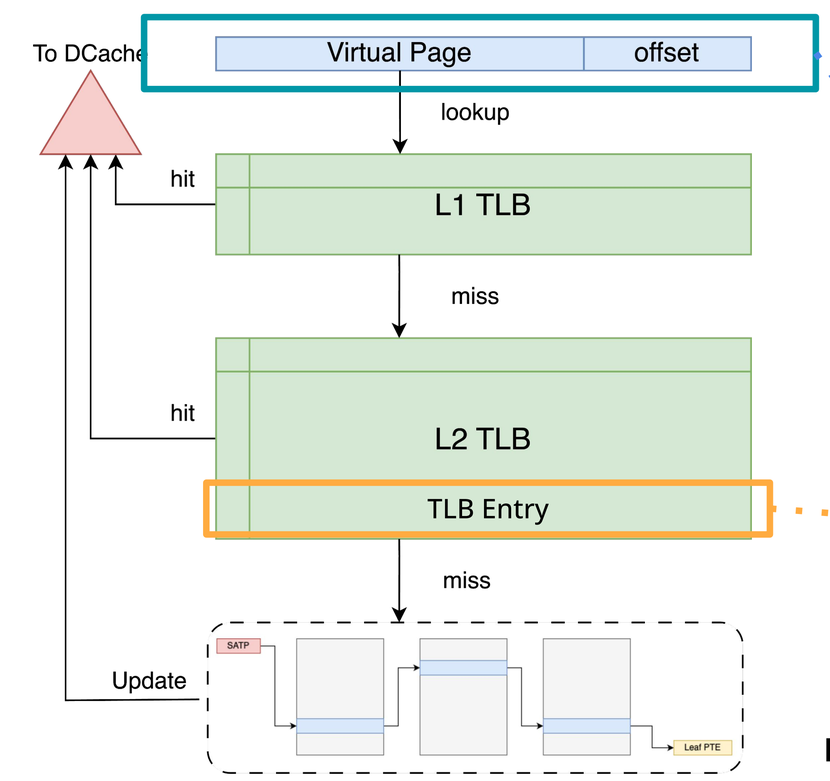
- **64KB is the default candidate as per the RISC-V Privileged Specification**
- **It is also implemented in Linux v6.8 with the MAP_HUGE_64KB flag**

Considerations on the Memory Management Unit

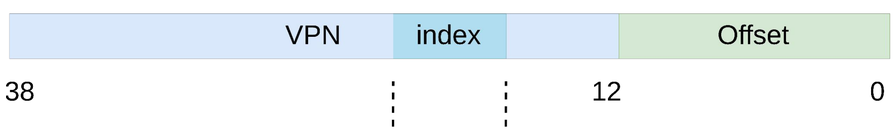
- Small (4K) pages may **stress the MMU & increase TLB misses**
- Huge (2M, 1G) pages may **alleviate TLB stress & reduce TLB misses**
  → but may cause **internal & external fragmentation**

Intermediate page sizes (64K) open up new research opportunities in MMU design trade-offs and offer potential for fine-tuning performance overheads and memory management

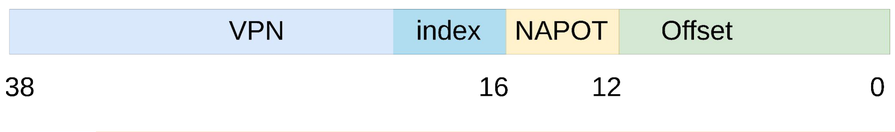
Design & Implementation



Virtual Address Format - 4K Page

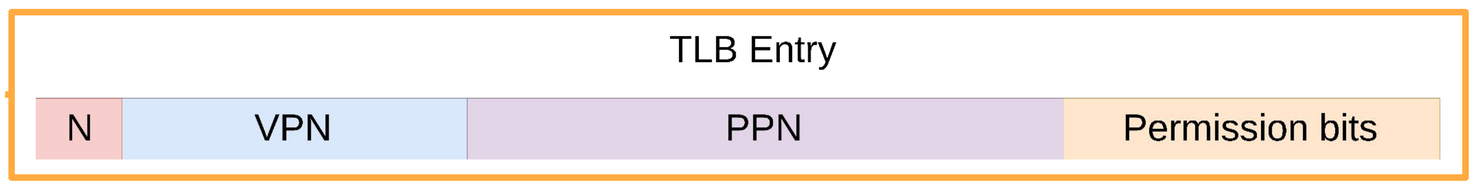


Virtual Address Format - 64K Page



- → Request: either 4K or 64K page
- → 64K page → 16 physically contig. 4K pages
- → 64K page offset: ***16 == 12 + NAPOT (4)***
- → Select index ***(log_2(#sets)+16,16)***
- → **N bit:** distinguish between 4K & 64K pages
- → If a page is 64K, then **N = 1** otherwise **N = 0**
- → Every 64K page **must be aligned** (end in 0000_2), if not → page fault

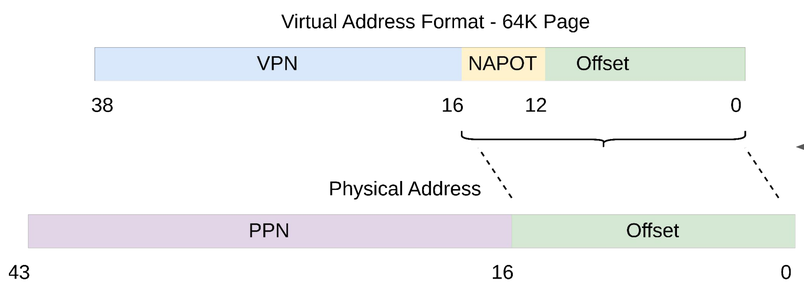
TLB Entry



Translation process

Hit && N == 1

Virtual Address Format - 64K Page

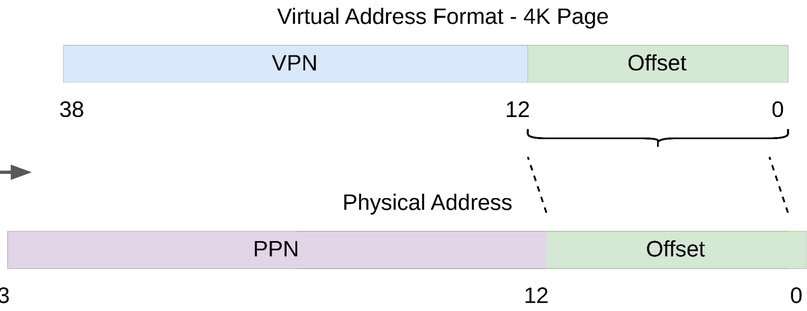


Miss

- Call Page Table Walker (PTW)
- During the PTW it is unknown if the page is 4K or 64K
- OS is responsible to correctly setup the result PPN & N bit

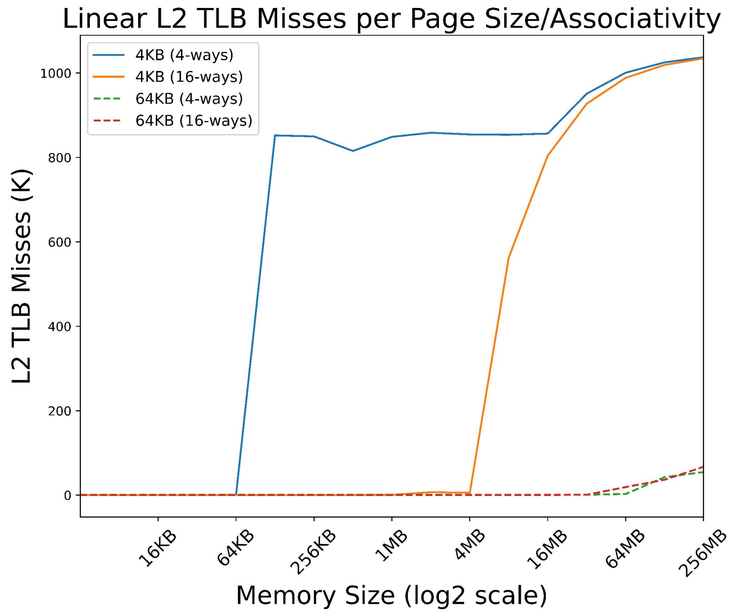
Hit && N == 0

Virtual Address Format - 4K Page



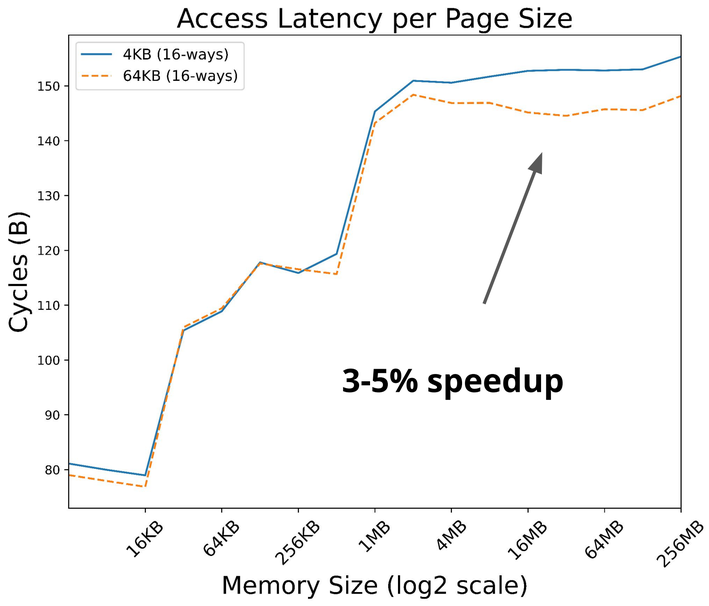
Preliminary Results

Linear L2 TLB Misses per Page Size/Associativity



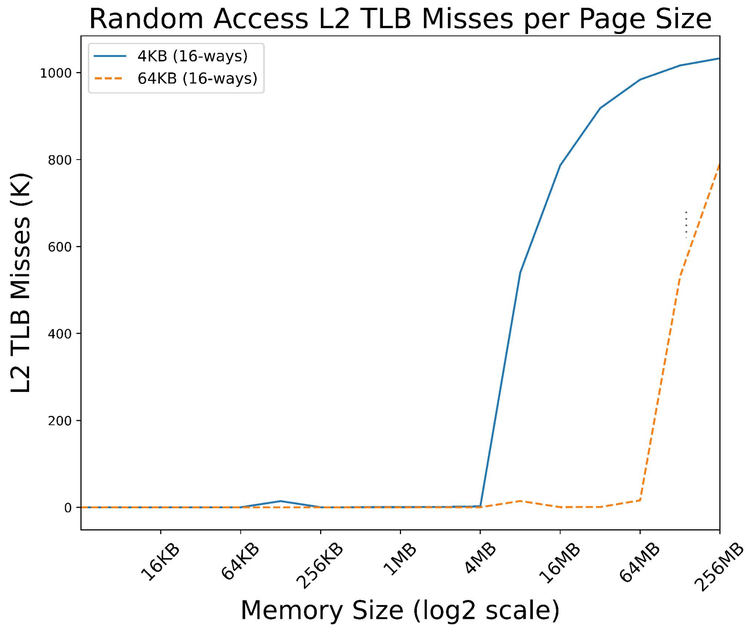
★ **Higher order INDEX selection impacts conflict misses for 4K pages**
★ **Higher associativity may eliminate TLB conflicts**

Access Latency per Page Size



★ **64K pages save 3-5% overall cycles compared to 4K pages due to less PTWs**
★ **One PTW for a 64K page → 16 PTWs for 4K pages**

Random Access L2 TLB Misses per Page Size



★ **64K Pages exhibit up to 16x larger reach**
★ **4K → 4MB reach, 64K → 64MB reach**

This work was supported in part by the EU funded Vitamin-V project #101093062

More info →

RISC-V SUMMIT EU JUNE 24 – 28 | MUNICH 2024