Performance Laws
"Find the bottleneck and go for the throat"

"Count your change" I.e., conversions are not 1-1
- **Cache access, memory-cycle, disk access, network access**
- **Multiplication/division is slower than shift**
- **The new frontier:**
  - **Power**
  - **Size**
  - **Bandwidth**
- **Doubling the voltage, quadruples the power**
- **ROM is smaller than RAM**

"Expect change"

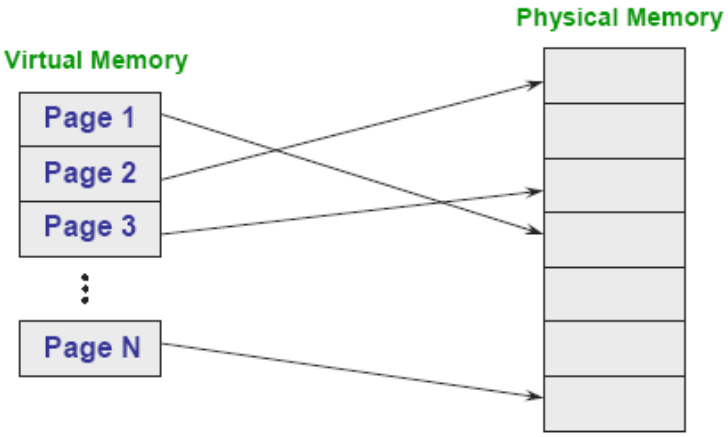
Laws of Happiness
1 "Live for today"
2 "Live for tomorrow"
3 "Make no excuses" "Tell the truth, be nice, be generous"
4 "Try until you succeed" "Decide to be great"
5 "The unexamined life is not worth living ..."
6 "Be courageous"
7 "Surround yourself with nice, smart people"

http://cseweb.ucsd.edu/classes/fa04/cse120/lectures/
http://courses.cs.vt.edu/~cs3204/spring2009/butta/local/lectures/lecture-16.pdf
http://courses.cs.vt.edu/~cs3204/spring2009/butta/local/lectures/lecture-17.pdf

Paging occurs as a hardware memory manager

Paging

- Paging solves the external fragmentation problem by using fixed sized units in both physical and virtual memory



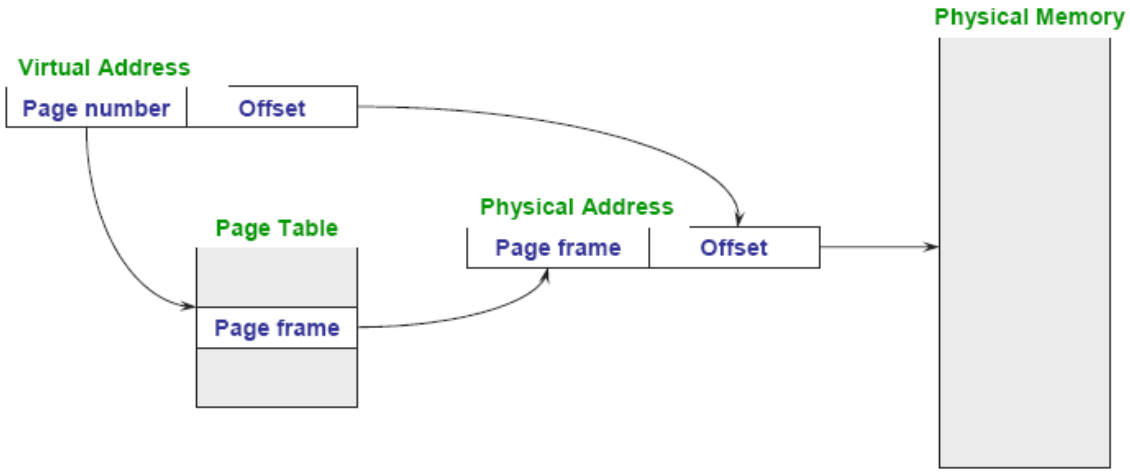
Show logical to physical address translation with a page table

What is a logical address?

What is a physical address?

How does the translation occur?

Page Lookups



How big does the page table need to be?

20-bit address

12 bit page offset (4 kibibyte page size)

8 bit page number

256 pages, each entry is 12 bits

32-bit address

14 bit page offset (16 kibibyte page size)

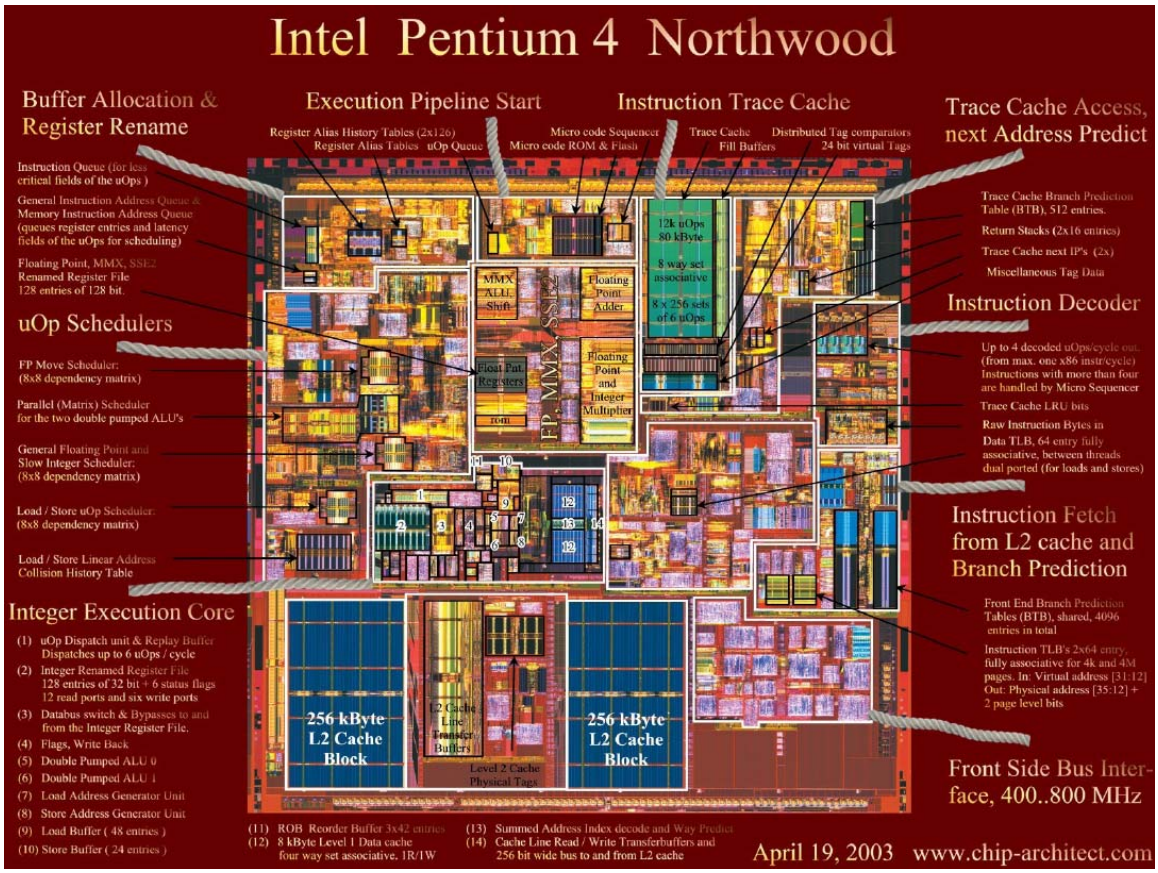
18 bit page number

2^{18} pages, each entry is 14 bits

Page Table Entries (PTEs)

- Page table entries control mapping
  - The Modify bit says whether or not the page has been written
    - » It is set when a write to the page occurs
  - The Reference bit says whether the page has been accessed
    - » It is set when a read or write to the page occurs
  - The Valid bit says whether or not the PTE can be used
    - » It is checked each time the virtual address is used
  - The Protection bits say what operations are allowed on page
    - » Read, write, execute
  - The page frame number (PFN) determines physical page

How do we choose page size?
How fast is it?
How to make it bigger?
Any internal fragmentation?
Any external fragmentation?

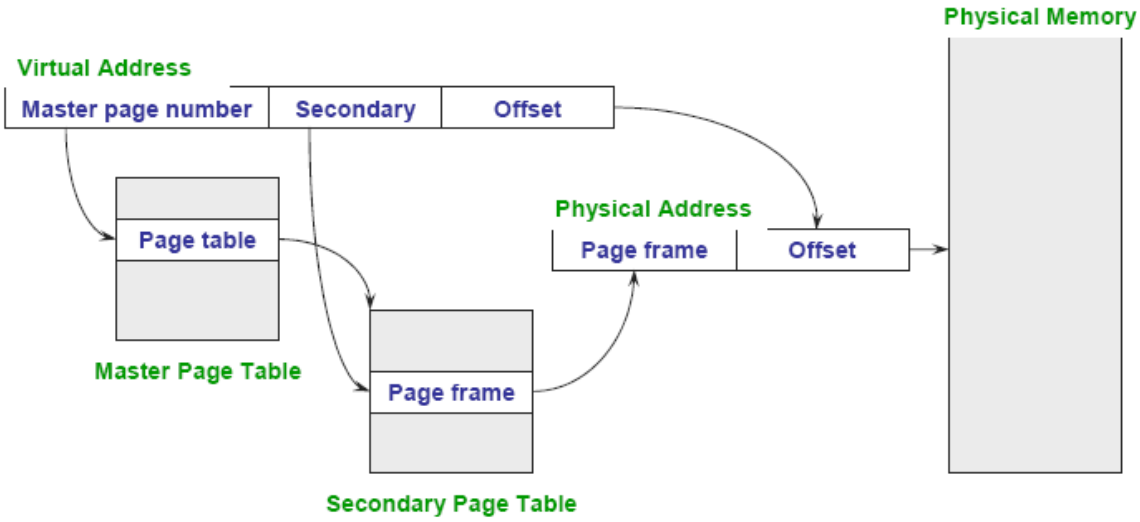


Paging Limitations

- Can still have internal fragmentation
  - Process may not use memory in multiples of a page
- Memory reference overhead
  - 2 references per address lookup (page table, then memory)
  - Solution – use a hardware cache of lookups (more later)
- Memory required to hold page table can be significant
  - Need one PTE per page
  - 32 bit address space w/ 4KB pages = 2^{20} PTEs
  - 4 bytes/PTE = 4MB/page table
  - 25 processes = 100MB just for page tables!
  - Solution – page the page tables (more later)

Concepts
Logical address to physical address translation
Page table
Input is logical page number (offset into table)
Output, contents is the physical page number
Two-level paging

Two-Level Page Tables



November 4, 2004 CSE 120 – Lecture 10 – Paging 5
© 2004 Geoffrey M. Voelker

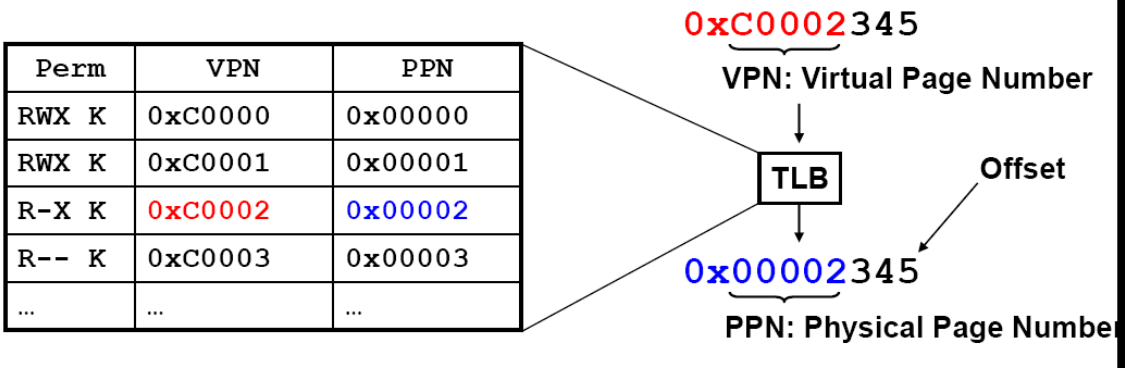
Content addressable memory (CAM)
Define CAM: what are inputs and outputs?
Compare to RAM: what are inputs and outputs?

Efficient Translations

- Our original page table scheme already doubled the cost of doing memory lookups
  - One lookup into the page table, another to fetch the data
- Now two-level page tables triple the cost!
  - Two lookups into the page tables, a third to fetch the data
  - And this assumes the page table is in memory
- How can we use paging but also have lookups cost about the same as fetching from memory?
  - Cache translations in hardware
  - Translation Lookaside Buffer (TLB)
  - TLB managed by Memory Management Unit (MMU)

TLB: Translation Look-Aside Buffer

- Virtual-to-physical translation is part of every instruction (why not only load/store instructions?)
  - Thus must execute at CPU pipeline speed
- TLB caches a number of translations in fast, fully-associative memory
  - typical: 95% hit rate (*locality of reference principle*)



Perm	VPN	PPN
RWX K	0xC0000	0x00000
RWX K	0xC0001	0x00001
R-X K	0xC0002	0x00002
R-- K	0xC0003	0x00003
…	…	…

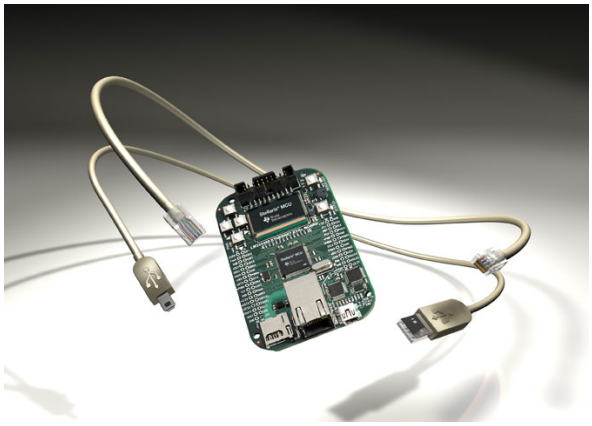
Translation lookaside buffer

CAM holding a subset of logical page numbers
RAM holding the physical page numbers for that page
Hit/miss (is the logical page number in TLB?)
Hit, use stored physical page number from RAM
Miss, access memory for regular page table

Locality of reference
Previous memory accesses can be used to predict future accesses

How can the memory manager increase effective size of memory?
How can the memory manager increase reliability/robustness?

EK-LM3S6965



MPU Region Base Address (MPUBASE)

Base 0xE000.E000
Offset 0xD9C
Type R/W, reset 0x0000.0000

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ADDR															
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ADDR											VALID	reserved	REGION		
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	WO	RO	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit/Field	Name	Type	Reset	Description
31:5	ADDR	R/W	0x0000.000	Base Address Mask Bits 31:*N* in this field contain the region base address. The value of *N* depends on the region size, as shown above. The remaining bits (*N*-1):5 are reserved. Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.

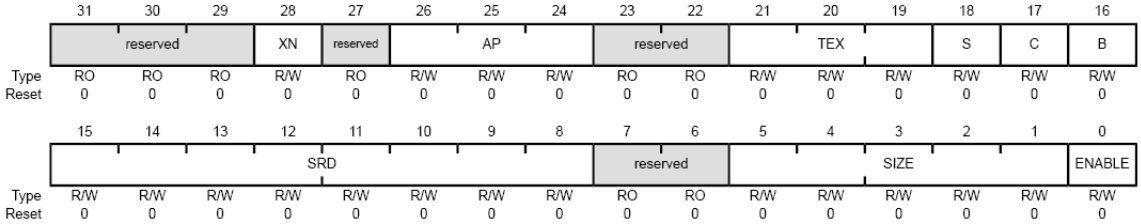
Table 3-9. Example SIZE Field Values

SIZE Encoding	Region Size	Value of N[a]	Note
00100b (0x4)	32 B	5	Minimum permitted size
01001b (0x9)	1 KB	10	-
10011b (0x13)	1 MB	20	-
11101b (0x1D)	1 GB	30	-
11111b (0x1F)	4 GB	No valid ADDR field in **MPUBASE**; the region occupies the complete memory map.	Maximum possible size

a. Refers to the N parameter in the **MPUBASE** register (see page 153).

MPU Region Attribute and Size (MPUATTR)

Base 0xE000.E000
Offset 0xDA0
Type R/W, reset 0x0000.0000



28 XN R/W 0 Instruction Access Disable

Value	Description
0	Instruction fetches are enabled.
1	Instruction fetches are disabled.

26:24 AP R/W 0 Access Privilege
For information on using this bit field, see Table 3-5 on page 101.

Table 3-5. AP Bit Field Encoding

AP Bit Field	Privileged Permissions	Unprivileged Permissions	Description
000	No access	No access	All accesses generate a permission fault.
001	R/W	No access	Access from privileged software only.
010	R/W	RO	Writes by unprivileged software generate a permission fault.
011	R/W	R/W	Full access.
100	Unpredictable	Unpredictable	Reserved.
101	RO	No access	Reads by privileged software only.
110	RO	RO	Read-only, by privileged or unprivileged software.
111	RO	RO	Read-only, by privileged or unprivileged software.

Table 3-3. TEX, S, C, and B Bit Field Encoding

TEX	S	C	B	Memory Type	Shareability	Other Attributes
000b	x[a]	0	0	Strongly Ordered	Shareable	-
000	x[a]	0	1	Device	Shareable	-
000	0	1	0	Normal	Not shareable	Outer and inner write-through. No write allocate.
000	1	1	0	Normal	Shareable	
000	0	1	1	Normal	Not shareable	
000	1	1	1	Normal	Shareable	
001	0	0	0	Normal	Not shareable	Outer and inner noncacheable.
001	1	0	0	Normal	Shareable	
001	x[a]	0	1	Reserved encoding	-	-
001	x[a]	1	0	Reserved encoding	-	-
001	0	1	1	Normal	Not shareable	Outer and inner write-back. Write and read allocate.
001	1	1	1	Normal	Shareable	

What is paging?

- Virtual to physical translation

Why is it important?

- Virtual memory
- Protection of one process from another
- Protection of the operating system
- Removes external fragmentation (adds internal fragmentation)