Review: (Write on the board before class)
Q1: Where can a block be placed in the upper level? (*Block placement*)
Q2: How is a block found if it is in the upper level? (*Block identification*)
Q3: Which block should be replaced on a miss? (*Block replacement*)
Q4: What happens on a write? (*Write strategy*)
Memory organizations; SRAM vs. DRAM

Break:
Survey sheets
Meeting times are moday (bring signup sheet?)
Talk by Bob Miller yesterday of MIPS
one IBM salesman: 3090 + VF = $9M + $6M to run it
another IBM salesman: POWERsystem 540: = $100,000
=> $14,900,000 to do software conversion
Next Assignment: Tentative on computer account 8.7 & 8.8
Discussion questions for class: 8.12, 8.22, 8.23, 8.24
Pizza & beer Wednesday 11:30-1:30 $5 (make reservation)

Today: memory & VM

DRAM specific
Square die
latch square root of size
Transfer from latch at much higher rate
Figure 8.20, p. 432

Virtual Memory
How many wrote programs without VM? solution?
block=page or segment
miss=address fault (
Hit = 1 to 10 clocks, miss = 100,000 to 500,000 clocks
Q1: Where placed? What is it called? FA. Why?
Q2: How is it found? Page table
Q3: Which replaced? LRU; computer needs?
Q4: What on write? WB

Fast address translation: is cache works for data & instructions, cache must work for **addresses** of data & instructions
=> TB or TLB
Q1,Q2,Q3: just like caches
Q4: rare, so flush TLB after change page table

Speed? TLB in parallel with data access to cache: tag uses translated address.

Protection Examples:
Process: multiprogramming means many at one time
1)can't afford to wait for page so switch
2) can't afford to copy full process memory, so protection to leave in physical memory but prevent access

Base & bound is simplest; protection so user can't change base & bounds

VAX page example:
process & system (bit 31)
process => P0 and P1 (bit 30)
Base & bound per 3 areas, allowing separate page tables
See 8.27
See 8.29: just like the VAX-11/780 data cache except for size (also divide into system/process bit: limits flushing of system by user, but also limits user or system TLB size)

Read Intel 80286 Example on your own

More examples
Instruction Prefetch Buffer
Variable sized instructions, when instruction memory bandwdith higher than CPU can consume instructions
See 8.31
FIFO
Register Windows
See 8.32, 8.33
8.35 shows that can reduce stores a lot, loads some
Compiler differences can overcome these differences

Putting it all together on the VAX

8.48

Ask qeustions what happens after a branch

1: PC virtual address sent to TLB

2: process/system bit + 5 bit index access both TLB banks

3: valid bit + page address (- 5 bits) compared

4: hit returns physical page address via 2:1 MUX

5: 2 CPU registers get virtual page address (VIBA) & physical address (PIBA)

6: physical address goes to cache; same steps occur

7: prefetch unit loaded with 4 bytes, with valid bits set; PIBA then asks for next 4 bytes

8: assume this instruction writes memory

9: address to TLB set dirty bit (trap in microcode if not set)

10: as before, except update cache block

11: write through, so update cache and send to write buffer

2.1 CPI out of 10.6 (OK out of 10; not so great from base of 1)