2 - Fundamental Concepts

Fundamental Concepts

1. What are three key components in a computer?

   Computation, communication, memory storage

2. Describe the two key properties in the Von Neumann Model

   a. Stored program

      Instructions stored in a linear memory array

      Memory is unified between instructions and data

   b. Sequential instruction processing

      One instructions processed at a time

      Program counter identifies the current instructions

      Program counter is advanced sequentially except for control transfer instructions

3. What are the differences between Von Neumann Model and the Dataflow model?

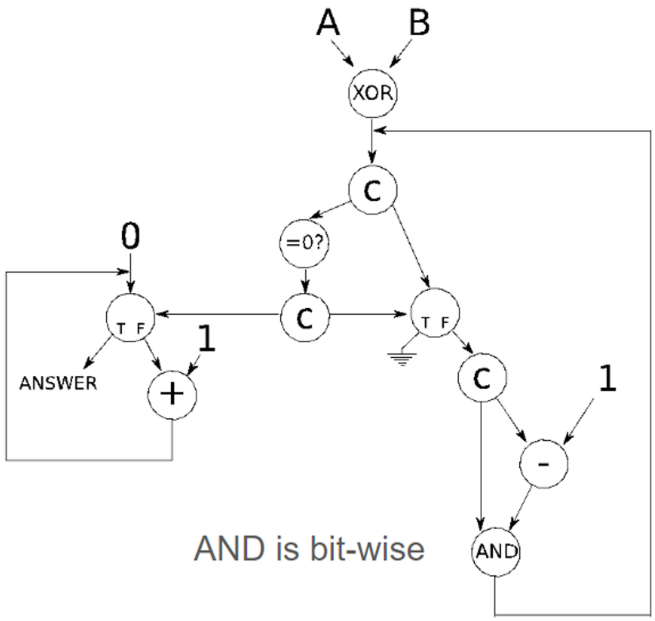
   In a Von Neumann model, the instructions are processed sequentially (in control flow order).

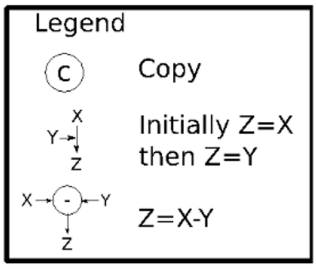
   In a dataflow model, the instructions are processed in data flow order. The instructions are fired as soon as the inputs are ready.

   In a Von Neumann model, the instructions are **sequential** (as specified by the instruction pointer) unless there is explicit control flow instruction.

   In a dataflow model, the instruction ordering is specified by **data flow dependence**. Each instruction specifies “who” should receive the result. An instruction can “fire” whenever all operands are received. Potentially many instructions can execute at the same time.

4. What algorithm does this dataflow represent?





Hamming distance.

Note: (X &= X - 1) clears the rightmost set bit

3 - ISA I

Instruction Set Architecture (ISA)

1. What is an ISA?

   Agreed upon interface between software and hardware. What the software writer needs to know to write system/user programs.

2. What is a microarchitecture?

   Implementation of ISA under constraints and specifications. Not visible to the software.

3. What are the differences between ISA and microarchitecture?

   ISA is exposed at the software level. Programmers need to know the ISA to write instructions.

   Microarchitecture is hidden to the programmers. It is related to the implementation of the ISA.

4. Why is data flow model said to be data-driven?

   The only real dependencies are the data.

   Nodes are operators. Arcs are data (I/O). No sequential I-stream and program counter. Execution triggered by the presence of data.

5. Advantage and disadvantage of stack machine?

   Advantage: short instructions ⇒ compact code

   Disadvantage: some computation is difficult to implement on stack machines

6. Convert (12 + 45) * 98 into instructions in a stack machine.

```
PUSH 98
PUSH 12
PUSH 45
ADD
MUL
```

7. What is the semantics of the following code?

```
PUSH B
PUSH X
ADD
POP C   ; C = B + X
PUSH C
PUSH Y
SUB
POP Z   ; Z = Y - C
```

4 - ISA II

ISA Elements & Tradeoffs

1. What are the advantages and disadvantages of having more or high-level data types in the ISA?

   Advantage: less work for programmers, denser code

   Disadvantage: more work for architects

2. Why is having registers a good idea?

   For temporal locality, data is likely accessed multiple times during a short interval time so it is better to have an access that is faster than memory

3. Describe the three instruction classes
   a. Operate instructions

      For arithmetic computations

   b. Data movement instructions

      Move data between registers and memory. Load or store

   c. Control flow instructions

      Control the PC counter

      Change the sequence of instructions that are executed

4. Describe the addressing modes:
   a. Absolute

      LW rt, 10000

      Use 10000 as address

   b. Register indirect

      LW rt, (r_{base})

      Use the value in r_{base} as address

   c. Displaced or based

LW rt, offset(r_{base})

Use the value in r_{base} plus an offset as address.

d. Indexed

LW rt, (r_{base}, r_{index})

Use the value in r_{base} plus the value in r_{index} as address

e. Memory indirect

LW rt ((r_{base}))

Double dereference (pointer to pointer)

f. Auto inc/dec

LW rt, (r_{base})

Use the value in r_{base} as the starting address and increment/decrement every time

5. Describe the interfaces with I/O devices. Which one is more general purpose?

   a. Memory mapped I/O

      Specific region in memory mapped for I/O

   b. Special I/O instructions

      Special in/out instructions

   Memory mapped I/O is more general purpose.

6. What are the advantages and disadvantages of complex instructions?

   Advantages: more complex instructions, denser code, smaller code size, simpler compiler

   Disadvantages: hardware architects need to do more work

7. What are the advantages and disadvantages of smaller semantic gap?

   Advantages: more complex instructions, denser code, smaller code size, simpler compiler

   Disadvantages: hardware architects need to do more work

8. What is the purpose of a translation layer?

A **translation layer** translates from one ISA to another to reduce the semantic gap between high-level language and ISA

5 - ISA & Intro to Microarchitecture

Instruction-level Tradeoffs

1. Advantages and disadvantages of fixed length vs. variable length?

   Fixed length: easier to decode multiple instructions parallelly, wasted bits

   Variable length: more compact code, easier to add new instructions but harder to decode multiple instructions parallelly, also more complex logic

2. Advantages and disadvantages of uniform code vs. non-uniform code?

   Uniform code: easier to decode multiple instructions parallelly

   Non-uniform code: more compact and powerful instruction format, more complex decode logic

3. Advantages and disadvantages of more registers?

   More registers: less load or store but larger register files

   Also larger instruction size

4. What is an addressing mode?

   Addressing mode specifies how to obtain an operand of an instruction

5. Advantages and disadvantages having more addressing modes?

   More flexibility? Using the most suitable addressing mode can reduce the code size and increase efficiency. However this means more work for architectures

Microarchitecture Basics

1. What is the difference between ISA and Microarchitecture from the perspective of two architectural states AS and AS'?

   ISA specifies how to transform AS to AS'

In a microarchitecture, there may be multiple states from AS to AS'

In a microarchitecture, there can be multiple programmer-invisible states to optimize the speed of instruction execution

2. In a single-cycle machine, what determines the cycle time?

   The slowest instruction

   Each instruction takes a single clock cycle

3. In a multi-cycle machine, what determines the cycle time? How does it compare with a single-cycle machine?

   The slowest stage; should be faster than single-cycle machine

4. What are the two components of an instruction processing engine? What are the three components of the first component?

   Datapath and control logic. The three components of datapath are functional units, hardware structures and storage units

5. How to calculate the execution time of an instruction?

   CPI $\times$ cycle time

Single-Cycle Microarchitecture

1. What are the five steps?

   IF, ID, EX, MEM, WB

2. Describe the semantics of these instructions

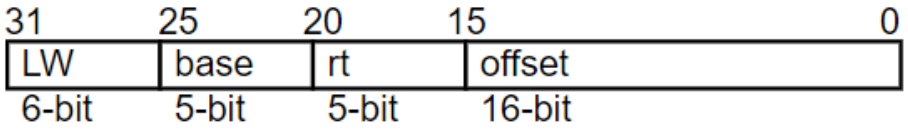
   a. R-Type

      rd ← rs + rt

   b. I-Type

      rt ← rs + sign_extend(immediate)

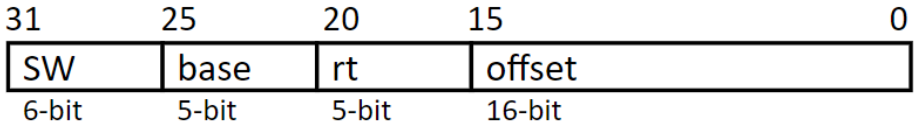
   c. Load



I-type

rt ← M[base + offset]

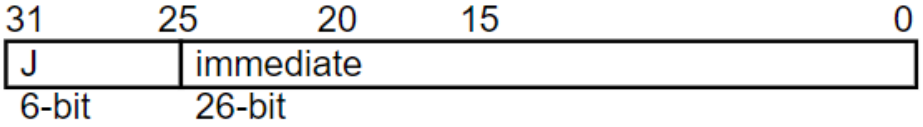
d. Store



I-type

M[base + offset] ← rt

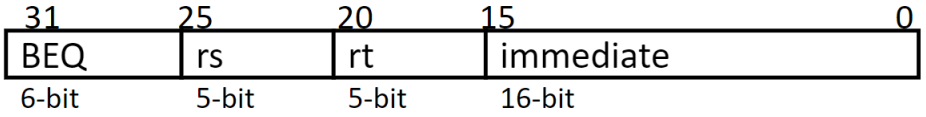
e. Unconditional Jump



J-type

PC ← PC[31:28] + immediate << 2

f. Conditional Branch



I-type

if rs == rt:

PC ← PC + 4 + sign_extend(immediate)

3. Describe the semantics of control signals

a. RegDest

0 for selecting 20-16 as the write register

1 for selecting 15-11 as the write register

b. ALUSrc

0 for selecting value as read register 2

1 for selecting value from the sign-extended immediate

c. MemToReg

   For load instructions

d. RegWrite

   For R-Type and load instructions

e. MemRead

   For load instruction

f. MemWrite

   For store instruction

g. PCSrc1

   For jump instructions

h. PCSrc2

   Branch and Zero

6 - Single Cycle & Pipelining

Single-Cycle Microarchitecture

1. Assume
   - memory units (read or write): 200 ps
   - ALU and adders: 100 ps
   - register file (read or write): 50 ps
   - other combinational logic: 0 ps

   Fill in the table

steps	IF	ID	EX	MEM	WB	Delay
resources	**mem**	**RF**	**ALU**	**mem**	**RF**	
R-type	200	50	100		50	400
I-type	200	50	100		50	400
LW	200	50	100	200	50	600
SW	200	50	100	200		550
Branch	200	50	100			350
Jump	200					200

2. Why is single-cycle microarchitecture inefficient? Give 3 reasons

   Some instructions do not have to go through all stages, but the cycle time is determined by the slowest instruction.

   Must provide worst-case combinational resources in parallel as required by any instruction

   Need to replicate a resource if it is needed more than once by an instruction during different parts of the instruction processing cycle

3. What are the following design principles?
   a. Critical path design

      Focus on the parts with maximum latency

   b. Bread and butter (common case) design

      Focus on cases that are the most common

   c. Balanced design

      Balance instruction / data flow through hardware components

Pipelining

1. Describe the basic idea of pipelining

   Divide the instructions into multiple stages so that multiple instructions in different stages can be processed at the same time

2. Describe the three principles of an ideal pipeline
   a. Repetition of identical operations

The same operation is repeated on a large number of different inputs

b. Repetition of independent operations

   The stages are independent of each other. There are no dependencies

c. Uniformly partitionable suboperations

   The stages have the same latency.

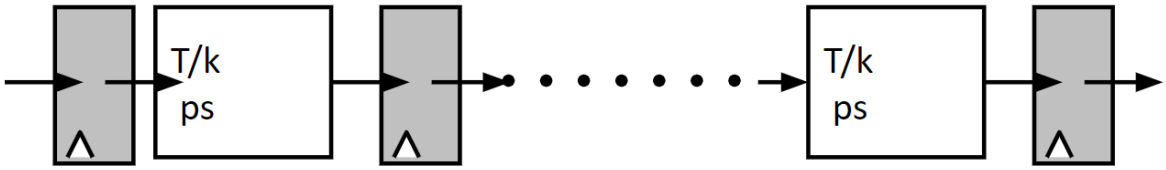
7 - Control & Data Dependency

Pipelining

1. Calculate the bandwidth where S = latch delay

BW = $\frac{1}{T+S}$

2. Calculate the bandwidth where S = latch delay



BW = $\frac{1}{T/k+S}$

3. What is a disadvantage of pipelining?

   Additional hardware cost and overhead.

4. Why is the instruction pipeline not an ideal pipeline, in terms of

   a. Identical operations?

      Not all instructions go through the same stages

      ⇒ external fragmentation

   b. Uniform suboperations?

      The stages do not have the same latency

      ⇒ internal fragmentation

   c. Independent operations?

      There can be data dependencies.

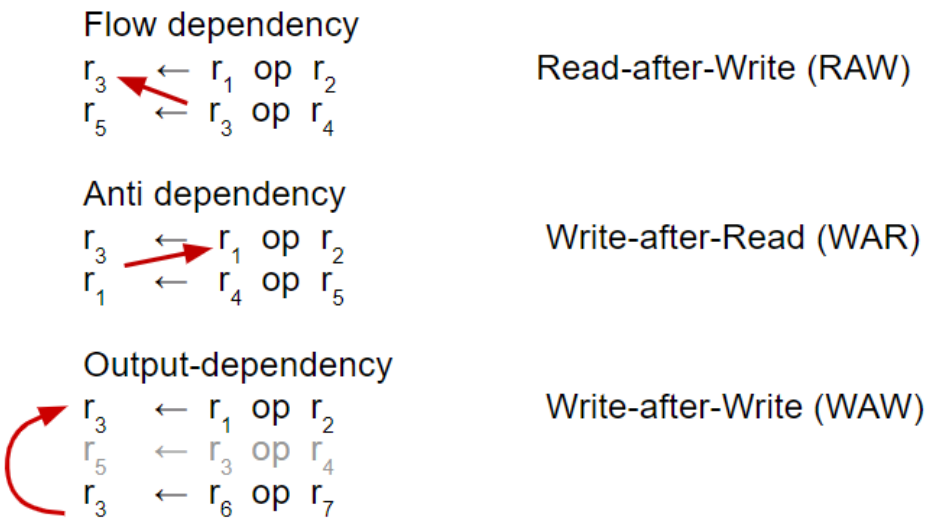
⇒ pipeline stalls

5. What are the three types of data dependencies? Which one is a true data dependency?

   The three types of data dependencies are **flow dependency** (RAW), **anti dependency** (WAR), **output dependency** (WAW)

   Flow dependency is the only true dependency.

   Anti and output dependencies exist due to a limited number of architectural registers.



6. Describe the following and list the advantages and disadvantages

   a. Scoreboarding

      Each register has a valid bit. When there is a write to register, the bit will be reset. An instruction in ID stage needs to check if the source and destination registers are valid.

      Advantage: easy to implement, each register only needs one bit

      Disadvantage: stalls for all types of dependencies, not only flow dependency

   b. Combinational dependency check logic

      Special logic that checks if any instruction in later stages is supposed to write to any source register of the instruction that is being decoded

Advantage: no need to stall for output and anti dependencies

Disadvantage: complex logic

7. Describe the idea of data forwarding

   Instructions in later stages feed the values to consumers before reaching the WB stage

   Additional dependency check logic and data forwarding paths (buses) to supply the producer's value to the consumer right after the value is available

8. Describe the difference between static scheduling and dynamic scheduling

   Static scheduling: the compiler orders instructions and hardware executes them

   Dynamic scheduling: the hardware can execute instructions out of compiler order

8 - Control Dependency & SIMD

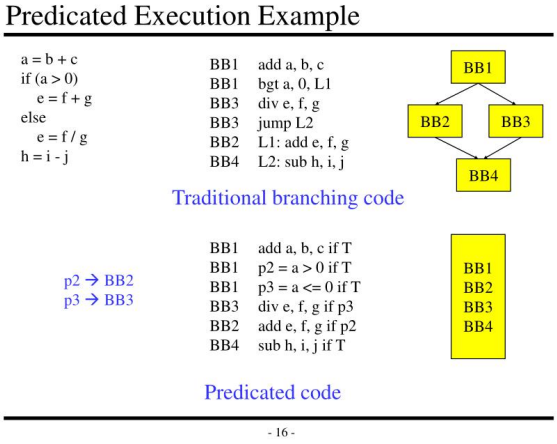
Control Dependency

1. Draw the pipeline for stall fetch (stall the pipeline until we know the next fetch address)
2. Describe the following and their advantages and disadvantages
   a. Predicated execution

      Eliminate branch ⇒ only straight line code

      Advantages: no need for branch prediction, more compiler optimization opportunities

      Disadvantages: may fetch useless instructions. Some instructions fetched/executed but discarded

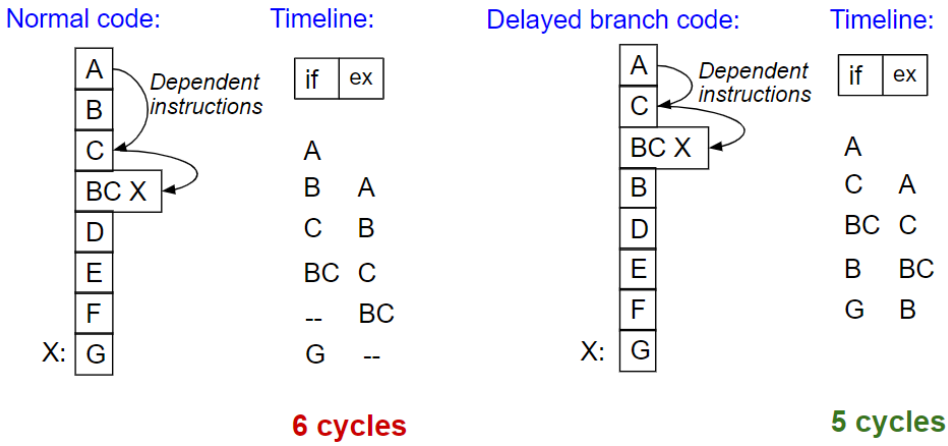


   b. Delayed branching

      Find instructions that do not have an dependencies to fill in the delay slots of branch instructions. Easier to find such instructions for unconditional branch

Advantages: do useful work during the delay slot ⇒ eliminate the delay slots, assuming number of delay slots = number of instructions to keep the pipeline full before the branch resolves

Disadvantages: might be difficult to find suitable instructions



c. Fine-grained multithreading

Multiple threads in hardware. In each cycle, an instruction from a different thread is fetched

Advantages: no data dependencies within threads, increased throughput, no need for branch prediction

Disadvantages: decreased single thread performance, more complex hardware for the threads, resource contention in caches and memory

SIMD

1. Give examples of

   a. Single Instruction Multiple Elements (SIMD)

   Array processor

   Vector processor

   b. Multiple Instruction Single Elements (MISD)

   Systolic array

   c. Multiple Instructions Multiple Elements (MIMD)

Multithreading

Multiprocessing

2. What is the difference between array processor and vector processor?

   Array processor: same operation on different input elements at the same time

   Vector processor: pipelined, different operations on different input elements at the same time

3. What are the advantages and disadvantages of vector processing?

   Advantages: parallelism, very good for regular program patterns

   Disadvantages: more sophisticated hardware

   Advantages: no dependencies within a vector, each instruction generates a lot of work, no need to explicitly code loops

   Disadvantages: works only if parallelism is regular

4. What are the basic requirements of vector processors?

   Vector registers, vector length register, vector stride register

5. Describe memory banking. What is the purpose?

   Memory is divided into banks that can be accessed independently

   Banks share address and data busses

   Can sustain N parallel accesses if all N go to different banks

6. In a vector memory system, how to calculate next address using previous address and stride?

   Next address = prev address + stride

9 - SIMD, Multicore & Accelerator

Vectorized Processing

1. How many dynamic instructions? How many cycles?

```
for i = 0 to 49
    C[i] = (A[i] + B[i]) / 2
```

```
   MOVI R0  = 50            ; 1
   MOVA R1 = A              ; 1
   MOVA R2 = B              ; 1
   MOVA R3 = C              ; 1
X: LD R4 = MEM[R1++]      ; 11
   LD R5 = MEM[R2++]      ; 11
   ADD R6   = R4 + R5     ; 4
   SHFR R7 = R6 >> 1        ; 1
   ST MEM[R3++] = R7      ; 11
   DECBNZ --R0, X           ; 2  (decrement and branch if not ze
```

$4 + 6 \times 50 = 304$ dynamic instructions

$4 + (11 + 11 + 4 + 1 + 11 + 2) \times 50 = 2004$ cycles

2. How many cycles (assuming no chaining and one memory port) ?

```
MOVI VLEN = 50     ; 1
MOVI VSTR = 1      ; 1
VLD V0 = A         ; 11+49=60
VLD V1 = B         ; 11+49=60
VADD V2 = V0 + V1  ; 4+49=53
```

```
VSHFR V3 = V2 >> 1 ; 1+49=50
VST C = V3         ; 11+49=60
```

Total = 285 cycles

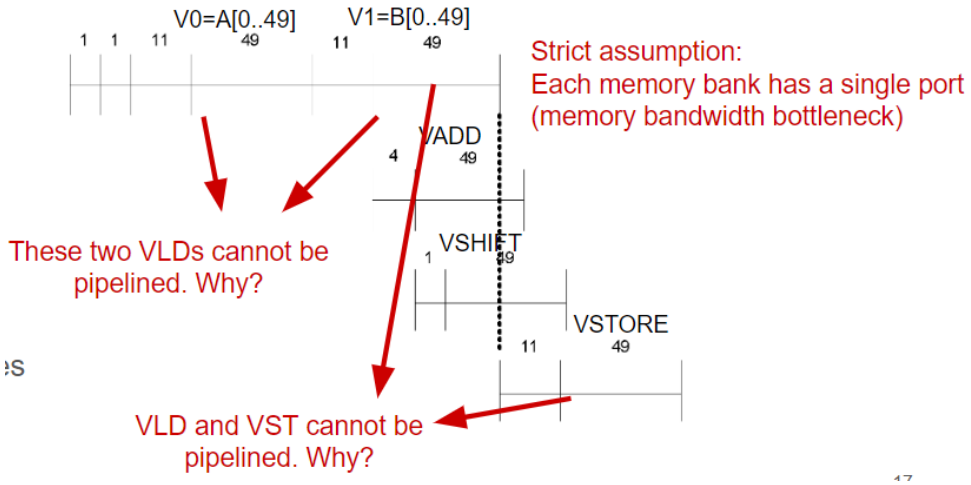
3. What is vector chaining?

   Forward the result from one vector instruction to the input of the next vector instruction

   Data forwarding from one vector functional unit to another

4. For the same code in Q2, how many cycles does it take if there is vector chaining but one memory port?

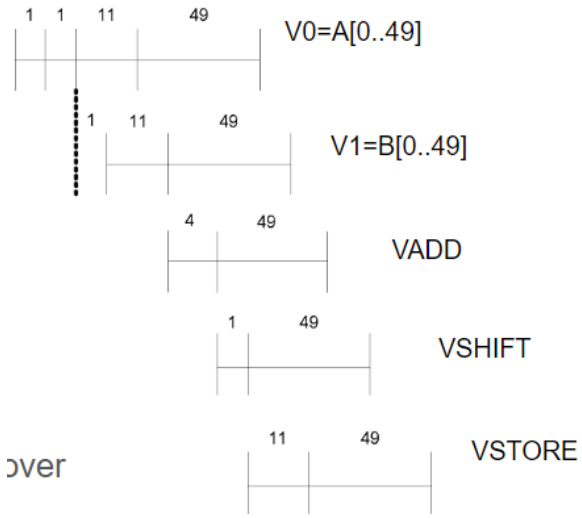
   182 cycles



5. For the same code in Q2, how many cycles does it take if there is vector chaining and 2 load ports and 1 store port?

   78 cycles

   79 cycles (V1 load starts one cycle later than V0 load)



Systolic Architectures

1. Describe the idea of systolic architectures

   Data passes through many processing elements in a rhythmic way

2. What are the advantages and disadvantages?

   Advantage: specialized to increase throughput for certain algorithms

   Disadvantage: not suitable for irregular parallelism

   Advantage: simple and regular design, high concurrency, balanced computation and I/O bandwidth

   Disadvantage: not generally applicable because computation needs to fit the PE functions

Multicore

1. Describe the idea of multicore

   Many processors on the same chip

2. What are the advantages and disadvantages?

   Advantage: increase overall performance

Disadvantage: more complex design and logic

Disadvantage: power hungry, diminishing returns, does not help memory-bound application, resource sharing can reduce single-thread performance, need to manage shared hardware resources

3. What are the advantages and disadvantages of these alternatives?
   a. Larger caches

      Advantage: increase performance

      Disadvantage: diminishing return

      Advantage: increase single-thread performance transparently to programmer

      Disadvantage: complicate memory hierarchy

   b. Integrating platform components on chip

      Advantage: speed up many system functions

      Disadvantage: not all application benefit

4. What are the differences between symmetric, asymmetric multicore, dynamic, and composed multicore?

   Symmetric multicore: many processors that are the same

   Asymmetric multicore: one large processor with many smaller processors

   Dynamic multicore: similar to asymmetric multicore, but may turn off some smaller processors depending on the situation

   Composed multicore: many small cores that can be logically fused to compose a large one

5. What are the implications of Amdahl's Law?

   f: Parallelizable fraction of a program

   N: Number of processors

$$\text{Speedup} = \frac{1}{1 - f + \frac{f}{N}}$$

The speedup is bottlenecked by the fraction of the program that is **not parallelizable**

10 - Branch Prediction I

1. Performance analysis.
   - No penalty for correct guess
   - 2 bubbles in the pipeline for the incorrect guess
   - 20% control flow instructions
   - 70% of control flow instructions are taken

   Calculate the CPI.

   $1 + 0.2 \times 0.7 \times 2 = 1.28$

2. What are the three things to be predicted at fetch stage?

   Whether the next instruction is a branch, branch direction (taken or not taken), branch target address

3. What is Branch Target Buffer (BTB)?

   It stores previous branch target addresses.

4. Describe the following static branch prediction methods
   a. Always not-taken

      Low accuracy

   b. Always taken

      Good if there are many loops; better accuracy than always not taken.

   c. BTFN

      Backward taken, forward not taken

   d. Profile-based

      Compiler determines likely direction for each branch using a profile run.
      Encodes that direction as a hint bit in the branch instruction format.
      Accurate if profile is representative

   e. Program-based

Use heuristic based on program analysis to determine statically-predicted direction. Example: predict BLEZ as NT.

f. Programmer-based

Programmer encodes extra information for the branches.

Programmer provides the statically-predicted direction via pragmas.

5. What are the common disadvantages of d-f?

   They require the encoding of extra information.

   They cannot adapt to dynamic changes in branch behaviour.

6. What are the advantages of dynamic branch prediction?

   No profiling is needed and no extra information to encode.

   Prediction can adapt to dynamic changes in branch behaviour.

7. Why is the last time predictor good for loops with large number of iterations and bad for loops with small number of iterations?

   The branch direction for a loop is TT...TN. If the last time predictor starts with not-taken, then it will always predict the wrong direction for the first and last iterations.

8. Give an example with 0% accuracy for the last time predictor, assuming it starts with not-taken.

   TNTNTN...

9. What is a disadvantage of the two-bit counter compared to the last time predictor?

   Higher hardware cost

11 - Branch Prediction II

1. What is global branch correlation? Give some examples.

   Global branch correlation refers to the branch outcome of one branch correlating with other branch's outcomes.

   Examples:

   ```
   // If the first branch is not taken, then the second branch i
   if (a)
   if (a && b)
   ```

   ```
   // If the first branch is taken, then the second branch is de
   if (a) b = 0;
   if (b == 1)
   ```

2. What does the Global History Register (GHR) store?

   It stores the last branch outcomes for the other branches.

   It keeps track of the global T/NT history of all branches.

3. What does the Pattern History Table (PHT) store?

   It stores a 2-bit saturating counter for each branch.

4. For Two Level Global Branch Prediction

   a. What is in the first level?

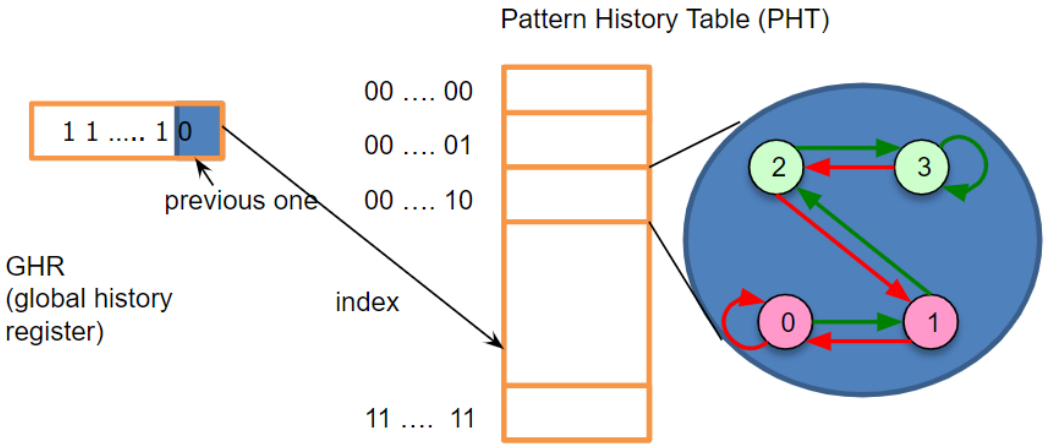
      N bits (corresponding to N branches) indicating the previous direction of the branches (the "history").

   b. What is in the second level?

      One 2-bit saturating counter for each branch.

   c. Describe the implementation using GHR and PHT

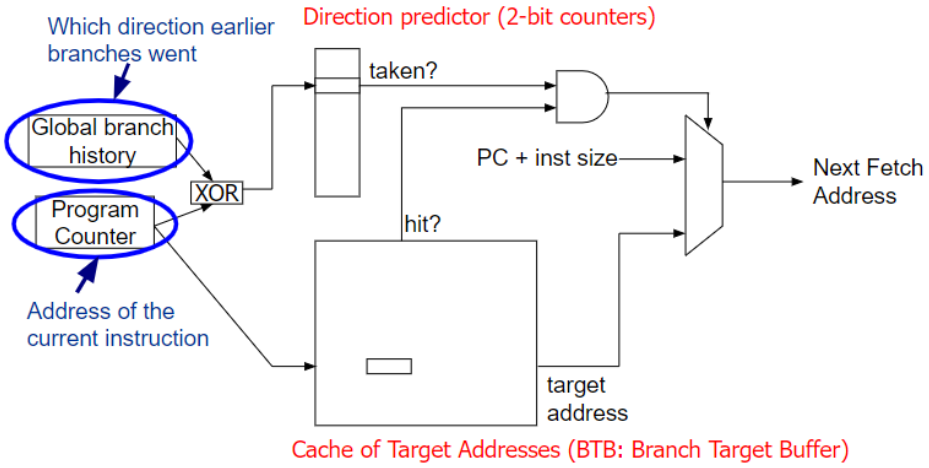
      We use the value in GHR to index into PHT, and use the 2-bit saturating counter to make prediction.



5. Describe the idea of Gshare predictor, its advantages and disadvantages.

   The Gshare predictor uses the value of GHR hashed with the PC counter to index into the table.

   This provides more context information and better utilization of PHT, but increases access latency.



6. What is local branch correlation?

   Local branch correlation refers to the outcome of one branch correlating with the previous outcomes of the same branch.

7. For Two Level Local Branch Prediction

   a. What is in the first level?

      The previous directions of the same branch.

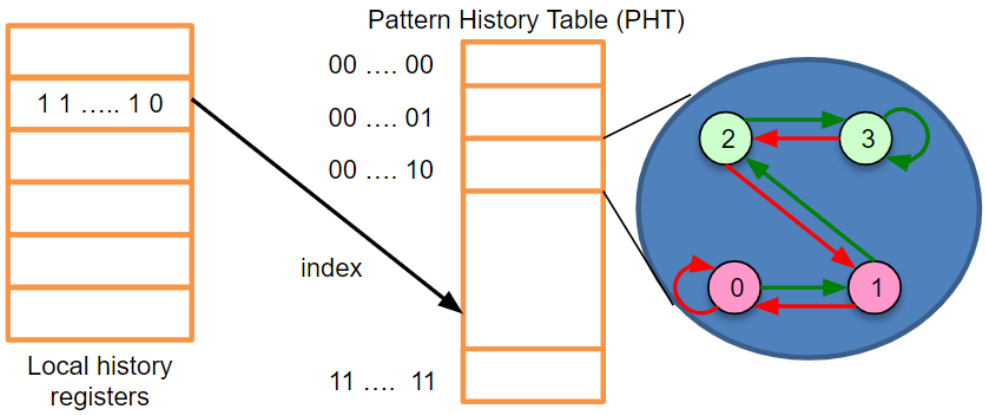
A set of local history registers. Select the history register based on the PC of the branch.

b. What is in the second level?

One 2-bit saturating counter for each branch history.

c. Describe the implementation using LHR and PHT

We use the value in LHR to index into PHT, and use the 2-bit saturating counter to make prediction.



Yeh and Patt, "Two-Level Adaptive Training Branch Prediction," MICRO 1991.

8. Describe the idea of a hybrid branch predictor

   Combine multiple branch predictors and select the best one

9. Advantages and disadvantages of a hybrid branch predictor

   Advantage: achieve the best performance out of different branch predictors

   Disadvantage: more complex

   Advantage: better accuracy, reduced warmup

10. Describe the idea of a perceptron branch predictor.

    Use a perceptron to learn the correlations between branch history register bits and branch outcome.

12 - Microarchitecture Simulation

Simulation

1. Why do we need simulation?

   Real HW is costly to build

2. Give some examples of simulators.

   CPU simulator, memory simulator, GPU simulator, etc.

3. Cycle-accurate simulator vs. trace-based simulator. Which one
   a. Has better accuracy?

      Cycle-accurate simulator

   b. Is simpler and faster?

      Trace-based simulator

4. Define the following
   a. Host

      The actual hardware

   b. Simulator

      Software that runs on the host, exposing hardware to the guest

   c. Simulator's performance

      Time it takes for the simulation to run, measured by the wall clock

   d. Simulated performance

      The predicted performance of the guest code

gem5

1. Briefly describe what it is.

The gem5 architecture simulator provides a platform for evaluating computer systems by modeling the behavior of the underlying hardware

FireSim

1. Briefly describe what it is.

   Cycle-accurate simulator for datacenter

2. What are three things it needs to do?

   Model HW at scale, run real SW, make it usable

13 - Cache I

DRAM & SRAM

1. What does DRAM it stands for?

   Dynamic random access memory

2. How is a value stored in DRAM?

   Through the charge in the capacitor

3. How many transistors are there in DRAM?

   1 transistor and 1 capacitor

4. Why doe DRAM cell needs to be refreshed?

   Because the capacitor loses charges over time

5. What does SRAM stands for?

   Static random access memory

6. How is a value stored in SRAM?

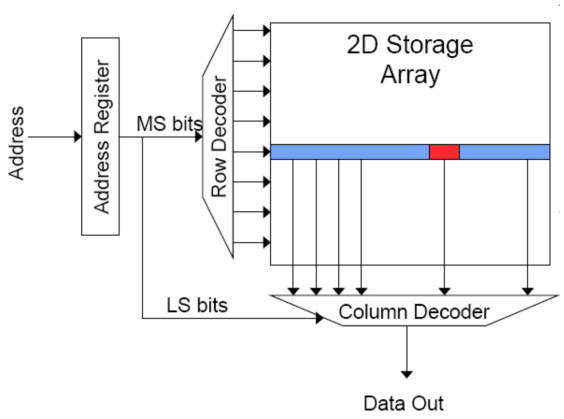
   Two inverters

   Two cross coupled inverters store a single bit

7. How many transistors for storage for SRAM? How many transistors for access for SRAM?

   4 transistors for storage. 2 transistors for access

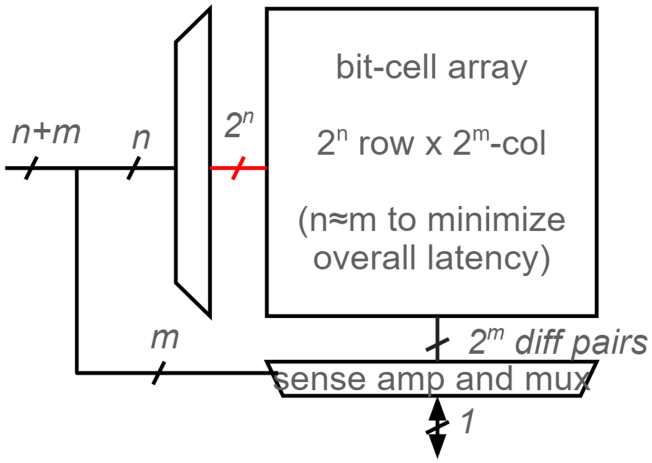
8. Describe the read access sequence



Decode address into row and column bits. Read the row and then the column?

1. Decode row address & drive word-lines.
2. Entire row read.
3. Amplify row data.
4. Decode column address address & select subset of row.
5. Precharge bit-lines for next address.

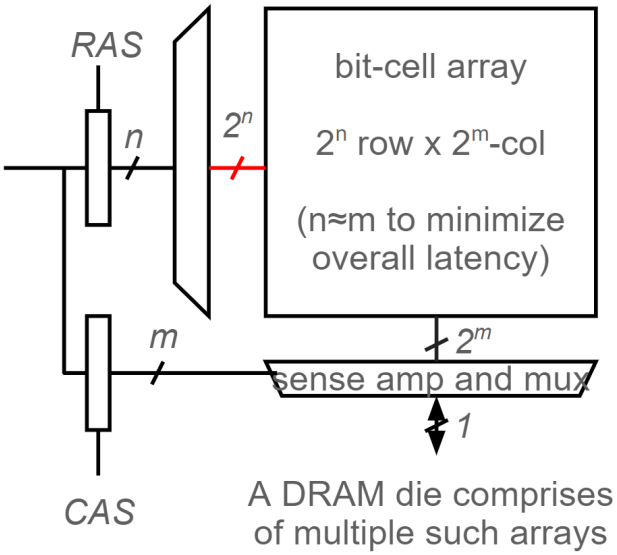
9. Describe the read sequence for SRAM



1. Address decode
2. Drive row select
3. Selected bit-cells drive bitlines (entire row is read together)

4. Differential sensing and column select (data is ready)
5. Precharge all bitlines (for next read or write)

10. Describe the read sequence for DRAM



1. Address decode
2. Drive row select
3. Selected bit-cells drive bitlines (entire row is read together)
4. A “flip-flopping” sense amplifies and regenerates the bitline, data bit is mux’ed out.
5. Precharge all bitlines (for next read or write)

11. DRAM vs. SRAM. Which one has
    a. faster access?

       SRAM

    b. higher density?

       ~~SRAM~~

       DRAM

    c. lower cost?

DRAM

d. refresh requirement?

DRAM

e. requirement of putting capacitor and logic together?

DRAM

Memory Hierarchy

1. Why memory hierarchy?

   We want data that is accessed most often to be closer to the processor (i.e., exploitation of temporal and spatial locality) so that it appears "fast", but we also want to store a large amount of data. By having a memory hierarchy, we can make memory appears to be "fast" and "large".

2. Define temporal locality.

   The same data is likely to be accessed many times within a small period of time.

   A program tends to reference the same memory location many times and all within a small windows of time

3. Define spatial locality. Give some examples.

   Data that is close together are likely to be accessed together. Examples include instructions and array-like data structures.

   A program tends to reference a cluster of memory locations at a time.

4. The recursive latency equation is $T_i = t_i + m_i \cdot T_{i+1}$.

   a. What is the goal?

      The goal is to make $T_i \approx t_i$

   b. How do we keep m_i low?

      Increase the hit rate by having higher set associativity and better replacement policies.

      Prefetching: anticipate what you will need

c. How do we keep T_{i+1} low?

   Introduce more levels of intermediate caches

5. Latency calculation

- 90nm P4, 3.6 GHz
- L1 D-cache
  - C_1 = 16K
  - t_1 = 4 cycles
- L2 D-cache
  - C_2 =1024 KB
  - t_2 = 18 cycles
- Main memory
  - t_3 = ~ 50ns or 180 cycles
- Notice
  - best case latency is not 1
  - worst case access latencies are into 500+ cycles

if m_1=0.1, m_2=0.1
T_1=? T_2=?

if m_1=0.01, m_2=0.01
T_1=? T_2=?

if m_1=0.05, m_2=0.01
T_1=? T_2=?

if m_1=0.01, m_2=0.50
T_1=? T_2=?

a. $T_1 = 7.6, T_2 = 36$

b. $T_1 = 4.198, T_2 = 19.8$

c. $T_1 = 4.99, T_2 = 19.8$

d. $T_1 = 5.08, T_2 = 108$

14 - Cache II

Cache Basics

1. Define a block (line) in cache.

   A block is a unit of storage in cache.

2. Define hit and miss in cache.

   Hit means the tag matches (or exists in a cache with higher associativity) and the valid bit is set. Otherwise, it's a miss.

   On a cache hit, use cached data instead of accessing memory.

   On a cache miss, have to replace a cached block.

3. How is average memory access time (AMAT) calculated, given hit rate, hit latency, miss rate and miss latency?

   AMAT = hit rate $\times$ hit latency + miss rate $\times$ miss latency

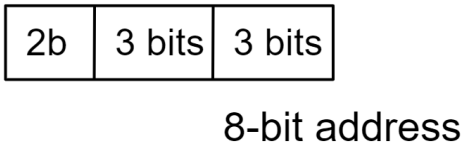
4. Can reducing AMAT reduce performance?

   Yes, reducing AMAT does not always lead to better performance.

   This metric only considers the overall latency of memory access (either cache hits or misses) but overlooks whether an access is latency-critical or not.

   For example, in throughput-oriented parallel applications, memory/cache latency is not as critical as throughput.

5. Identify the types of bits and their purposes for a cache block.



   The least significant 3 bits are the **block bits**. They identify the bytes within a block. Block size = 2^b where b is the number of block bits.

The middle 3 bits are the **index bits**. They are used to map a block into the cache. Number of entries in cache = 2^i where i is the number of index bits, for direct mapped cache.

The most significant bits are the t**ag bits**. They are used to distinguish blocks with the same index bits.

6. Each block maps to a location in the cache, determined by the ___ bits in the address.

   Index

7. Define a direct-mapped cache.

   A block can only be mapped to one location in cache.

8. Why can there be conflict misses for direct-mapped cache?

   Addresses with the same index bits can cause conflict misses because they are mapped to the same location in cache.

9. Give an example (of a sequence of addresses) that leads to 0% hit rate for a direct-mapped cache.

   00 000 000, 01 000 000, 00 000 000, 01 000 000... The index is the same but the tags are different.

Set Associativity

1. Define set associativity.

   Having a set associativity of n means that a maximum of n blocks with the same index can be placed into the cache at the same time.

2. Give an advantage and disadvantage of having higher set associativity?

   Advantage: reduce conflict misses

   Disadvantage: more complex, takes longer to check if the tag matches

3. Define full associativity

   A block can go to any location in cache.

4. Is the increase in the hit rate proportional to associativity?

No, there is a diminishing return.

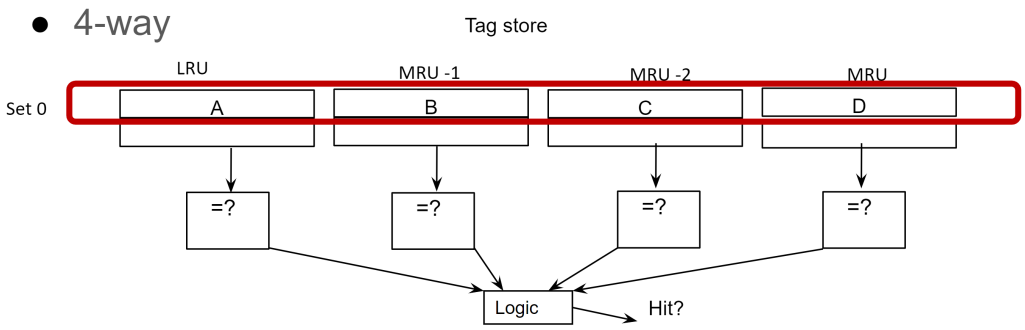
Cache Eviction Policy

1. Describe the LRU cache eviction policy.

   The cache block that is accessed furthest from now is replaced.

2. Given the initial state with accesses ACBD. What happens after

   - 4-way



   - E is accessed?

     A is replaced, so LRU=C, MRU-1=D, MRU-2=B, MRU=E

   - B is accessed?

     B becomes MRU, so LRU=C, MRU-1=E, MRU-2=D, MRU=B

3. How many bits are needed for the orderings of the n-way cache?

   $\log_2(n!)$

4. Why do most modern processors not implement “true LRU” in highly-associated caches?

   Because the implementation of true LRU is very complicated, and LRU is an approximation anyways.

5. What are some approximations of LRU?

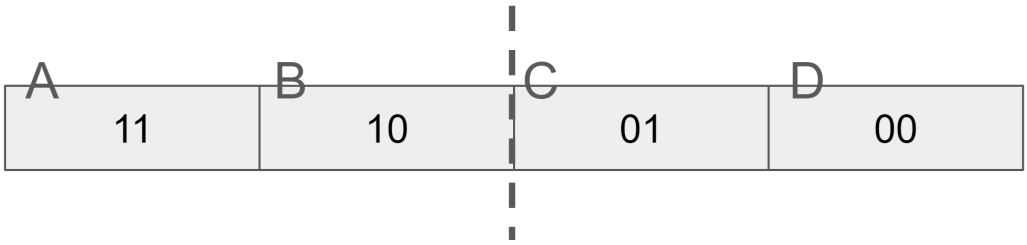
   Hierarchical LRU, etc...

   Not MRU, hierarchical LRU, Victim-NextVictim

6. Describe the idea of hierarchical LRU (not MRU). On replacement, how do we select victim?

   We only keep track of **MRU group** and **MRU block within the MRU group**. Victim is randomly chosen from a non-MRU block in a non-MRU group.

7. Given the initial state - two bits associated with 4 cached addresses (ABCD). Write the state after each access



   - Access to B

     A=10, B=11, C=01, D=00

   - Access to D

     A=00, B=01, C=10, D=11

   - Access to X (Who is the victim?)

     ~~We can either replace A or B.~~ Suppose A is replaced.

     X=11, B=10, C=00, D=01.

     Only A can be replaced since B was the MRU block in the non-MRU group.

15 - Cache III

Victim/Next-Victim Policy

1. Describe the victim/next-victim policy.
    a. How many blocks' status are tracked in each set?

        Only 2 blocks (the victim and the next victim) are tracked.

        Other blocks are ordinary (O) blocks.

    b. What happens on a cache miss?

        V is evicted. NV becomes V. A block is randomly selected to be NV.

    c. What happens on a cache hit to V?

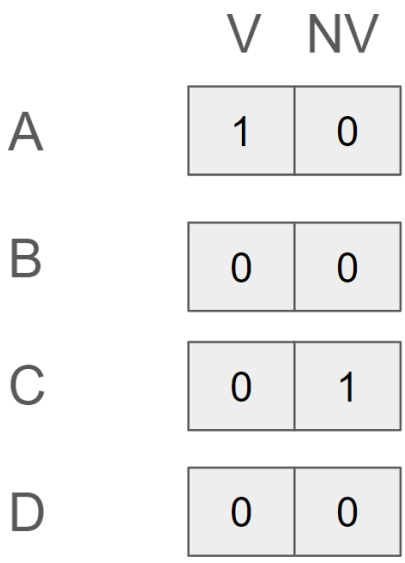
        V becomes O. NV becomes V. A block is randomly selected to be NV.

    d. What happens on a cache hit to NV?

        NV becomes O. A block is randomly selected to be NV.

    e. What happens on a cache hit to O?

        Nothing happens.
2. The initial state is given below. What happens after an access to A?



A becomes O (00).

B remains the same (00).

C becomes V (10).

D is randomly selected to be NV (01).

3. For a 4-way cache, describe a scenario where there is 0% hit rate with LRU policy.

   A, B, C, D, E, A, B, C, D, E, ...

4. What is set thrashing?

   When the program's working set is larger than the degree of associativity

5. When there is set thrashing, which cache replacement policy is better, LRU or random?

   Random is better.

6. Consider Belady's OPT

   a. What is this?

      Replace the cache block that is accessed the furthest in the future. It is the optimal replacement policy.

   b. Is it possible to implement?

No, it is only theoretical. It is impossible to predict the future.

c. Is this optimal for minimizing miss rate?

   Yes?

d. Is this optimal for minimizing execution time?

   No.

   Why? Because cache miss latency/cost from block to block

Cache vs. Page Replacement

1. What is the “tag store” for physical memory data store.

   The page table.

2. What are some differences between cache and page replacement?

   Page replacement is slower

   Page replacement is role of hardware?

Handling Writes

1. For writing the modified data in a cache to the next level, define

   a. Write through

      Write to next level after each write to cache

   b. Write back

      Write to next level after the cache block is evicted

2. Advantage and disadvantage of write-through?

   Advantage: easy to implement, consistency is ensured

   Disadvantage: not efficient?

   Disadvantage: more bandwidth intensive since there is no coalescing of writes

3. Advantage and disadvantage of write-back?

   Advantage: can combine multiple writes into one

Disadvantage: ?

Disadvantage: need a bit in the tag store indicating the block is "dirty/modified"

4. Advantages and disadvantages of
    a. allocate on write miss

        Advantage: easier to implement since it can be treated the same as reads.

        Disadvantage: requires transfer of the entire cache block

    b. no-allocate on write miss

        Better for programs that do not have a lot of writes since if we allocate on write miss, the write cache blocks will occupy the cache.

5. For sectored caches
    a. Describe the idea

        The cache blocks are divided into subblocks.

        Separate valid and dirty bits for each sector

    b. List the advantages

        Higher flexibility for writes since there is no need to write the entire block

    c. List the disadvantages

        Not useful for reads

        May not exploit spatial locality fully when used for reads

6. Advantages and disadvantages of unified instruction + data caches?

    Advantage: ?

    Disadvantage: thrashing

    Advantage: dynamic sharing of cache space

    Disadvantage: I and D caches are accessed in different places in the pipeline. Where do we place the unified cache for fast access?

Performance

1. What are the problems of
    a. Too small caches

        High miss rate (cannot hold a lot of data)

    b. Too large caches

        Longer access latency

    c. Too small blocks

        Does not exploit spatial locality well

        Large tag overhead

    d. Too large blocks

        Does not exploit temporal locality well

        Waste of cache space and bandwidth if spatial locality is not high

2. Define and explain how to reduce each type
    a. Compulsory miss

        The first access to cache is always a miss.

    b. Capacity miss

        The cache is filled so there is no space for new blocks.

    c. Conflict miss

        When two addresses have the same index bits.

3. What are some ways to reduce miss rate?

    Higher set associativity, smarter replacement policies, larger cache capacity, etc.

4. What are some ways to reduce miss latency/cost?

    Same as above?

    Multi-level caches, critical word first, subblocking/sectoring, etc.

5. Describe the idea of
    a. Victim cache

Evicted blocks are placed into victim cache to prevent blocks going back and forth between the cache and the memory.

b. Hashing associativity

   Use hash function for indexing the memory blocks into cache so that they are more evenly spread out.

   More complex to implement

c. Pseudo associativity

   Use a different index function again after each miss.

d. Skewed associative caches

   Use different index functions for each cache way.

   Advantage: indices are more randomized.

   Disadvantage: additional latency of hash functions

Software Approaches

1. Suppose the memory is column-major. Why is the following code bad? Give a solution.

   Poor code

```
for i = 1, rows
    for j = 1, columns
        sum = sum + x[i,j]
```

   M[i, j] is very far from M[i, j+1] in a column-major memory. Spatial locality is not exploited.

   To solve this, use a “loop interchange” to exchange the inner and outer loops.

```
for j = 1, columns
    for i = 1, rows
        sum = sum + x[i,j]
```

2. Why does the cache have poor hit rate? Give a solution.

```
struct Node {
     struct Node* node;
     int key;
     char [256] name;
     char [256] school;
}
while (node) {
     if (node->key == input_key) {
       // access other fields of node
     }
     node = node->next;
}
```

   The “name” and “school” fields occupy too much space. We can use a pointer to make it more compact.

```
struct Node {
    struct Node *node;
    int key;
    struct NodeInfo *node_info;
}
```

3. What does Miss Status Handling Registers (MSHRs) do?

   MSHRs buffer cache misses so that no new request will be generated if a miss is already in the buffer.

4. A cache access checks MSHRs to see if a miss to the same block is already pending. What happens if

   a. Pending?

      A new request will not be generated

   b. Pending and the needed data available?

      The data will be forwarded to buffered requests that need this piece of data

      Data forwarded to later load

5. What is memory level parallelism (MLP)?

   Processing multiple memory requests in parallel

6. Advantages and disadvantages of resource sharing?

   Advantage: no need to duplicate resource, save costs, improve utilization, reduce communication latency

   Disadvantage: resource contention, eliminates performance isolation

7. Advantages and disadvantages of shared caches?

   Advantages: no need to duplicate cache, save costs, high effective capacity, easier to maintain coherence (a cache block is in a single location)

   Disadvantage: misses can be caused due to cache access from other programs

16 - Memory I

1. What are the major trends affecting main memory?

   Modern applications have higher demands of latency and bandwidth.

   Power consumption is a key bottleneck.

   Need for main memory capacity, bandwidth, QoS increasing

   Main memory energy/power is a key system design concern.

   DRAM technology scaling is ending.

2. What is the DRAM scaling problem?

   We want large capacitors to counter the problem of capacitor losing charge over time, but it is difficult to scale beyond a certain point.

   Capacitor must be large enough for reliable sensing. Access transistor should be large enough for low leakage and high retention time. However, scaling beyond 40-35nm is challenging.

3. A DRAM cell stores data as _____. A DRAM cell needs to be _____ every x ms

   Charge; refreshed.

4. DRAM subsystem organization from top to bottom (7 layers)?

   Channel, module, rank, chip, bank, row/column, cell

5. For page mode DRAM, what are the steps to access

   a. A “closed row”

      Activate to open the row.

      Read/write the column.

      Precharge to close the row.

   b. An “open row”

      No need to activate. Just read/write.

17 - Memory II

1. What does the DRAM chip consist of?

   A DRAM chip consists of multiple banks.

2. What does the DRAM rank consist of?

   A DRAM rank consists of multiple chips.

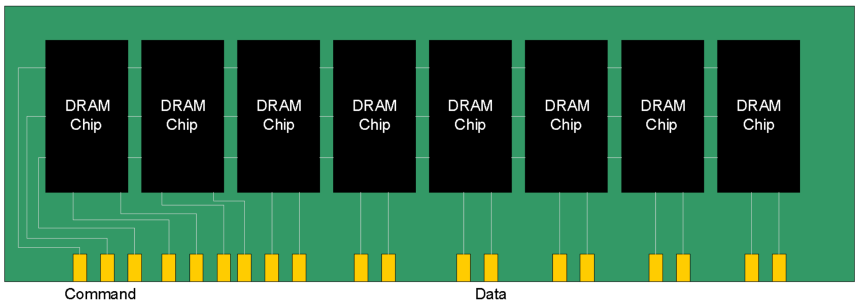
3. True or false. All chips comprising a rank are controlled at different times.

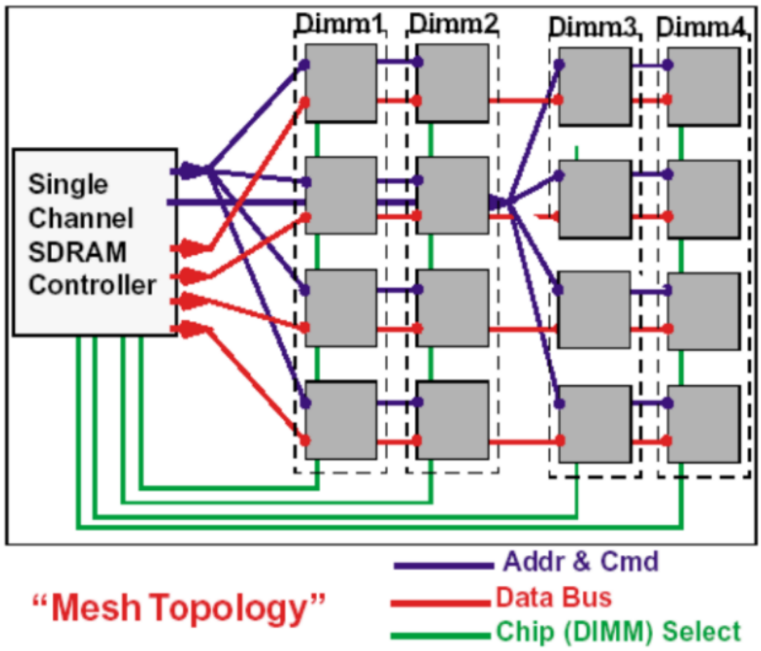
   False. All chips comprising a rank are controlled at the same time.

4. What does a DRAM module consist of? Give an example of a module.

   A DRAM module consists of multiple ranks. An example is Dual Inline Memory Module (DIMM)

5. What are the advantages and disadvantages of a 64-bit wide DIMM? Multiple DIMMs?





64-bit wide DIMM

- Advantage: High capacity, high flexibility (memory controller does not need to deal with individual chips)
- Disadvantage: Low granularity (access cannot be smaller than the interface width)

Multiple DIMMs

- Advantage: even higher capacity
- Disadvantage: interconnect complexity

6. How to calculate DRAM latency? Which one is dominant?

   DRAM latency = subarray latency + I/O latency. Subarray latency is dominant.

7. Explain the latency components. Suppose CAS stands for column address strobe, RAS stands for row address strobe, and PRE stands for precharge.

   a. Controller latency

      Queueing and scheduling of requests

      Convert commands into requests

   b. DRAM bank latency, if row is "open"

CAS only

c. DRAM bank latency, if array is precharged

   RAS + CAS

d. DRAM bank latency, if array is not precharged

   PRE + RAS + CAS

8. What are the purposes of multiple banks/independent channels? Which one is better?

   They enable concurrency ⇒ higher bandwidth

   Multiple independent channels are better because they have separate data buses ⇒ increased bus bandwidth

9. Advantages and disadvantages of having multiple channels?

   Advantages: higher throughput

   Disadvantage: increased cost

10. What is row interleaving? How are accesses to consecutive cache blocks serviced?

    Row interleaving means consecutive rows are placed together in the bank.

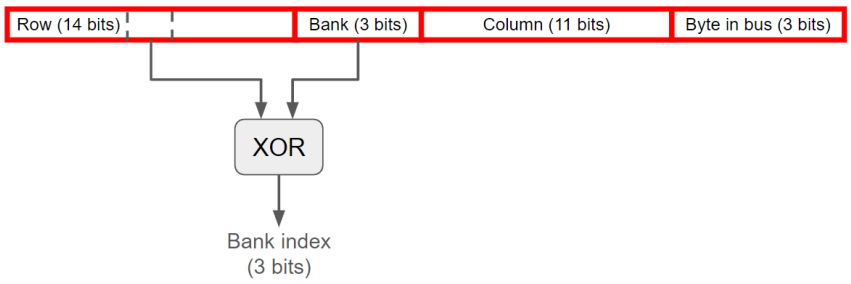
    Consecutive cache blocks are serviced in a pipelined manner.

11. What is cache block interleaving? How are accesses to consecutive cache blocks serviced?

    Cache block interleaving means consecutive cache blocks are placed together in the bank.

    Consecutive cache blocks are serviced in a parallel manner.

12. What is bank mapping randomization?



Bank mapping randomization means hashing the row with the bank index to form a random bank index. This can reduce the chance of having a bank conflict.

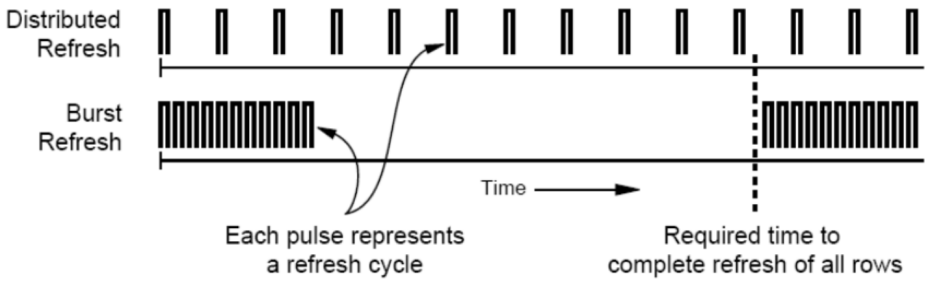
13. How do you refresh for DRAM?

    Activate and precharge each row

14. What is burst refresh and distributed refresh?

    Burst refresh: all rows refreshed immediately after one another

    Distributed refresh: each row refreshed at a different time, at regular intervals



15. What is one benefit of distributed refresh?

    It prevents long stall time caused by refresh

16. List four functions of a DRAM controller.

    Process memory requests, schedule memory request, power management

    Ensure **correct operation** of DRAM

    Service DRAM requests while obeying **timing constraints** of DRAM chips

    Buffer and schedule requests to improve performance

    Manage **power consumption**.

17. Describe the following DRAM scheduling policies
    a. FCFS

       First come first serve. Oldest first.

    b. FR-FCFS

       First ready, first come first serve. Row ready (open) first, then oldest first.

18. Describe the following row buffer management policies
    a. Row open

       Leave the row open after access. Good if the next access is the same row. Bad if the next access is a different row.

    b. Row close

       Close the row after access. Good if the next access is a different row. Bad if the next access is the same row.

    c. Adaptive policies

       Make predictions about the next access.

18 - New Memory Technologies I

Hybrid Memory System & PCM

1. Describe charge memory and resistive memory.

   Charge memory uses a charge to store a value, and uses the voltage to read the value.

   Resistive memory uses current to store a value, and uses the resistance to read the value.

   Charge memory writes data by capturing charge Q and read voltage V.

   Resistive memory writes data by pulsing current dQ/dt and reads data by detecting resistance R

2. What is the purpose of hybrid memory systems?

   To get the best from both systems

3. What are two limits of charge memory?

   ~~The capacitor can leak charge. It is also difficult to scale.~~

   Difficult charge placement and control.

   Reliable sensing becomes difficult as charge storage unit size reduces.

4. How does Phase Change Memory (PCM) work?

   a. Write

      Use current to set/reset values

      Change phase via current injection

   b. Read

      Use special material to detect current?

      Detect phase via material resistance

5. What are some advantages of PCM?

   Better scalability, non-volatility, no need to refresh, can be denser than DRAM

6. Is PCM faster or slower than DRAM?

   Slower

7. PCM vs. DRAM. Which one has
   a. Better technology scaling

      PCM

   b. Lower active energy

      DRAM

   c. Better endurance

      DRAM

   d. Non-volatility

      PCM

   e. Better reliability

      ~~PCM~~

      DRAM

   f. Low idle power (no need to refresh)

      PCM

   g. Lower latency

      DRAM

8. What are some challenges of hybrid memory systems related to
   a. Partitioning

      Should DRAM be cache or main memory, etc.

      What fraction? How many controllers?

   b. Data allocation/movement

      Who should control this? What algorithm should be used?

   c. Design

How should we design it to hide the disadvantages of PCM and exploit the advantages of PCM?

Merging of Memory and Storage

1. Briefly describe the goal of coordinated memory and storage with NVM.

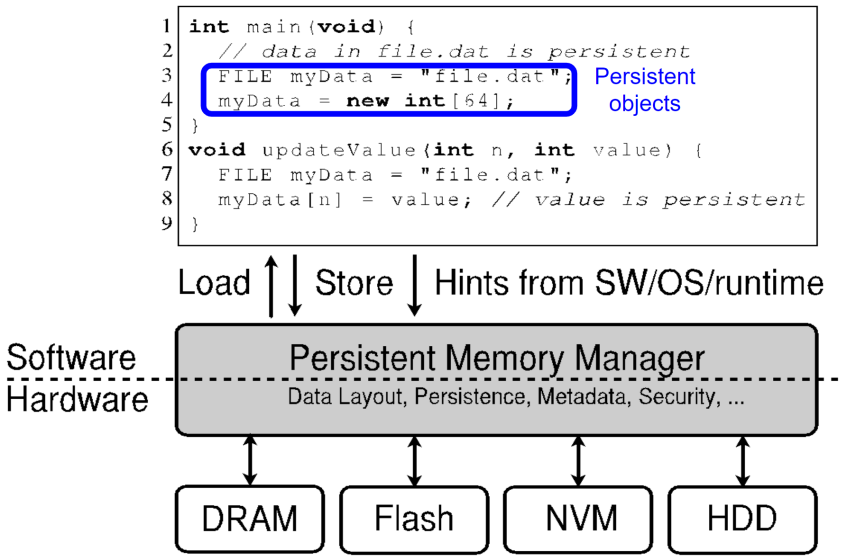
   The goal is to eliminate the need of transferring data back and forth between primary memory and secondary storage.

2. What are the benefits of coordinated memory and storage?

   It has the benefits of main memory and secondary storage - fast, byte addressable and non-volatile.

   It provides the opportunity to update data in-place in memory with load/store interface.

3. What does a Persistent Memory Manager (PMM) do?



PMM uses access and hint information to allocate, locate, migrate and access data in the heterogeneous array of devices

19 - New Memory Technologies II & HW Security

Processing in Memory

1. Why are today's computing systems said to be overwhelmingly processor centric?

   ~~Processing is done in one place. Data moves a lot.~~

   All data processed in the processor. Data storage units are dumb and are largely unoptimized

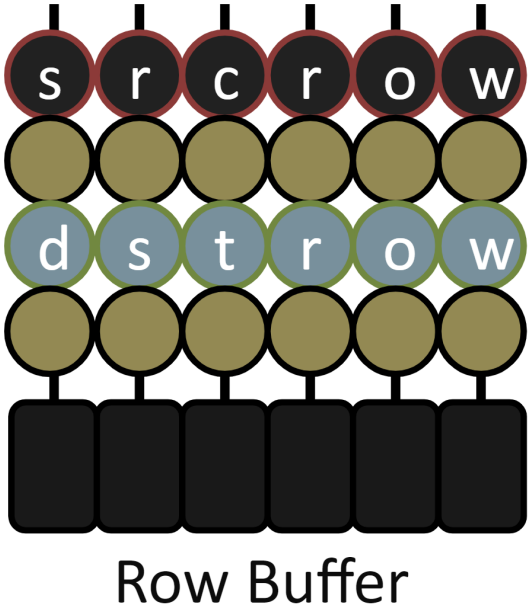
2. Data movement or computation, which one is a major system energy bottleneck?

   Data movement

3. Describe three ways we can make a paradigm shift to mitigate the effect of this bottleneck.

   Minimize data movement, make computation closer to memory, make the systems more data-centric

4. Describe the implementation of fast parallel mode:



Activate srcrow by putting it into row buffer. Then read the values from the srcrow

Activate dstrow by putting it into row buffer. Then transfer the values into the dstrow

5. What are the benefits of fast parallel mode?

   No bandwidth consumption. Very little changes to the DRAM chip.

6. Give an example of large-scale graph processing.

   Google's PageRank

7. What is the bottleneck of graph processing?

   Random memory accesses

8. Do conventional systems fully utilize high memory bandwidth?

   No

Side Channel Attacks

1. What are side channel attacks?

   Attacks that use extra information to reveal secrets.

2. What are some examples of side channel attacks in computer systems?

   Cache, memory, power consumption, latency, etc.

3. What is a covert channel?

   A secret channel to convey information

4. Briefly describe the following (bold = mentioned in final review)

   a. **Cache side channel attack**

      The attacker sets the cache to a known state, then checks the changes in the state of the cache after victim accesses.

   b. Flush + Reload attack (shared memory)

      The attacker flushes the data in the shared memory from cache. After victim's operations, the attacker tries to access the shared memory. Fast ⇒ victim accessed. Slow ⇒ victim did not access.

   c. Flush + Reload RSA attack

      RSA uses square-and-multiply algorithm to compute. For clear bits (0), there is only a square operation. For set bits (1), there is a square and a multiply operation, so the power consumption will be higher. Attackers can thus infer the secret exponent from the power consumption.

   d. Prime + Probe attack (shared cache)

      ~~Similar to Flush + Reload. Fast ⇒ victim accessed. Slow ⇒ victim did not access.~~

      Assumption: cache eviction may happen when multiple memory locations are mapped to the same block.

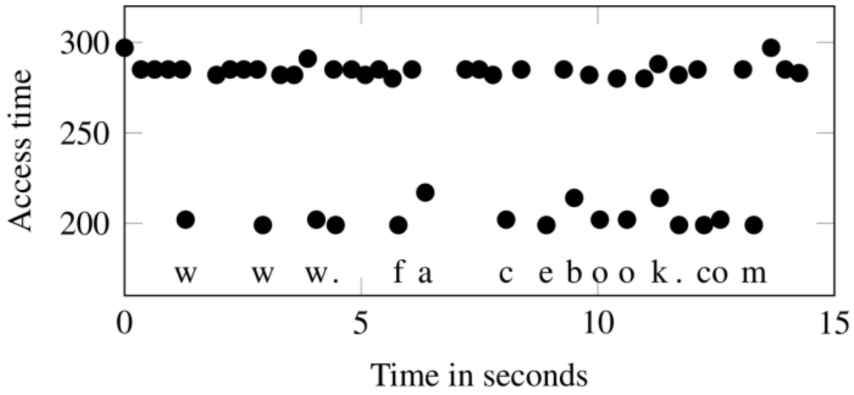
      Attacker primes: fill up the cache with its only memory. After victim's operations, the attacker checks the cache. Fast ⇒ victim not accessed. Slow ⇒ victim accessed.

   e. **Memory side channel attack**

      Similar to Flush + Reload. Fast ⇒ victim accessed. Slow ⇒ victim did not access.

DRAM is organized as a hierarchy. The row buffer (or sense amplifiers) in DRAM is like a cache. Fast ⇒ row buffer hit. Slow ⇒ row buffer missed.

f. Spy on memory access



Keystrokes in Firefox address bar

Spy activates conflict row. Victim computes and possibly accesses shared row.

Similar to Flush + Reload. Fast ⇒ victim accessed. Slow ⇒ victim did not access.

g. **Power side channel attack**

Different applications can have different power consumption, which can be useful information to attackers.

h. Leaking AES keys

The power consumption of S-box computation is data-dependent.