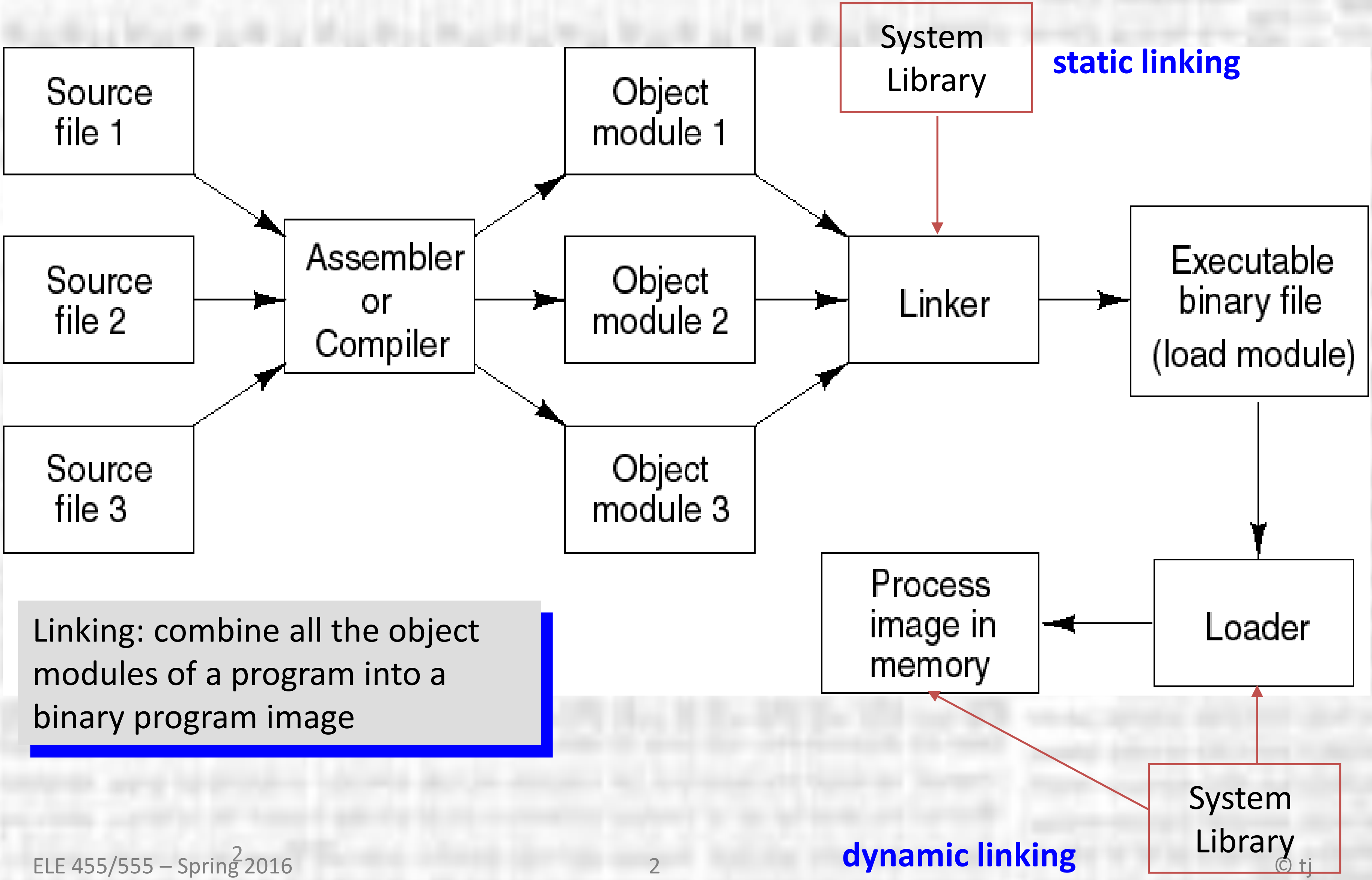
ELE 455/555
Computer System Engineering

Section 3 – Memory

Class 5 – Cache Control

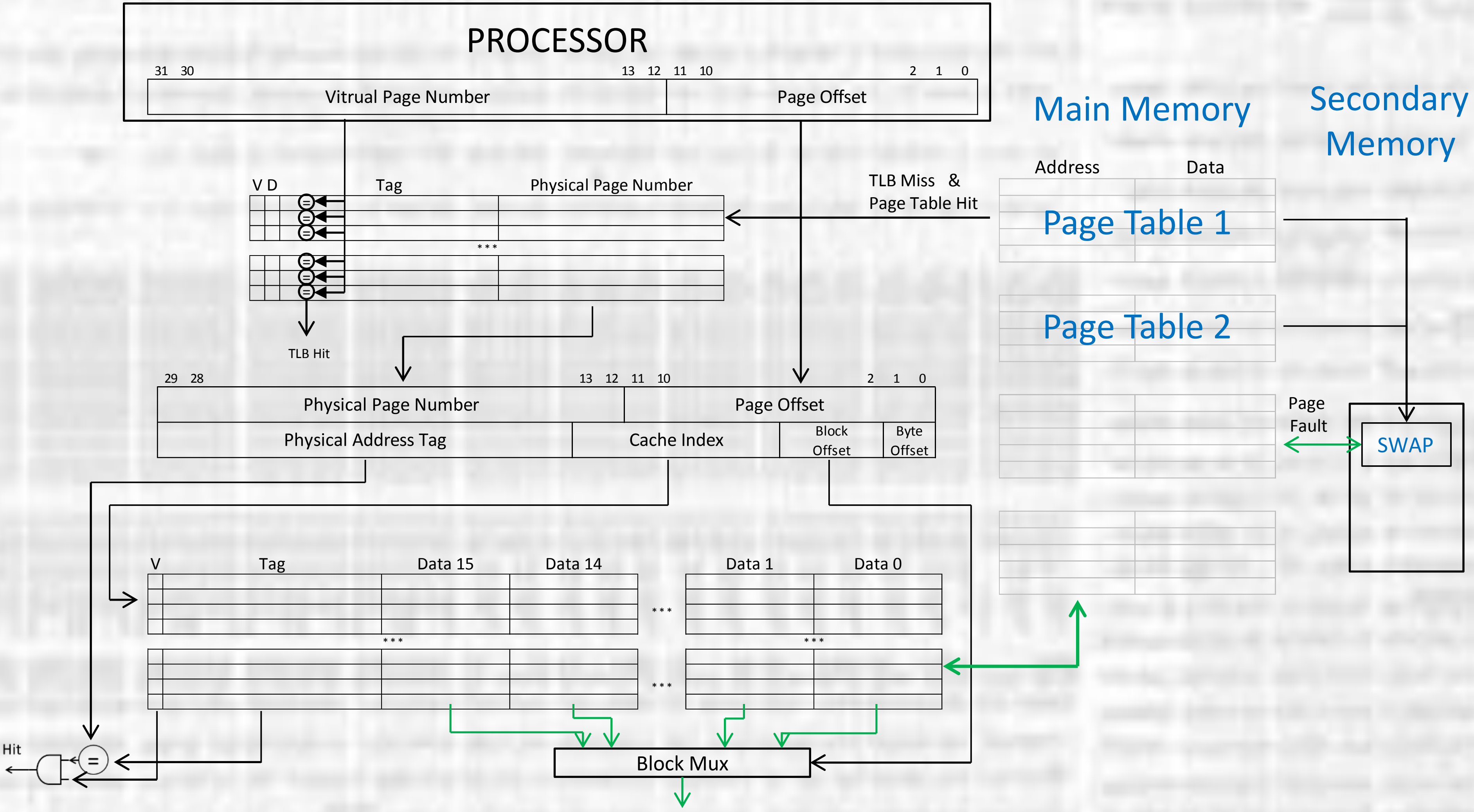
Steps for Loading a Process in Memory



Linking: combine all the object modules of a program into a binary program image

Virtual Memory

Cache Review



Cache Control

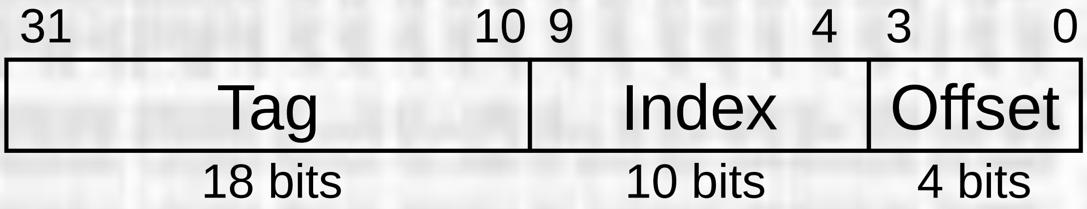
Sources of Misses

- Miss Definitions
  - Compulsory misses (aka cold start misses)
    - First access to a block
  - Capacity misses
    - Due to finite cache size
    - A replaced block is later accessed again
  - Conflict misses (aka collision misses)
    - In a non-fully associative cache
    - Due to competition for entries in a set
    - Would not occur in a fully associative cache of the same total size

Cache Control

Finite State Machine

- Example cache characteristics
  - Direct-mapped, write-back, write allocate
  - Block size: 4 words (16 bytes)
  - Cache size: 16 KB (1024 blocks)
  - 32-bit byte addresses
  - Valid bit and dirty bit per block
  - Blocking cache
    - CPU waits until access is complete

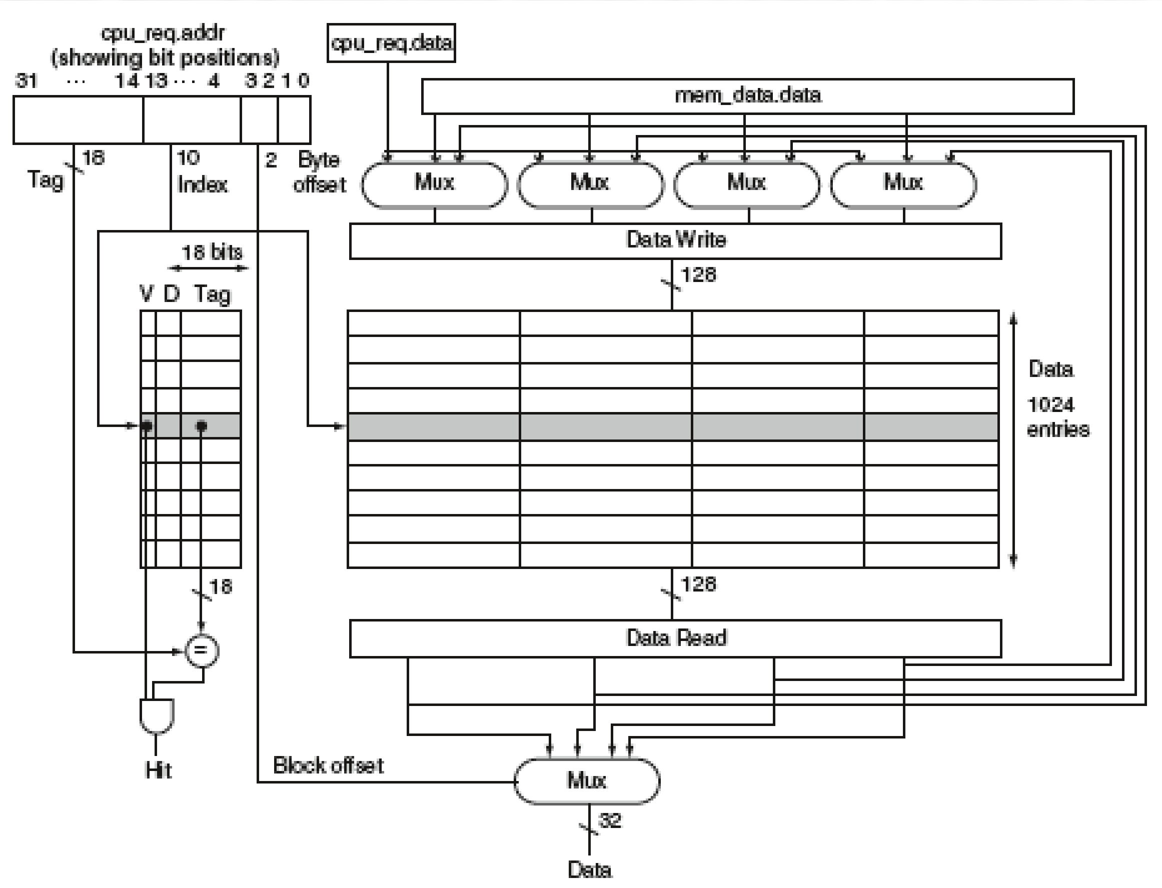


31 … 10	9 … 4	3 … 0
Tag	Index	Offset
18 bits	10 bits	4 bits

Cache Control

Finite State Machine

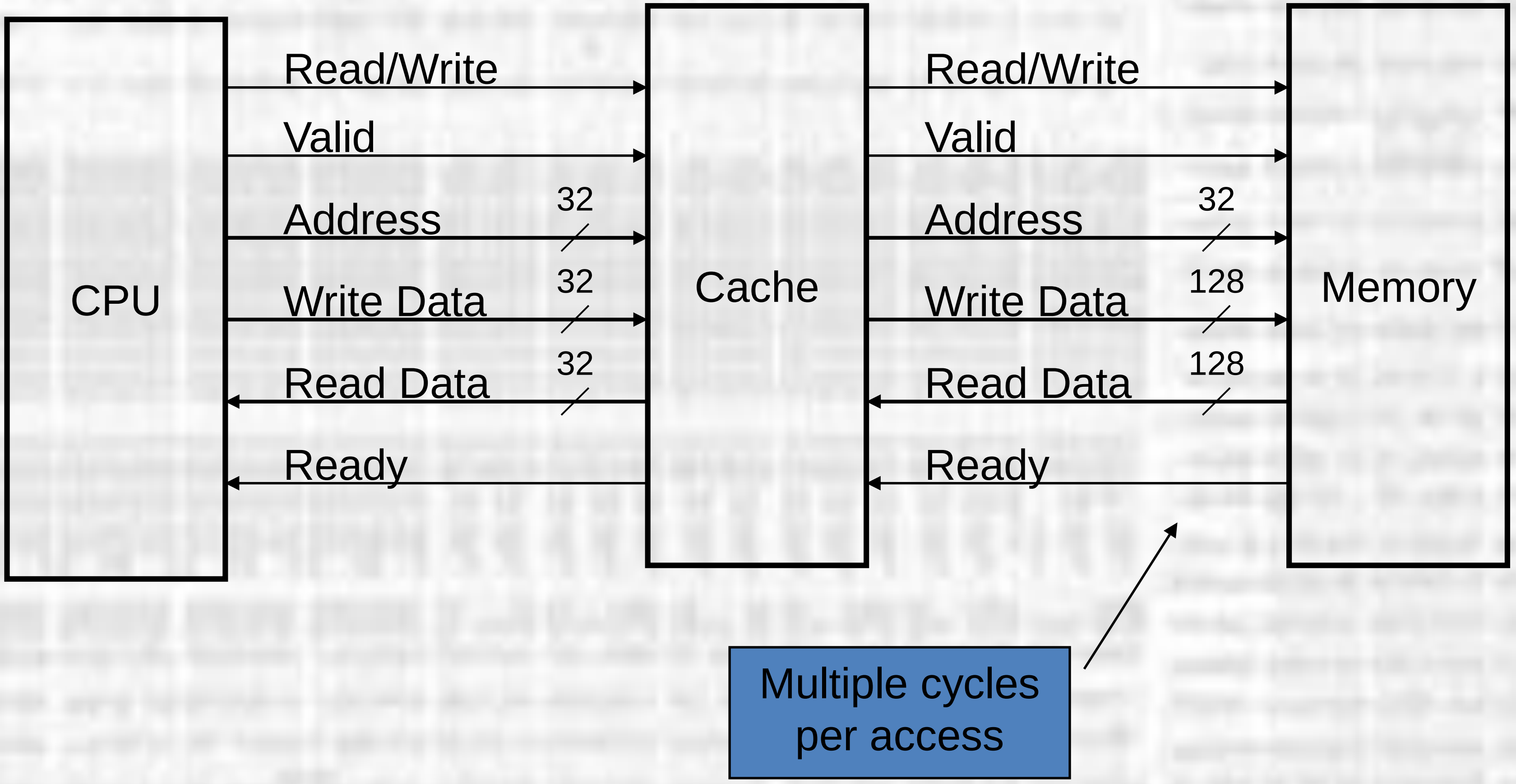
- Example cache characteristics



Cache Control

Finite State Machine

- Example cache characteristics

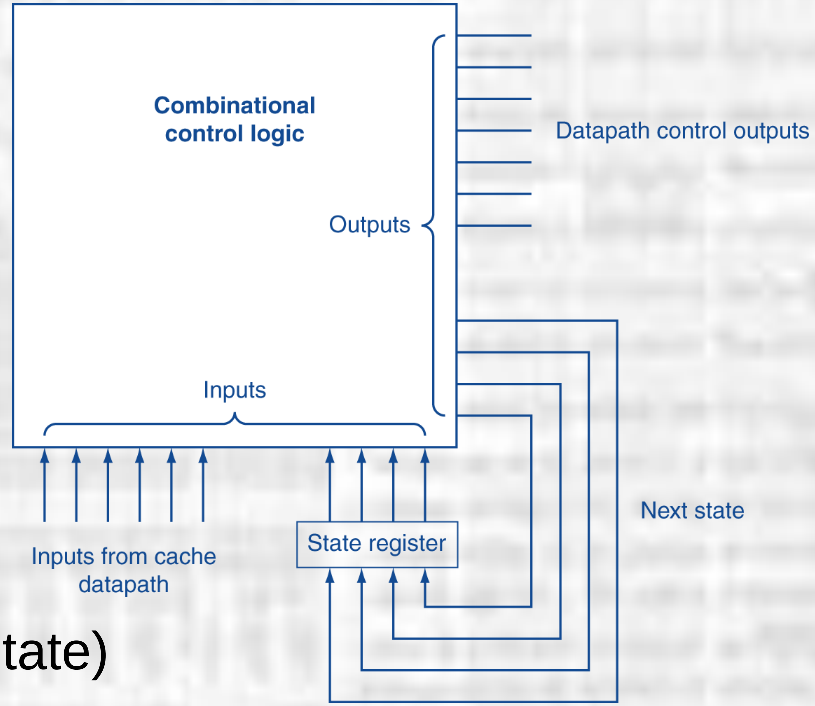


Cache Control

Finite State Machine

- Finite State Machine

- Use an FSM to sequence control steps
- Set of states, transition on each clock edge
  - State values are binary encoded
  - Current state stored in a register
  - Next state
    = fn (current state, current inputs)
  - Control output signals = fn (current state)

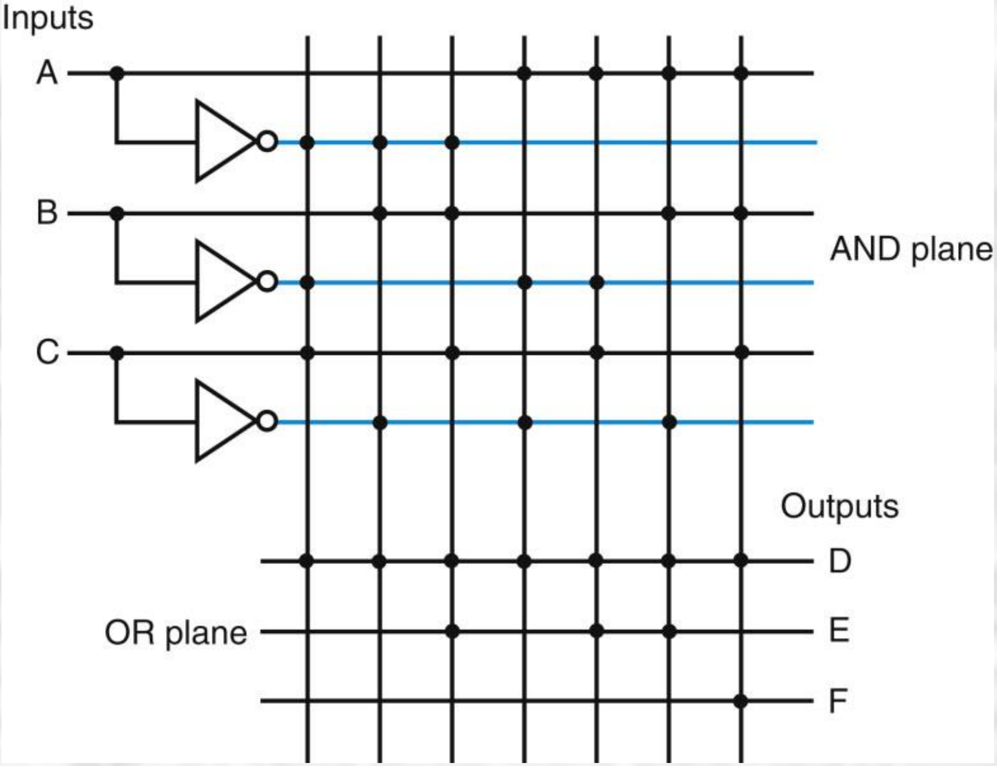


Cache Control

Finite State Machine

- Finite State Machine

- Programmable Logic Array



Cache Control

Finite State Machine

- FSM for Cache Controller
  - 4 states

  - Idle
    - Wait for a valid read or write request

  - Compare Tag
    - Tests the requested read/write address tag
    - Hit
      - Assert cache ready signal
      - Read / write word
      - On write – set dirty bit
      - On both – write valid and tag bits - Why?
      - Return to Idle
    - Miss
      - If block is dirty → Write Back
      - If not dirty → Allocate

Cache Control

Finite State Machine

- FSM for Cache Controller
  - 4 states

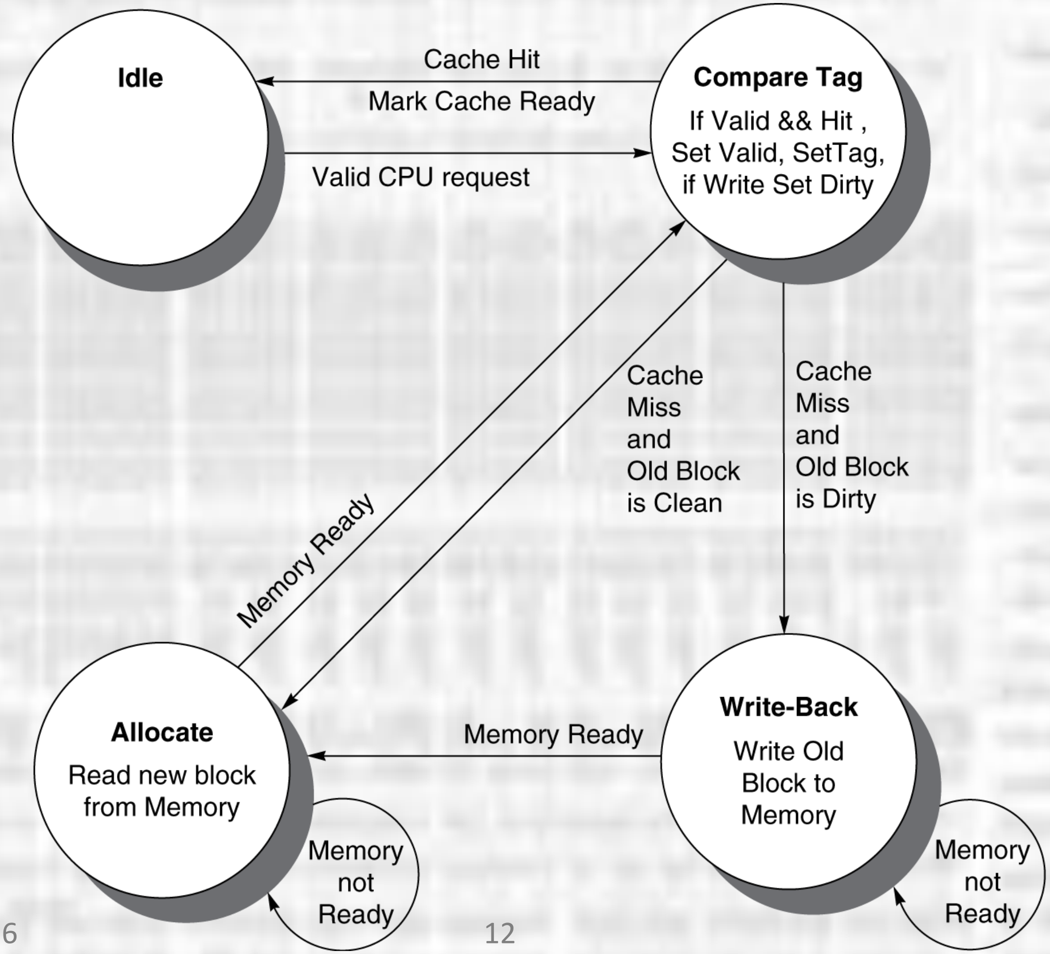
  - Write Back
    - Write the block back to memory – 128 bits
    - When memory ready signal asserted (ready for access) → Allocate

  - Allocate
    - Fetch new block from memory
    - When memory ready signal asserted (read complete) → Compare Tag

Cache Control

Finite State Machine

- FSM for Cache Controller
  - 4 states



Cache Control

Coherency

- Multi-processor Systems and Caches
  - Two CPU cores sharing a common higher level of memory

Time step	Event	CPU A's cache	CPU B's cache	Memory
0				0
1	CPU A reads X	0		0
2	CPU B reads X	0	0	0
3	CPU A writes 1 to X	1	0	1

Cache Control

Coherency

- Multi-processor Systems and Caches
  - Coherency
    - Informally: Reads return most recently written value
    - Formally:
      - P writes X; P reads X (no intervening writes)
        → read returns written value
      - P1 writes X; P2 reads X (sufficiently later)
        → read returns written value
      - P1 writes X, P2 writes X
        → all processors see writes in the same order
    - End up with the same final value for X

Cache Control

Coherency

- Multi-processor Systems and Caches
  - Operations performed by caches in multiprocessors to ensure coherence
    - Migration of data to local caches
      - Reduces bandwidth for shared memory
    - Replication of read-shared data
      - Reduces contention for access

Cache Control

Coherency

- Multi-processor Systems and Caches
  - 2 approaches
    - Snooping protocols
      - Each cache monitors bus reads/writes
    - Directory-based protocols
      - Caches and memory record sharing status of blocks in a directory

Cache Control

Coherency

- Multi-processor Systems and Caches

  - Write Invalidate Protocol
  - Cache gets exclusive access to a block when it is to be written
    - Broadcasts an invalidate message on the bus
    - Subsequent read in another cache misses
    - Owning cache supplies updated value

CPU activity	Bus activity	CPU A's cache	CPU B's cache	Memory
				0
CPU A reads X	Cache miss for X	0		0
CPU B reads X	Cache miss for X	0	0	0
CPU A writes 1 to X	Invalidate for X	1		0
CPU B read X	Cache miss for X	1	1	1

Cache Control

Coherency

- Multi-processor Systems and Caches
  - Block Size Impacts on snooping protocols
    - Most protocols invalidate whole blocks
      - 2 processors accessing different words in a common block → dithering
      - → smaller blocks desirable

Cache Control

Memory Dependability

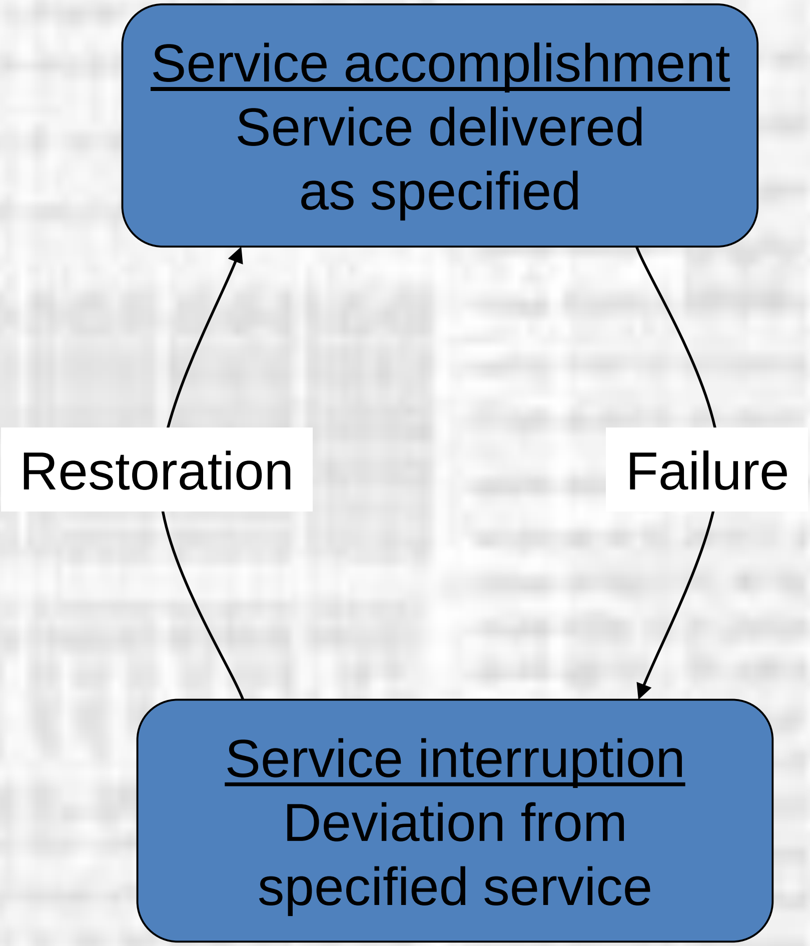
- Failure

  - Service Accomplishment
    - Service provided as specified

  - Service Interruption
    - Deviation from specified service

  - Failure
    - Transition from accomplishment to interruption

  - Restoration
    - Transition from interruption to accomplishment


Service accomplishment
Service delivered as specified
Restoration
Failure
Service interruption
Deviation from specified service

Cache Control

Memory Dependability

- Dependability Measures
  - Reliability: mean time to failure (MTTF)
  - Service interruption: mean time to repair (MTTR)
  - Mean time between failures
    - MTBF = MTTF + MTTR
  - Availability = MTTF / (MTTF + MTTR)
  - Improving Availability
    - Increase MTTF: fault avoidance, fault tolerance, fault forecasting
    - Reduce MTTR: improved tools and processes for diagnosis and repair

Cache Control

Memory Dependability

- Dependability Measures
  - 9’s measure
  - 90%, 99%, 99.9%, 99.99%
  - Five 9’s availability
    - 365 days/yr x 24hrs/day x 60min/hr = 526,000 min/yr
    - 99.999% availability → 0.001% repair
    - Five 9’s → 5.29 min/year of repair time

Cache Control

Memory Dependability

- Faults
  - Failure of a component
  - To improve MTTF
    - Fault Avoidance
      - Prevent faults by construction
      - High reliability parts
    - Fault Tolerance
      - Use redundancy to allow the system to continue
    - Fault Forecasting
      - Predict faults and resolve before occurrence
      - Early detection measures – current drain, voltage levels
      - Periodic maintenance

Cache Control

Memory Dependability

- Redundancy in Memories
  - Chip level
    - Extra rows and columns
    - Non-functioning portions of the circuit are replaced
    - Decode circuitry is “re-directed” to the replacement row/column
  - Flash
    - Flash can wear out after 100,000 r/w cycles
    - Wear leveling is used to ensure data is moved around in the flash
  - Bit Level
    - Error Detection
    - Error Correction

Cache Control

Memory Dependability

- Bit Level Error Detection
  - Parity Checking
  - Add a bit to every byte – parity bit
  - Force the parity bit to be either 1 or 0 depending on the eveness or oddness of the other bits
  - Even parity – sum of bits is even
  - Odd parity – sum of bits is odd
  - 1001 0110 → 1001 0110 0 To create even parity
  - 1001 0111 → 1001 0111 1 To create even parity
  - On reads – check to make sure parity is correct, if not → error

Cache Control

Memory Dependability

- Bit Level Error Detection
  - Richard Hamming
    - Hamming distance
    - # of bits that are different between 2 numbers
    - 1001 0110 and 1000 0110 differ by 1 bit → Hamming distance = 1
    - 1001 0110 and 1000 0010 differ by 2 bits → Hamming distance = 2
  - Parity can find 1,3,5,7 bit errors
    - 3,5,7 bit errors are very rare
  - More sophisticated schemes are required to detect even number bit errors and to correct some kinds of errors

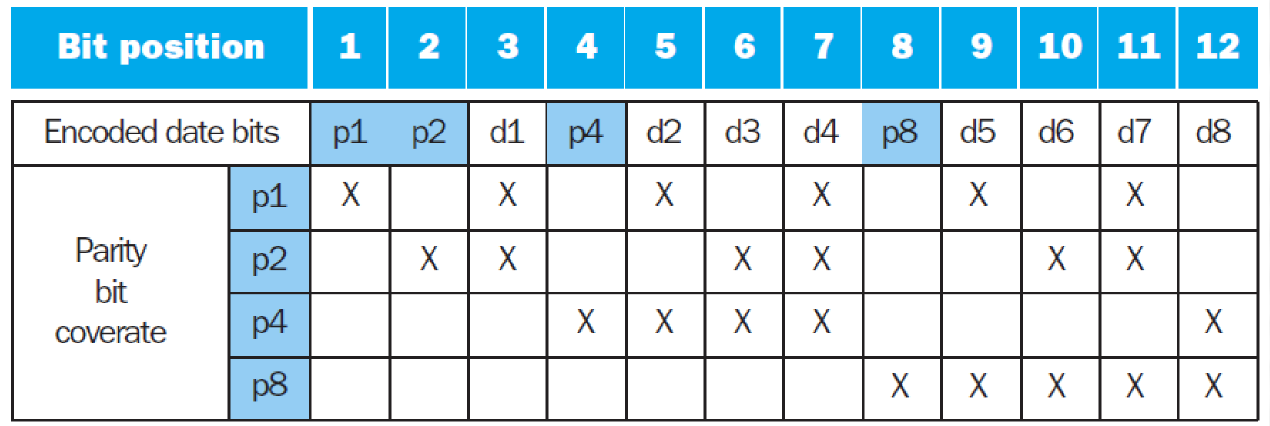
Cache Control

Memory Dependability

- Bit Level Error Correction

  - Hamming Error Correction Code (ECC)
    - Inset 4 parity bits into each byte → Hamming distance = 3
    - Each parity bit matches up to 5 bits in the new 12 bit structure

    - Parity code indicates which bit is flipped in a single bit error
      - The bit can then be fixed



Bit position		1	2	3	4	5	6	7	8	9	10	11	12
Encoded date bits		p1	p2	d1	p4	d2	d3	d4	p8	d5	d6	d7	d8
Parity bit coverate	p1	X		X		X		X		X		X	
	p2		X	X			X	X			X	X	
	p4				X	X	X	X					X
	p8								X	X	X	X	X

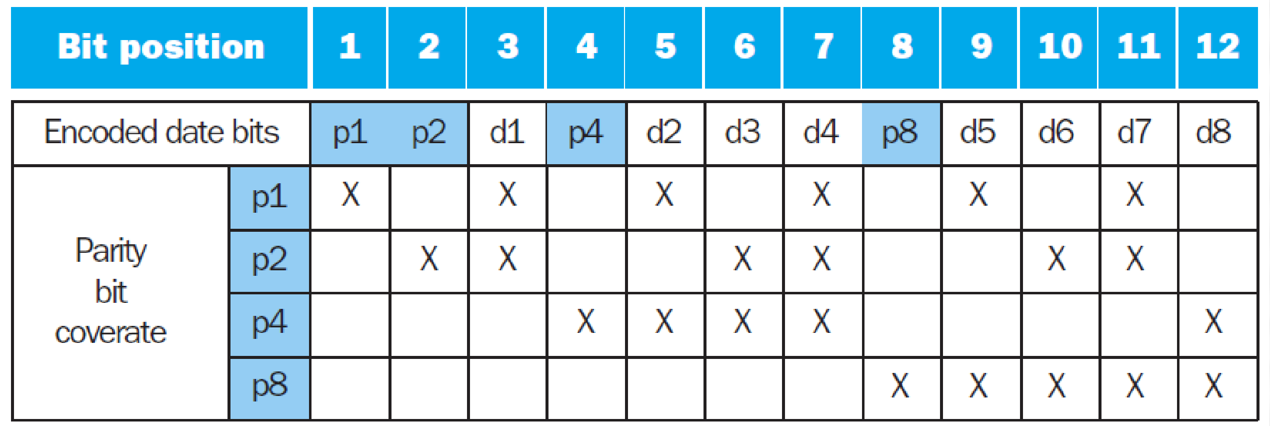
Cache Control

Memory Dependability

- Bit Level Error Correction

  - Example - 1001 1010 → _ _ 1_001_1010

    - With ECC : p1 → _ _1_001_1010 → 0 _1_001_1010
    - With ECC : p2 → 0 _1_001_1010 → 0 11_001_1010
    - With ECC : p4 → 0 11_001_1010 → 0 111001_1010
    - With ECC : p8 → 0 111001_1010 → 0 111001010 10



Bit position		1	2	3	4	5	6	7	8	9	10	11	12
Encoded date bits		p1	p2	d1	p4	d2	d3	d4	p8	d5	d6	d7	d8
Parity bit coverate	p1	X		X		X		X		X		X	
	p2		X	X			X	X			X	X	
	p4				X	X	X	X					X
	p8								X	X	X	X	X

Cache Control

Memory Dependability

- Bit Level Error Correction

  - Example - 1001 1010 → 011100101010

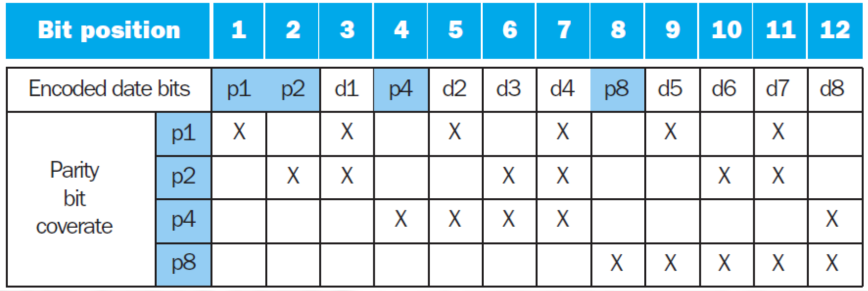
    assume we actually read 011100101110

    p1 = 011100101110 = 4 – OK - 0

    p2 = 011100101110 = 5 – not OK - 1

    p4 = 011100101110 = 2 – OK - 0

    p8 = 011100101110 = 3 – not OK - 1



Bit position		1	2	3	4	5	6	7	8	9	10	11	12
Encoded date bits		p1	p2	d1	p4	d2	d3	d4	p8	d5	d6	d7	d8
Parity bit coverate	p1	X		X		X		X		X		X	
	p2		X	X			X	X			X	X	
	p4				X	X	X	X					X
	p8								X	X	X	X	X

Parity result is 1010b = 10 → d6 is wrong

Flip d6

011100101110 → 011100101010, correct

Cache Control

Memory Dependability

- Bit Level Error Detection

  - Dual Error Detection
    - Add an additional parity bit for the whole word (pn)
    - Make Hamming distance = 4
  - Decoding:
    - Let H = SEC parity bits
      - H even, pn even, no error
      - H odd, pn odd, correctable single bit error
      - H even, pn odd, error in pn bit
      - H odd, pn even, double error occurred

  - ECC DRAM uses SEC/DEC with 8 bits protecting each 64 bits