Microprocessor & Computer Architecture (µpCA)

UE19CS252

Dr. D. C. Kiran

Department of
Computer Science and Engineering

Microprocessor & Computer Architecture (µpCA)

Unit 3: Read and Write Policy

Dr. D. C. Kiran
Department of Computer Science and Engineering

Microprocessor & Computer Architecture (µpCA)

Syllabus

~~**Unit 1: Basic Processor Architecture and Design**~~

~~**Unit 2: Pipelined Processor and Design**~~

Unit 3: Memory

- ~~Memory Hierarchy~~
- ~~Principles of Locality~~
- ~~Cache Design Principles~~

Mapping Functions

- ~~Direct Mapping~~
- ~~Full Associative Mapping~~
- ~~Set Associative Mapping~~
- ~~Cache Replacement Policy~~
- **Read & Write Policy**

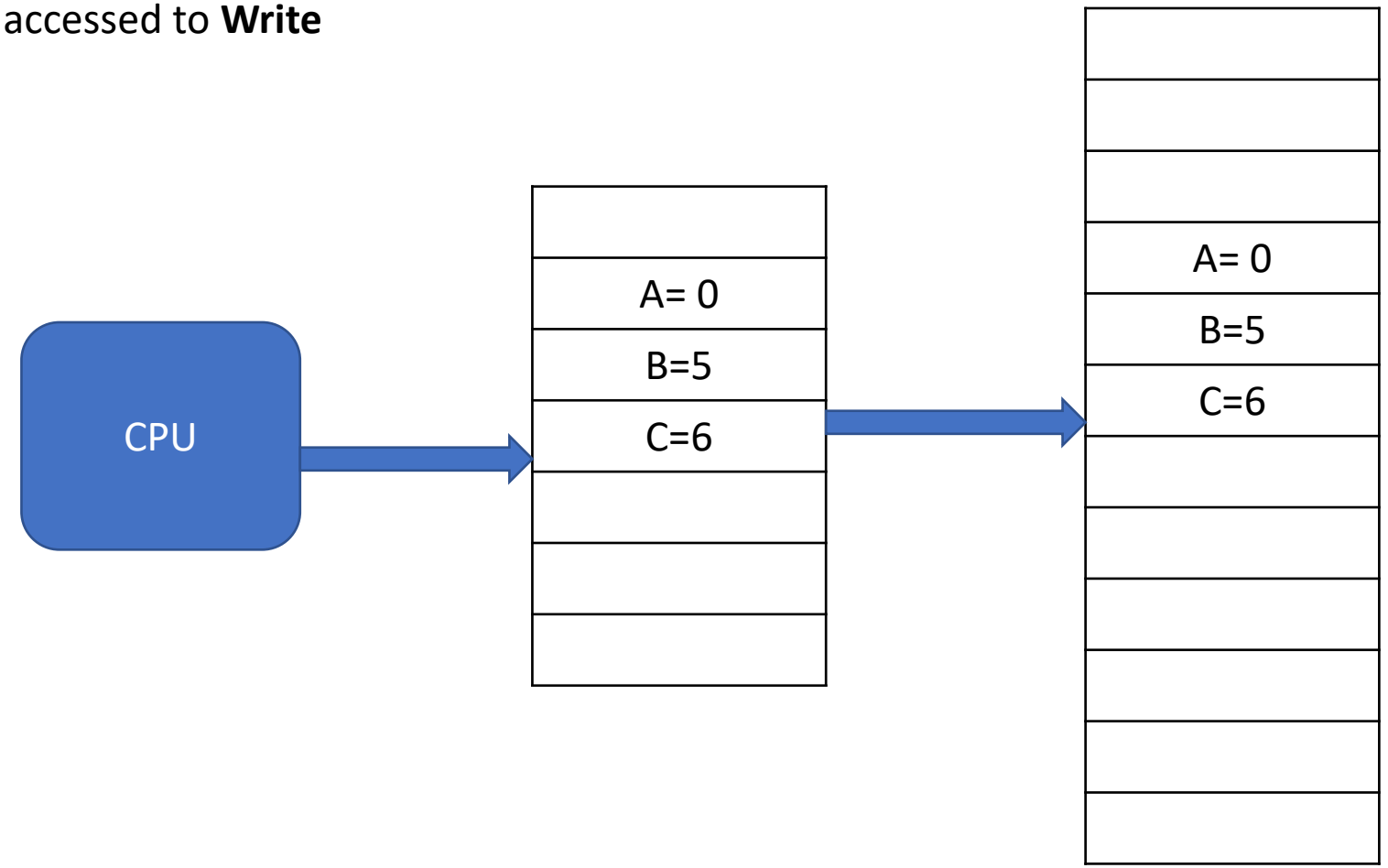
Unit 4: Input/Output Device Design

Unit 5: Advanced Architecture

Read vs Write

Consider the statement A=B+C

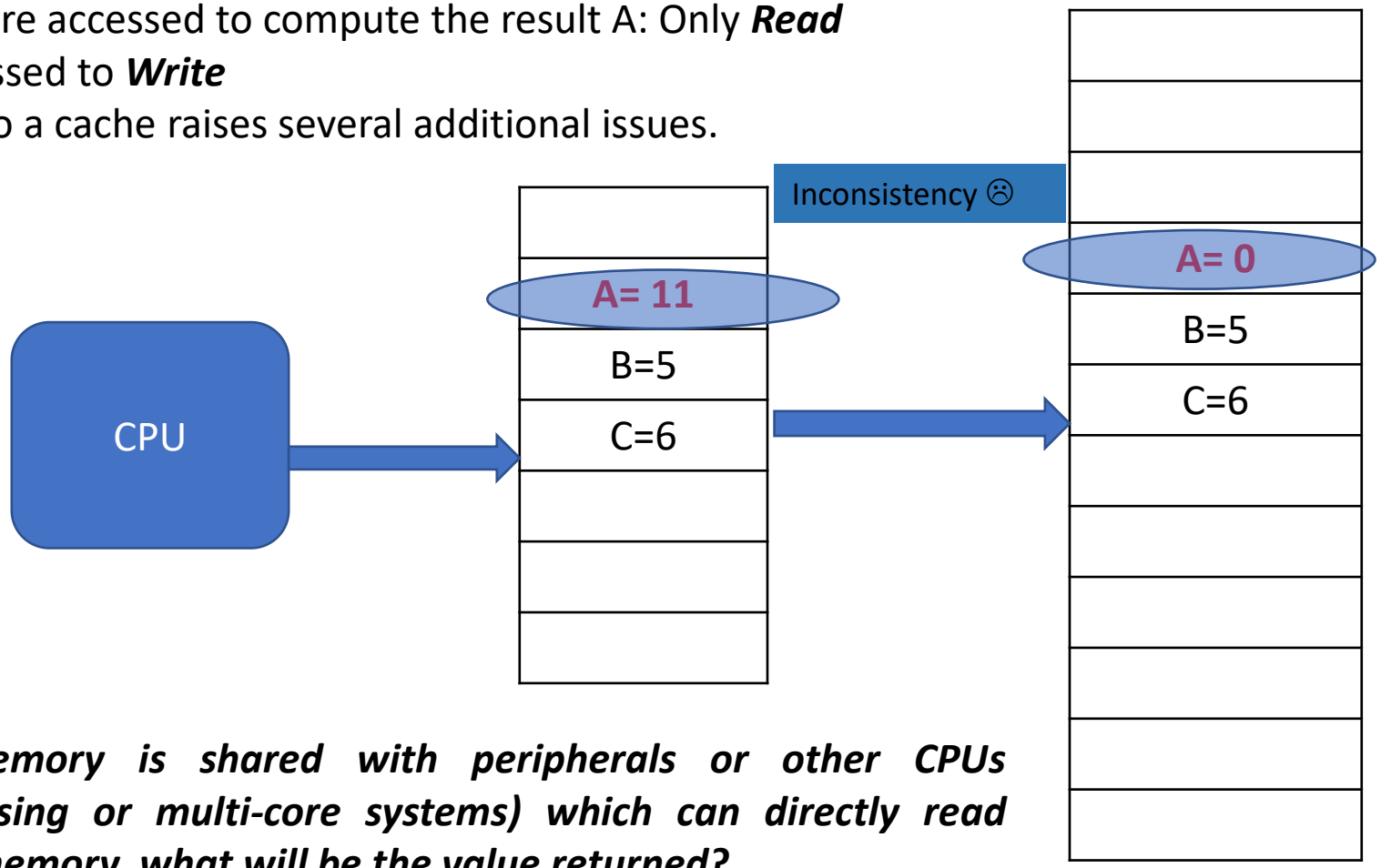
B and C are accessed to compute the result A: Only **Read**
A is accessed to **Write**



Read vs Write

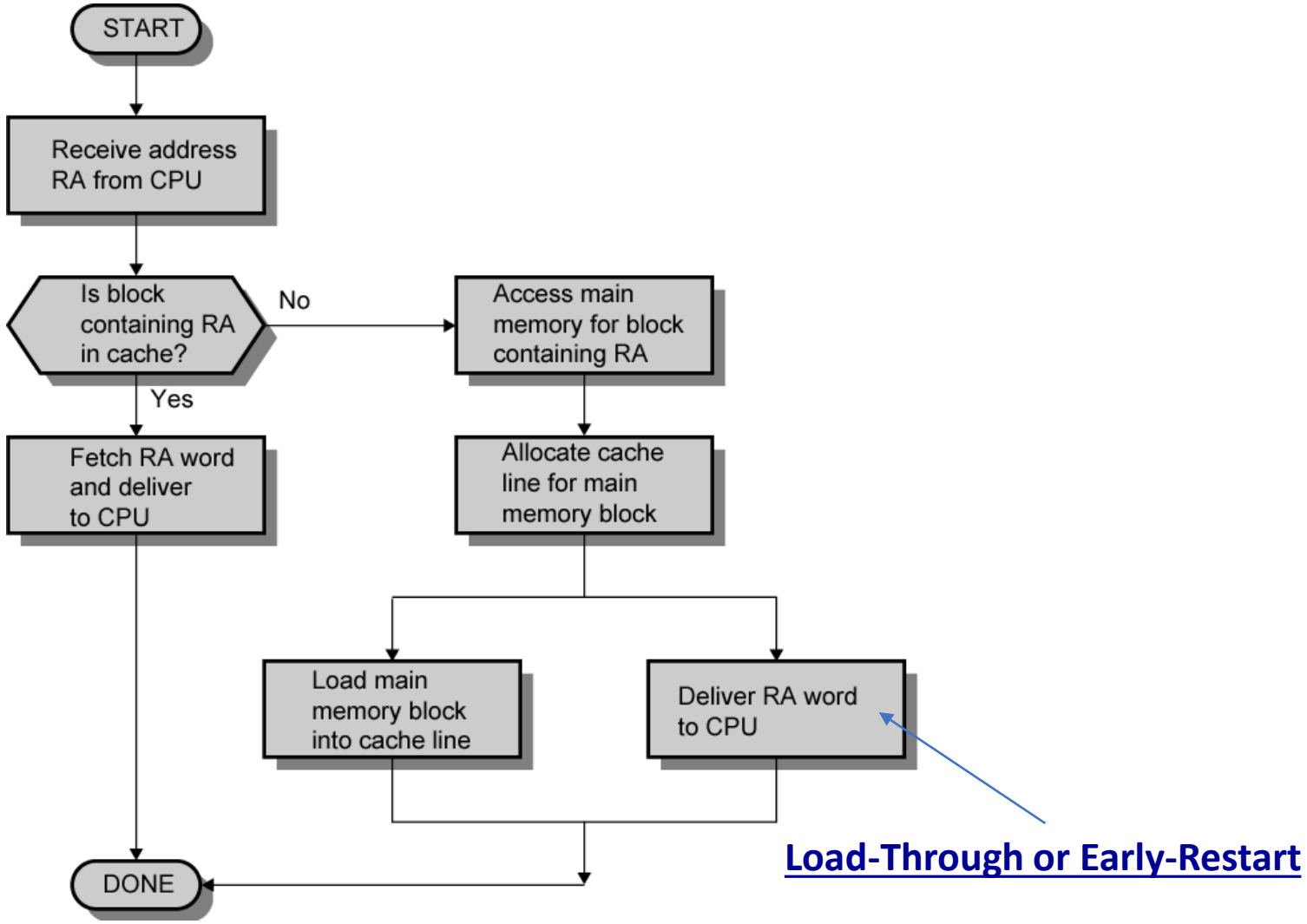
Consider the statement A=B+C

B and ***C*** are accessed to compute the result A: Only ***Read***
A is accessed to ***Write***
Writing to a cache raises several additional issues.



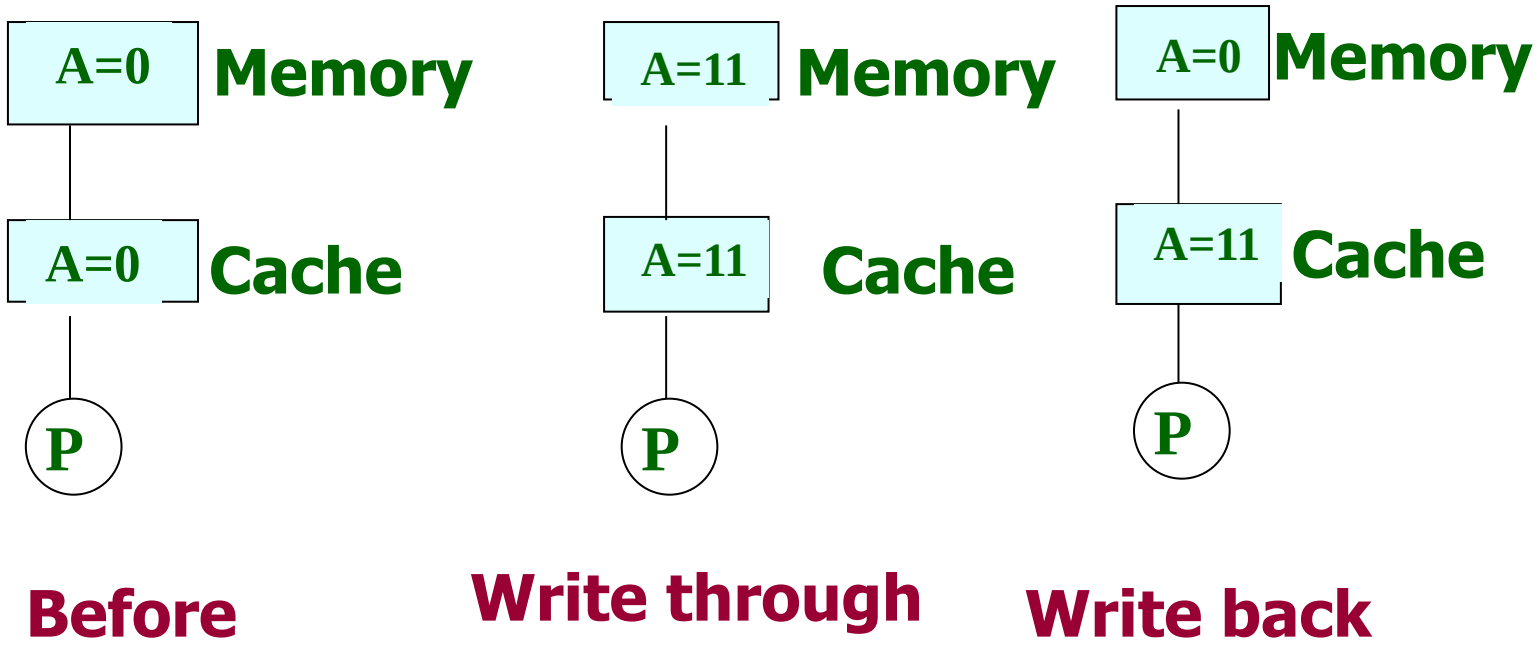
If main memory is shared with peripherals or other CPUs (multiprocessing or multi-core systems) which can directly read from main memory, what will be the value returned?

Read: Hit or Miss

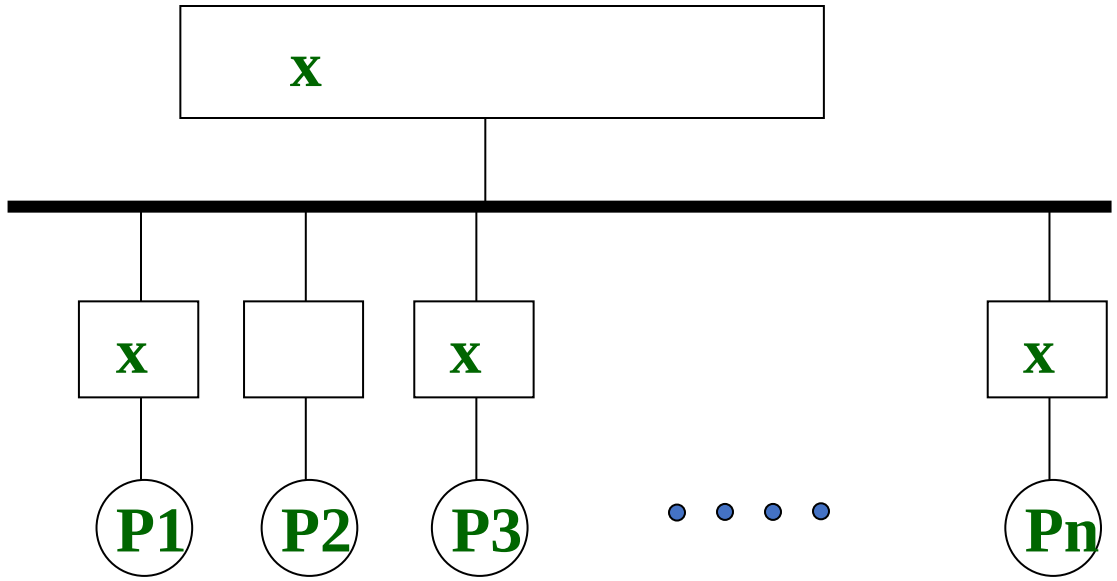


Load-Through or Early-Restart

Write On Hit


A=0
Memory
A=0
Cache
P
A=11
Memory
A=11
Cache
P
A=0
Memory
A=11
Cache
P
Before
Write through
Write back

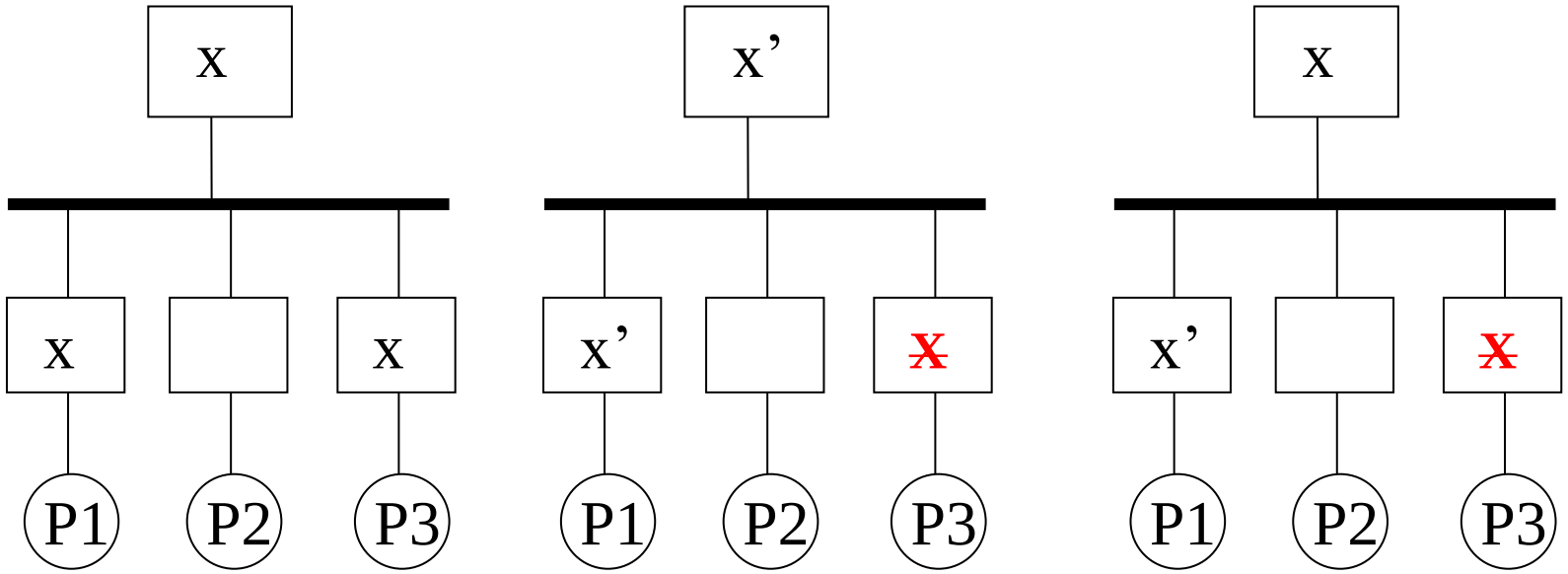
Write on Hit: Cache Coherence



-Multiple copies of x

-What if P1 updates x?

Write on Hit-Invalidate



Before

Write Through

Write back

X= Original X
X'= Updated X
~~X~~ = Invalid X

Write Through on Hit

Valid Bit	TAG	Data
1	10100	A=0
1	10100	B=5
1	10100	C=6

- Each Line in the cache has Valid Bit.
- **Valid Bit** indicates that no other program has changed the Data.

10100…..1010..0000	**A= 0**
10100…..1010..0001	B=5
10100…..1010..0010	C=6
00010…..1010..0000	P=100

Write Through on Hit

Updated Simultaneously

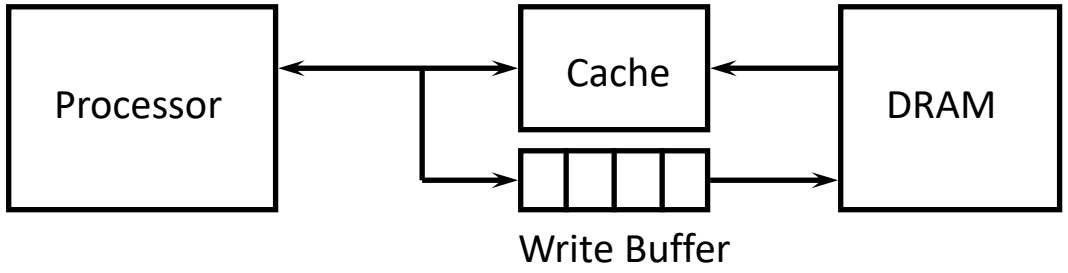
Valid Bit	TAG	Data
1	10100	A=11
1	10100	B=5
1	10100	C=6

10100…..1010..0000	**A= 11**
10100…..1010..0001	B=5
10100…..1010..0010	C=6
00010…..1010..0000	P=100

- Avoid Inconsistency.
- Purpose of having Cache is dishonored.
- Some design, buffer the update and continue executing other instruction

Write Buffer for Write Through on Hit

- Some design, buffer the update and continue executing other instruction



Write Back on Hit: With Validation

Before Write A

Valid Bit	Dirty Bit	TAG	Data
1	**0**	**10100**	A=0
1	**0**	**10100**	B=5
1	**0**	**10100**	C=6

- Each Line in the cache has Valid Bit and Dirty Bit
- **Valid Bit** indicates that no other program has changed the Data
- **Dirty Bit** indicates that Data is written or not written

10100…..1010..0000	**A= 0**
10100…..1010..0001	B=5
10100…..1010..0010	C=6
00010…..1010..0000	P=100

Write Back on Hit: With Validation

After Write A

Valid Bit	Dirty Bit	TAG	DAta
1	**1**	**10100**	A=11
1	**0**	**10100**	B=5
1	**0**	**10100**	C=6

10100…..1010..0000	**A= 0**
10100…..1010..0001	B=5
10100…..1010..0010	C=6
00010…..1010..0000	P=100

Write Back on Hit: With Validation

To Replace A by P

Valid Bit	Dirty Bit	TAG	DAta
1	0	00010	P=11

Step 1: Update A in memory to 11
Step 2: Replace A with P
Step3: Set Dirty Bit to 0

10100…..1010..0000	**A= 11**
10100…..1010..0001	B=5
10100…..1010..0010	C=6
00010…..1010..0000	P=100

Write Back on Miss: With Validation

On Miss : Two Possibility

1: Bring the Block on the Line **(Write Allocate)**
2: Update the Block in the Memory Directly **(Write No Allocate)**

Write Allocate On Miss

P need to be written?

Valid Bit	Dirty Bit	TAG	DAta
1	1	10100	A=11
1	0	10100	B=5
1	0	10100	C=6

Replace P by A

10100.....1010..0000	**A= 11**
10100.....1010..0001	B=5
10100.....1010..0010	C=6
00010.....1010..0000	P=100

Write Allocate On Miss

Update P t0 200
P need to be written?

Valid Bit	Dirty Bit	TAG	DAta
1	1	00010	P=200
1	0	10100	B=5
1	0	10100	C=6

Replace P by A

10100.....1010..0000	A= 11
10100.....1010..0001	B=5
10100.....1010..0010	C=6
00010.....1010..0000	P=100

Write No Allocate On Miss

P need to be written?

Valid Bit	Dirty Bit	TAG	DAta
1	1	10100	A=11
1	0	10100	B=5
1	0	10100	C=6

Update P directly in Memory

10100.....1010..0000	A= 11
10100.....1010..0001	B=5
10100.....1010..0010	C=6
00010.....1010..0000	P=200

Write No Allocate On Miss

This is good when data is written but not immediately used again, in which case there's no point to load it into the cache yet.

```
for(i=0;i<10;i++)
a[i]=i*2;
```

Where ith Location is referred to write only once.

Think About it!!!

Writing Cache Friendly Code

- Two major rules:
- Repeated references to data are good (temporal locality)
- Stride-1 reference patterns are good (spatial locality)
- Example: 4-byte words, 4-word cache blocks

```
int sum_array_rows(int a[M][N])
{
  int i, j, sum = 0;

  for (i = 0; i < M; i++)
     for (j = 0; j < N; j++)
        sum += a[i][j];
  return sum;
}
```

```
int sum_array_cols(int a[M][N])
{
  int i, j, sum = 0;

  for (j = 0; j < N; j++)
     for (i = 0; i < M; i++)
        sum += a[i][j];
  return sum;
}
```

Miss rate = 1/4 = 25%

Miss rate = 100%

Next Session

Performance Analysis

THANK YOU

Dr. D. C. Kiran
Department of Computer Science and Engineering

dckiran@pes.edu

9829935135