GLOBAL EDGE

Intelligence Of Things

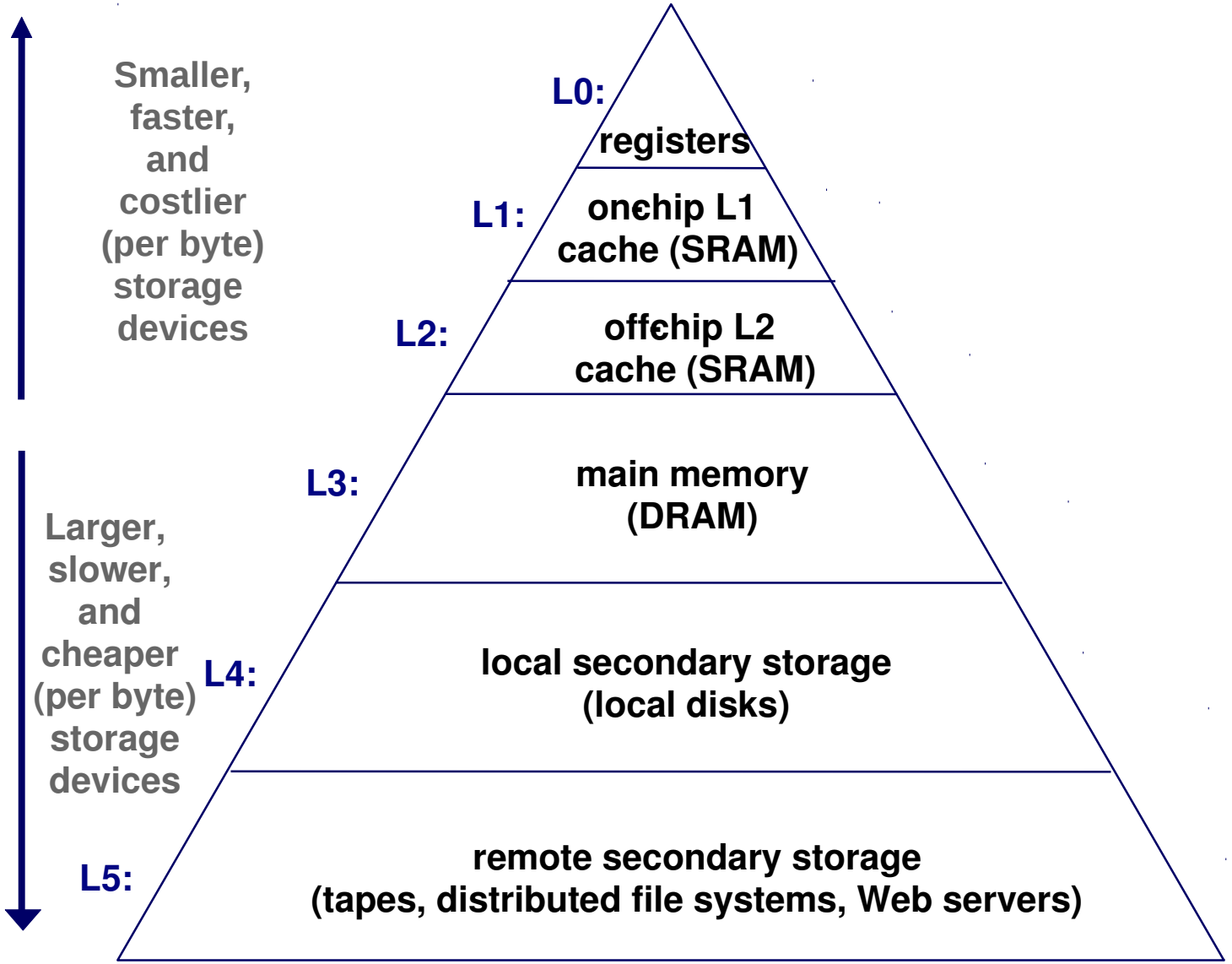
ARM Memory Management Unit

By,
CM Naveen Kumar Reddy.

Overview

Memory Hierarchy
Memory Management Unit
page
Cache

Memory Hierarchy



Memory Hierarchy (cont..)

DRAM/SRAM is “volatile”: contents lost if power lost.

The primary level is *main memory*. It includes volatile components like SRAM and DRAM, and nonvolatile components like flash memory.
The purpose of main memory is to hold programs while they are running on a system.

Both retain data till the time they are supplied with power.

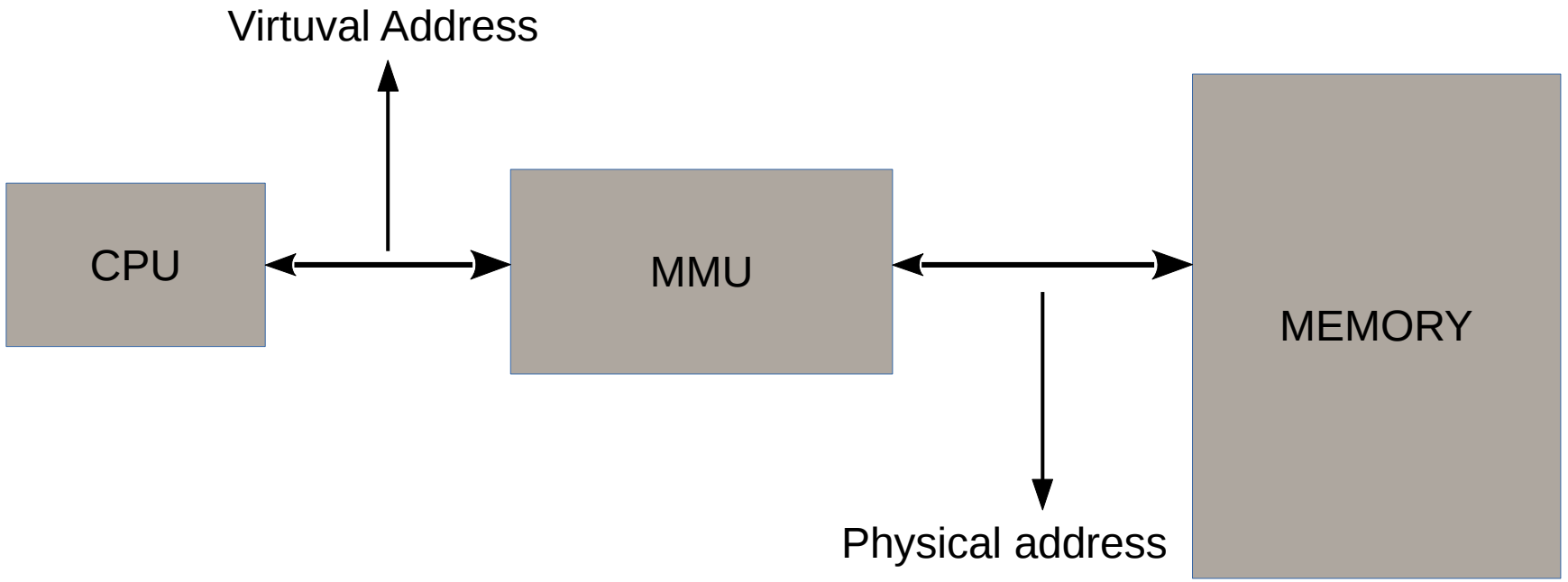
SRAMs are used in Caches because of higher speed.

DRAMs are used for main memory in a PC because of higher densities.

Disks are “non-volatile”: contents survive power outages.

The next level is *secondary storage*—large, slow, relatively inexpensive mass storage devices such as disk drives or removable memory.

Memory Management Unit (MMU)

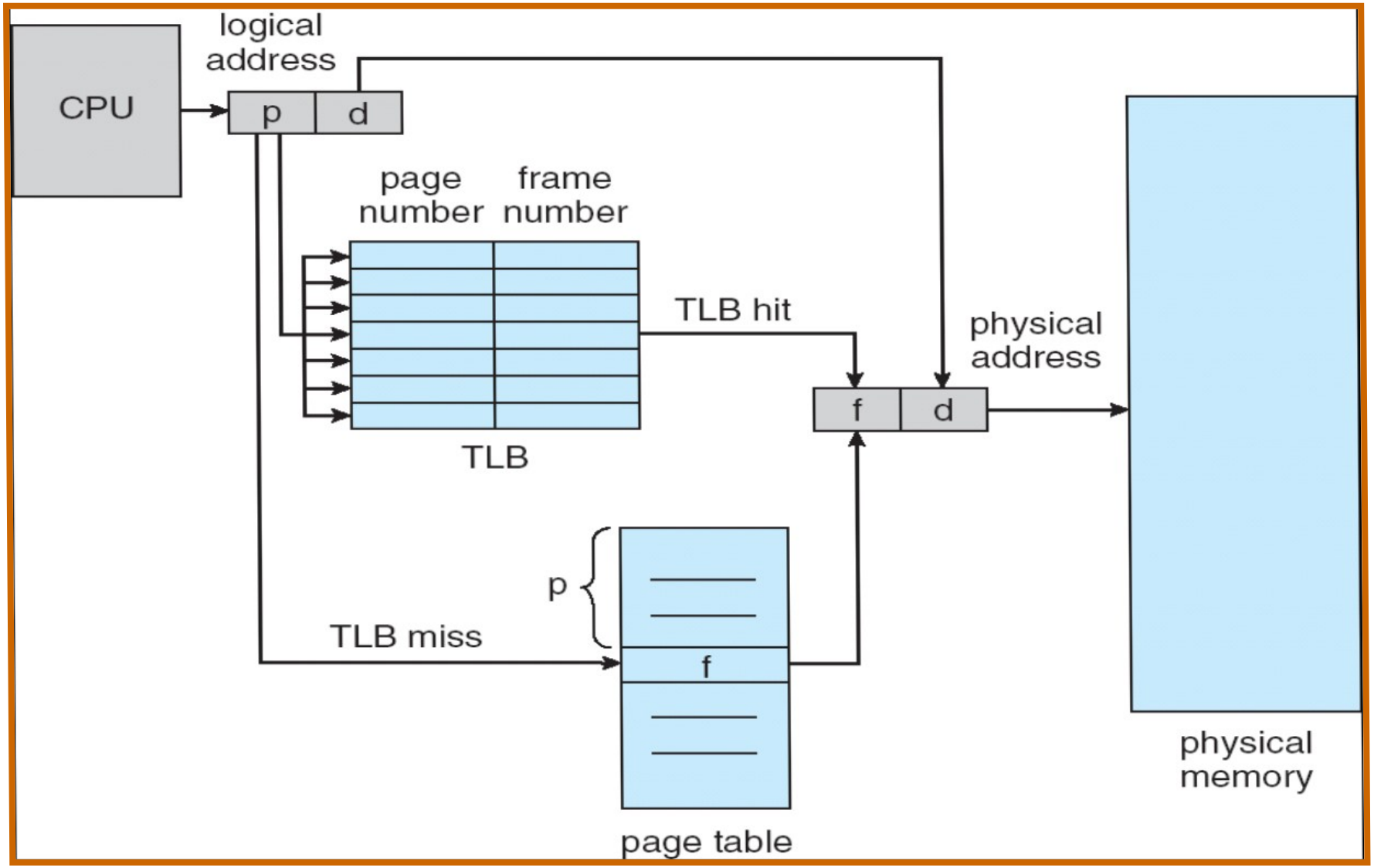


The Memory Management MMU it translates virtual addresses into physical addresses, and it controls memory access permissions.

Translation Look-aside Buffer (TLB) as part of the MMU hardware.

Translation Lookaside Buffer

TLB is consulted by the MMU when the CPU accesses a virtual address



Translation Lookaside Buffer

Translation Look-aside Buffer :In order to speed up the translation of virtual to physical addresses.

When an address must be translated, if it is found in the TLB then it can be very quickly mapped to the correct physical address.

On a TLB miss, then the address translation must be done using the full set of *page tables* stored in the system's main memory.

If the virtual address is in the TLB,the MMU can look up the physical resource (RAM or hardware).

If the virtual address is not in the TLB,the MMU will generate a ***page fault*** exception and interrupt the CPU.

If the address is in the TLB, but the permissions are insufficient,the MMU will generate a ***page fault.***

Page

Paging: divide Memory into fixed sized **Pages** for both Virtual and physical memory

•Another terminology
–A Virtual Page: ***page***
– A physical page: ***frame***

The ARM MMU supports four page sizes. The largest sizes are called sections and the smaller sizes are called pages:

Supersections: 16 MB memory blocks (24-bit offsets)
Sections: 1 MB memory blocks (20-bit offsets)
Large pages: 64 KB pages (16-bit offsets)
Small pages: 4 KB pages (12-bit offsets)

Paging and Translation addresses

Paging:-

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

Translation addresses:-

–Virtual address has two parts: *Virtual page number* and *offset*

–Virtual page number(VPN) is an index into a page table

–Page table determines page frame number (PFN)

–Physical address is PFN: :offset

Page fault

page table entry(PTE) tells the memory management unit whether a specific page is mapped into physical memory ("valid") or is not ("invalid"). An attempt to access an invalid page will cause the MMU to generate a **page fault.**

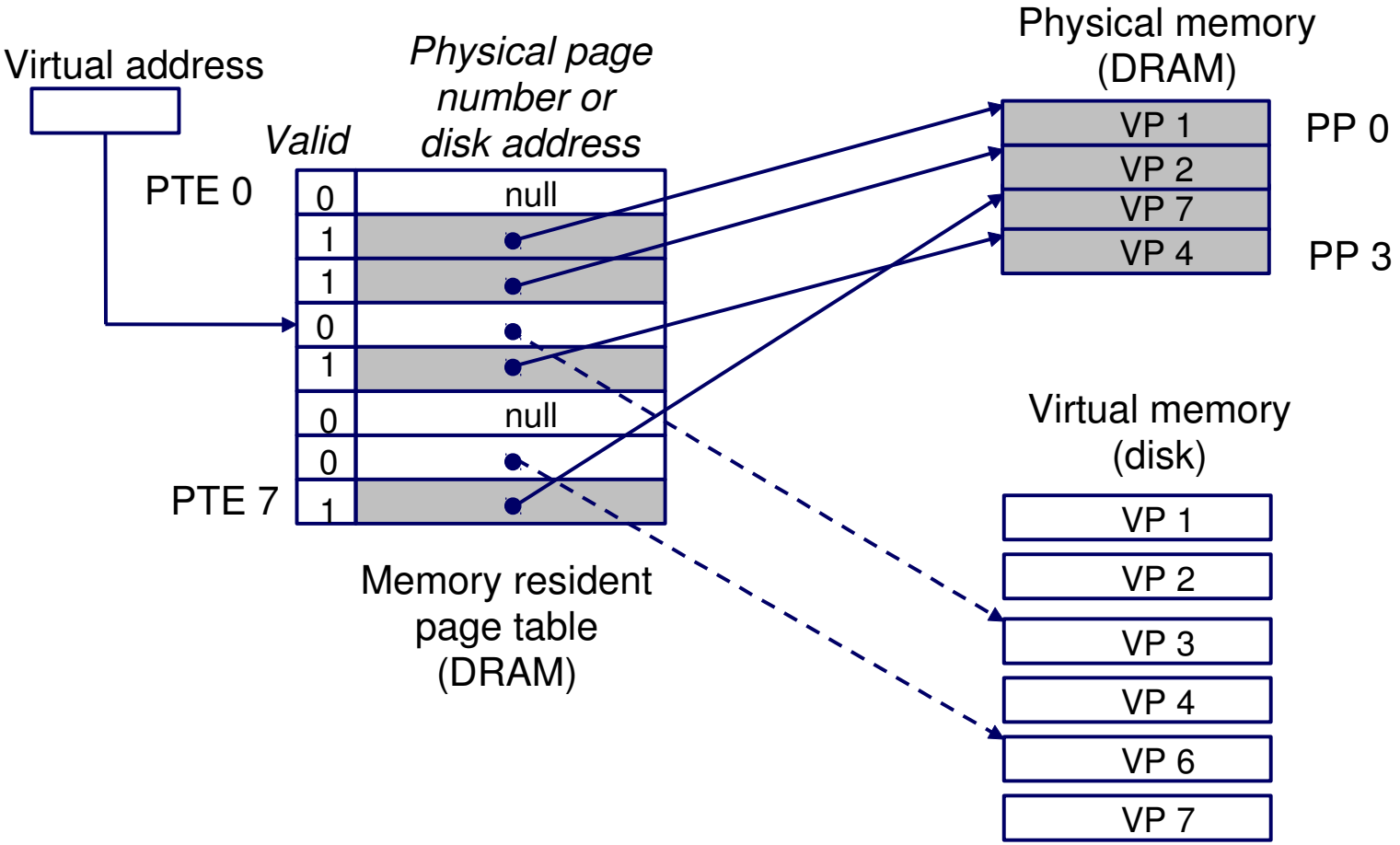
The virtual address is not mapped for the process requesting it.

The processes has insufficient permissions for the address.

The virtual address is valid, but swapped out.

Page Faults

- Example: An instruction references a word contained in VP 3, a miss that triggers a page fault exception



Demand paging

The operating system copies a disk page into physical memory only if an attempt is made to access it and that page is not already in memory (if a page fault occurs).

when a process starts executing and tries to load its first instruction, the operating system will get a page fault because the required page has not been loaded and mapped onto a page frame.

Bring a page into memory only when it is needed"

- Less I/O needed"
- Less memory needed "
- Faster response"
- More users"

Page is needed ⇒ reference to it"

- invalid reference ⇒ abort"
- not-in-memory ⇒ bring to memory

Caches

- Caches
  - Non-cached system
  - cached system
  - write buffer

Why Caches?

A cache holds this information to decrease the time required to access both instructions and data.

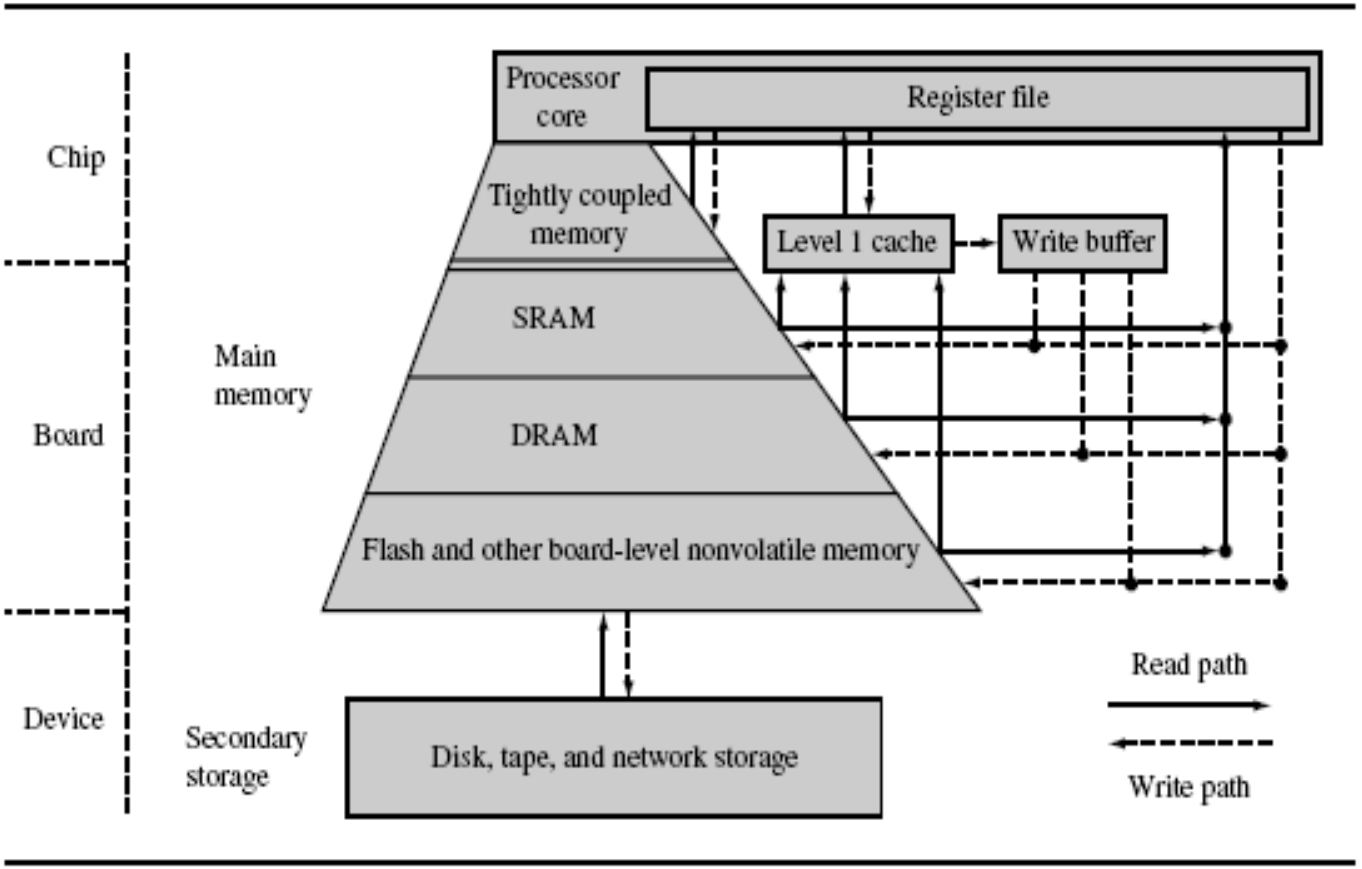
- To increase system performance.

- To hide the slower main memory.

- To handle large amounts of code and data at a faster and more efficient way.

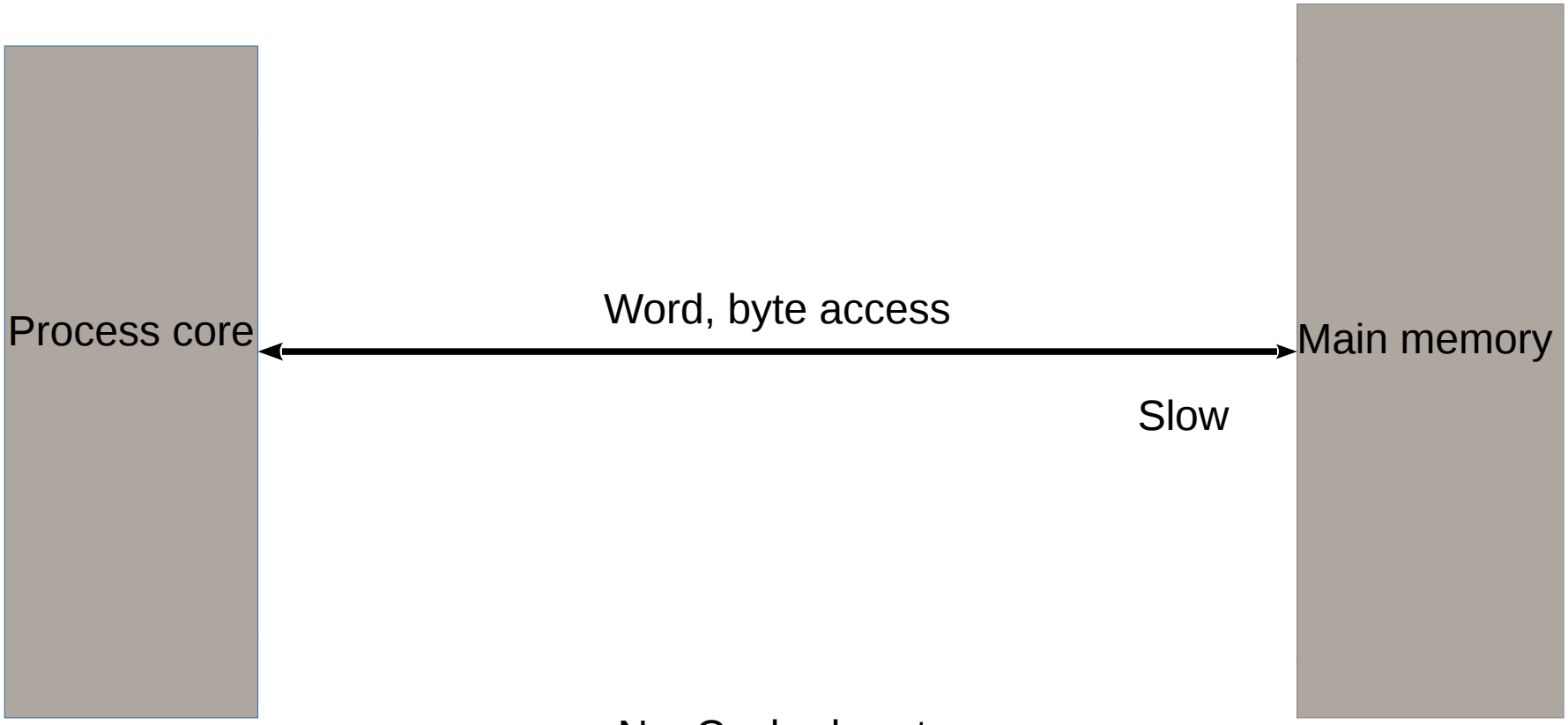
Caches

Cache is the array of high speed, very small memory sitting between processor(ARM) core and the main memory



Non-Cached system

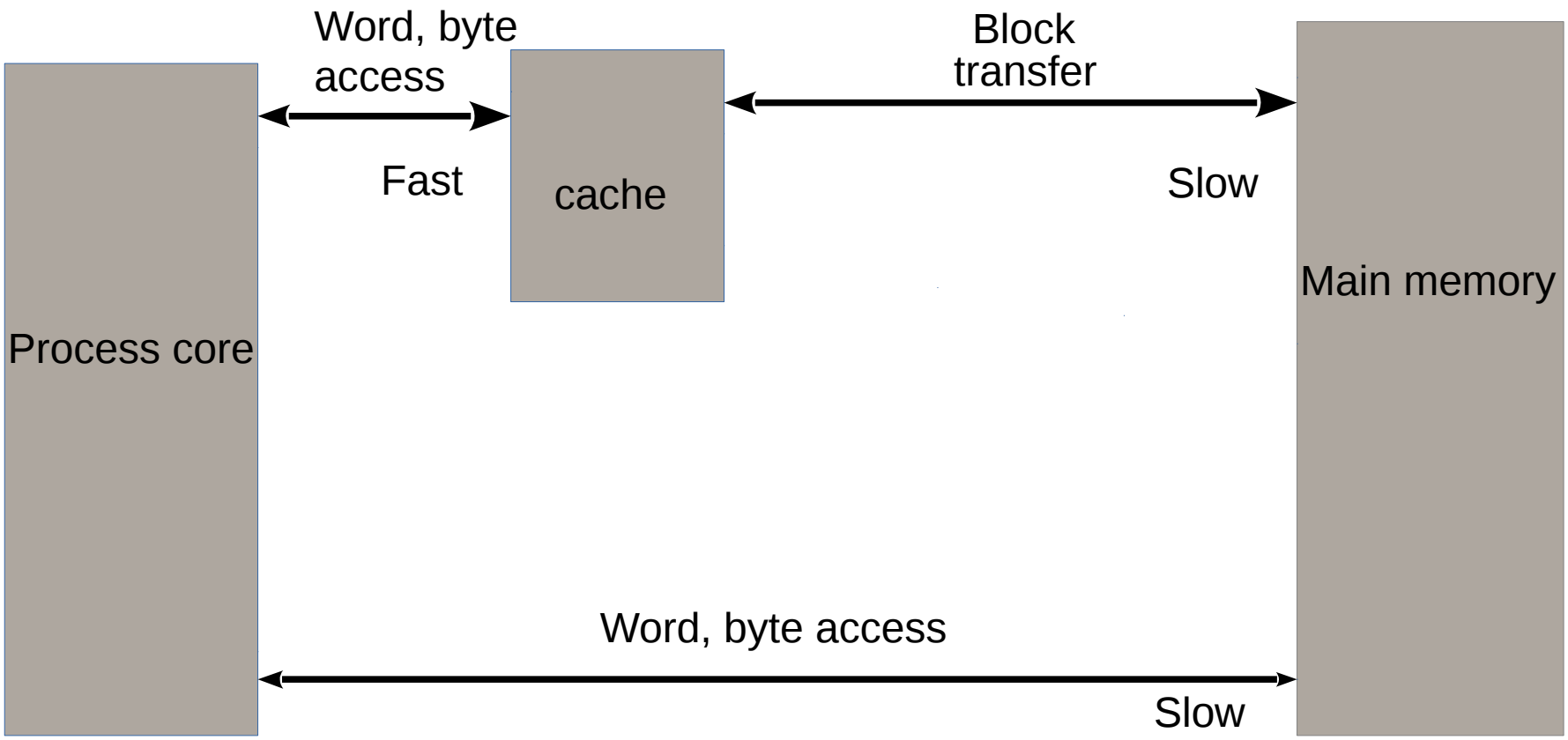
- Main memory is accessed directly by the processor core using the data types supported by the processor core.



NonCached system

Cached system

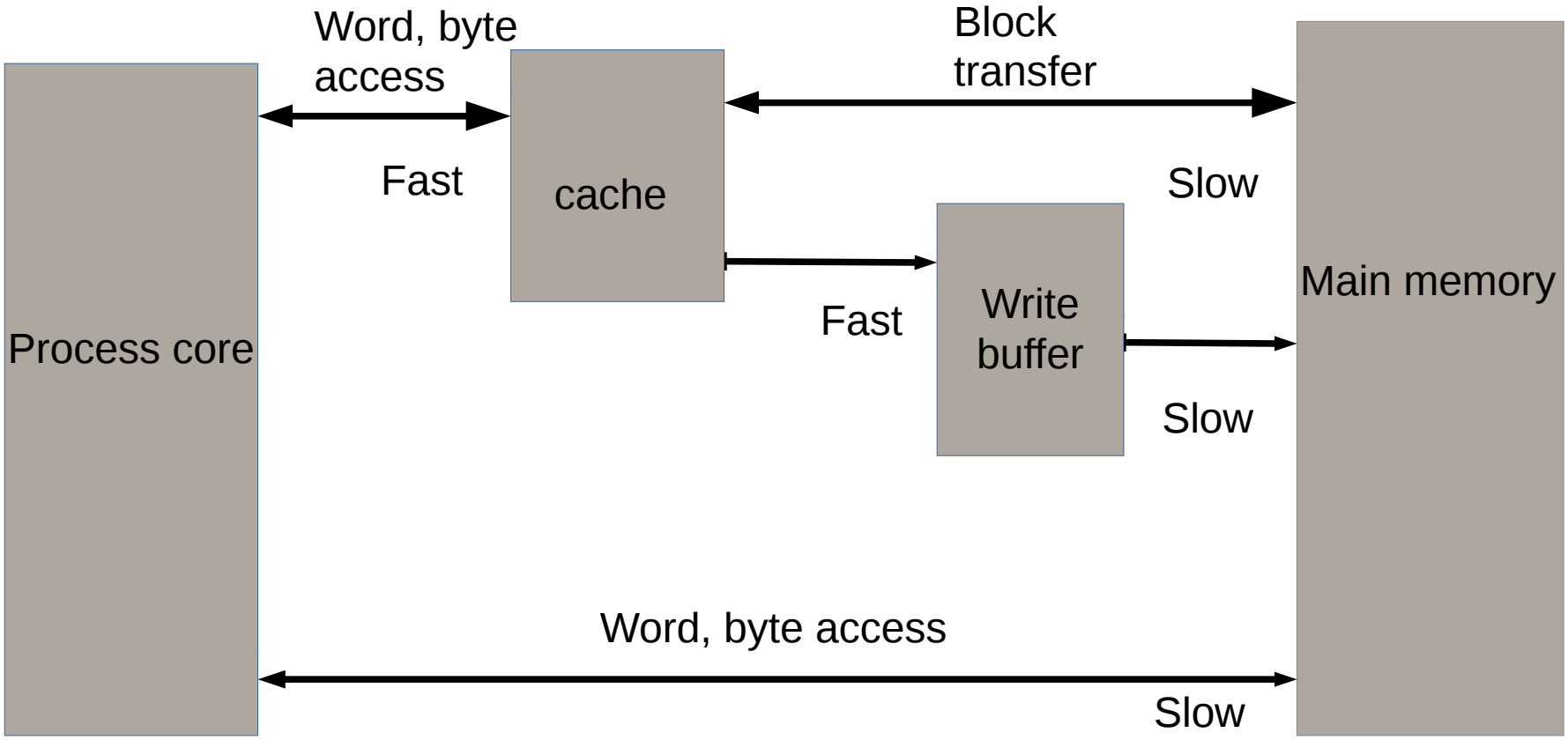
- Small memory placed between processor and main memory to store recently accessed transactions and make memory access faster.



Cached system

Write Buffer

- The write buffer acts as a temporary buffer that frees available space in the cache memory.
- The cache transfers a cache line to the write buffer at high speed and then the write buffer drains it to main memory at slow speed.



Cached system

Cache Architecture

- ARM uses two bus architectures in its cached cores, the Von-Neumann and the Harvard.
- In processor cores using the Von Neumann architecture,there is a single cache used for instruction and data.

- This type of cache is known as a unified cache. A unified cache memory contains both instruction and data values.

- The Harvard architecture has separate instruction and data buses to improve overall system performance, but supporting the two buses requires two caches
- instruction cache (I-cache) and data cache (D-cache). This type of cache is known as a **split cache.**

Cache Elements

Cache memory

Cache memory holds the data frequently requested by CPU

Cache controller

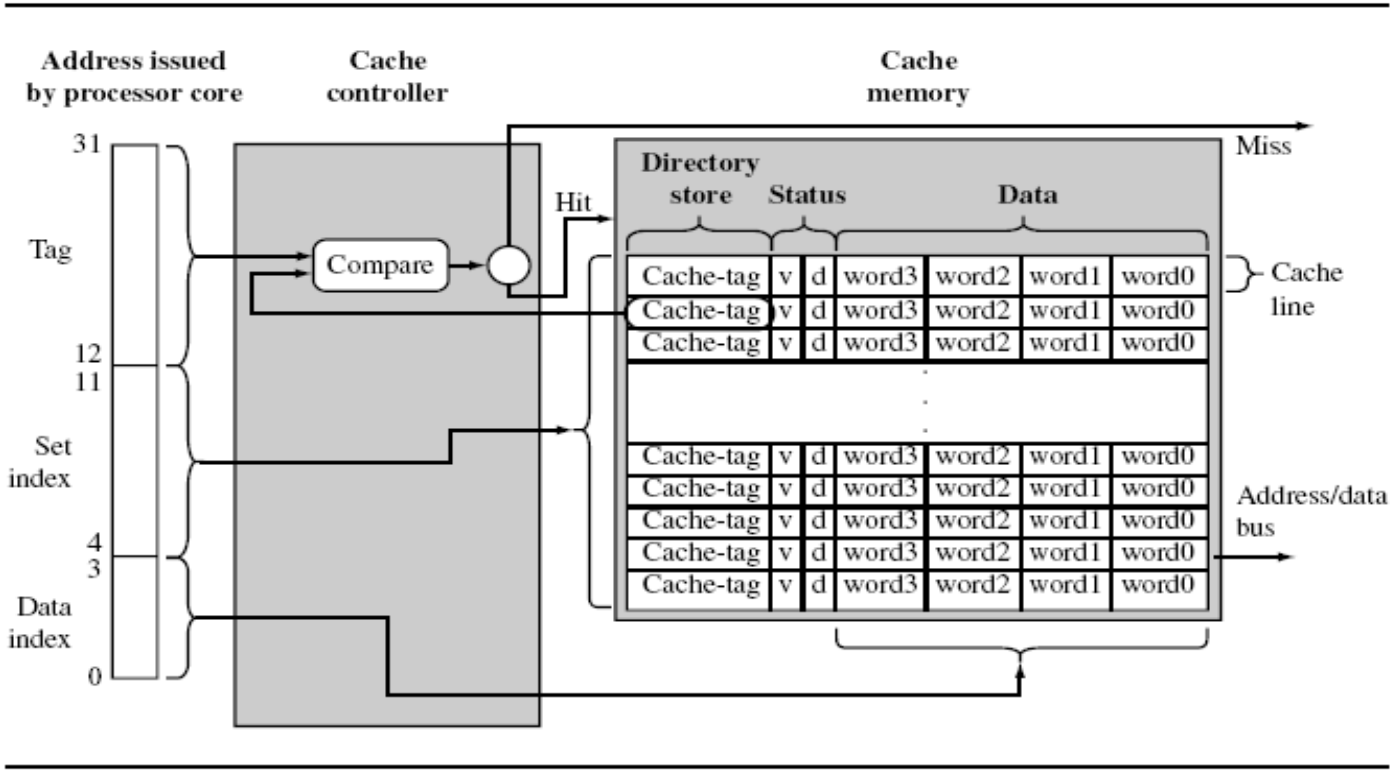
The cache controller is hardware that copies code or data from main memory to cache memory automatically.

Cache Memory

The cache memory is a dedicated memory array accessed in units called **cache lines**.

Cache memory has three main parts Directory store, Status information, and Data section.

All three parts of the cache memory are present for each cache line.



About Cache Line

Cache tag is a directory store which holds the address, identifying where the cache line was copied from main memory. In this way it keeps track of information from main memory being stored in cache line of cache memory.

Data section holds the data read from main memory.

Status bits maintain state information. Two common status bits are the valid bit and dirty bit.

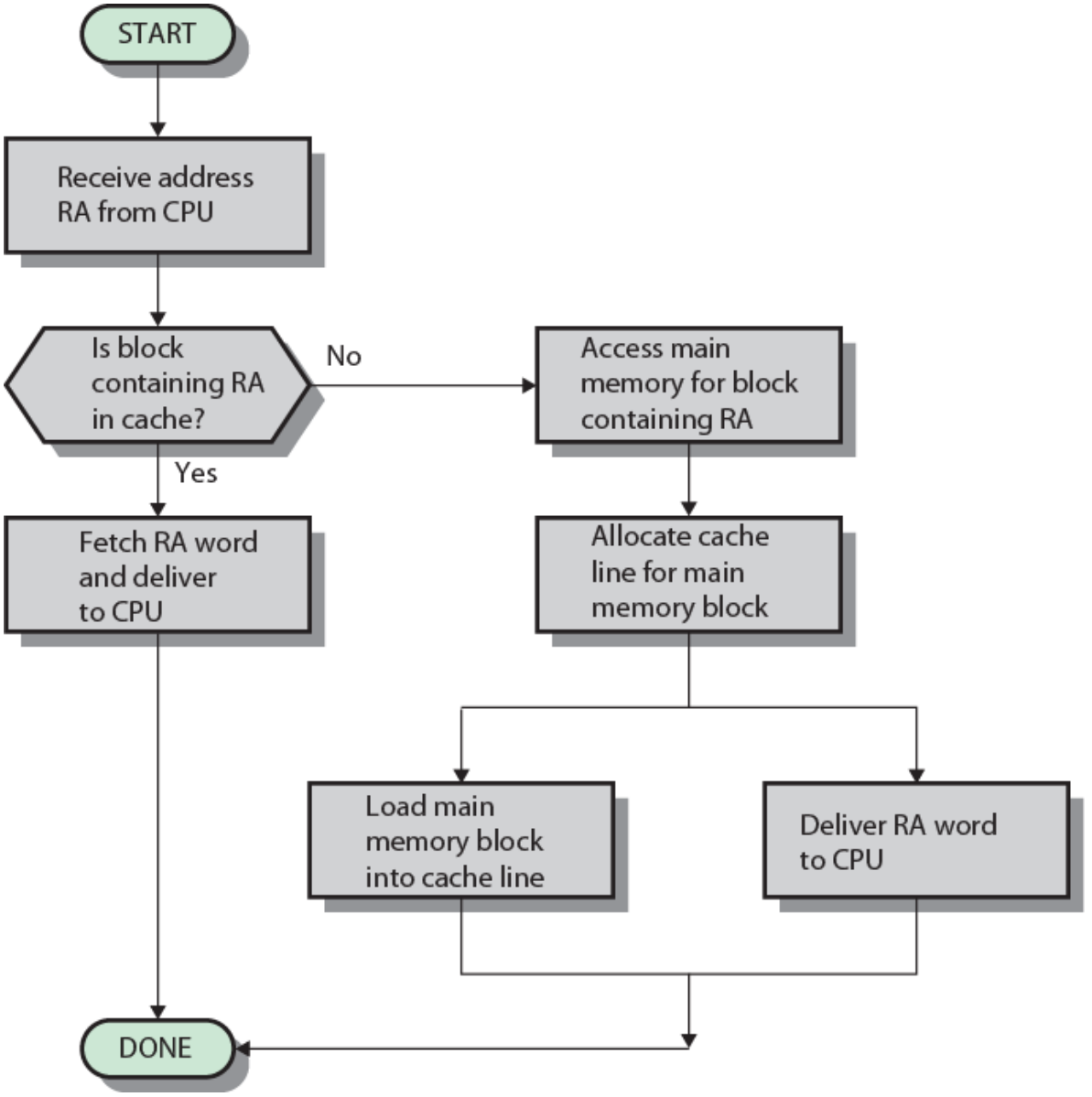
A **valid bit** marks a cache line as active, meaning it contains live data originally taken from main memory and is currently available to the processor core on demand.

A **dirty bit** defines whether or not a cache line contains data that is different from the value it represents in main memory.

Cache operation

- CPU requests contents of memory location
- Check cache for this data
- If present, get from cache (fast)
- If not present, ***read required block from main memory to cache***
- Then deliver from cache to CPU
- Cache includes tags to identify which block of main memory is in each cache slot

Cache Read Operation - Flowchart



Cache hit and cache miss

The cache controller divides the address of the request from processor core into three fields i.e . the ***tag field***, the ***set index field***, and the ***data index field***.

The controller then checks the *valid bit* to determine if the cache line is active, and compares the cache-tag to the tag field of the requested address.

If both the status comparison succeed, it is a ***cache hit***. If either the status comparison fails, it is a ***cache miss***.

IDC operation:-

The MMU must never be disabled when the cache is on.

However, you can enable the two devices simultaneously with a single write to the “Control Register”.

Control Register

31 … 14	13	12 11 10	9	8	7	6	5	4	3	2	1	0
UNP/SBZ	V	UNP /SBZ	R	S	B	L	D	P	W	C	A	M

The C bit determines if data being read can be placed in the IDC and used for subsequent read operations. Typically, main memory is marked as cachable to improve system performance,and I/O space is marked as noncachable to stop the data being stored in the ARM720T cache.

Cachable reads (C=1)

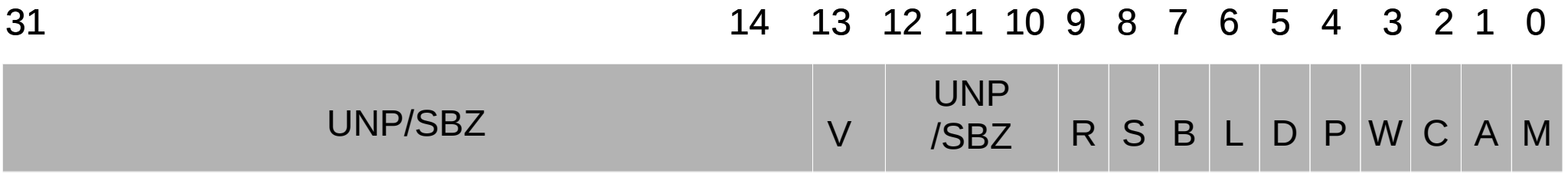
A line fetch of eight words is performed when a cache miss occurs in a cachable area of memory, and it is randomly placed in a cache bank.

Note:- Memory aborts are not supported on cache line fetches and are ignored.

Uncachable reads (C=0)

An external memory access is performed and the cache is not written.

Control Register



M Bit :-MMU enable/disable: 0 = MMU disabled 1 = MMU enabled.

A Bit :- Alignment fault enable/disable:
0 = Address Alignment Fault Checking disabled
1 = Address Alignment Fault Checking enabled.

C Bit :-Cache enable/disable:
0 = Instruction and/or Data Cache (IDC) disabled
1 = Instruction and/or Data Cache (IDC) enabled.

W Bit :-Write buffer enable/disable:
0 = Write Buffer disabled
1 = Write Buffer enabled.

P Bit :- When read, returns 1. When written, is ignored.

Control Register

31 … 14	13	12 11 10	9	8	7	6	5	4	3	2	1	0
UNP/SBZ	V	UNP /SBZ	R	S	B	L	D	P	W	C	A	M

D Bit :- When read, returns 1. When written, is ignored.

L Bit :- When read, returns 1. When written, is ignored.

B Bit :- Big-endian/little-endian:
0 = Little-endian operation
1 = Big-endian operation.

S Bit :- System protection: Modifies the MMU protection system.

R Bit :- ROM protection: Modifies the MMU protection system.

UNP/SBZ Bits :- When read, this returns an Unpredictable value. When written, it Should Be Zero, or a value read from these bits on the same processor.

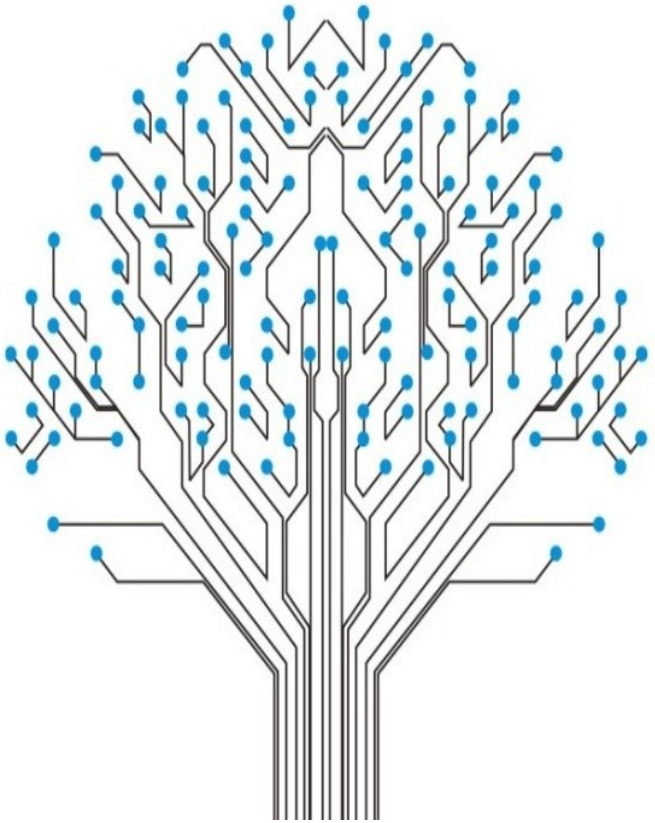
V Bit :-Location of exception vectors:
0 = low addresses 1 = high addresses.
The value of the V bit reflects the state of the VINITHI external input, sampled while HRESETn is LOW.

Any Questions ?

GLOBAL EDGE

Intelligence Of Things

Thank you



Fairness

Learning

Responsibility

Innovation

Respect