BVM Engineering College, VV Nagar

Gujarat Technological University

ADVANCED MICROPROCESSOR

SEM 6 PRESENTATION

VM, MultiTasking, PageTables

Electronics & Communication Dept.

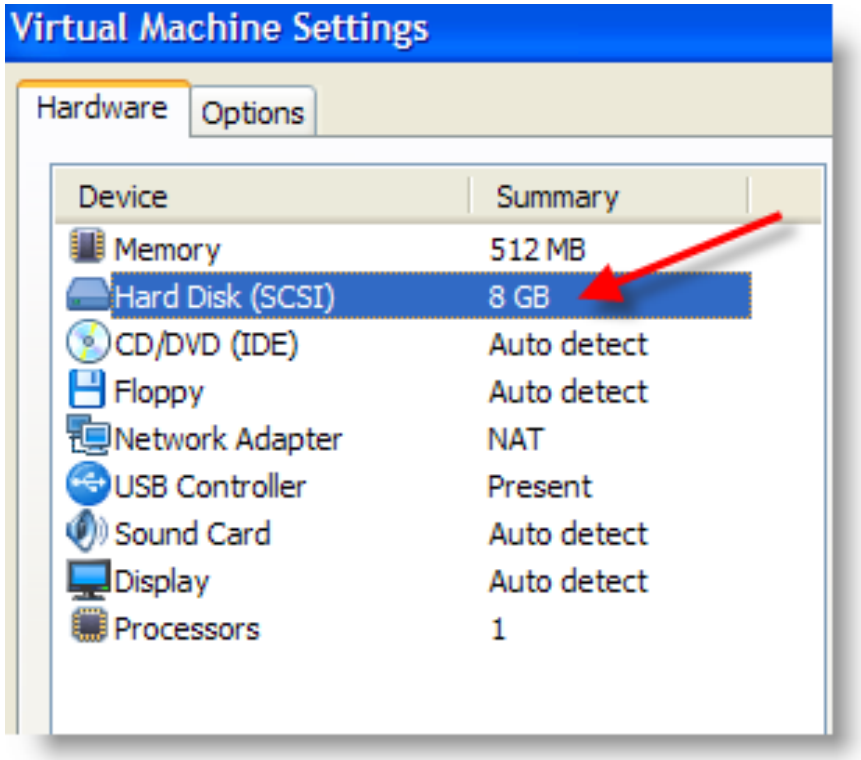
Presented By :

- Dhrumil Shah (130080112015)
- Chinmay Raval (140080111044)
- Chaitanya Tejaswi (140080111013)

Virtual Memory

Virtual Memory: What is it?

"Memory that appears to exists as main storage although most of it is supported by data held in secondary storage , transfer between the two being made automatically as required."



Virtual Memory: Mapping

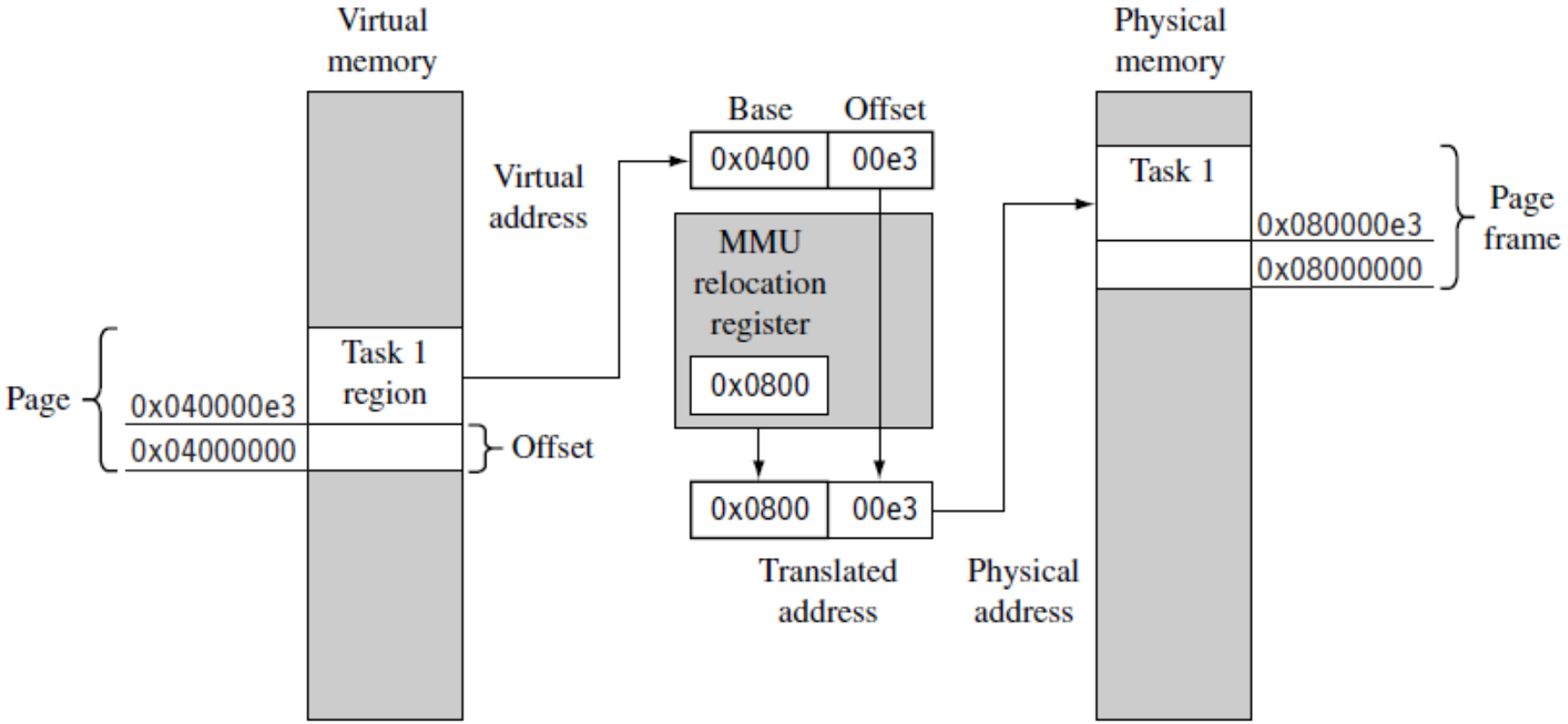
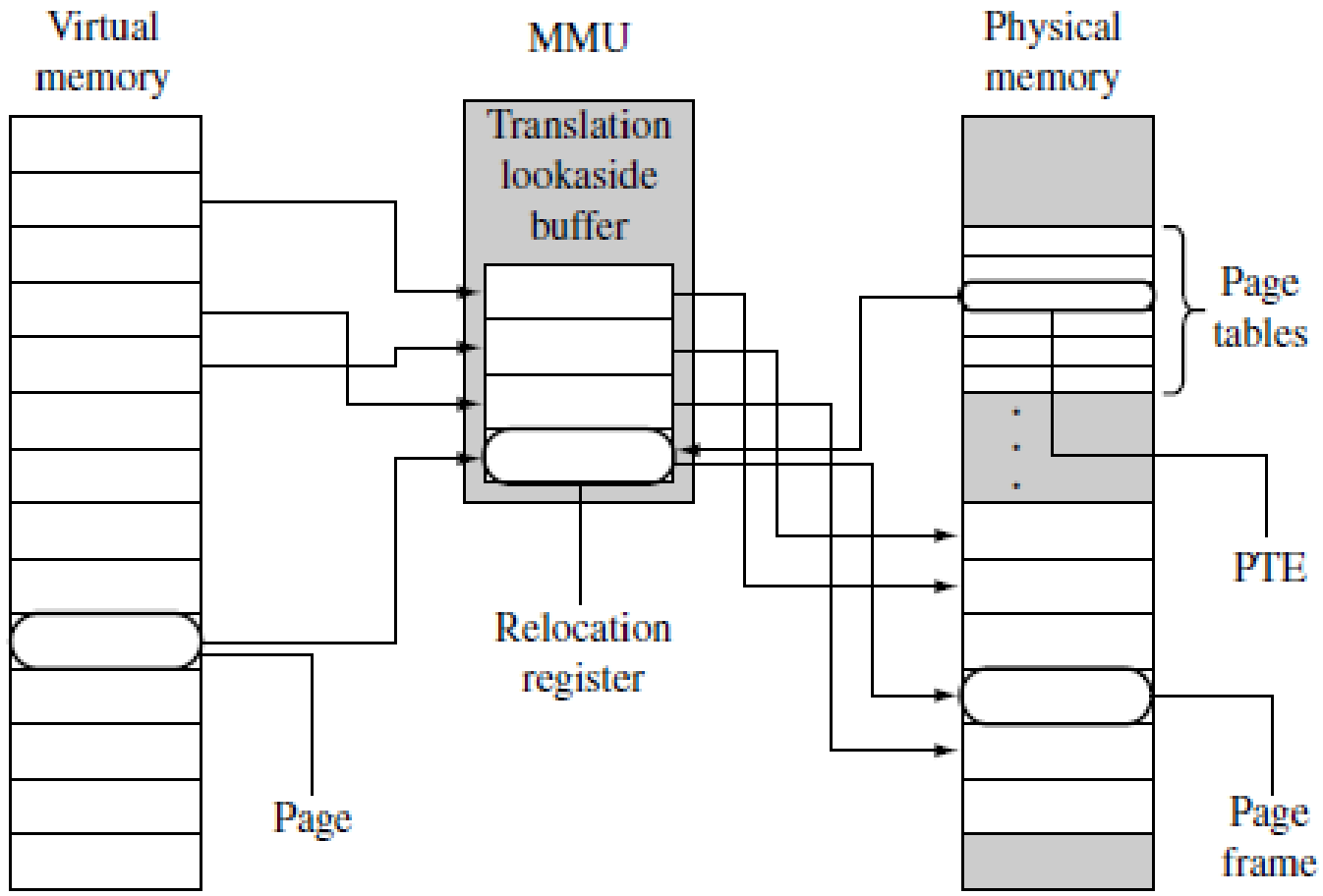


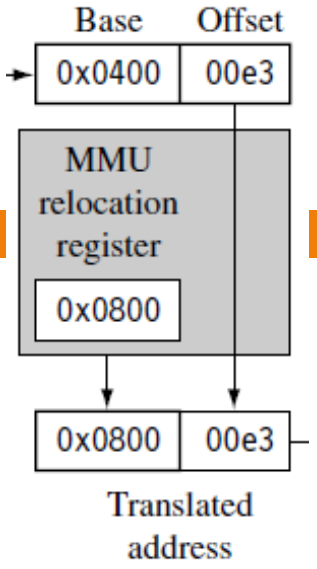
Figure 14.1 Mapping a task in virtual memory to physical memory using a relocation register.

Virtual Memory: Components



The components of a virtual memory system.

Page Table: What is it?



$00e3 = 15.16 + 3 = 243\ locations$

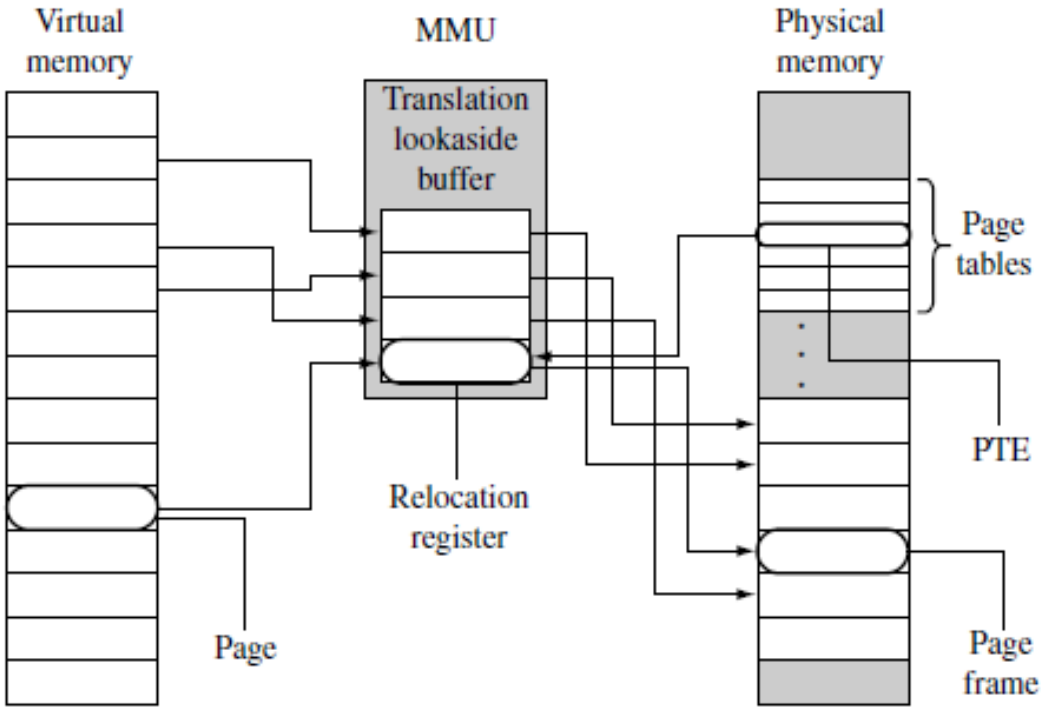
With reference to ARM terminology,

- A single relocation register can only translate a single area of memory, which is set by the number of bits in the offset portion of the virtual address. **This area of virtual memory is known as a *page*.**

- **The area of physical memory pointed to by the translation process is known as a *page frame*.**

  - The MMU uses tables in main memory to store the data describing the virtual memory maps used in the system.

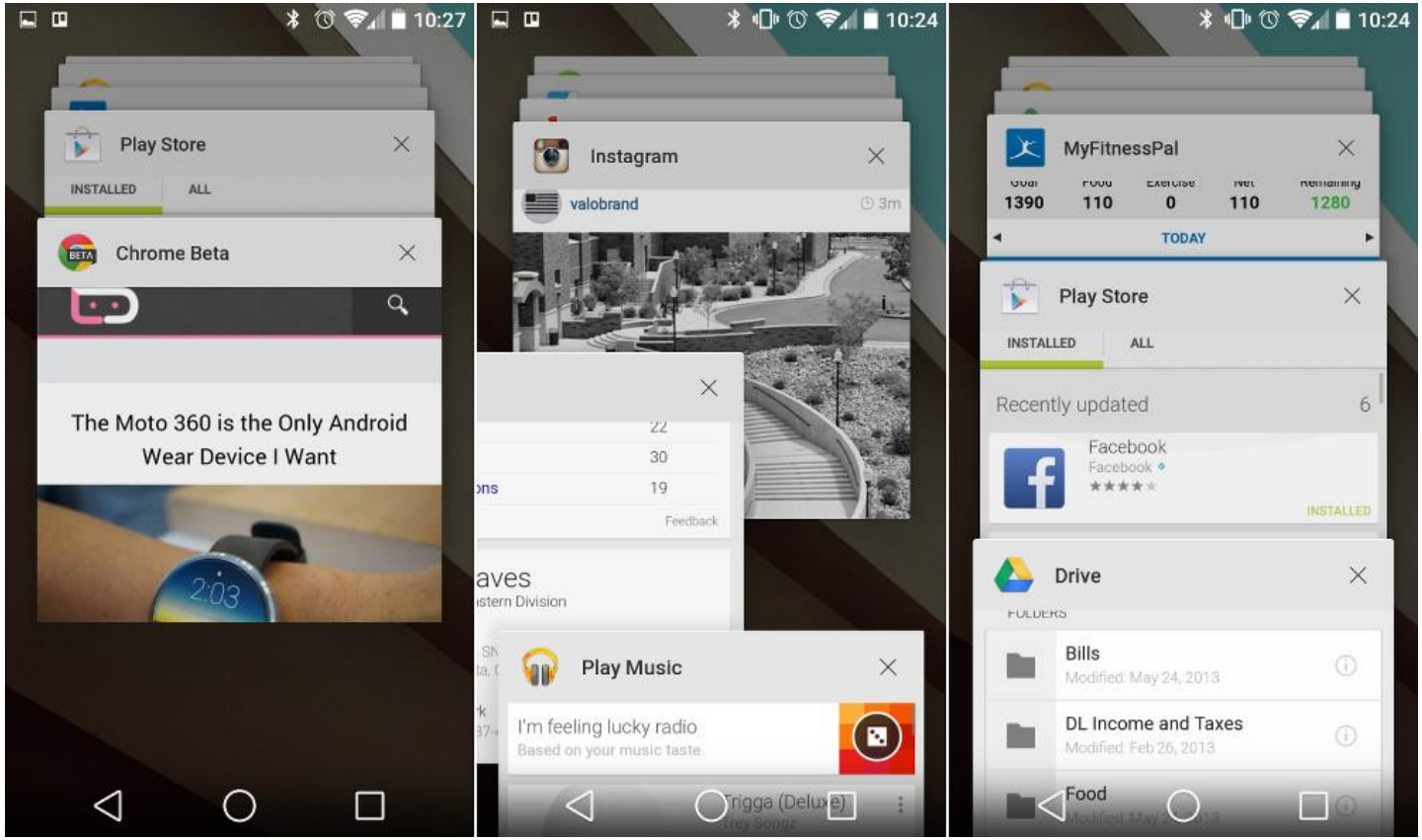
These tables of translation data are known as page tables.



MultiTasking

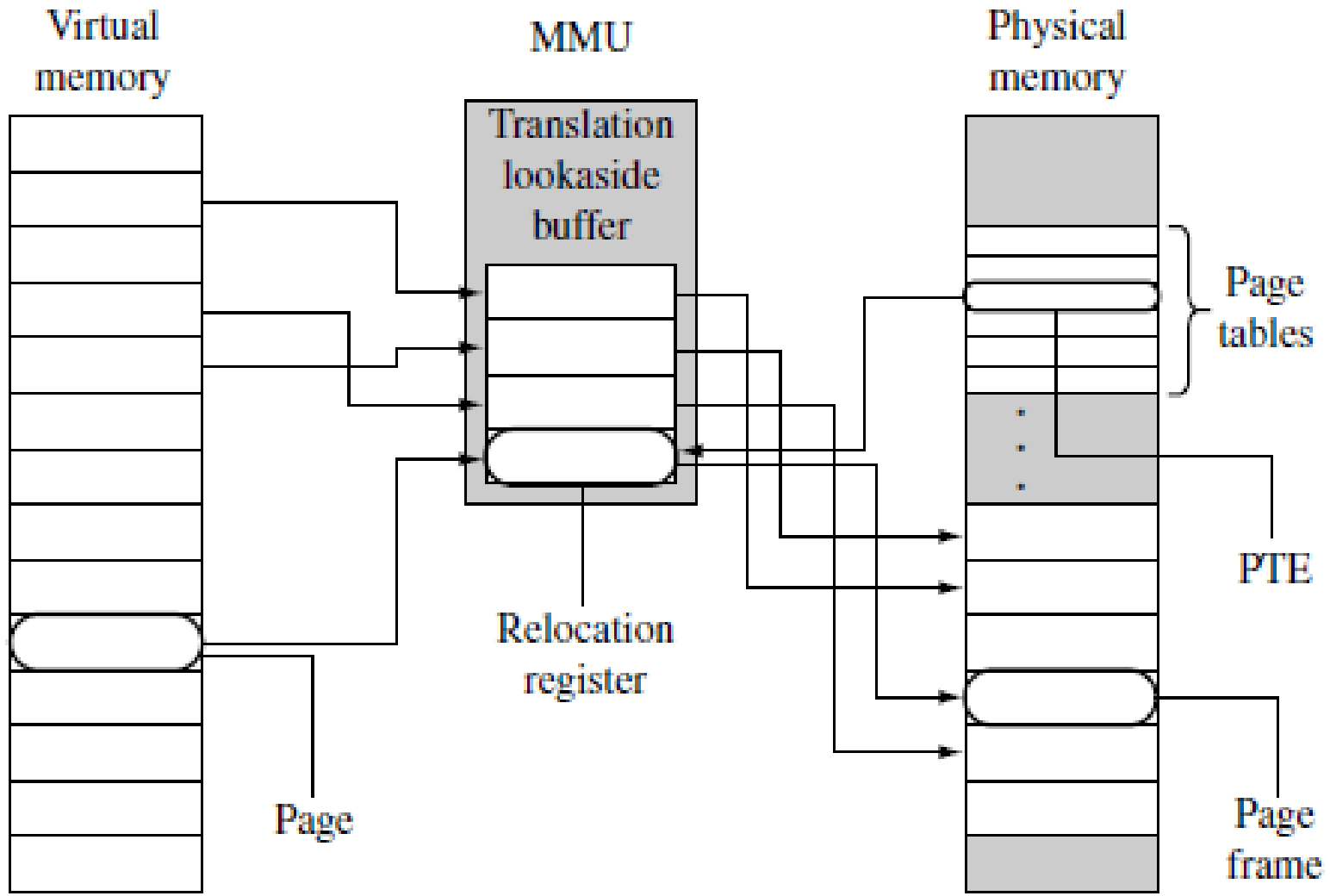
MultiTasking : What is it?

"Executing more than one program (or task) at the same time"



MMU: MultiTasking

“Page tables can reside in memory and not be mapped to MMU hardware.”



The components of a virtual memory system.

MMU: MultiTasking

Page tables can reside in memory and not be mapped to MMU hardware.

- One way to build a multitasking system is to create separate sets of page tables, each mapping a unique virtual memory space for a task.
- To activate a task, the set of page tables for the specific task and its virtual memory space are mapped into use by the MMU. The other sets of inactive page tables represent dormant tasks. This approach allows all tasks to remain resident in physical memory and still be available immediately when a context switch occurs to activate it.
- By activating different page tables during a context switch, it is possible to execute multiple tasks with overlapping virtual addresses.

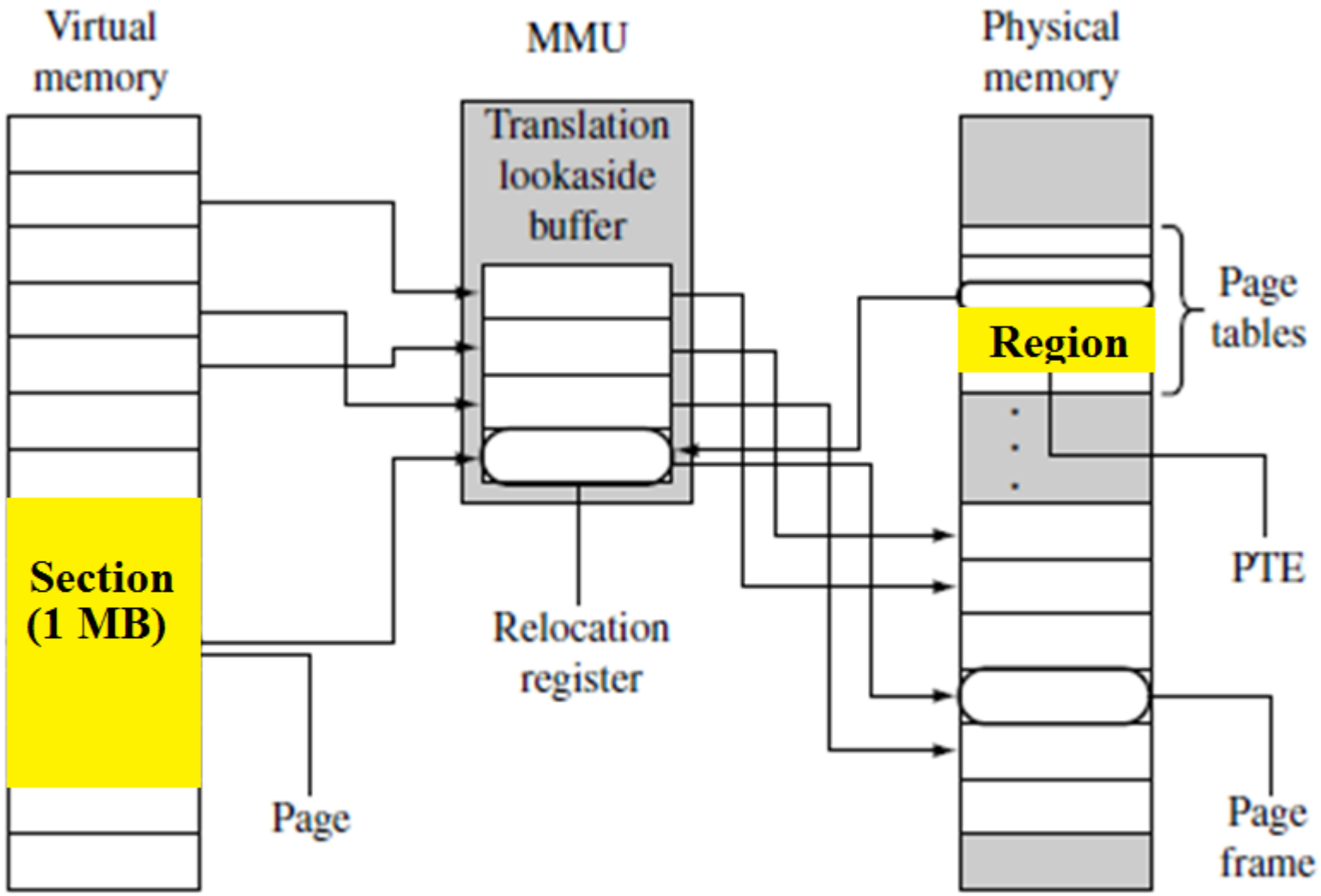
Regions & Sections:

Since a page in virtual memory has a corresponding entry in a page table, a block of virtual memory pages map to a set of sequential entries in a page table.

- **A region is a sequential set of page table entries.**

The location and size of a region can be held in a software data structure while the actual translation data and attribute information is held in the page tables.

- **A section is 1MB block of pages.**



The components of a virtual memory system.

Regions can be “Active” or “Dormant”!

VM: User Perspective

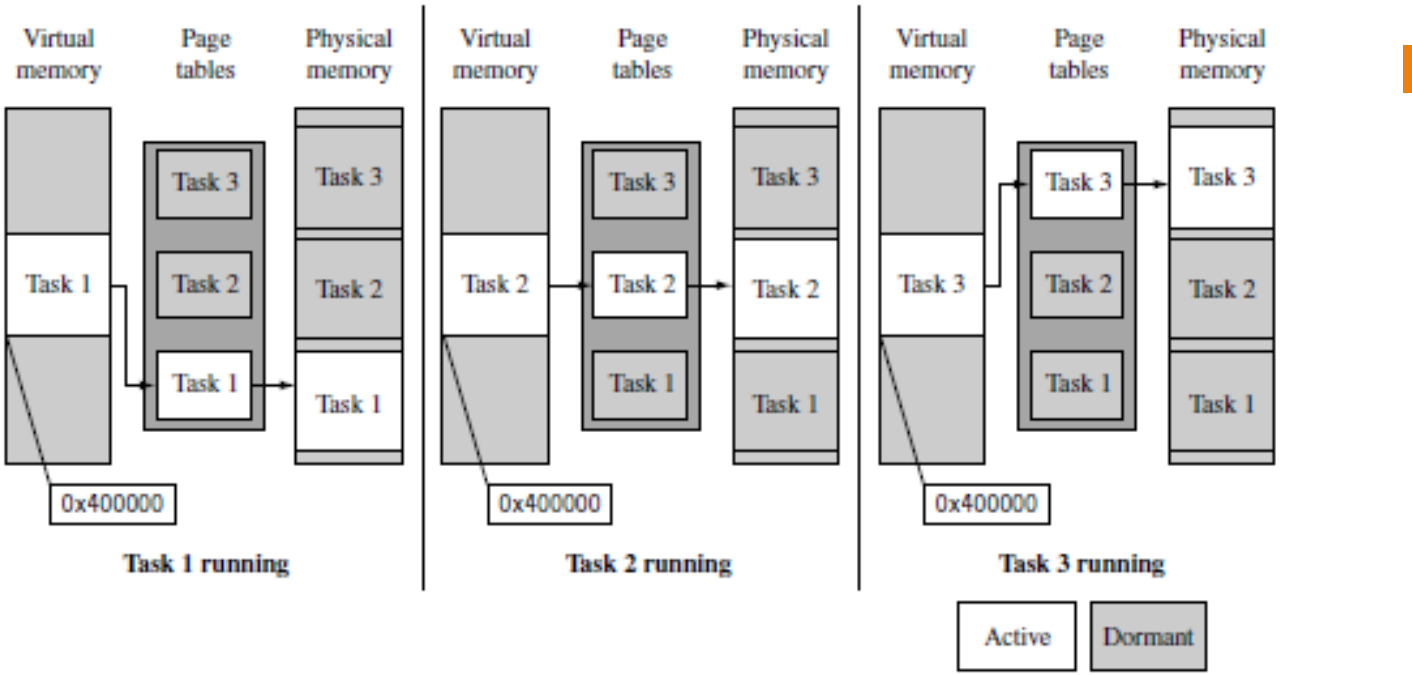


Figure 14.4 Virtual memory from a user task context.

- In the first view, Task 1 is running, and Task 2 and Task 3 are dormant.
- In the second view, Task 2 is running, and Task 1 and Task 3 are dormant.
- In the third view, Task 3 is running, and Task 1 and Task 2 are dormant.

The virtual memory in each of the three views represents memory as seen by the running task. The view of physical memory is the same in all views because it represents the actual state of real physical memory.

Page Tables

Page Table: What is it?

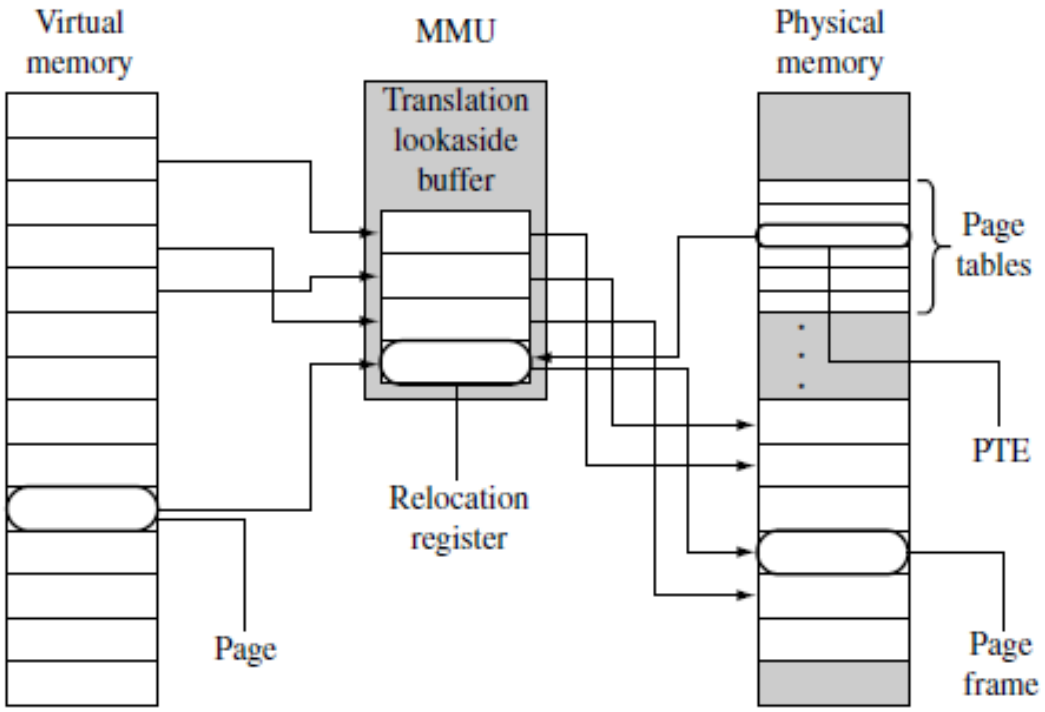
With reference to ARM terminology,

- A single relocation register can only translate a single area of memory, which is set by the number of bits in the offset portion of the virtual address. **This area of virtual memory is known as a *page*.**

- **The area of physical memory pointed to by the translation process is known as a *page frame*.**

- The MMU uses tables in main memory to store the data describing the virtual memory maps used in the system.

These tables of translation data are known as page tables.



Page Table: Architecture

The ARM MMU hardware has a multilevel page table architecture. **There are two levels of page table: L1 and L2**.

1. **L1 (master) page table that can contain two types of page table entry**. It can hold pointers to the starting address of L2 page tables, and page table entries for translating 1 MB pages = 4096 entries.
2. **A coarse L2 page table has 256 entries** consuming 1 KB of main memory. Each PTE in a coarse page table translates a 4 KB block of virtual memory to a 4 KB block in physical memory.
3. **A fine page table has 1024 entries** consuming 4 KB of main memory. Each PTE in a fine page translates a 1 KB block of memory. A fine page table supports 1, 4, or 64 KB pages in virtual memory.

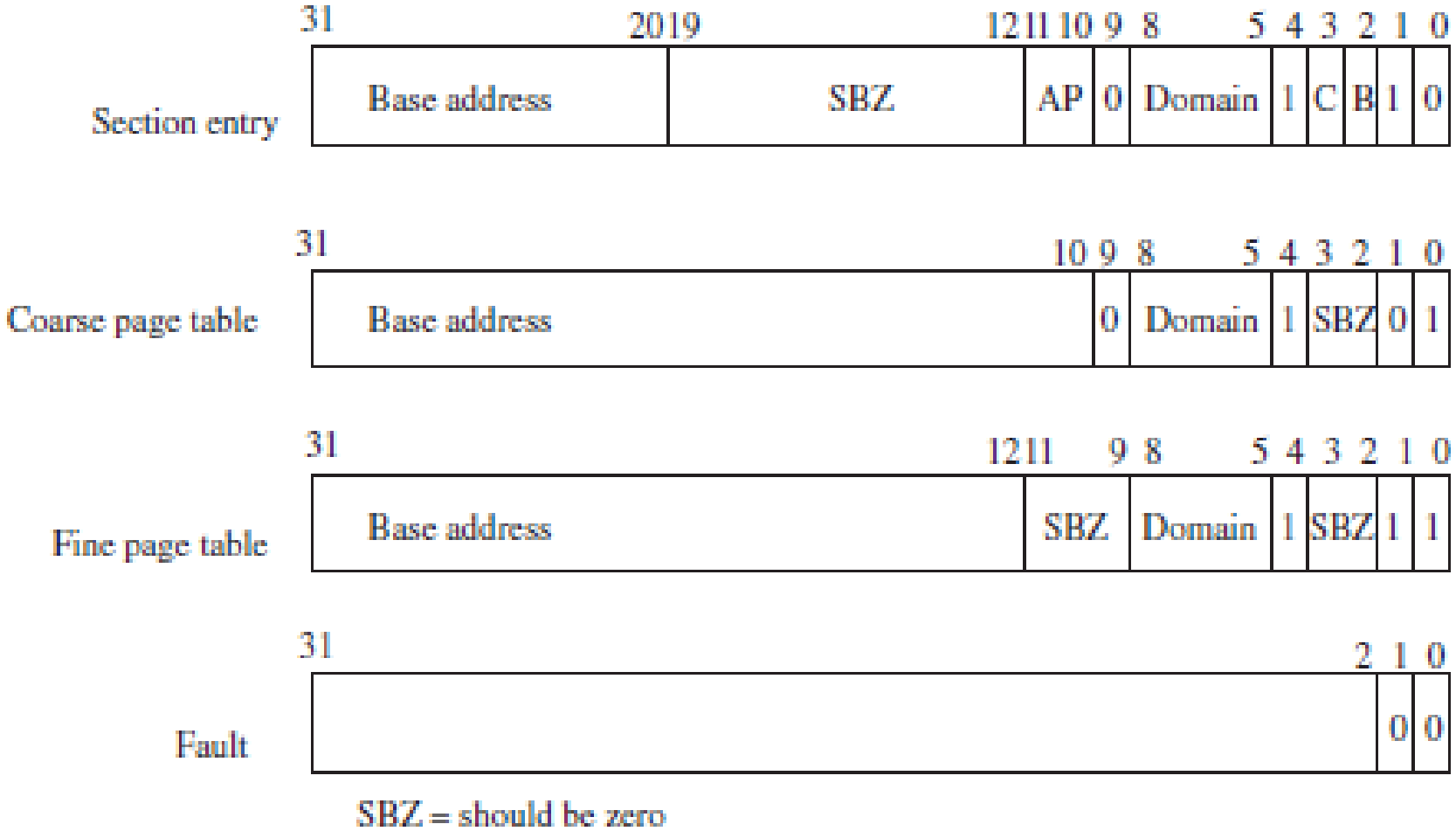
Table 14.2 Page tables used by the MMU.

Name	Type	Memory consumed by page table (KB)	Page sizes supported (KB)	Number of page table entries
Master/section	level 1	16	1024	4096
Fine	level 2	4	1, 4, or 64	1024
Coarse	level 2	1	4 or 64	256

Page Table: L1-Entries

The level 1 page table accepts **four types of entries**:

- A 1 MB **section translation** entry
- A directory entry that points to a **fine L2** page table
- A directory entry that points to a **coarse L2** page table
- A **fault** entry that generates an abort exception



L1 page table entries.

Page Table: L2-Entries

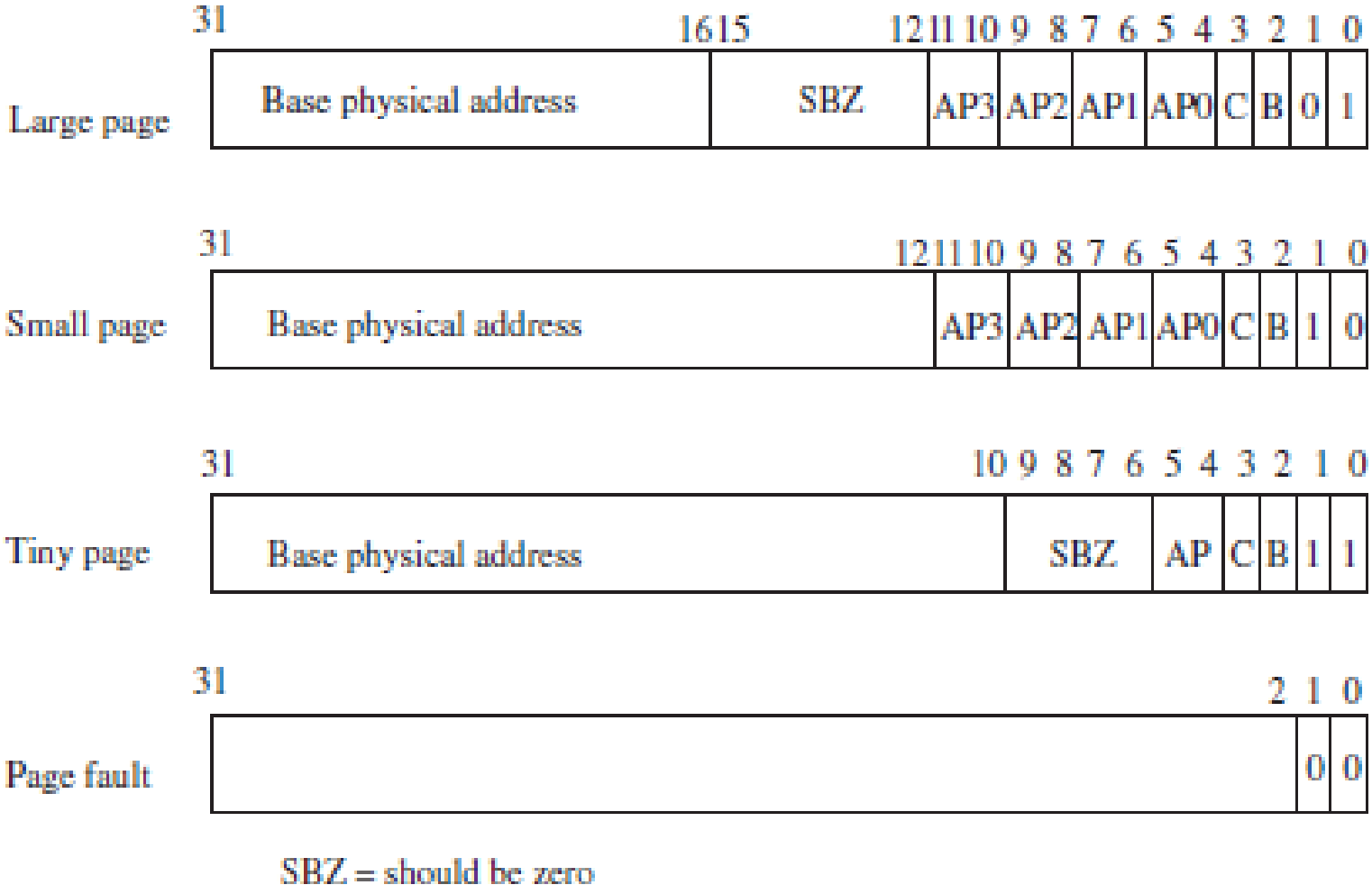
The level 1 page table accepts **four types of entries**:

■ A **large** page entry defines the attributes for a 64 KB page frame.

■ A **small** page entry defines a 4 KB page frame.

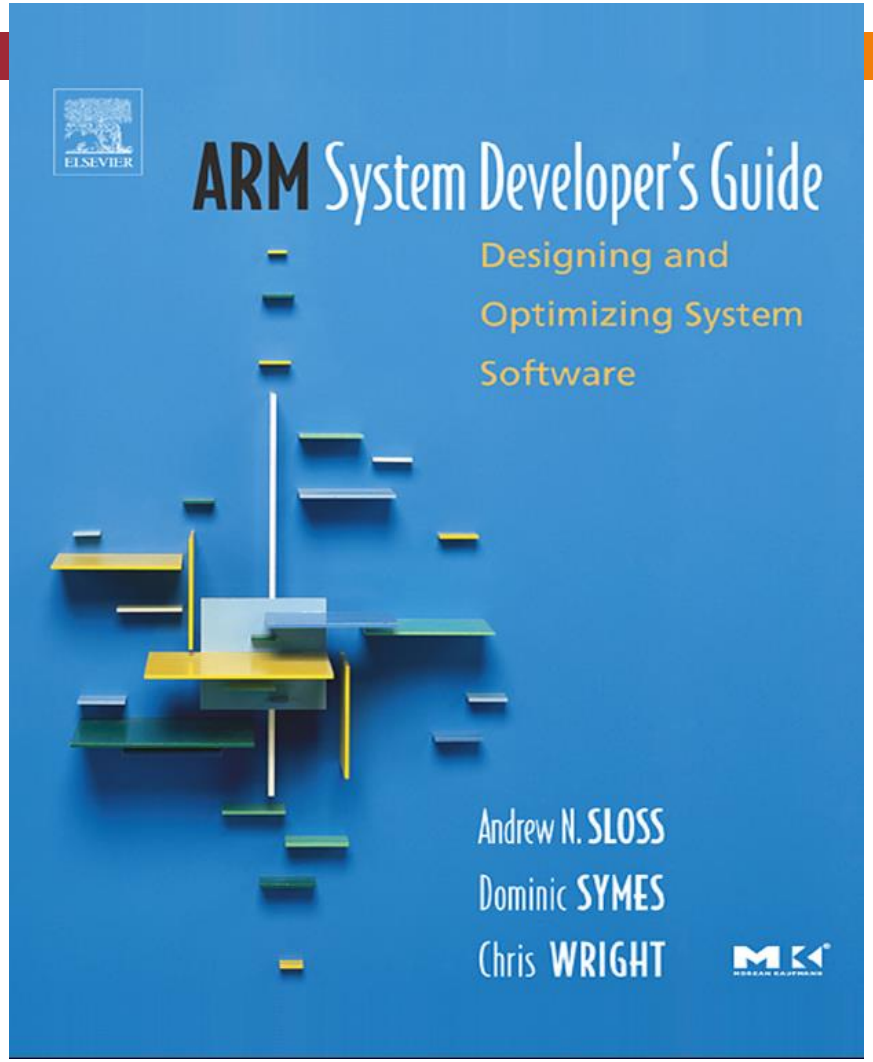
■ A **tiny** page entry defines a 1 KB page frame.

■ A **fault** page entry generates a page fault abort exception when accessed.



L2 page table entries.

Bibliography:



Book:

ARM System Developer's Guide

Website:

http://www.wikipedia.org

THANK YOU !