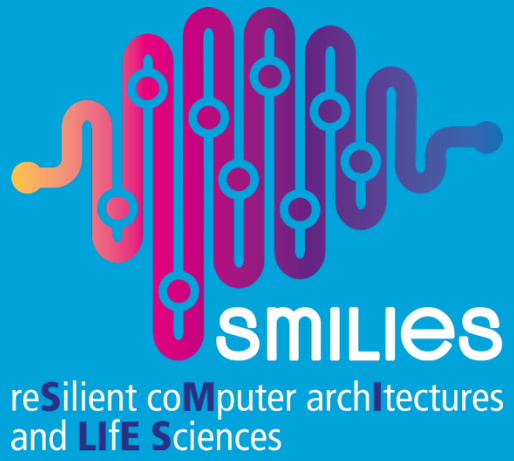


Politecnico di Torino

Department of Control and Computer Engineering

ARM OS SUPPORT

STEFANO DI CARLO

ARCHITECTURAL SUPPORT FOR OPERATING SYSTEMS

- ARM system control coprocessor
- CP15 protection unit registers
- CP15 MMU registers
- ARM MMU architecture
- Context switching
- Input/Output

ARM SYSTEM CONTROL COPROCESSOR

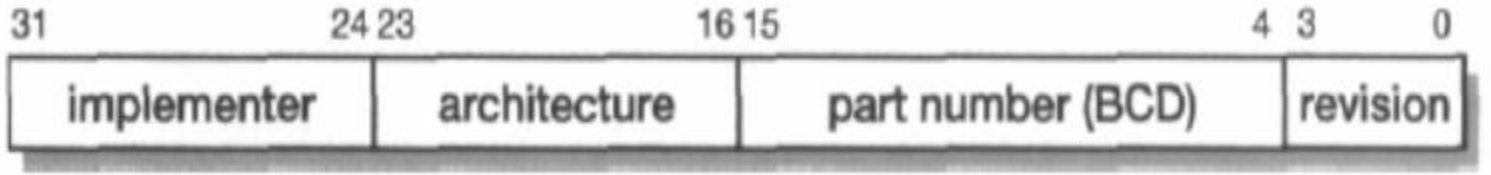
- ARM system control coprocessor is an on-chip coprocessor, using coprocessor number 15 (CP15)
- It controls the operation of the
  - On chip cache
  - Memory management
  - Protections unit
  - Write buffer
  - Prefetch buffer
  - Branch target cache
  - System configurations signals

CP15 PROTECTION UNIT REGISTERS

Register	Purpose
0	ID Register
1	Configuration
2	Cache Control
3	Write Buffer Control
5	Access Permissions
6	Region Base and Size
7	Cache Operations
9	Cache Lock Down
15	Test
4,8,10-14	UNUSED

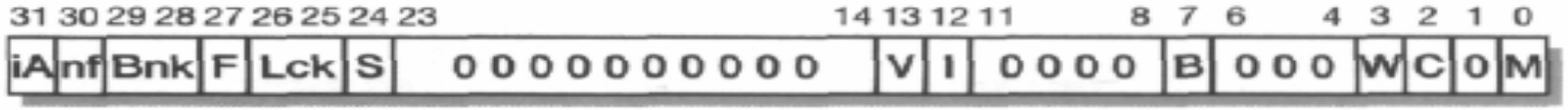
CP15 PROTECTION UNIT REGISTERS

- Register 0 (ID Register)
- Bits [3:0] ➔ revision number,
  - bits [15:4] ➔3-digit part number
  - bits [23:16] ➔architecture version
  - (0 for version 3,
  - 1 for version 4,
  - 2 for version 4T,
  - 4 for version 5T)
  - bits [31:24] ➔ASCII code of an implementer's trademark



CP15 PROTECTION UNIT REGISTERS

- Register 1 (Configuration)
  - All bits are cleared on reset.
  - M ➔ Protection unit,
  - C ➔ data or unified cache,
  - W ➔ write buffer,
  - B switches from little- to big-endian byte ordering,
  - I enables the instruction cache when this is separate from the data cache,
  - V causes the exception vectors to move to near the top of the address space,
  - S, Lck, F and Bnk are used to control the cache (on the ARM740T), and
  - nf and iA control various clock mechanisms (on the ARM940T).



CP15 PROTECTION UNIT REGISTERS

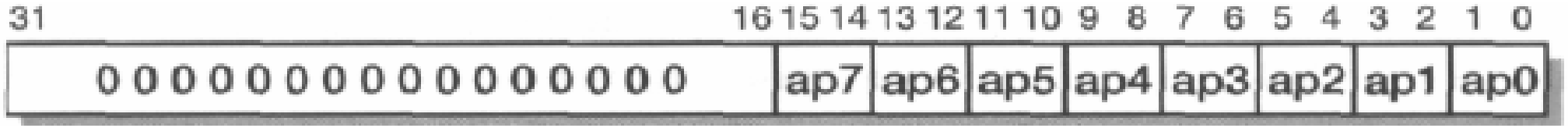
- Register 2 (Cache Control)
  - It controls the cache ability of the eight individual protection regions
  - Bit 0 enables the cache for loads within region 0,
  - Bit 1 likewise for region 1, and so on.
  - The ARM940T has separate protection units on its instruction and data ports
  - Cop2 is used to determine which unit is accessed:
  - Cop2 = 0 gives access to the protection unit on the data port;
  - Cop2 = 1 gives access to the protection unit on the instruction port

CP15 PROTECTION UNIT REGISTERS

- Register 3 (Write Buffer Control)
  - It defines whether or not the write buffer should be used for each of the protection regions.
  - The ARM940T instruction port is read-only,
  - The write buffer can only be enabled for the data port
  - and so Cop2 should always be zero

CP15 PROTECTION UNIT REGISTERS

- Register 5 (Access Permission)
  - 00 ➔ No access
  - 01 ➔ Privileged modes
  - 10 ➔ Privileged full access and user read only
  - 11 ➔ Full access.
  - Again the ARM940T uses the Cop2 field to differentiate
    - 1➔ instruction protection units
    - 0 ➔ data protection units.



CP15 PROTECTION UNIT REGISTERS

- Register 6 (Region Base and Size)
  - It defines the start address and size of each of the eight regions.

CP15 PROTECTION UNIT REGISTERS

- Register 7 (Cache Operation)
  - It controls various cache operations and
  - its operation is different for the ARM740T and the ARM940T.
- Register 9 (Cache Lock Down)
  - It is used in the ARM940T to lock down areas of the cache.

CP15 PROTECTION UNIT REGISTERS

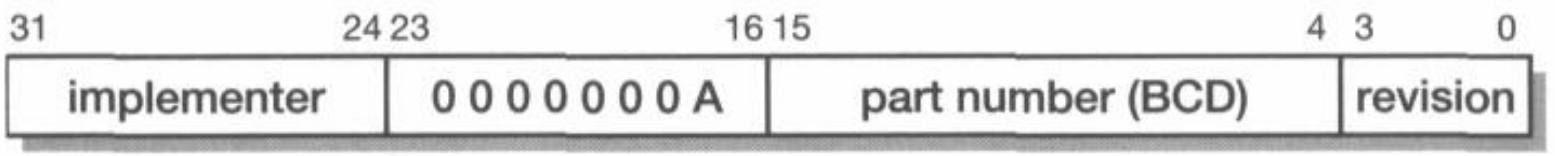
- Register 15 (Test)
  - It is used in the ARM940T to modify the cache allocation algorithm from random to round-robin.
  - This is intended for use only during silicon production testing.

CP15 MMU REGISTERS

Register	Purpose
0	ID Register
1	Control
2	Translation Table Base
3	Domain Access Control
5	Fault Status
6	Fault Address
7	Cache Operations
8	TLB Operations
9	Read Buffer Operations
10	TLB Lockdown
13	Process ID Mapping
14	Debug Support
15	Test and Clock Control
4, 11–12	UNUSED

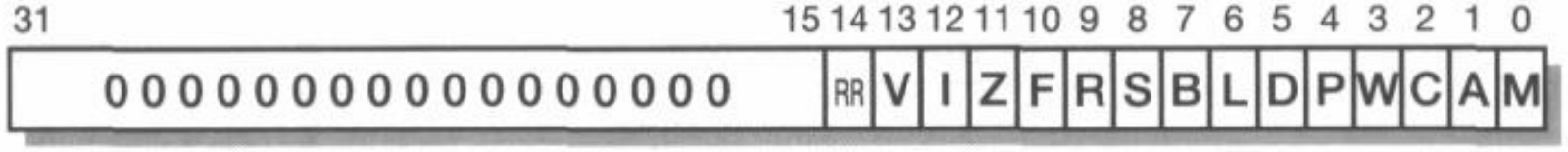
CP15 MMU REGISTERS

- Register 0
  - Bits [3:0] ➔ revision number,
  - bits [15:4] ➔3-digit part number
  - bits [23:16] ➔architecture version
  - (0 for version 3,
  - 1 for version 4)
  - bits [31:24] ➔ASCII code of an implementer's trademark



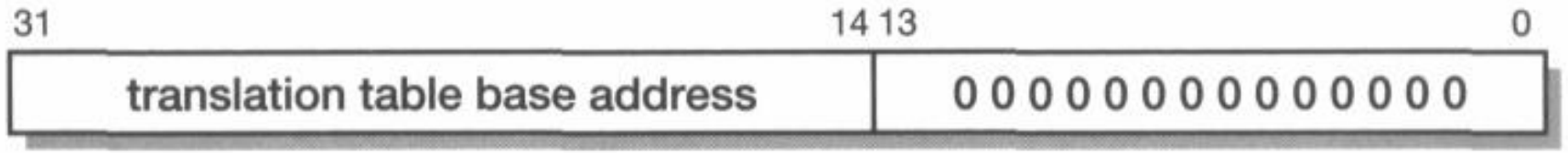
CP15 MMU REGISTERS

- Register 1 (Control)
  - All bits are cleared on reset.
  - M ➔ MMUunit,
  - A ➔ Address Alignment fault checking,
  - C ➔data or unified cache
  - W ➔ write buffer,
  - P ➔ switches from 26 to 32 bit address range
  - L ➔ switches to late abort timeing
  - B ➔switches from little- to big-endian byte ordering,
  - S & R ➔ modify the MMU system and ROM protection states
  - F ➔ controls the external coprocessor communications
  - Z ➔ enables branch prediction
  - I ➔ enables the instruction cache when this is separate from the data cache,
  - V causes the exception vectors to move to near the top of the address space,
  - RR ➔ enables cache replacement algorithm



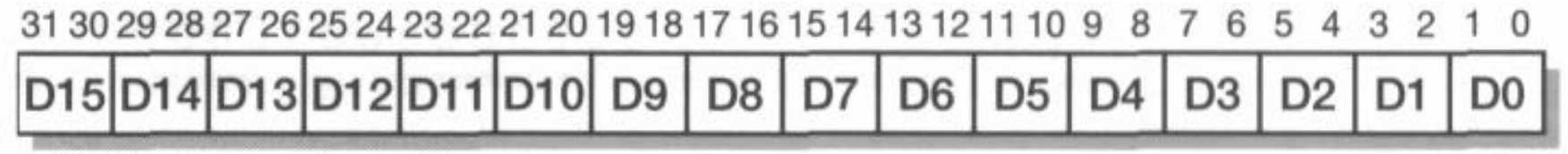
CP15 MMU REGISTERS

- Register 2 (Translation Table Base)
  - It contains the address of the start of the currently active first-level translation table



CP15 MMU REGISTERS

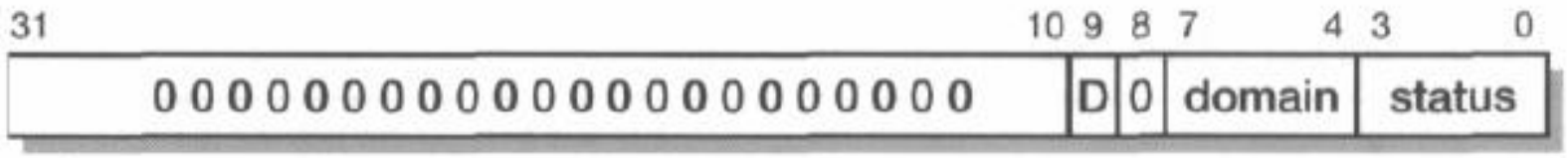
- Register 3(Domain Access Control)



Value	Status	Description
00	No access	Any access will generate a domain fault
01	Client	Page and section permission bits are checked
10	Reserved	Do not use
11	Manager	Page and section permission bits are not checked

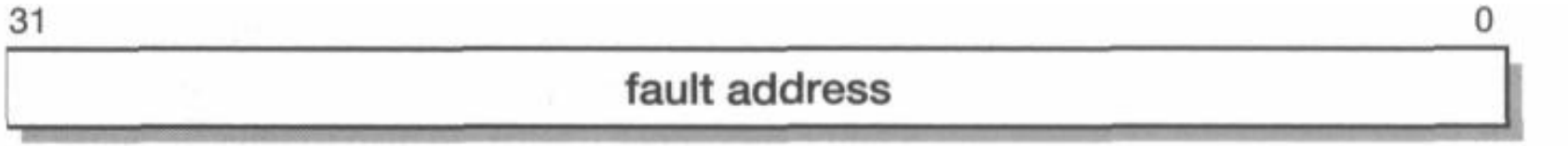
CP15 MMU REGISTERS

- Register 5 (Fault Status)
  - It indicates the type of fault and the domain of the last data access that aborted.
  - D is set on a data breakpoint.



CP15 MMU REGISTERS

- Register 6 (Fault Address)
  - It contains the address of the last data access that aborted.



CP15 MMU REGISTERS

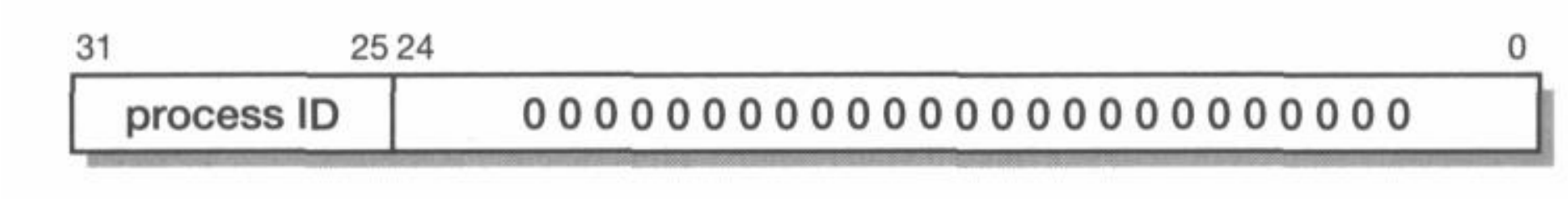
- Register 7 (Cache Operation)
- It is used to perform a
  - Number of cache,
    - Write buffer,
    - Prefetch buffer and
    - Branch target cache clean and/or
    - Flush operations.
  - The data supplied should be either zero or a relevant virtual address.

CP15 MMU REGISTERS

- Register 8 (TLB Operations)
  - It is used to perform a number of
    - TLB operations,
    - Flushing single entries or the whole TLB and
    - Supporting unified or separate instruction and data TLBs

CP15 MMU REGISTERS

- Register 9 (Read Buffer Operation)
  - It is used to control the read buffer
- Register 10 (TLB Lockdown)
  - It is used to control TLB lockdown functions
- Register 13 (Process ID Mapping)
  - It is used to remap virtual addresses through a process ID register.



ARM MMU ARCHITECTURE

- An MMU performs two primary functions:
  - It translates virtual addresses into physical addresses.
  - It controls memory access permissions, aborting illegal accesses.

MEMORY GRANULARITY

- The units that can be used are:
  - Sections.
    - These are 1 Mbyte blocks of memory.
  - Large pages.
    - These are 64 Kbyte blocks of memory, and within a large page access control is applied to individual 16 Kbyte subpages.
  - Small pages.
    - These are 4 Kbyte blocks of memory, and within a small page access control is applied to individual 1 Kbyte subpages.
  - Tiny pages.
    - Some of the latest CPUs also support 1 Kbyte 'tiny' pages.

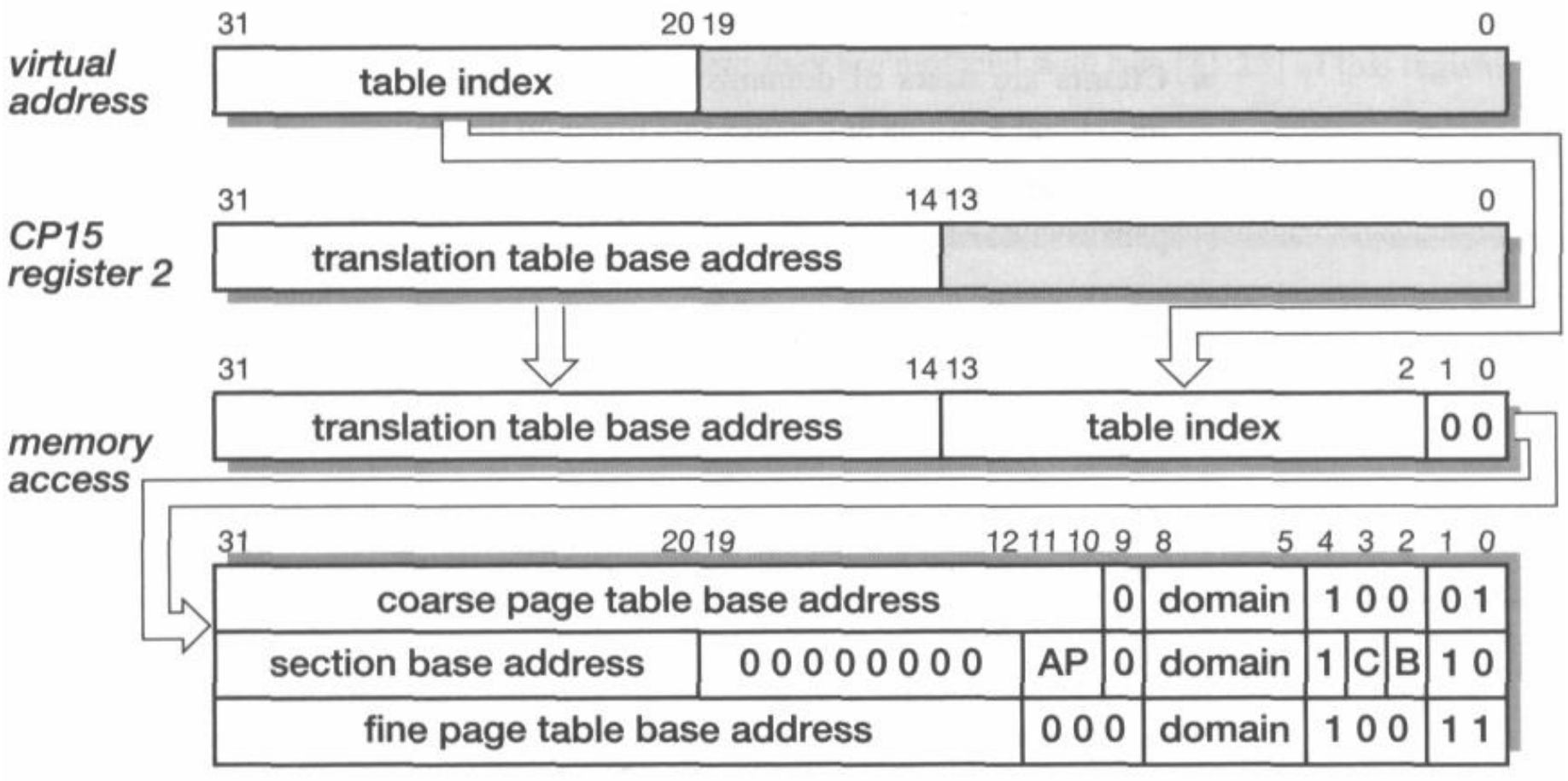
DOMAINS

- Domain is a group of sections or pages which have particular access permission
- The access control is based on two sorts of programs
  - Clients
    - Clients are users of domains and must observe the access permissions of the individual sections and pages that make up the domain.
  - Managers
    - Managers are the controllers of the domain and can bypass the access permissions of individual sections or pages

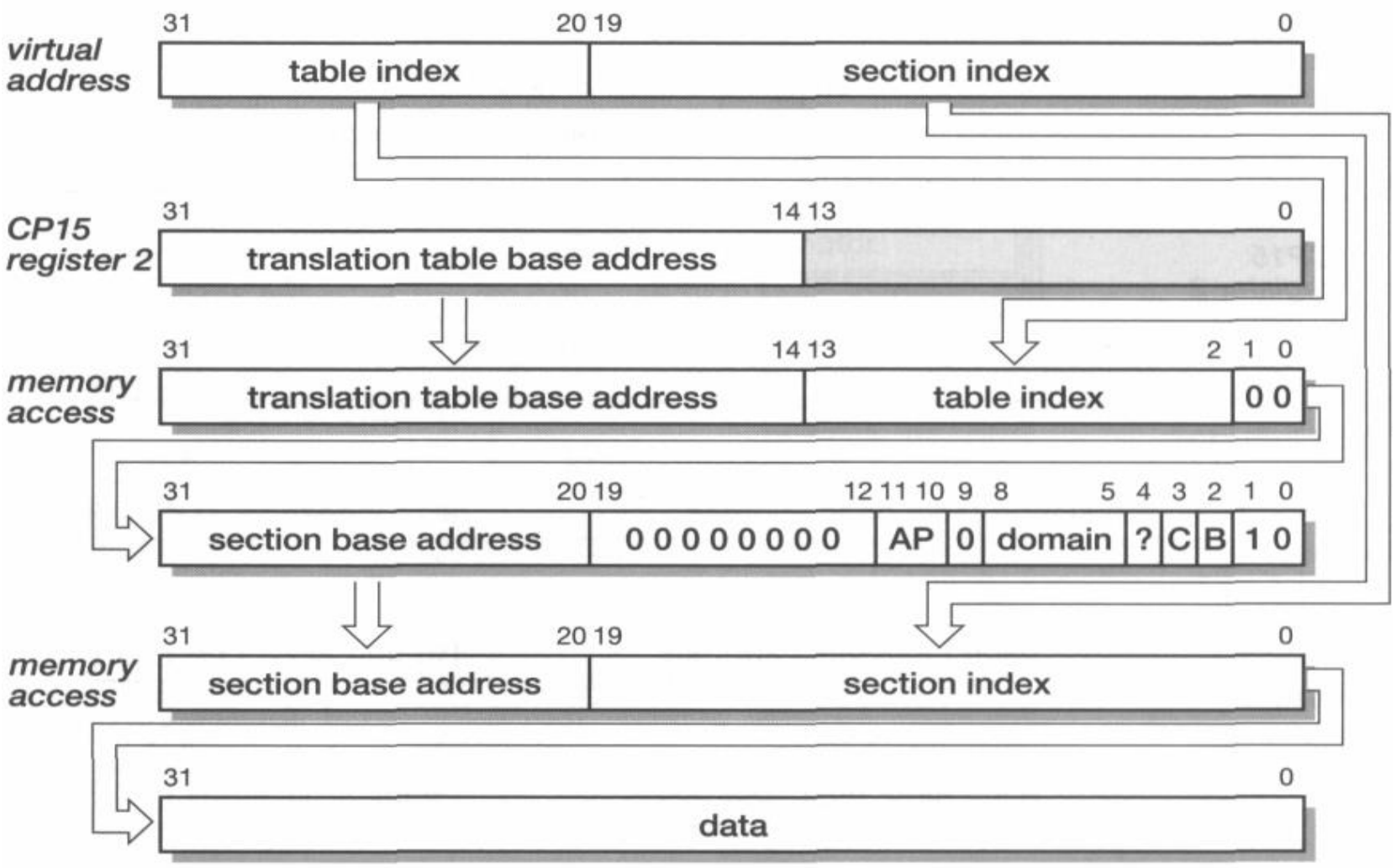
TRANSLATION PROCESS

- First Translation fetch
- Section Translation
- Page Translation
- Access Permissions

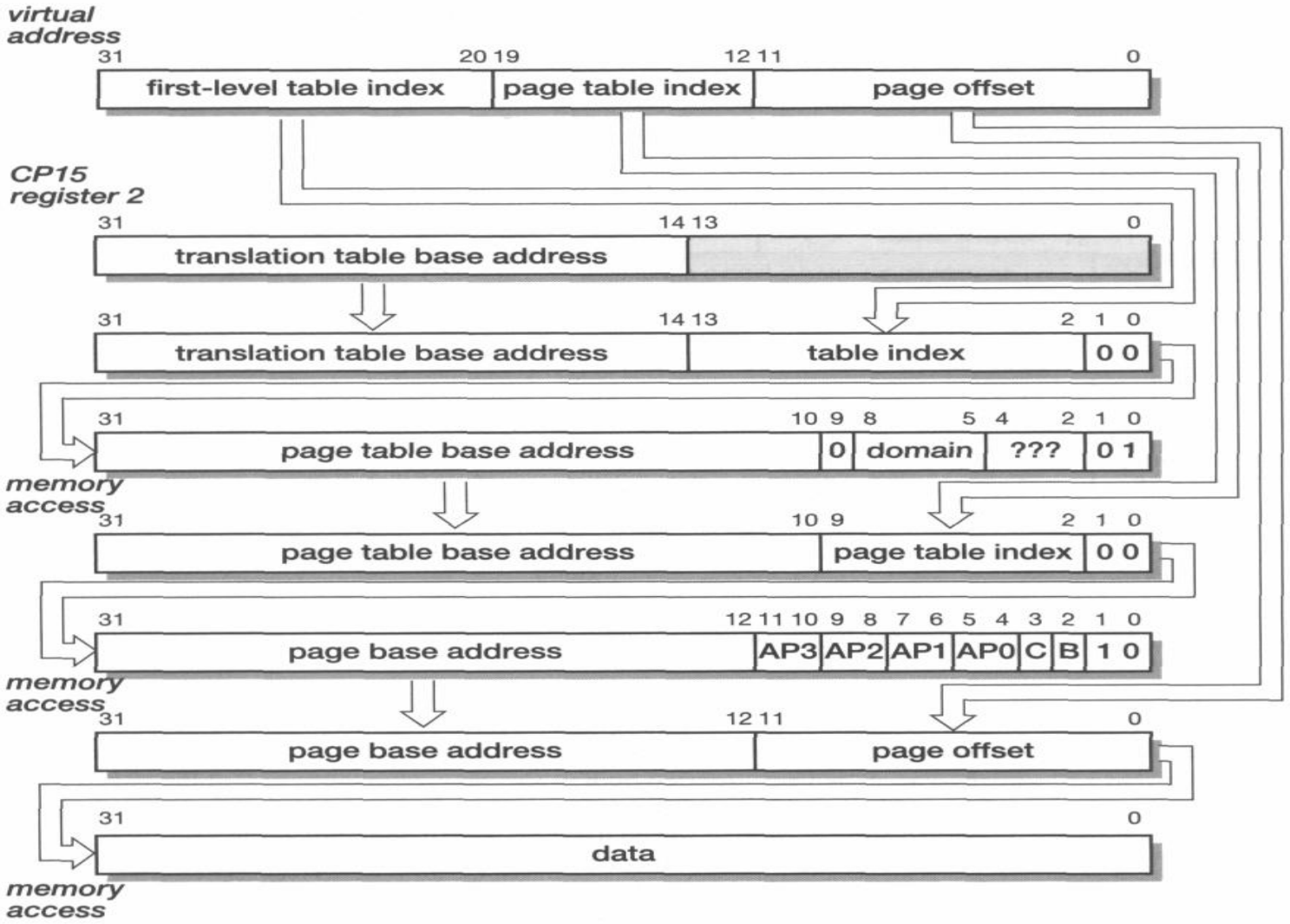
FIRST TRANSLATION FETCH



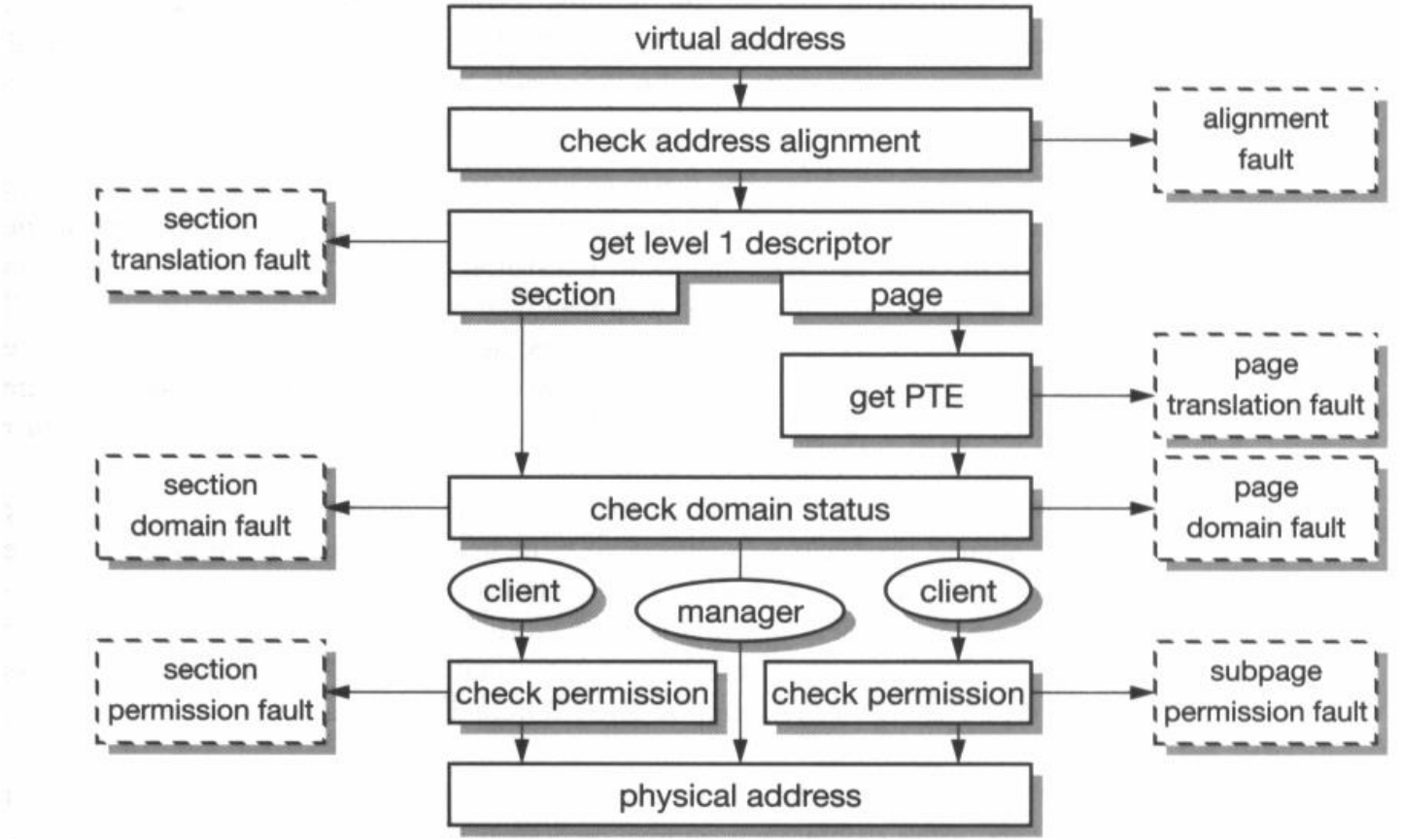
SECTION TRANSLATION



PAGE TRANSLATION



ACCESS PERMISSION CHECKING SCHEME



ACCESS PERMISSION CHECKING SCHEME

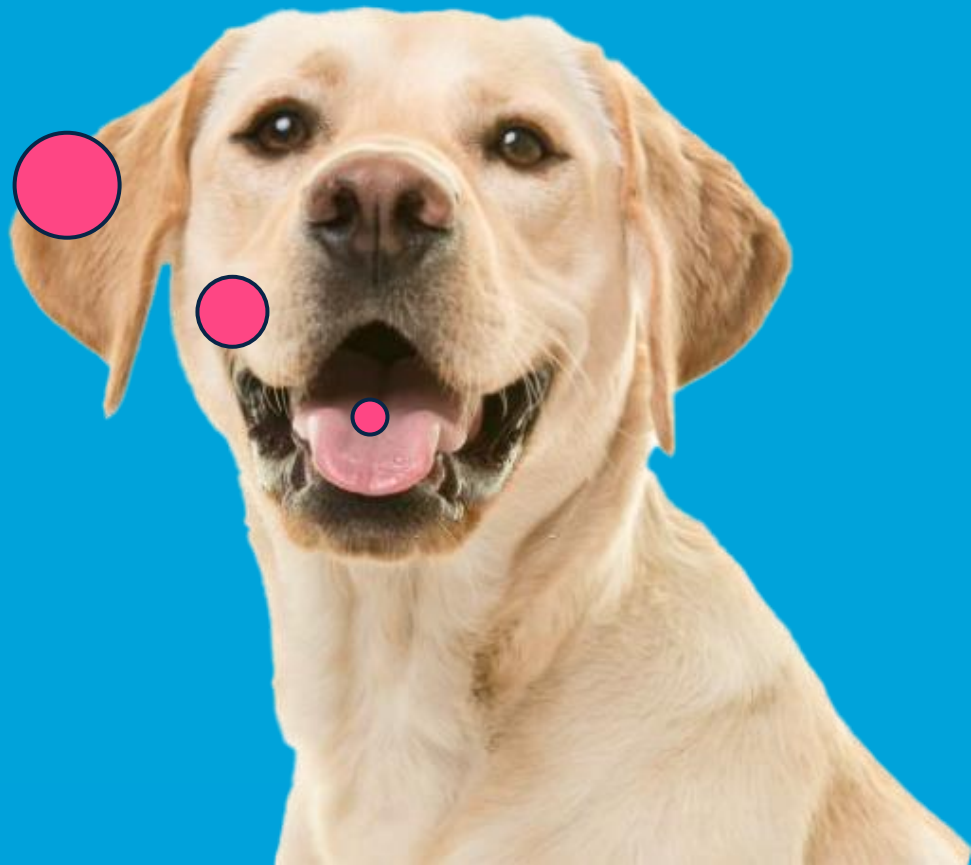
AP	S	R	Supervisor	User
00	0	0	No access	No access
00	1	0	Read only	No access
00	0	1	Read only	Read only
00	1	1	Do not use	
01	-	-	Read/write	No access
10	-	-	Read/write	Read only
11	-	-	Read/write	Read/ write

SMILIES
reSilient coMputer archItectures and LIfE Sciences

Politecnico di Torino
Department of Control and Computer Engineering

DAUIN

QUESTIONS?



THANK YOU!