

Computation structures  
Tutorial 4:  $\mu$ -code for ULg03

# ULg02 - constant ROM and XP register

- For handling exceptions, ULg02 introduces **the XP (R30) register** in which is stored the user program address at which the execution must return after the system have finished executed an exception handler.
- ULg02 introduces a control ROM storing useful constants:
  - Address of XP
  - Address of exception handlers

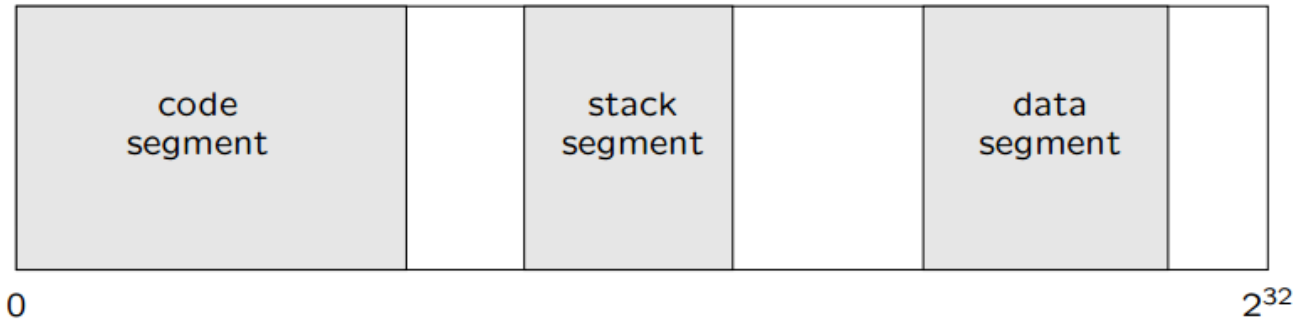
0x00	⋮
	⋮
0xFA	10000000 00000000 01000000 00000000 (Address of the handler "IRQ" — 0x2012)
0xFB	10000000 00000000 00100000 00000000 (Address of the handler "Supervisor Call" — 0x2008)
0xFC	10000000 00000000 01100000 00000000 (Address of the handler "Illegal Operation" — 0x2016)
0xFD	(Address of the handler "Cache Miss Code" — 0x2004)
0xFE	(Address of the handler "Cache Miss Data" — 0x2000)
0xFF	00000000 00000000 11110000 00000000 (Address of register XP)

# ULg03, introducing virtual memory

- **Virtual memory:** user programs have virtually access to a large contiguous space of dynamic memory (larger than the actual amount of available physical memory). In practice, the **memory is fragmented into pages** which can be stored on disk.
- **Goal:**
  - Make it easier to have several processes
  - A process cannot mess with the memory of other processes, nor the system memory
  - More memory available than actual amount of available physical memory
  - No need for reserving a fixed amount of memory for processes execution

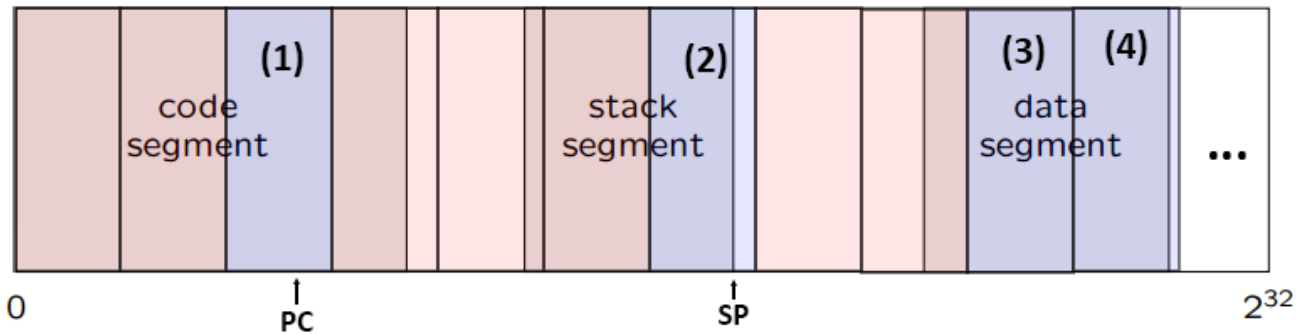
# Virtual memory – how it works

Virtual memory  
view of a user  
program



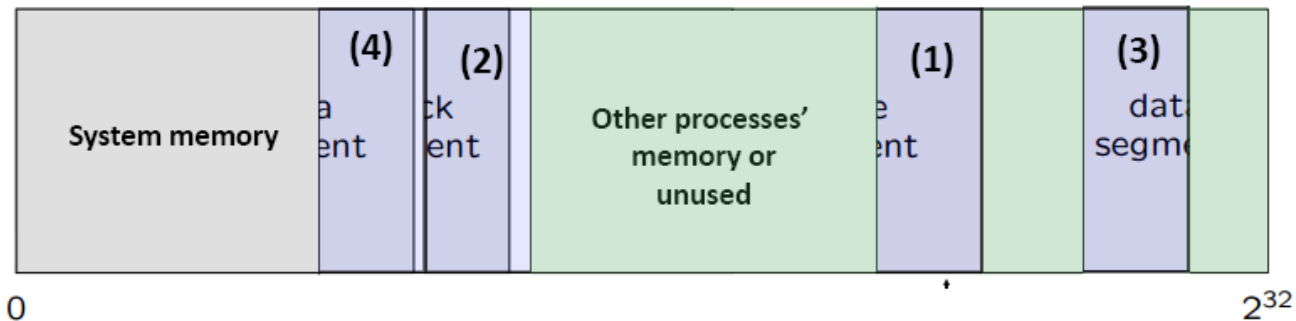
**System operates in physical memory while user programs operate in virtual memory**

Splitting in pages



A **page** is a fixed-size set of contiguous memory locations that can contain anything (code, stack, data,...)

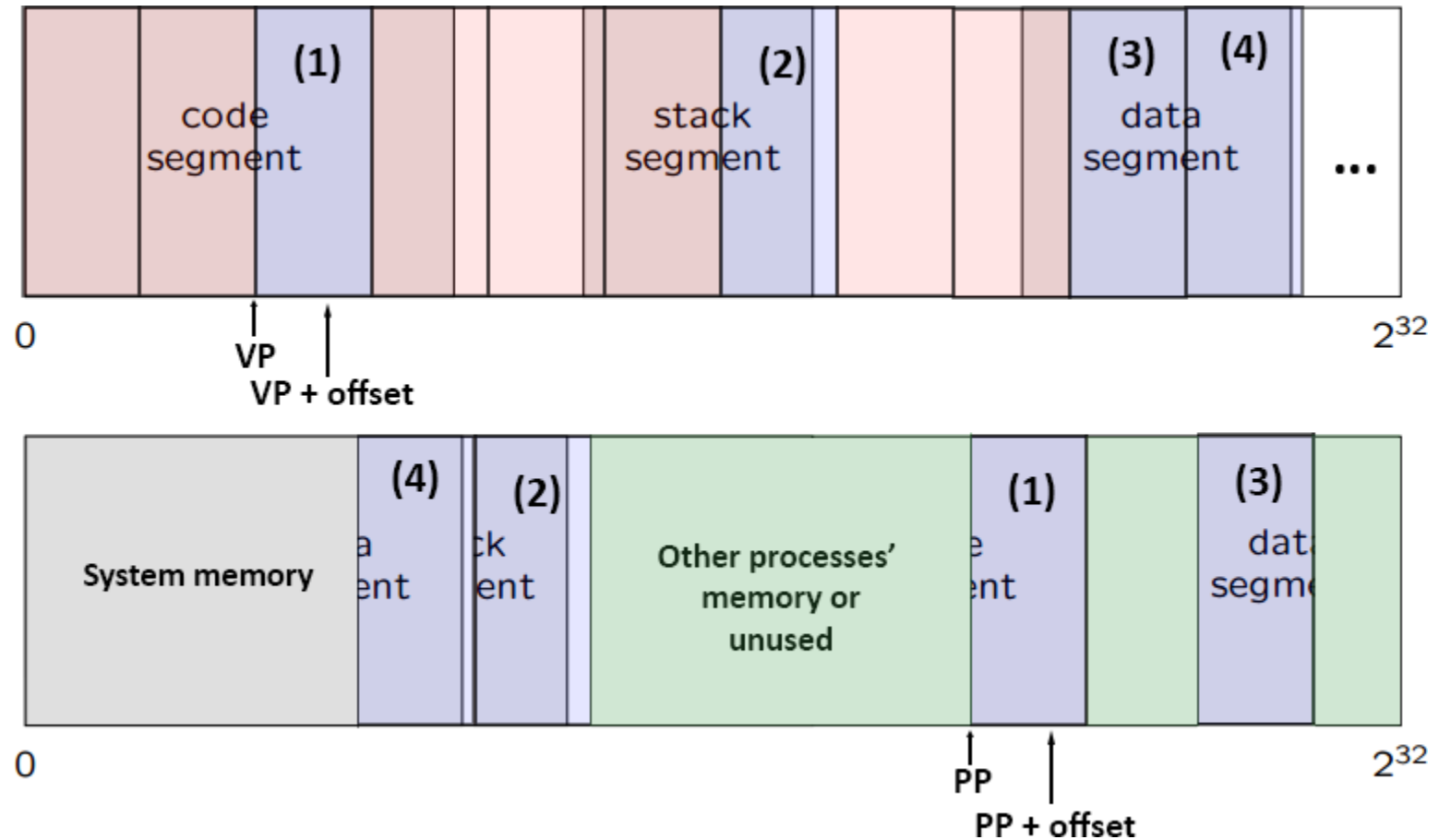
Physical (actual)  
memory



Useful pages are stored in memory while other can be stored on disk  $\Rightarrow$  virtually more memory available than actual physical memory

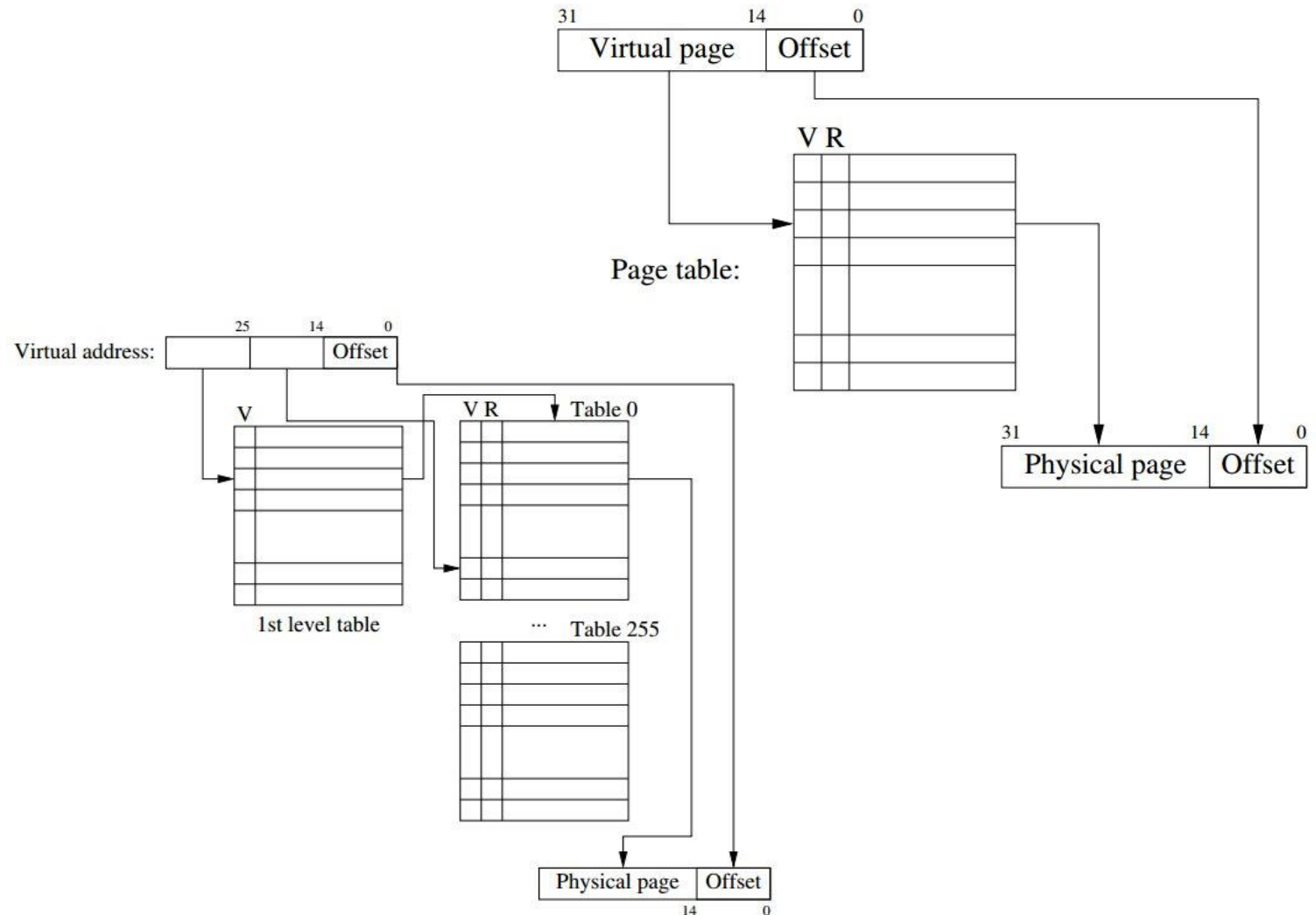
# Virtual memory – how it works

- **Virtual address:** an address in the virtual space
- **Physical address:** the address in the actual memory
- All addresses have two components:
  - Page address: VP (virtual page) or PP/SP (physical page/system page)
  - Offset: the address offset in the page



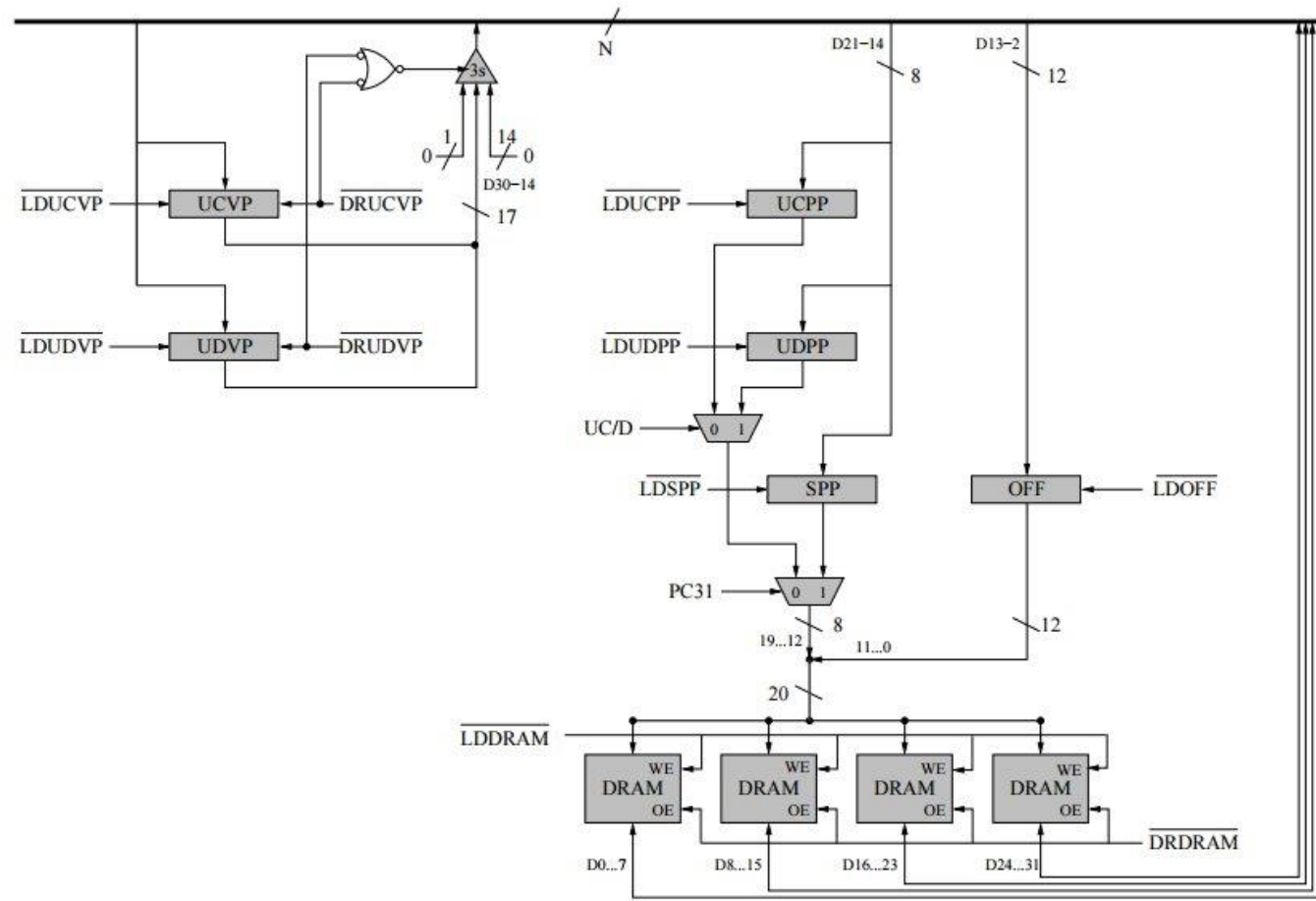
# Virtual memory – page table

- The system translates virtual pages (VP) into physical pages (PP) using a **page table**
- The page table is stored in the system memory
- Sometimes uses a two-level table for storage efficiency
- Need to access memory for translating virtual addresses ⇒ **very inefficient**



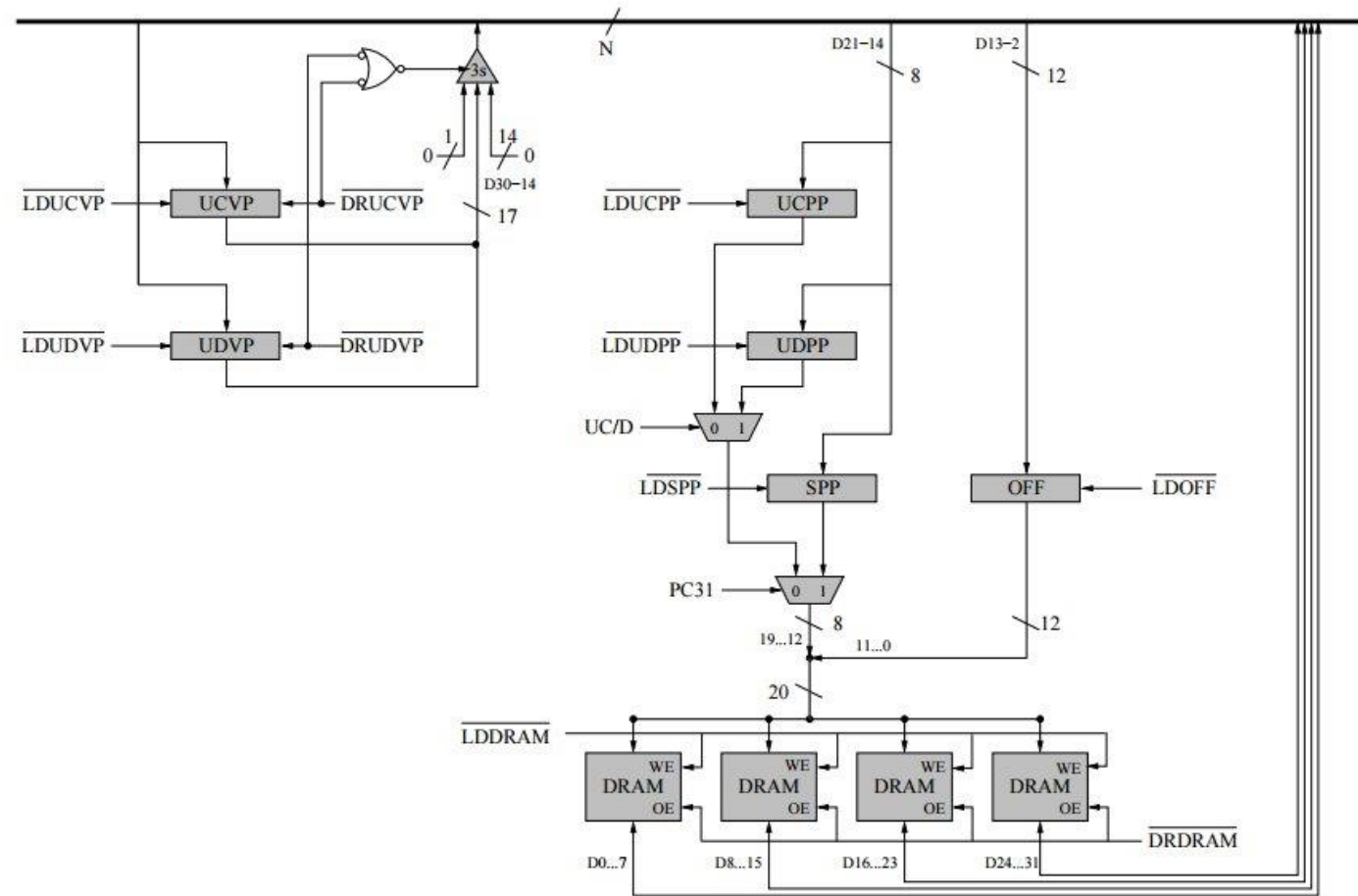
# Virtual memory – optimize look-up with caching

- **Locality principle:** accesses that are time-wise close are often address-wise close
- We can store PP, the result of the last translation of VP, in a register because the next translation will probably yield the same PP !
- Frequent memory accesses:
  - **Code access:** instructions
  - **Data access:** actual data or stack
- Both types of accesses are cached



# Virtual memory – optimize look-up with caching

- In user mode:
  - **U\*VP and U\*PP** contains respectively virtual page address and translated physical address
  - **Code:** UCVP and UCPP
  - **Data:** UDVP and UDPP
  - UC/D is a flag allowing to choose the cache among those two
- In SVR mode (PC31 = 1), uses physical memory:
  - SPP: System Physical Page
- In both cases, the offset is stored in register OFF





# Virtual memory – cache hit and miss

- **Cache hit:** if the cache register contains the expected address.
- **Cache miss:** if the cache register doesn't contain the expected address.
- Cache miss triggers a « Cache miss code » or a « Cache miss data » **exception** that is handled at the  $\mu$ -code level.
- The handlers of those exceptions update the corresponding cache with the expected address

0x00	⋮
0xFA	10000000 00000000 01000000 00000000 (Address of the handler "IRQ" — 0x2012)
0xFB	10000000 00000000 00100000 00000000 (Address of the handler "Supervisor Call" — 0x2008)
0xFC	10000000 00000000 01100000 00000000 (Address of the handler "Illegal Operation" — 0x2016)
0xFD	(Address of the handler "Cache Miss Code" — 0x2004)
0xFE	(Address of the handler "Cache Miss Data" — 0x2000)
0xFF	00000000 00000000 11110000 00000000 (Address of register XP)

Control Inputs						Output
F3	F2	F1	F0	C	M	
0	0	1	1	1	1	0
1	1	0	0	1	1	0xffffffff
1	1	1	1	1	1	A
1	0	1	0	1	1	B
1	1	1	1	1	0	A-1
0	0	0	0	0	0	A+1
1	0	0	1	1	0	A+B
0	1	1	0	0	0	A-B
1	0	1	1	1	1	A and B
1	1	1	0	1	1	A or B
0	1	1	0	1	1	A xor B
0	0	0	0	1	1	not A
1	1	0	0	1	0	A+A
0	1	1	0	1	0	A-B-1
1	0	0	1	x	1	not (A xor B)
0	1	0	0	x	1	not (A and B)

flags:  
E=1 <=> Alu out = 0xffffffff  
C : negated carry out  
N = most sign. Alu out: a31

