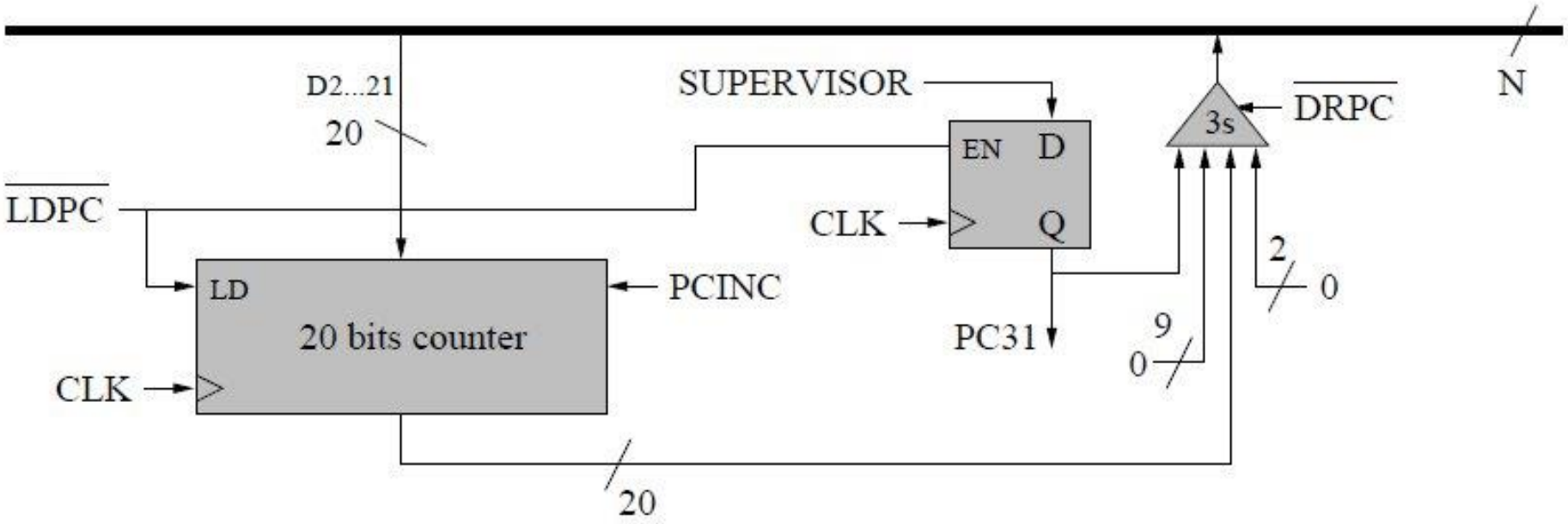
Computation structures
Tutorial 3: μ-code for ULg02

ULg02, a shared machine

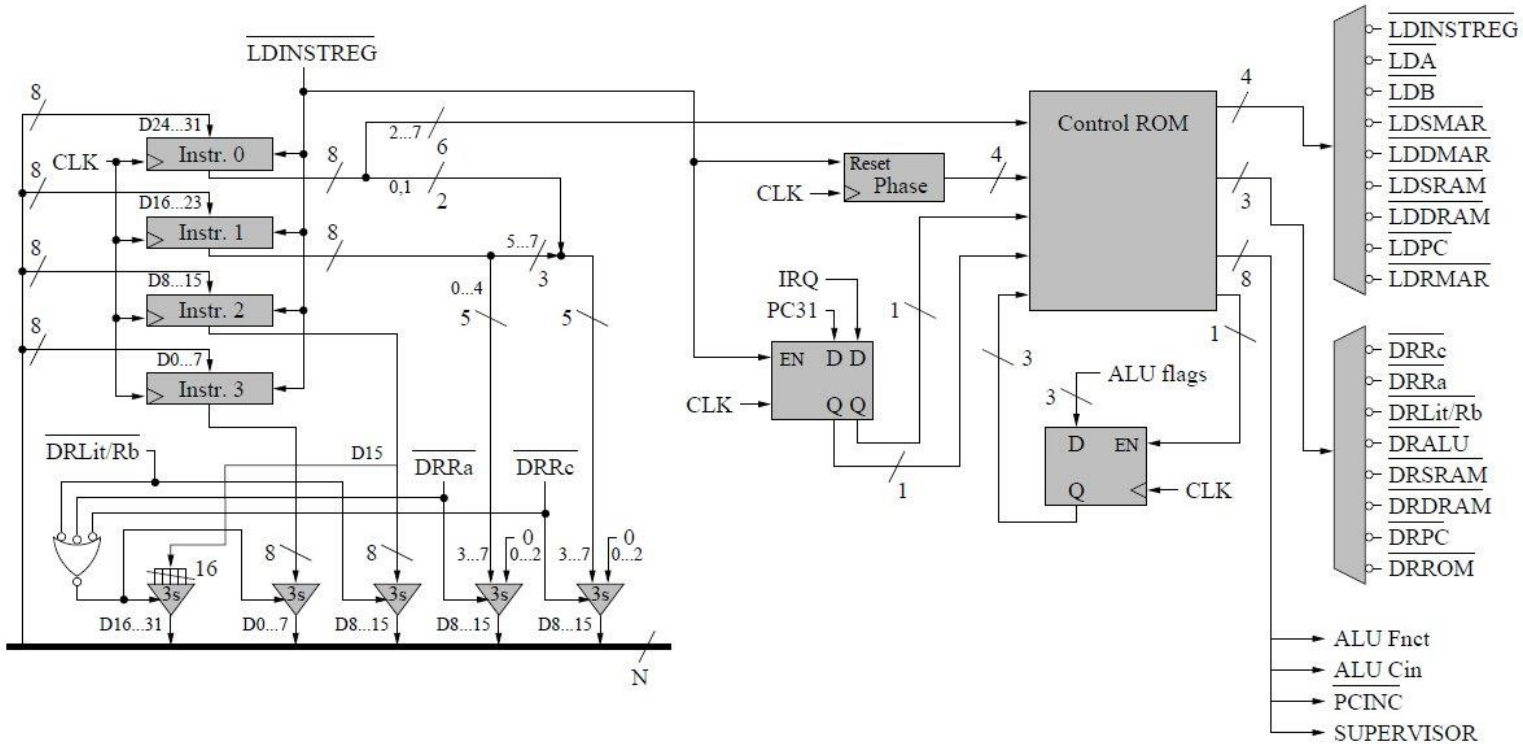
- **Goal**:
  - allowing user programs developers not having to care about low level details
  - having a system which can provide basic functions to user programs (Input / Output,…)
  - preventing a user program to corrupt the machine and put it in an incoherent state
- ULg01 is modified to feature two execution contexts:
  - **User mode**: for executing users' (your) programs
  - **Supervisor mode (SVR)**: for executing system code
- Supervisor mode implemented in the hardware

Supervisor mode – Program Counter



- PC31, the sign bit of PC indicates the mode (0 for user mode, 1 for SVR mode)
- Supervisor mode can be set by **explicitely activating** the **SUPERVISOR** signal **when PC is loaded**.
- If SVR is not activated when PC is loaded, the context is changed to user mode !

Supervisor mode – Control Unit



Control ROM

New inputs:

- **PC31**: supervisor mode enabled ?
- IRQ: received an interrupt ?

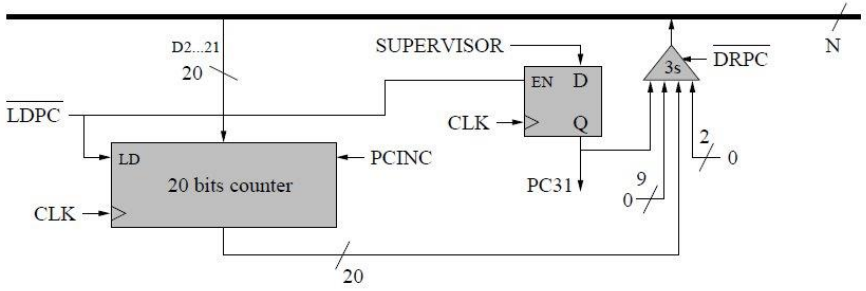
New outputs:

- **SUPERVISOR** (SVR): enable SVR mode
- LDRMAR and DRROM: constant ROM

SVR mode and µ-code

Two cases:

- PC31 == 1:
  - SUPERVISOR is always set to 1 for all instructions (except JMP and variants) so that program remain in supervisor mode
  - For jump instructions, SUPERVISOR is set with the sign bit of the jumping address enabling switching to user mode
- PC31 == 0:
  - SUPERVISOR is always set to 0 (**switching manually from user mode to SVR mode is forbidden**)
  - Few exceptions: SVC() and interrupts



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