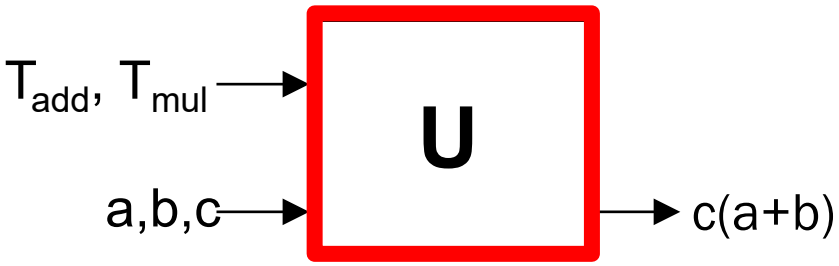
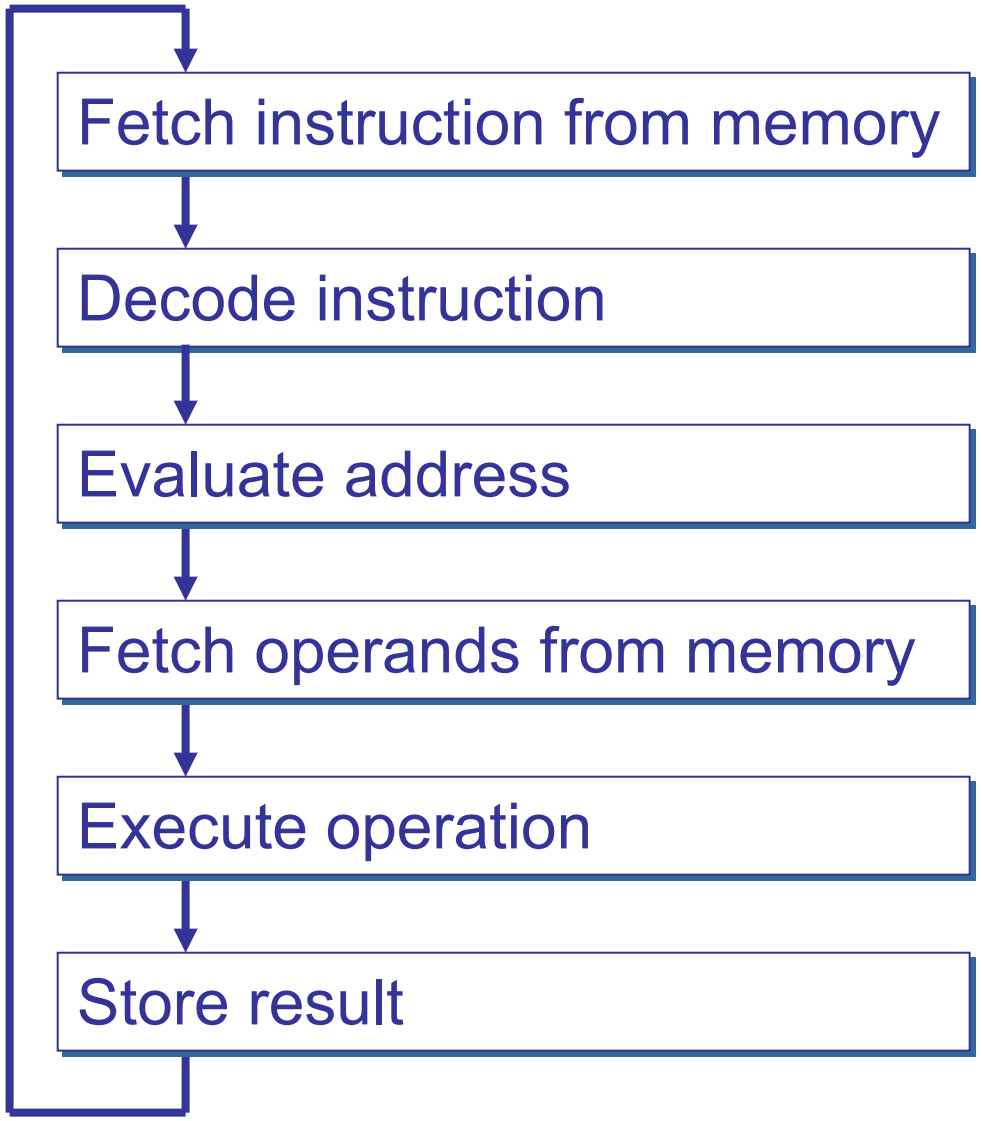
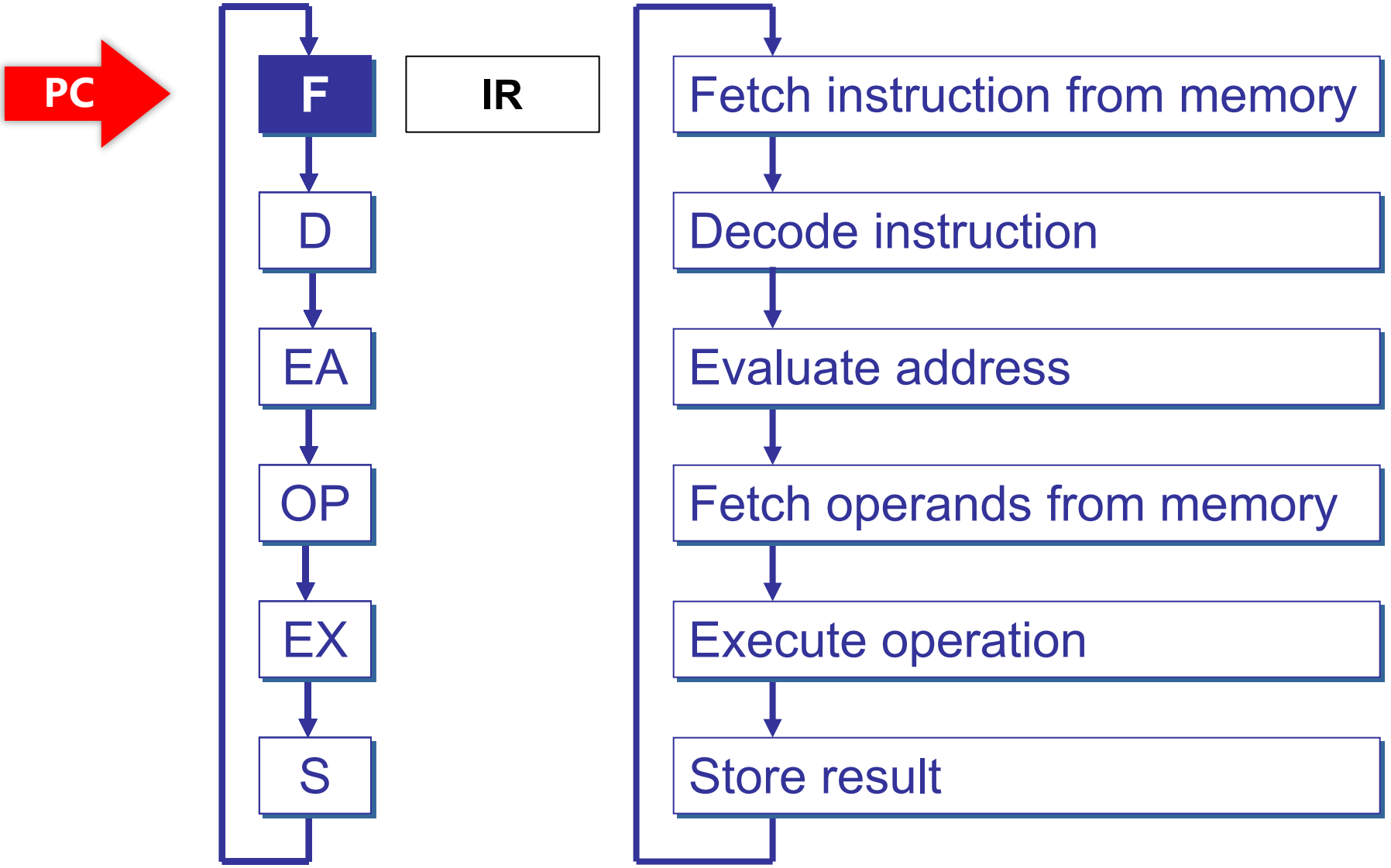
Great Idea #1 Turing Machine (Computataional Model)



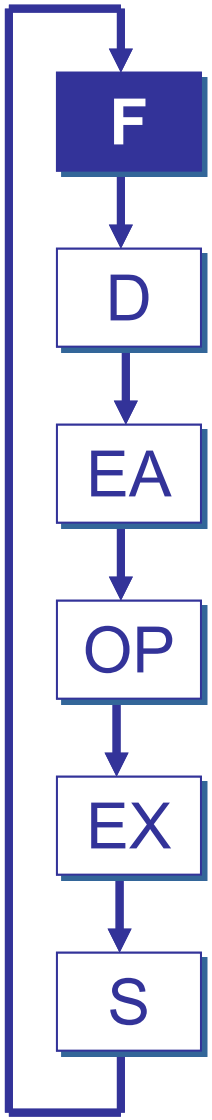
Universal Turing Machine

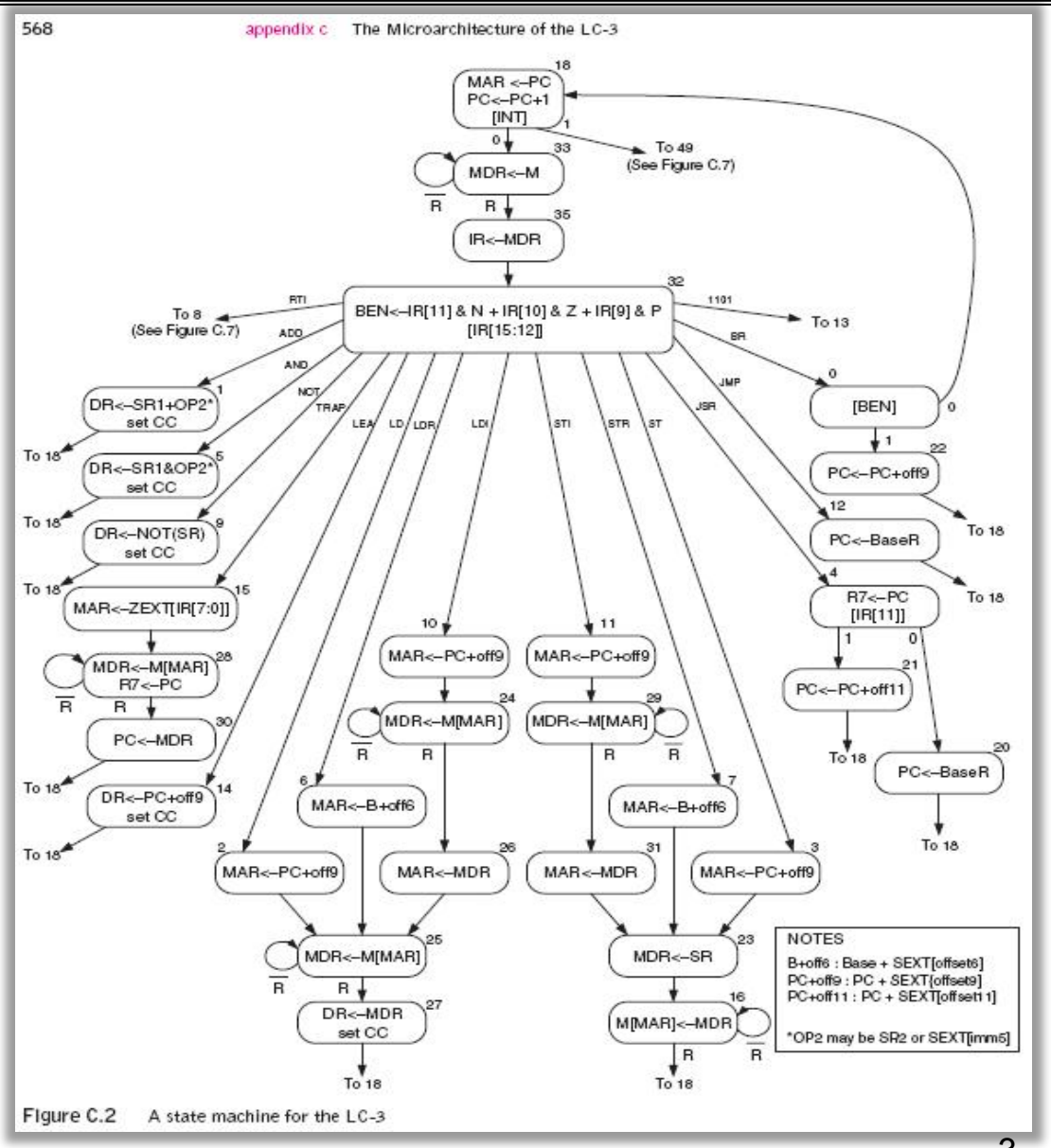


Instruction Processing: State Transtion

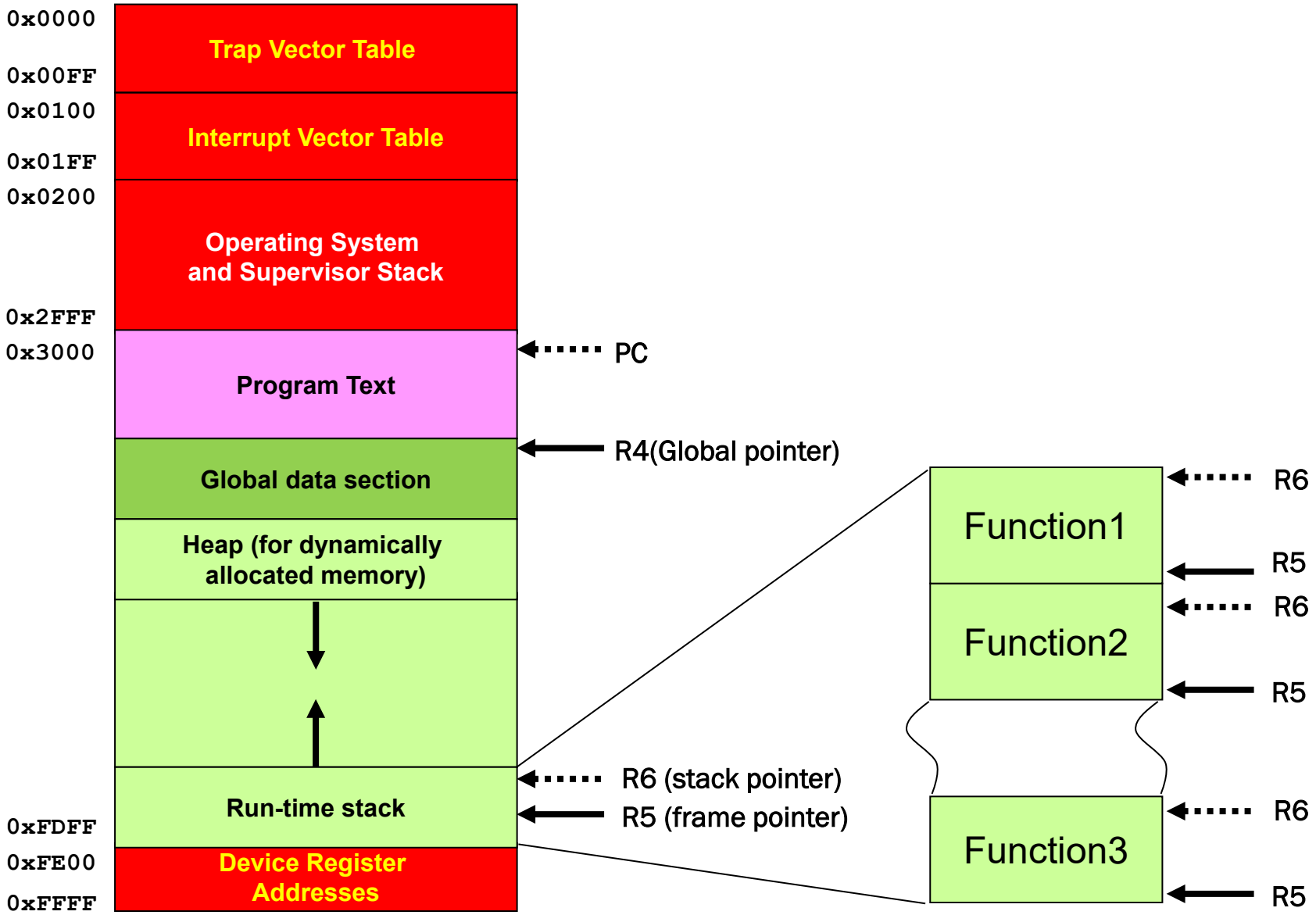


Instruction Processing: Finite State Automata





LC-3 Overview: Memory Map



LC-3 ISA Overview

运算指令(Operate Instructions)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADD	0	0	0	1	DR			SR1			0	0	0	SR2		
ADD	0	0	0	1	DR			SR1			1	Imm5				
AND	0	1	0	1	DR			SR1			0	0	0	SR2		
AND	0	1	0	1	DR			SR1			1	Imm5				
NOT	1	0	0	1	DR			SR1			1	1	1	1	1	1
Reserved	1	1	0	1												

控制指令(Control Instructions)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BR	0	0	0	0	n	z	p	PCoffset9								
JSR	0	1	0	0	1	PCoffset11										
JSRR	0	1	0	0	0	0	0	BaseR			0	0	0	0	0	0
RTI	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
JMP	1	1	0	0	0	0	0	BaseR			0	0	0	0	0	0
RET	1	1	0	0	0	0	0	1	1	1	0	0	0	0	0	0
TRAP	1	1	1	1	0	0	0	0	TrapVector8							

数据移动指令 (Data Movement Instructions)

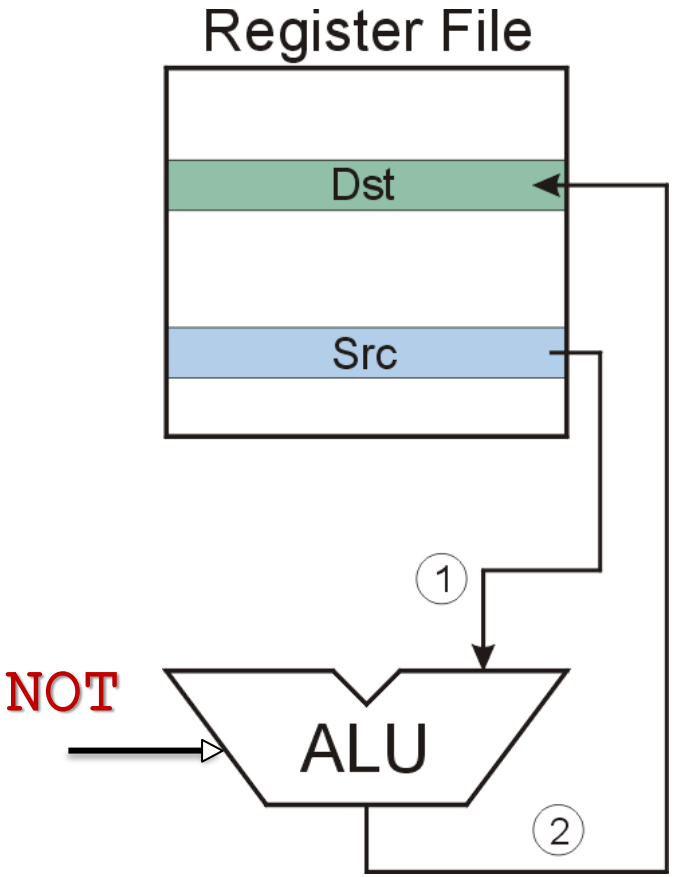
取数指令(Load)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LD	0	0	1	0	DR			PCoffset9								
LDR	0	1	1	0	DR			BaseR			PCoffset6					
LDI	1	0	1	0	DR			PCoffset9								
LEA	1	1	1	0	DR			PCoffset9								

存数指令(Store)

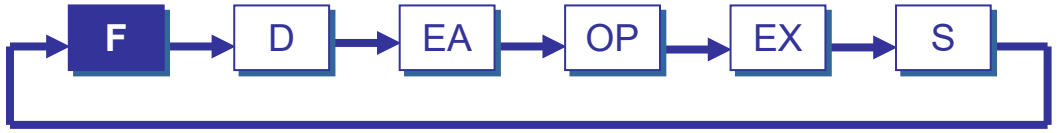
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ST	0	0	1	1	SR			PCoffset9								
STR	0	1	1	1	SR			BaseR			PCoffset6					
STI	1	0	1	1	SR			PCoffset9								

NOT (Register)

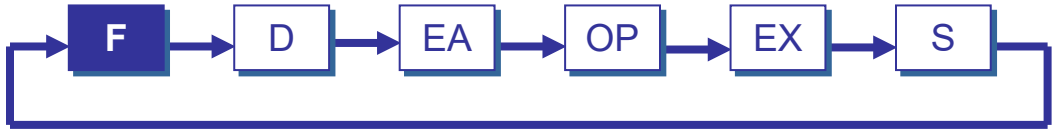


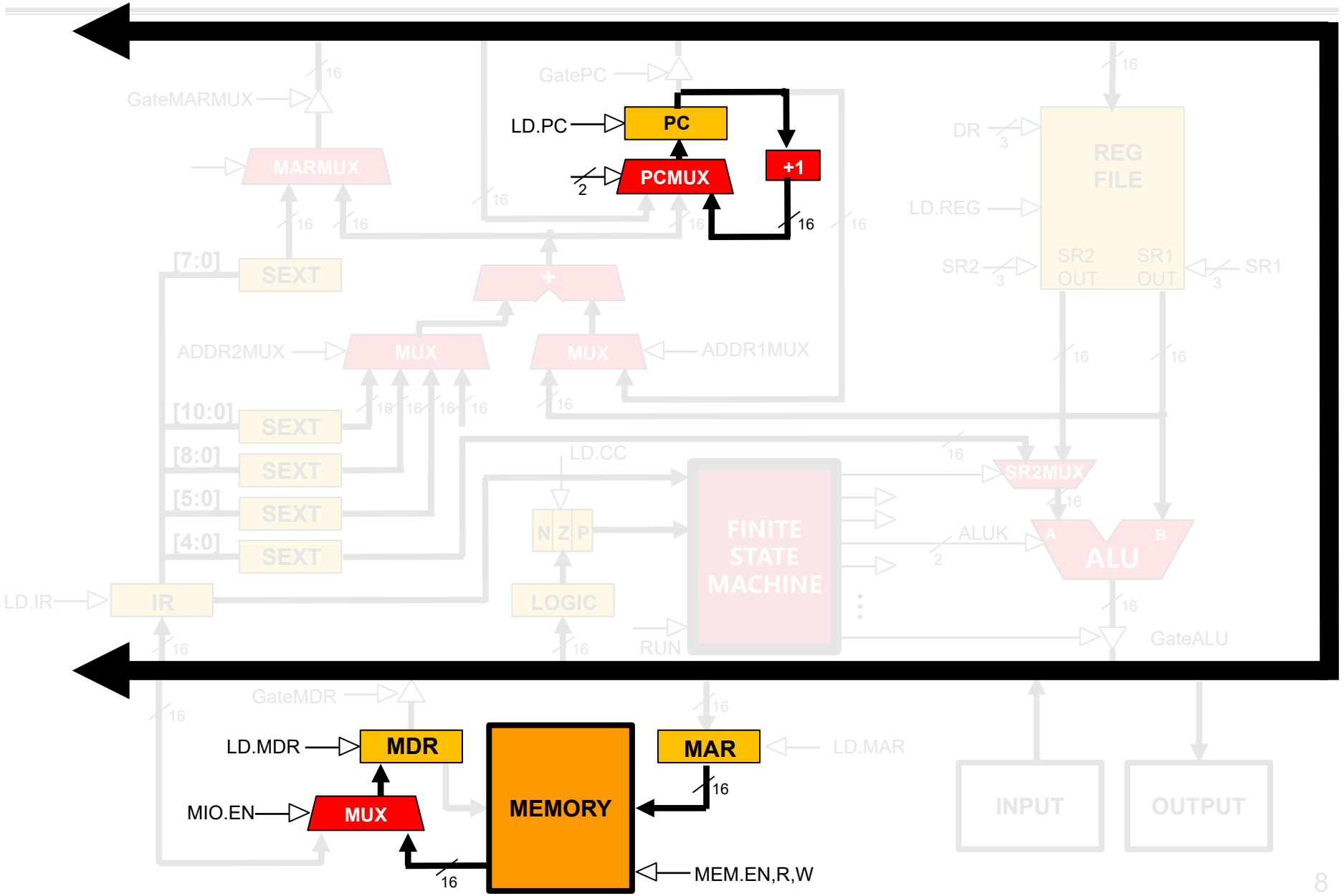
Note: Src and Dst could be the same register.

NOT (Register):

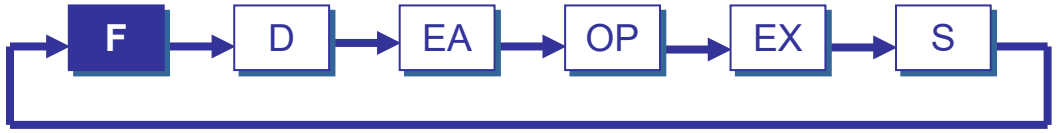


NOT (Register):

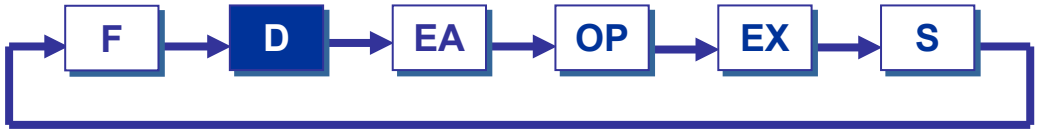


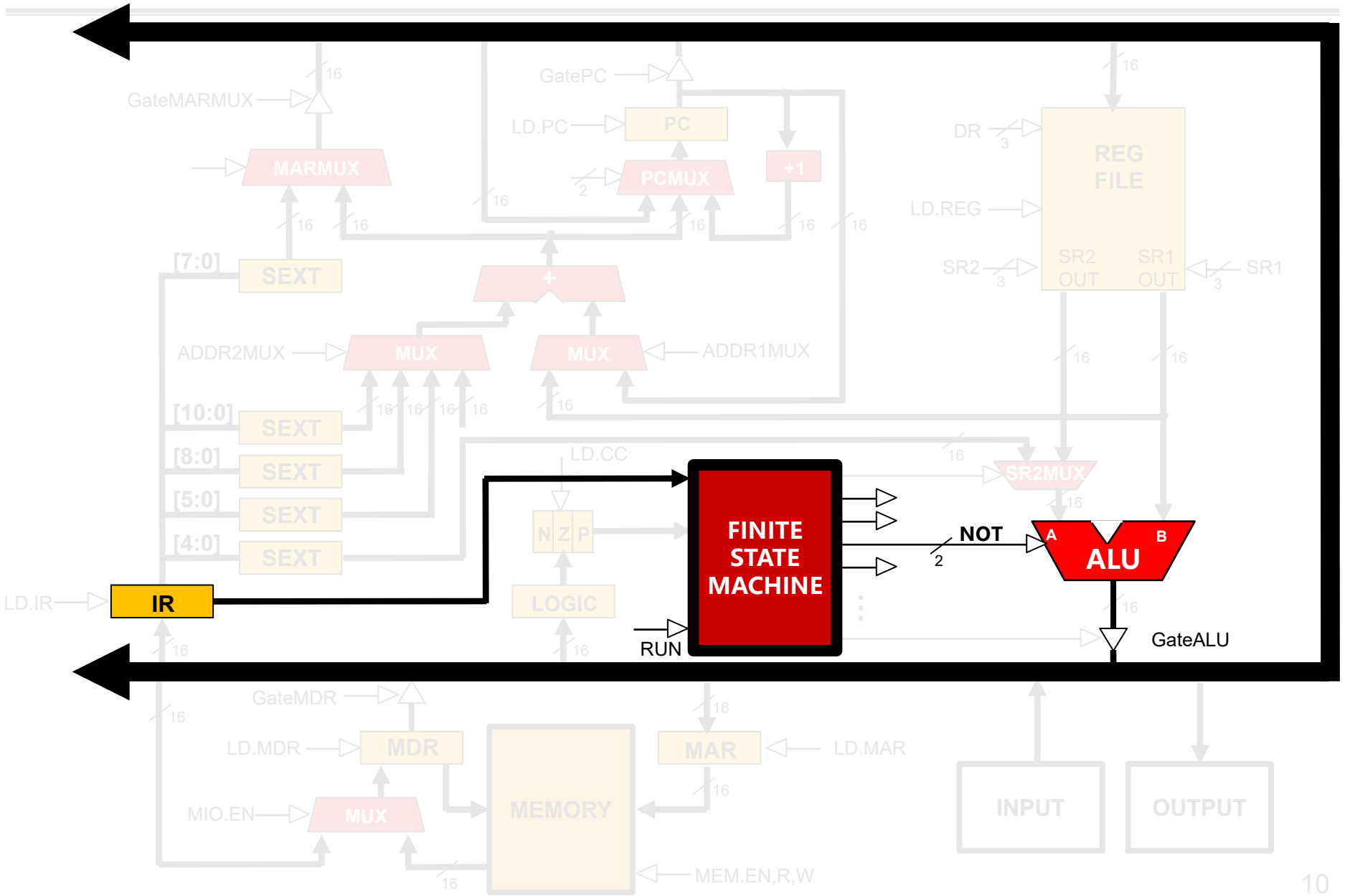


NOT (Register):

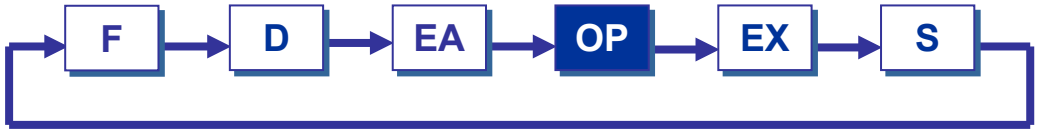


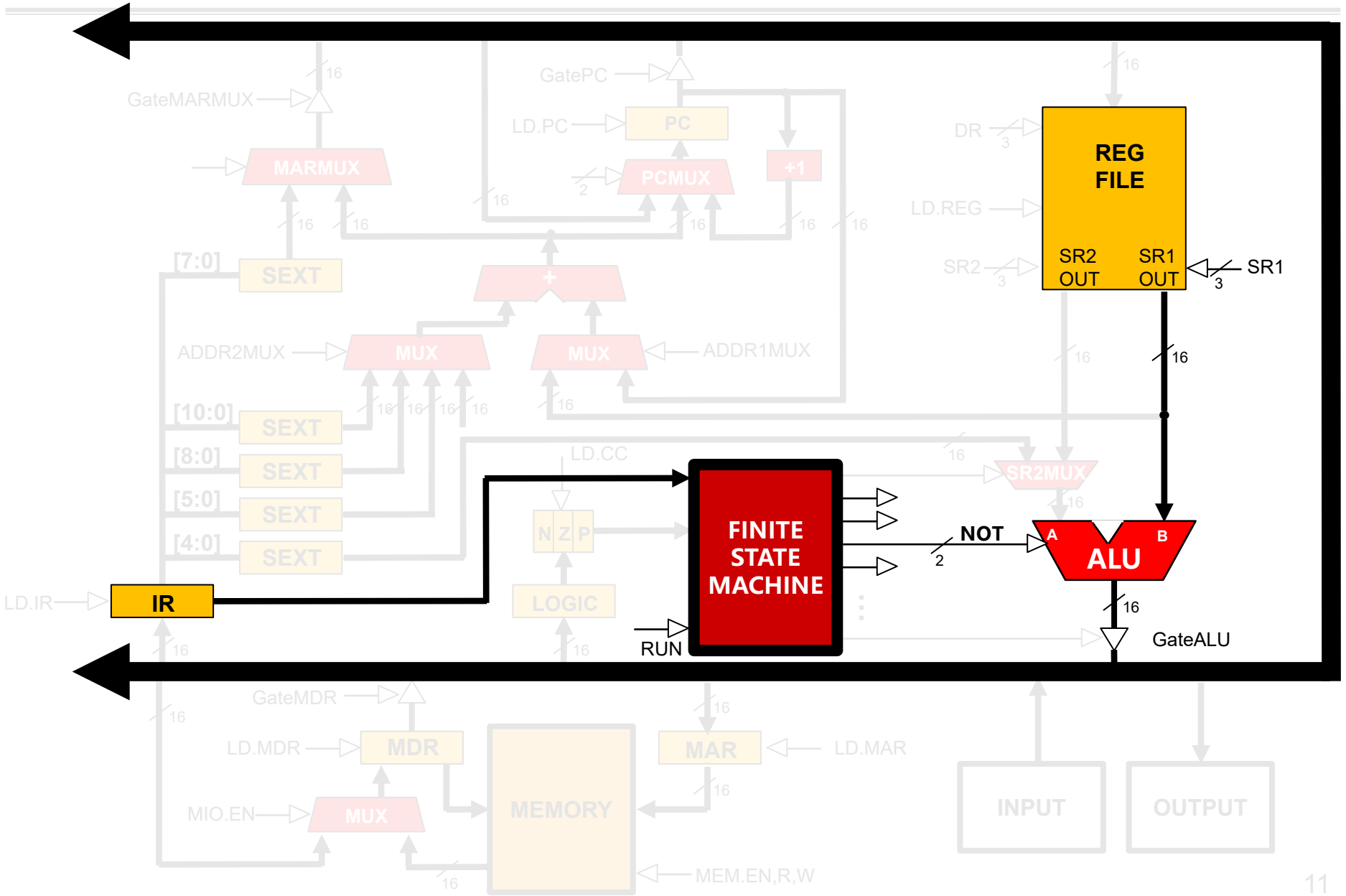
NOT (Register):



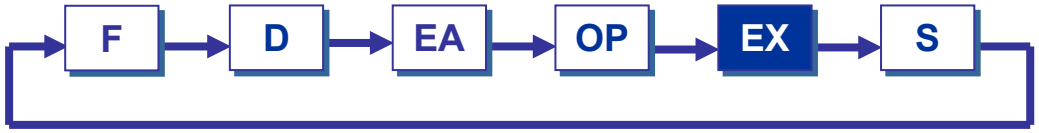


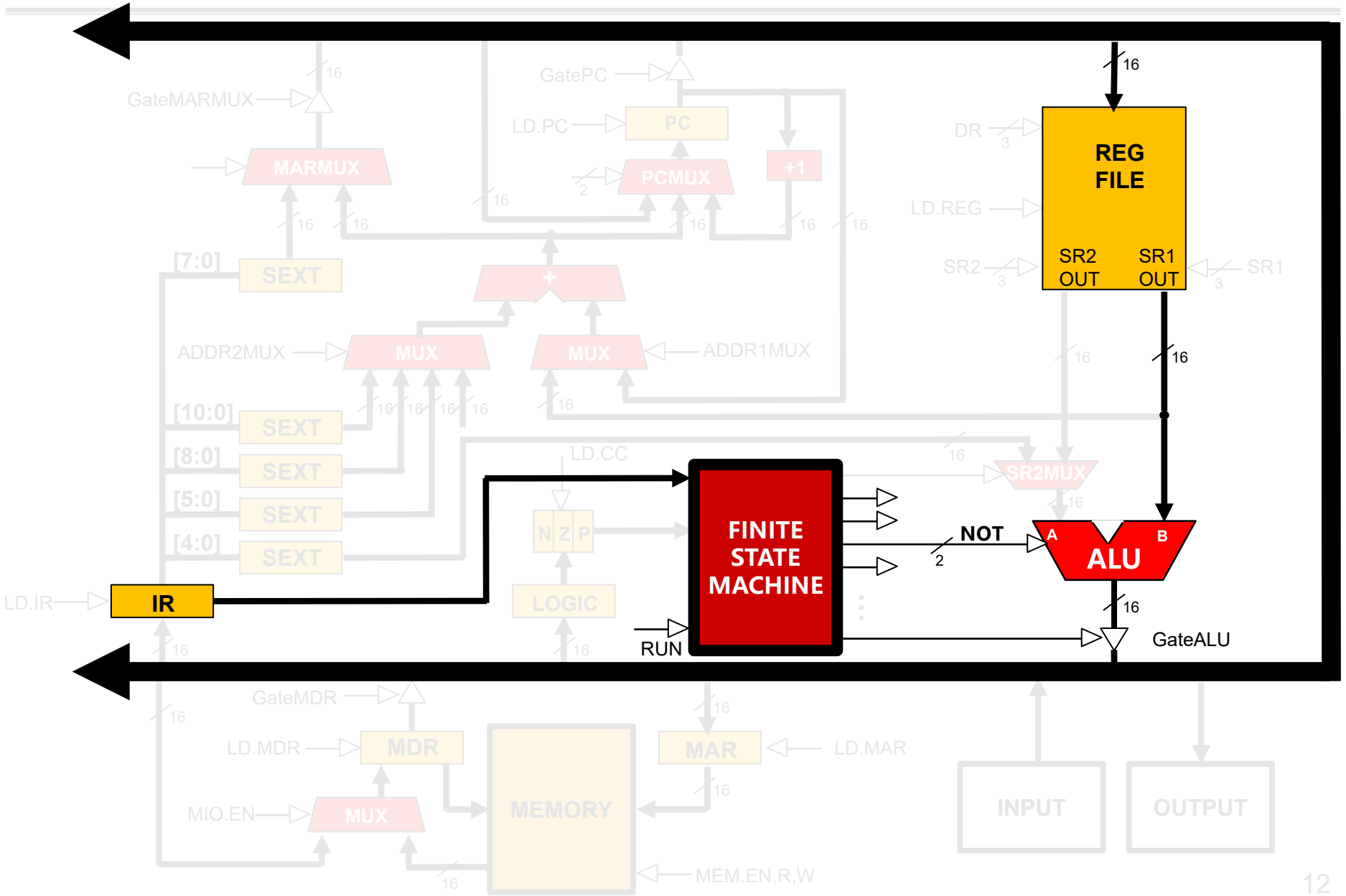
NOT (Register):



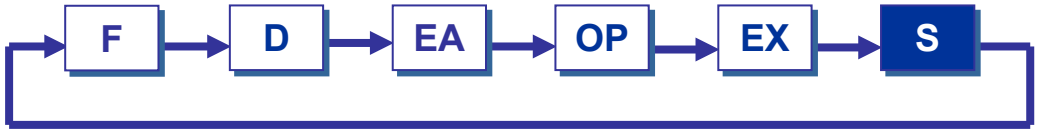


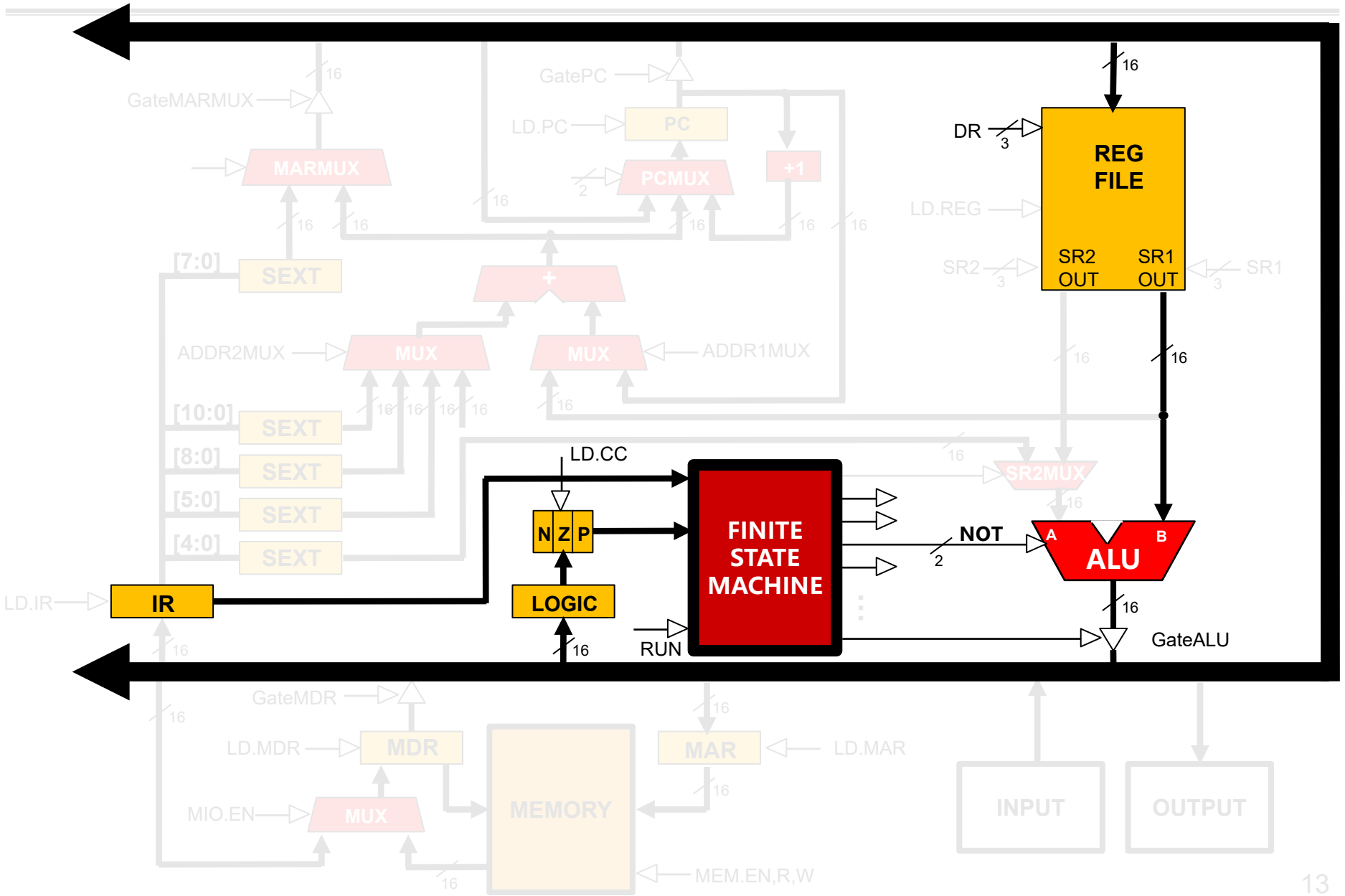
NOT (Register):





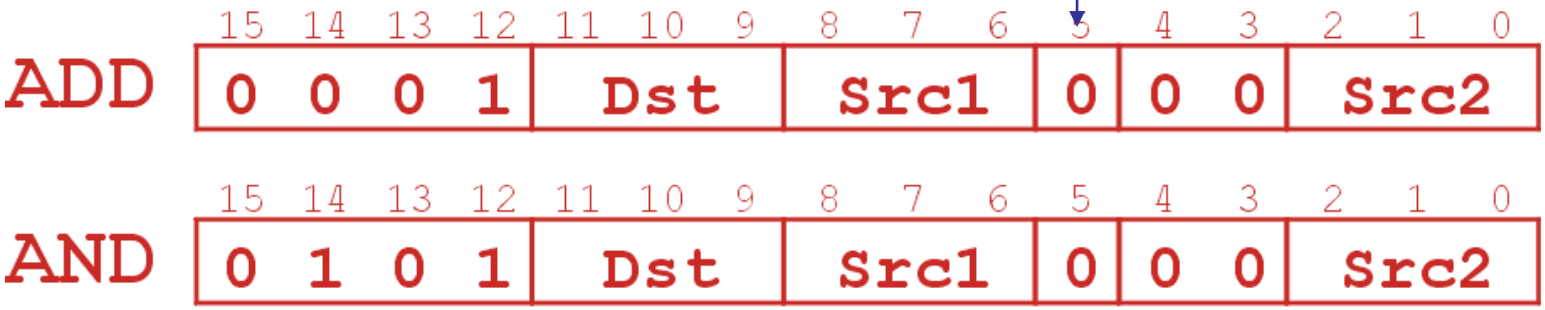
NOT (Register):

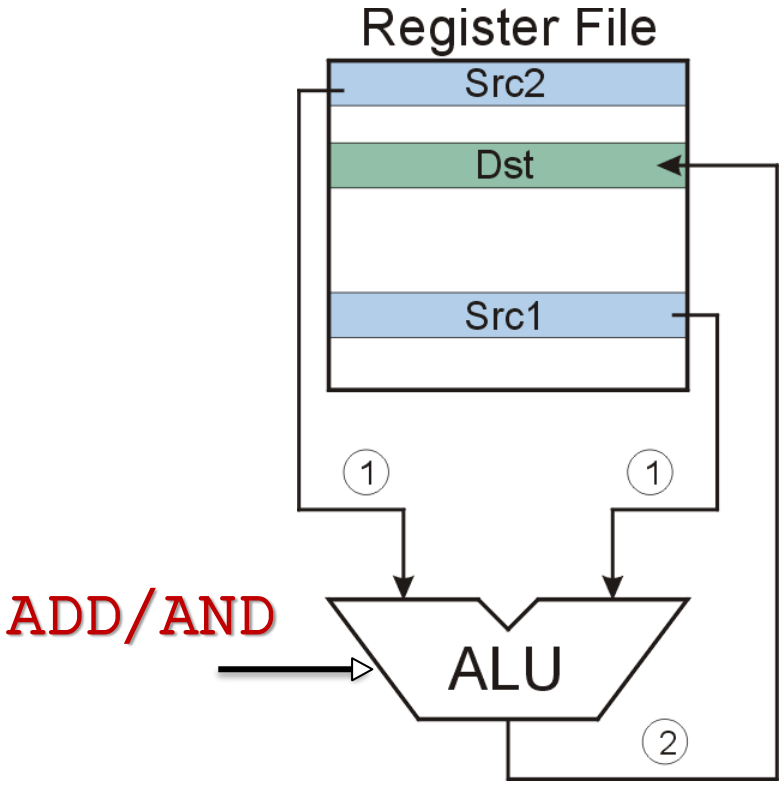




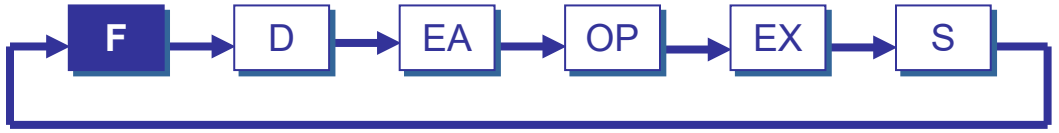
ADD/AND (Register)

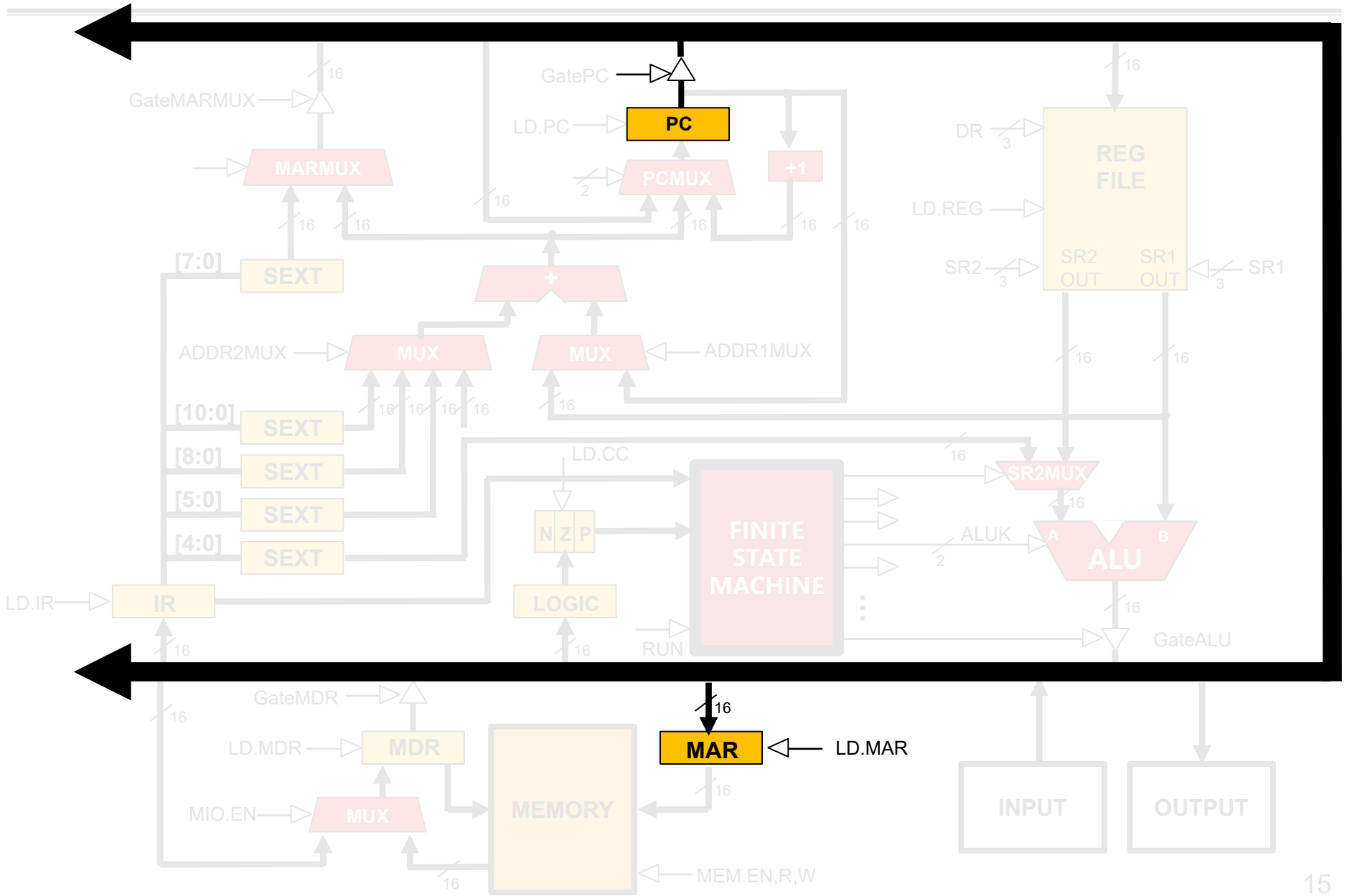
this zero means "register mode"



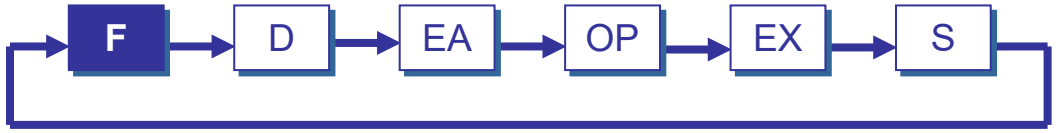


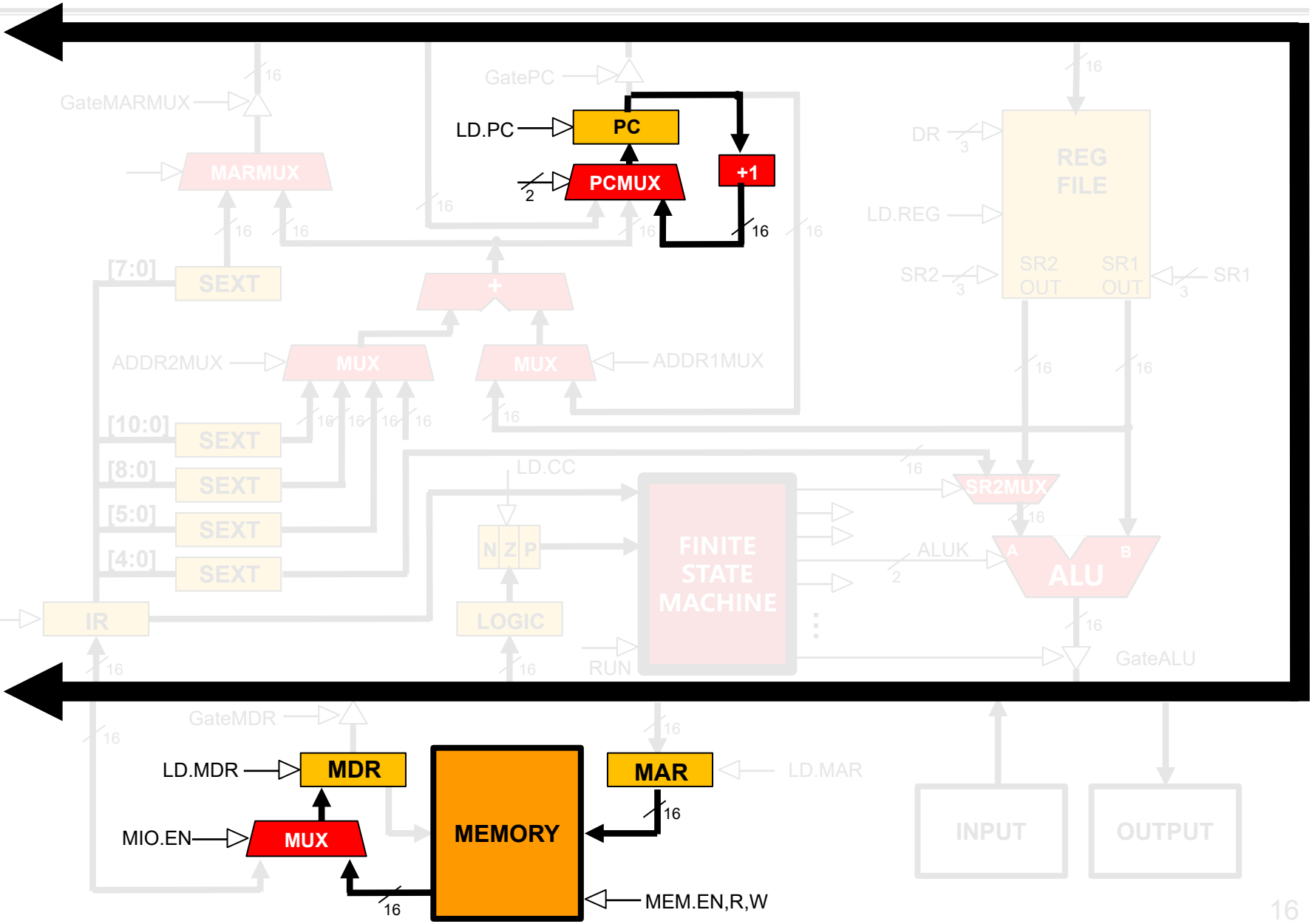
ADD/AND (Register)



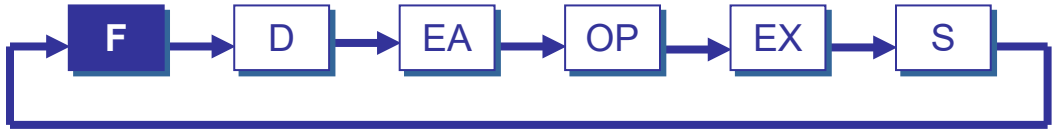


ADD/AND (Register)

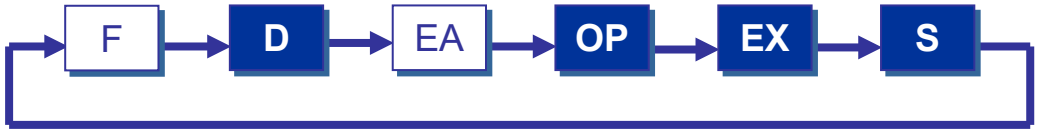


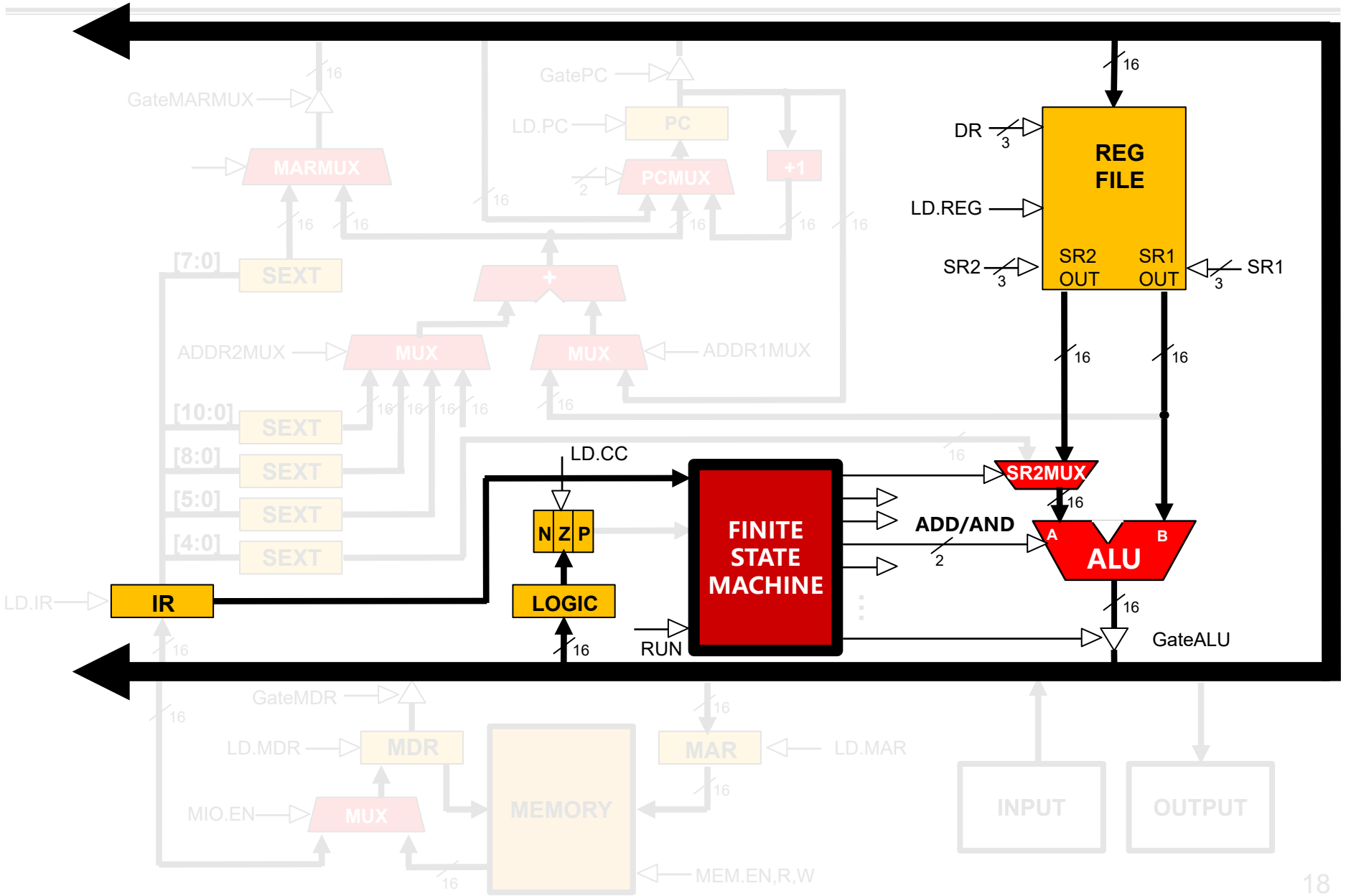


ADD/AND (Register)



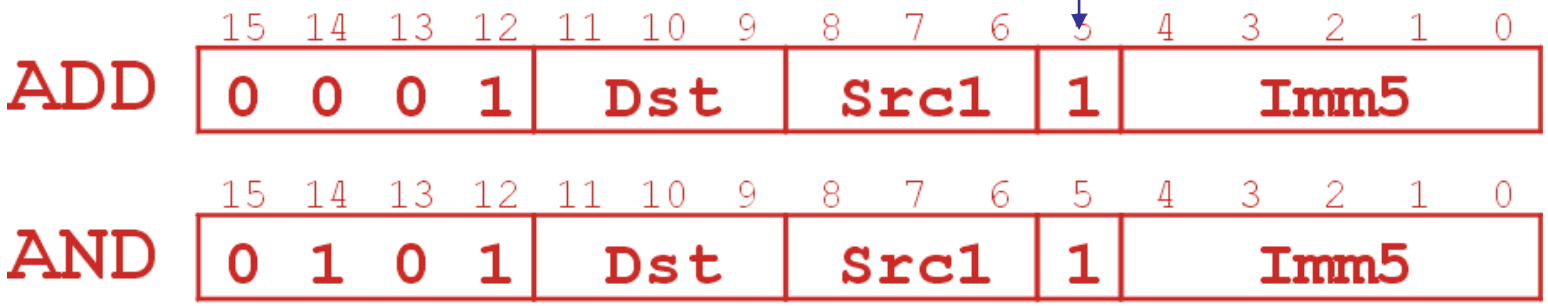
ADD/AND (Register)



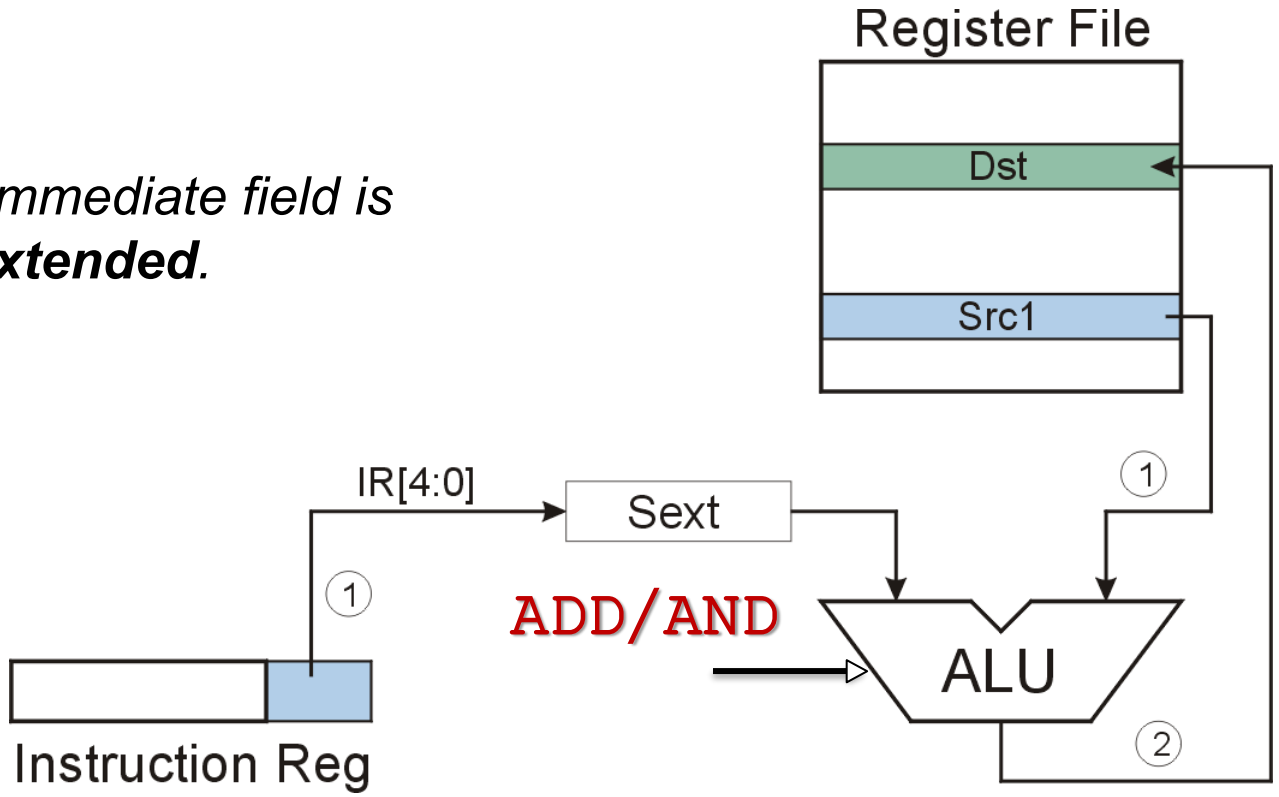


ADD/AND (Immediate)

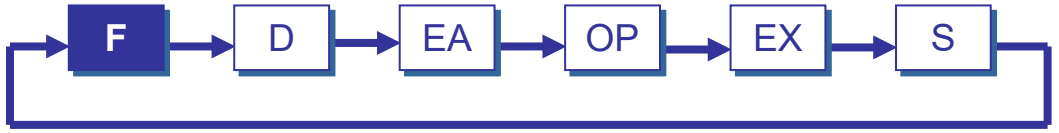
this one means "immediate mode"

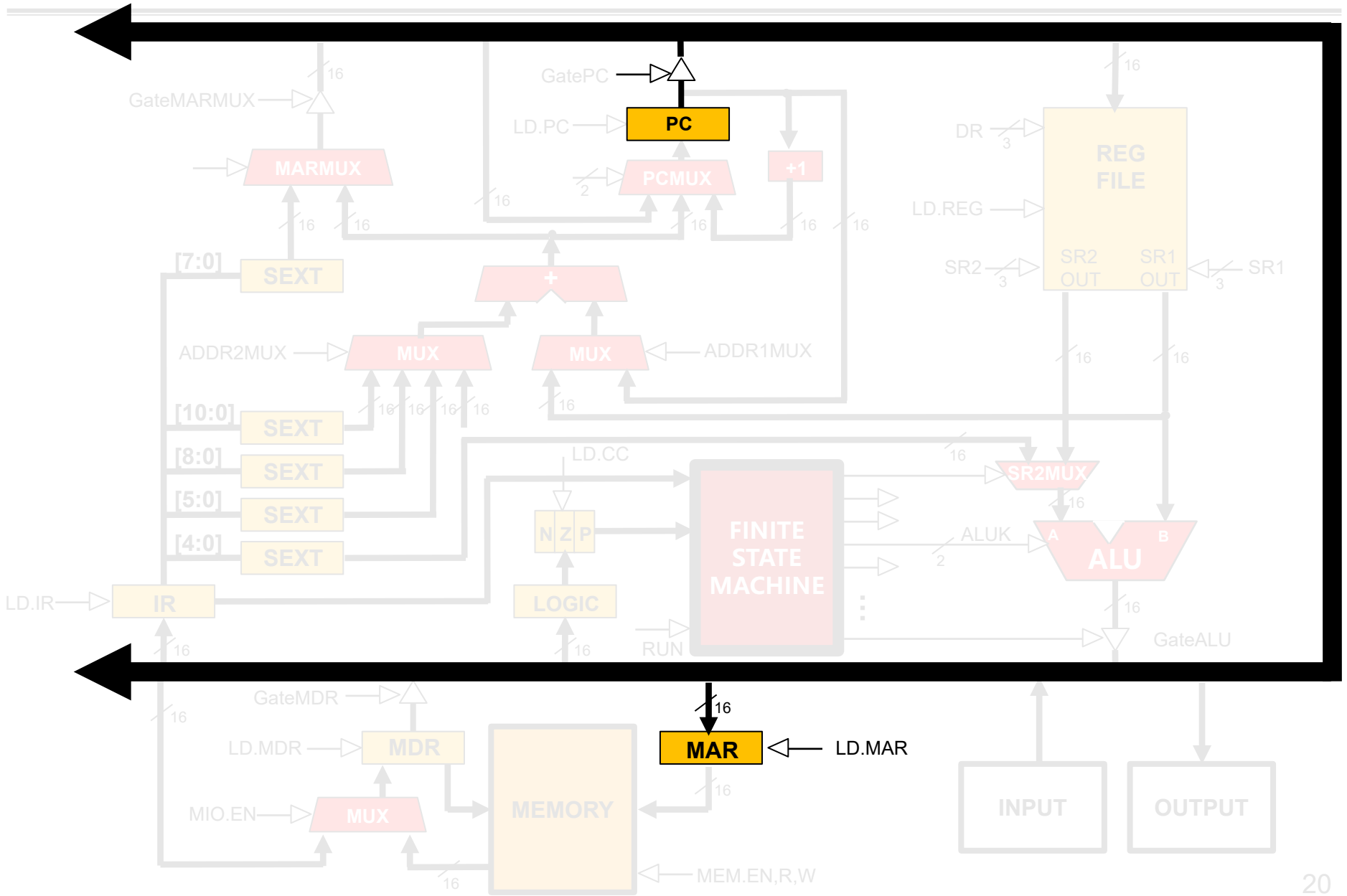


Note: Immediate field is ***sign-extended****.*

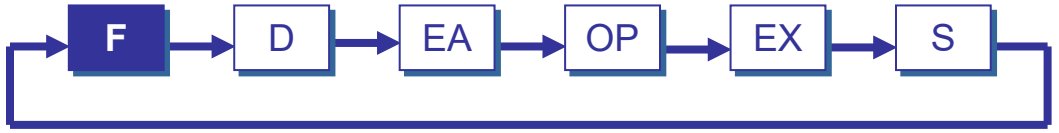


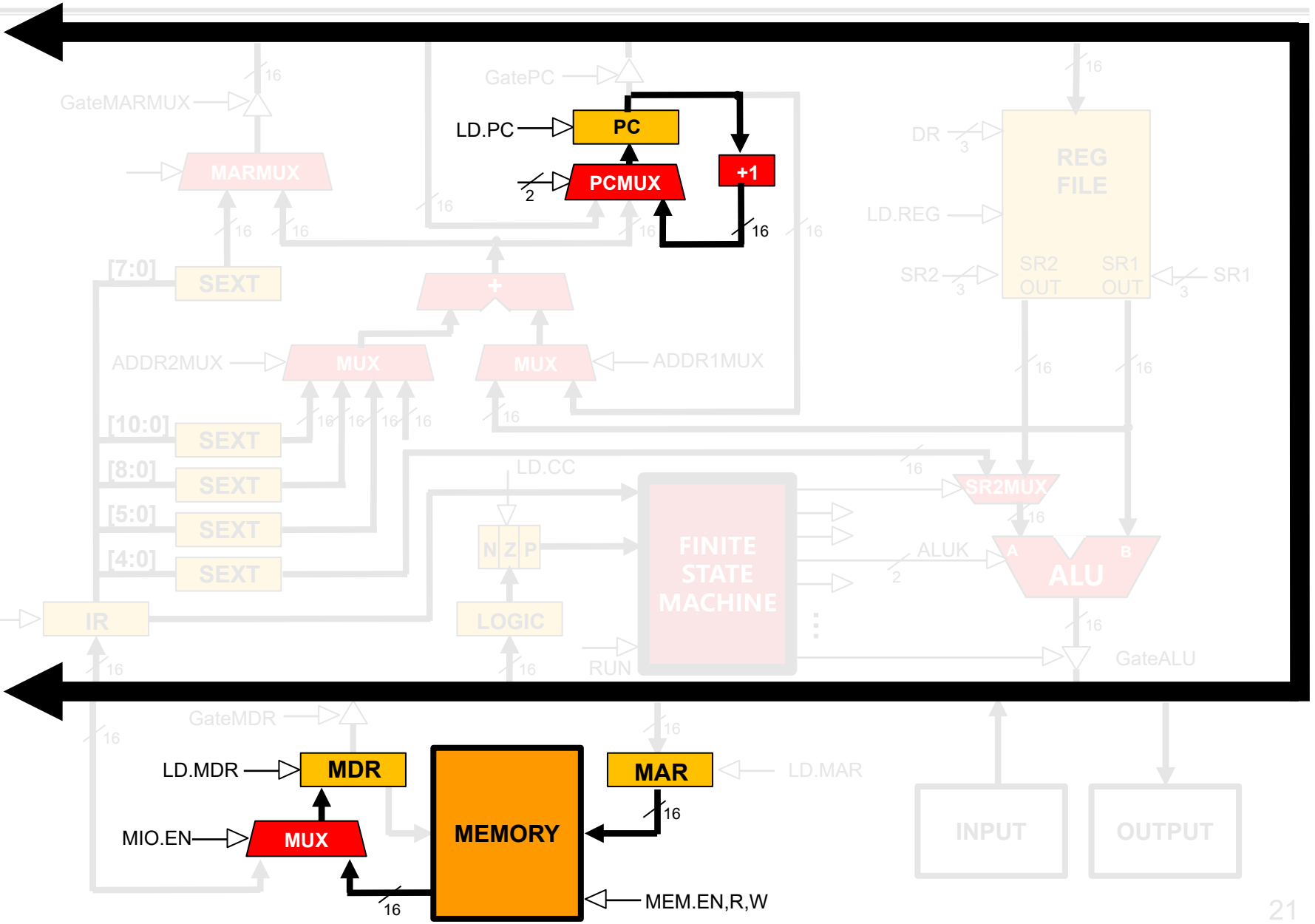
ADD/AND (Immediate)



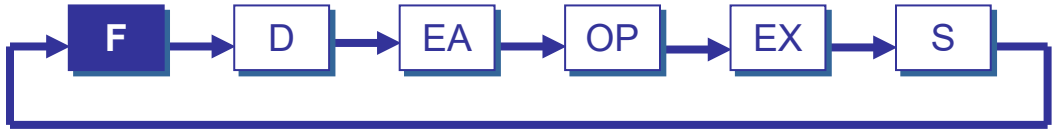


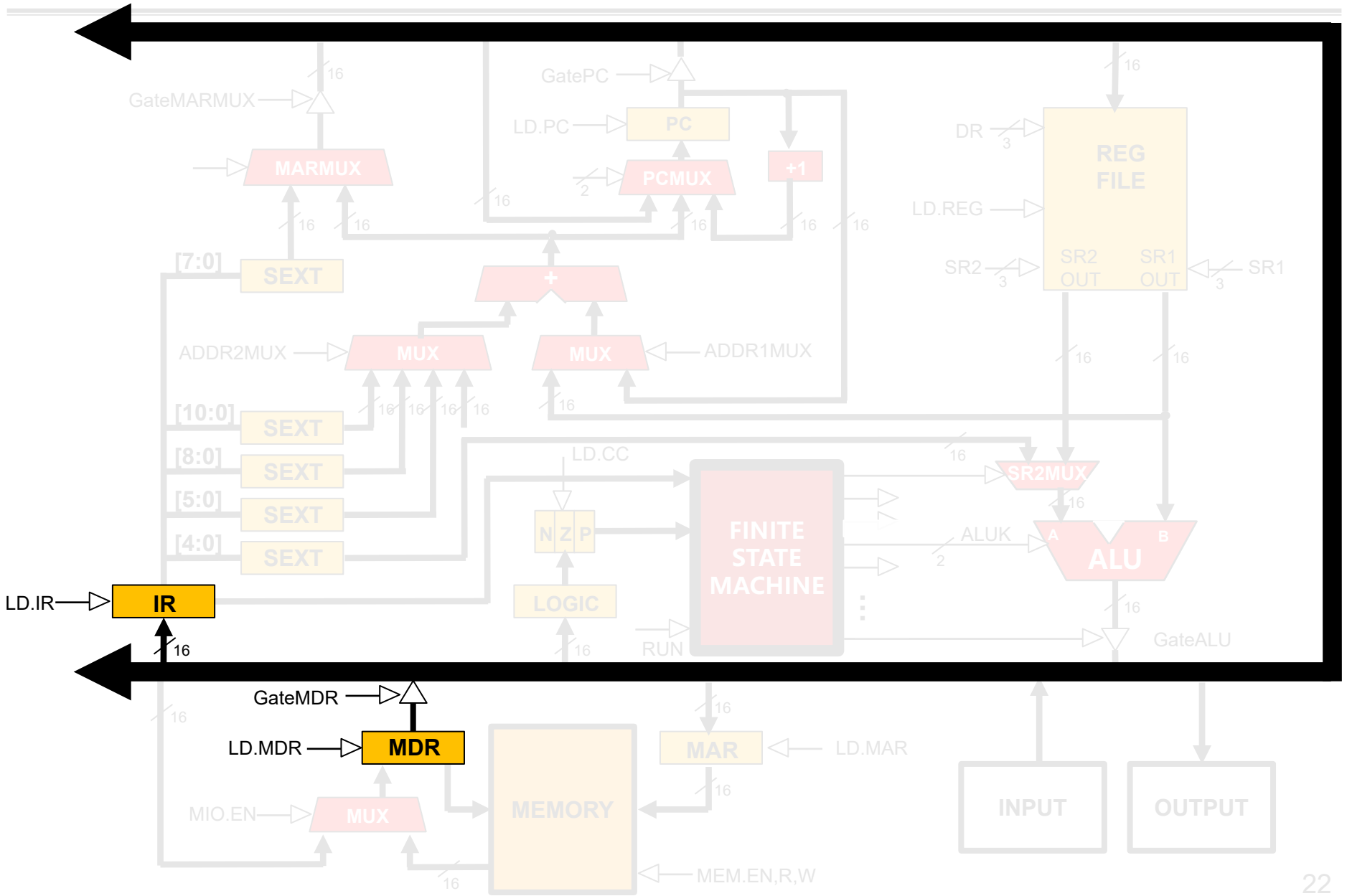
ADD/AND (Immediate)



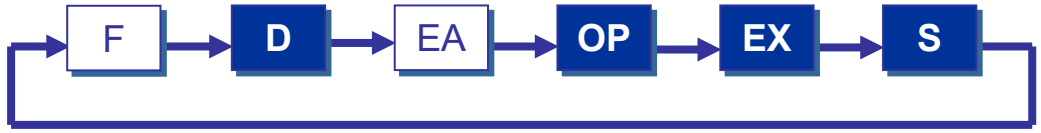


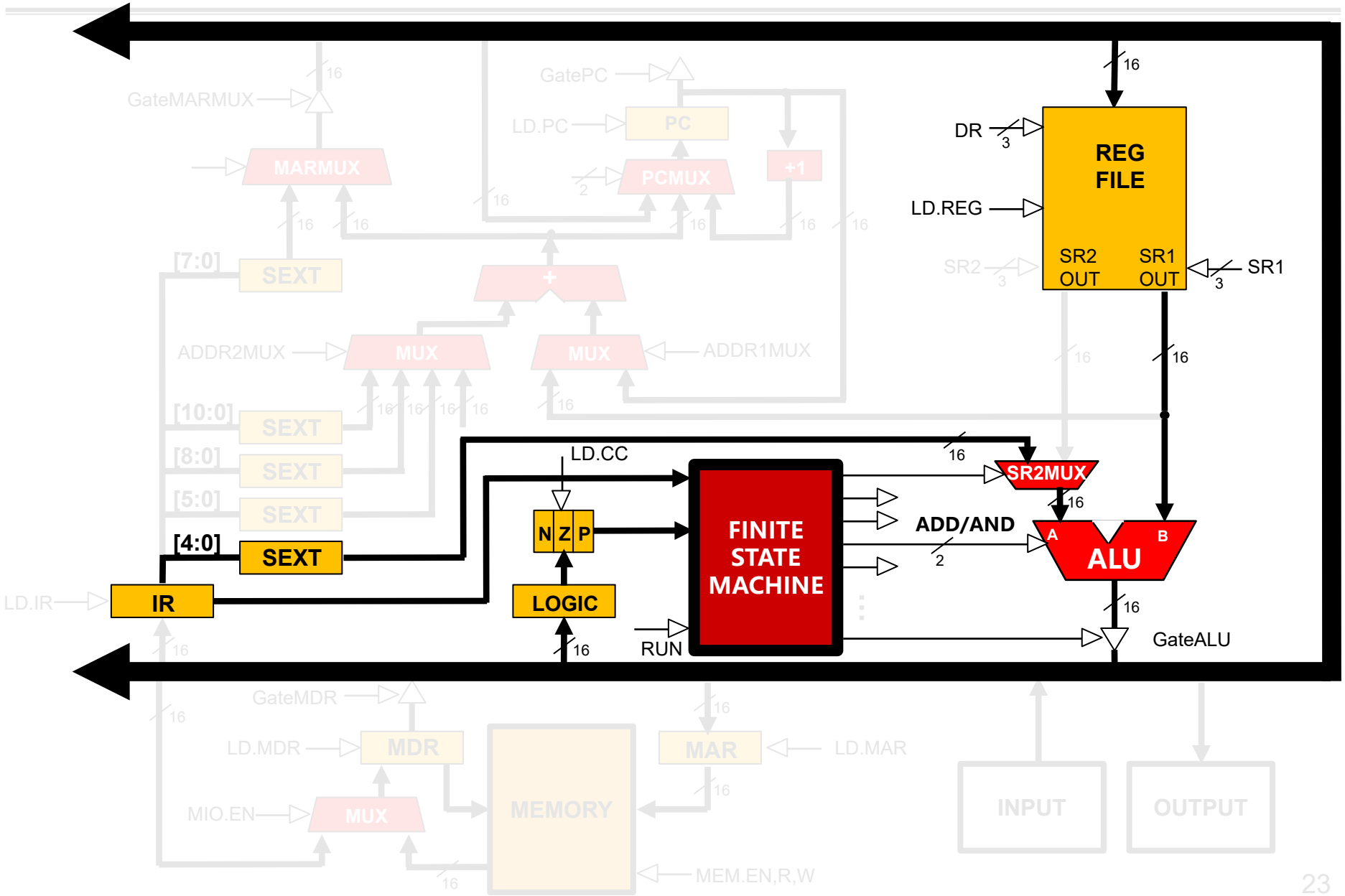
ADD/AND (Immediate)





ADD/AND (Immediate)





Data Movement Instructions

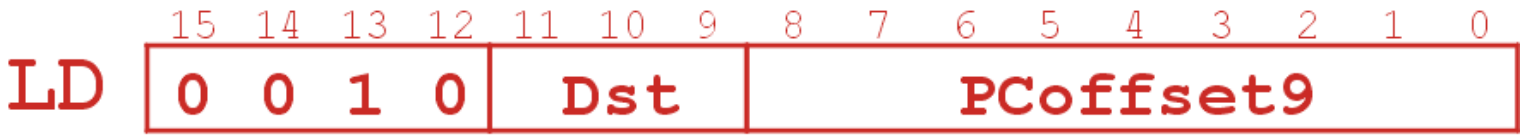
取数指令

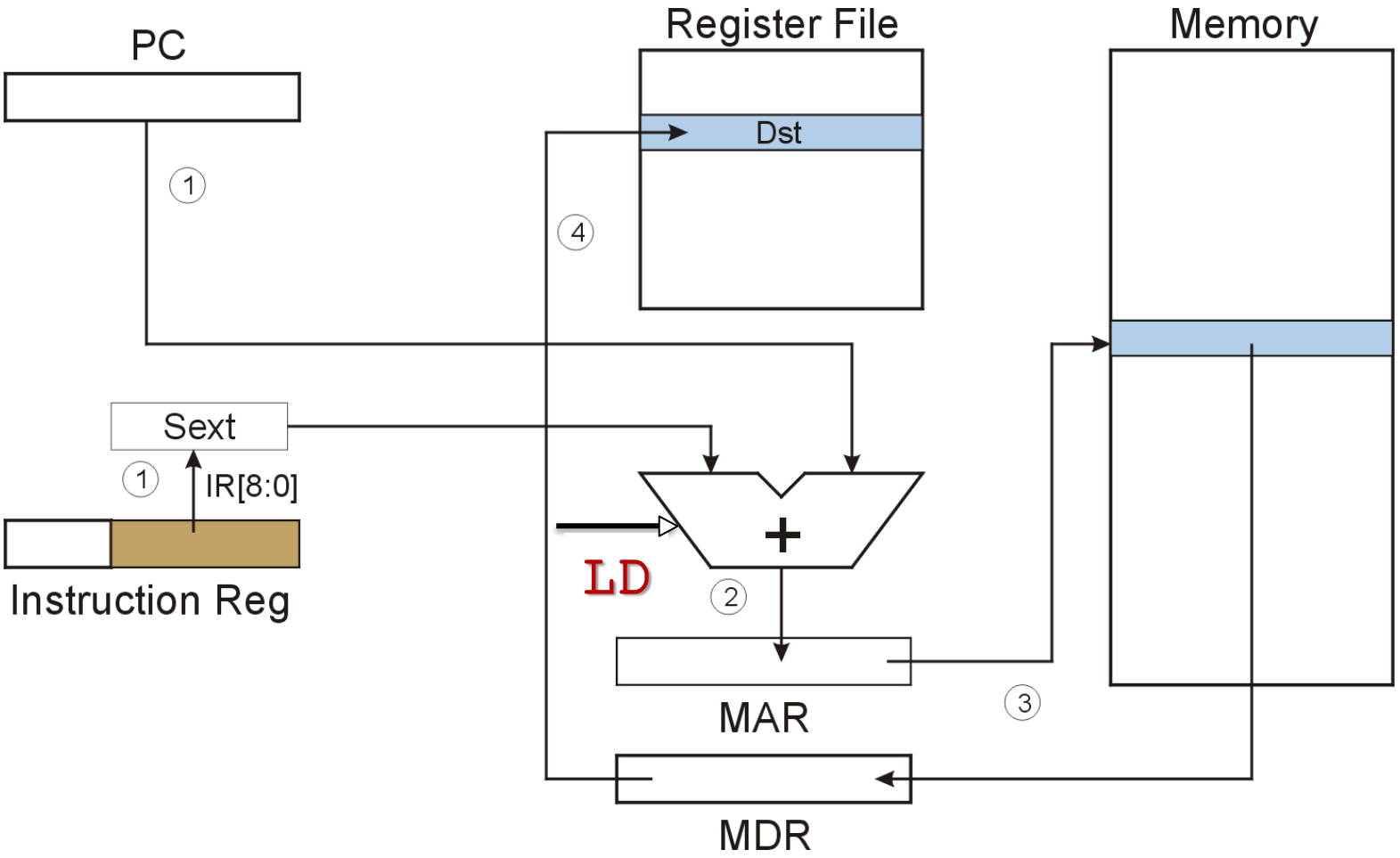
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LD	0	0	1	0	DR			PCoffset9								
LDR	0	1	1	0	DR			BaseR			PCoffset6					
LDI	1	0	1	0	DR			PCoffset9								
LEA	1	1	1	0	DR			PCoffset9								

存数指令

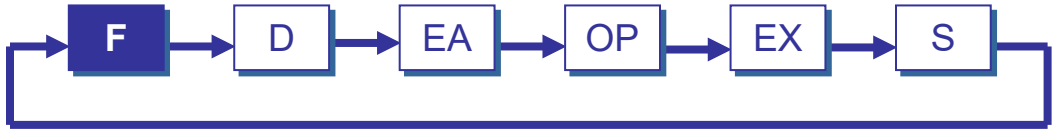
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ST	0	0	1	1	SR			PCoffset9								
STR	0	1	1	1	SR			BaseR			PCoffset6					
STI	1	0	1	1	SR			PCoffset9								

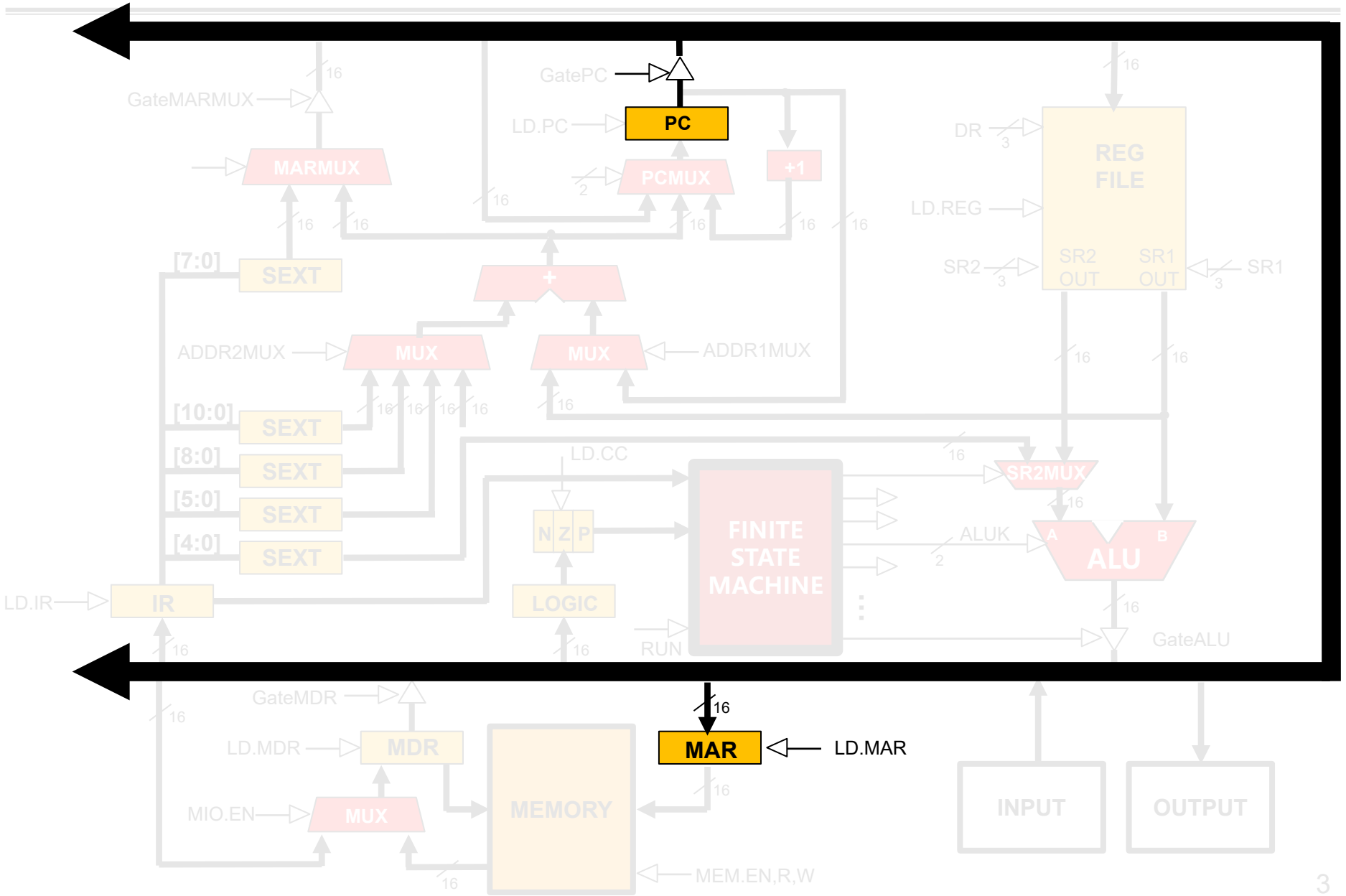
LD (PC-Relative) LD DR, PCoffset9



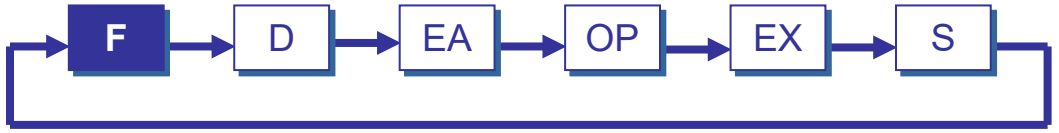


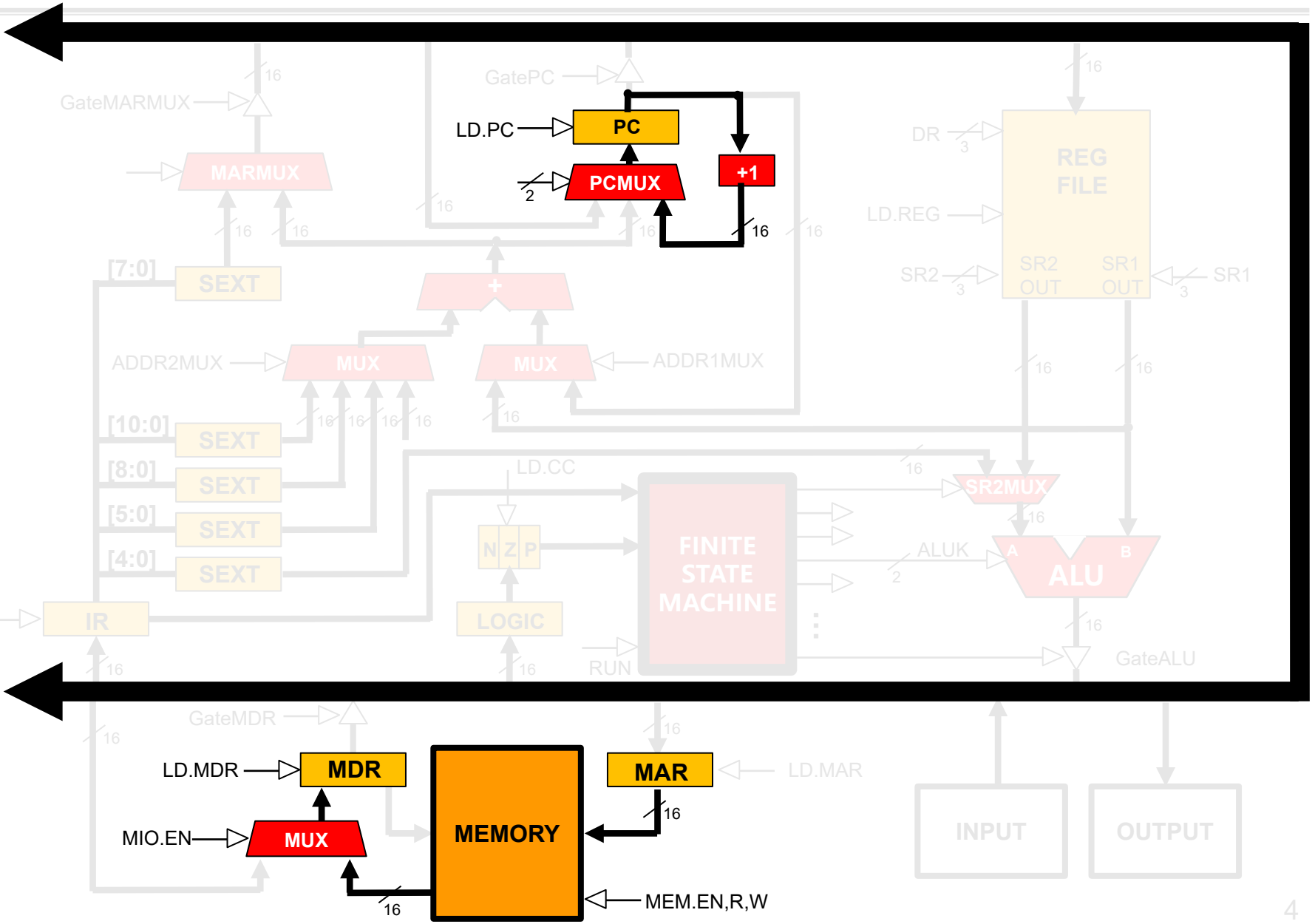
LD (PC-Relative)



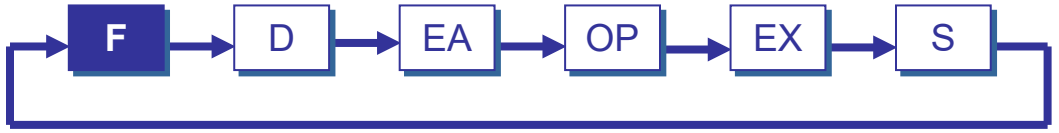


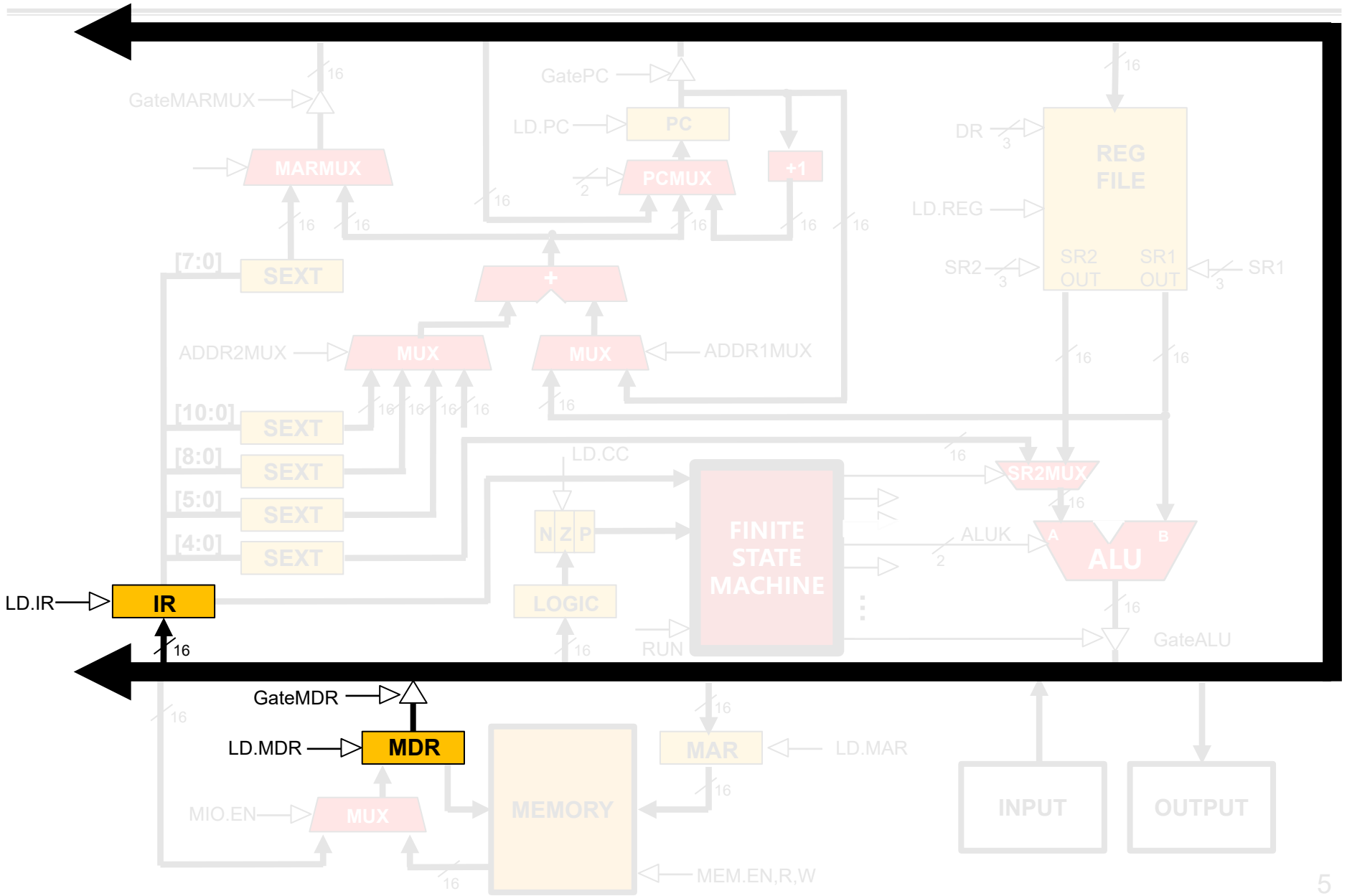
LD (PC-Relative)



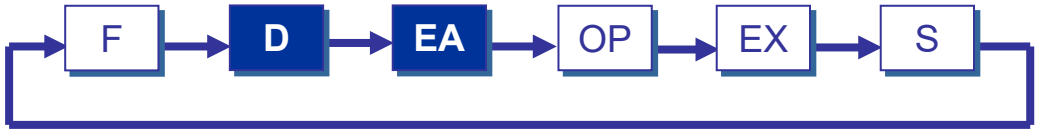


LD (PC-Relative)



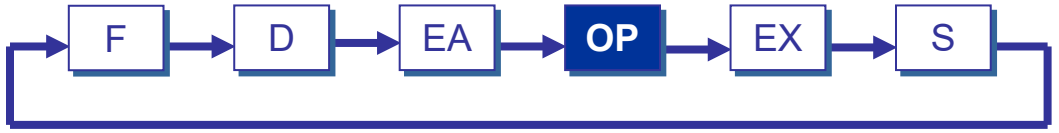


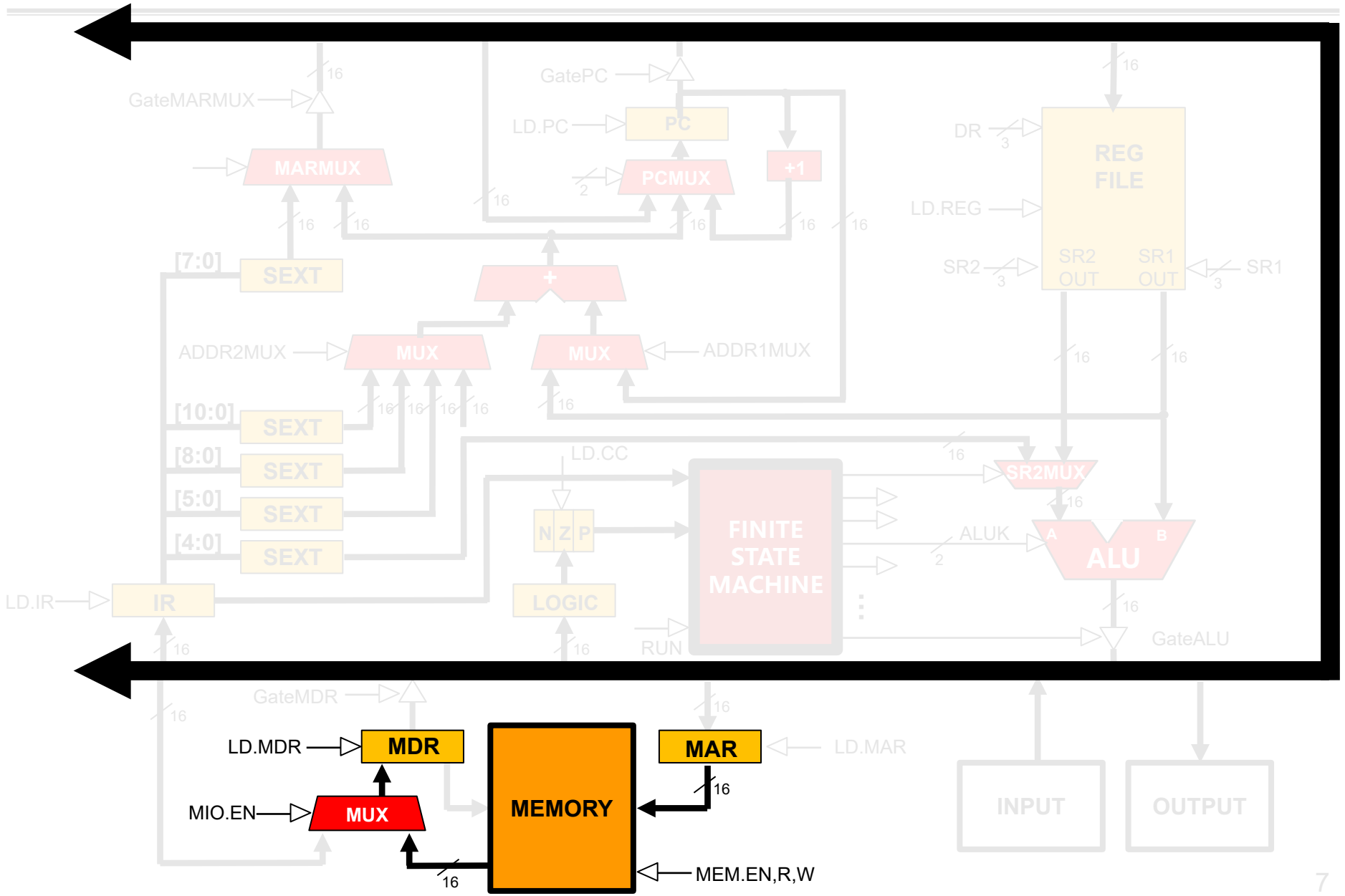
LD (PC-Relative)



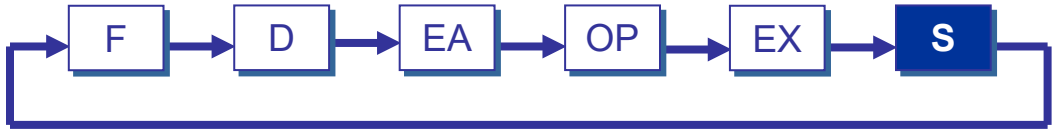
GateMARMUX
MARMUX
16
PC
LD.PC
GatePC
PCMUX
+1
+
ADDR2MUX
MUX
MUX
ADDR1MUX
[7:0]
SEXT
[10:0]
SEXT
[8:0]
SEXT
[5:0]
SEXT
[4:0]
SEXT
IR
LD.IR
LD.CC
N Z P
LOGIC
FINITE STATE MACHINE
RUN
REG FILE
DR
LD.REG
SR2
SR1
SR2 OUT
SR1 OUT
SR2MUX
ALUK
ALU
GateALU
MAR
LD.MAR
GateMDR
MDR
LD.MDR
MIO.EN
MUX
MEMORY
MEM.EN,R,W
INPUT
OUTPUT

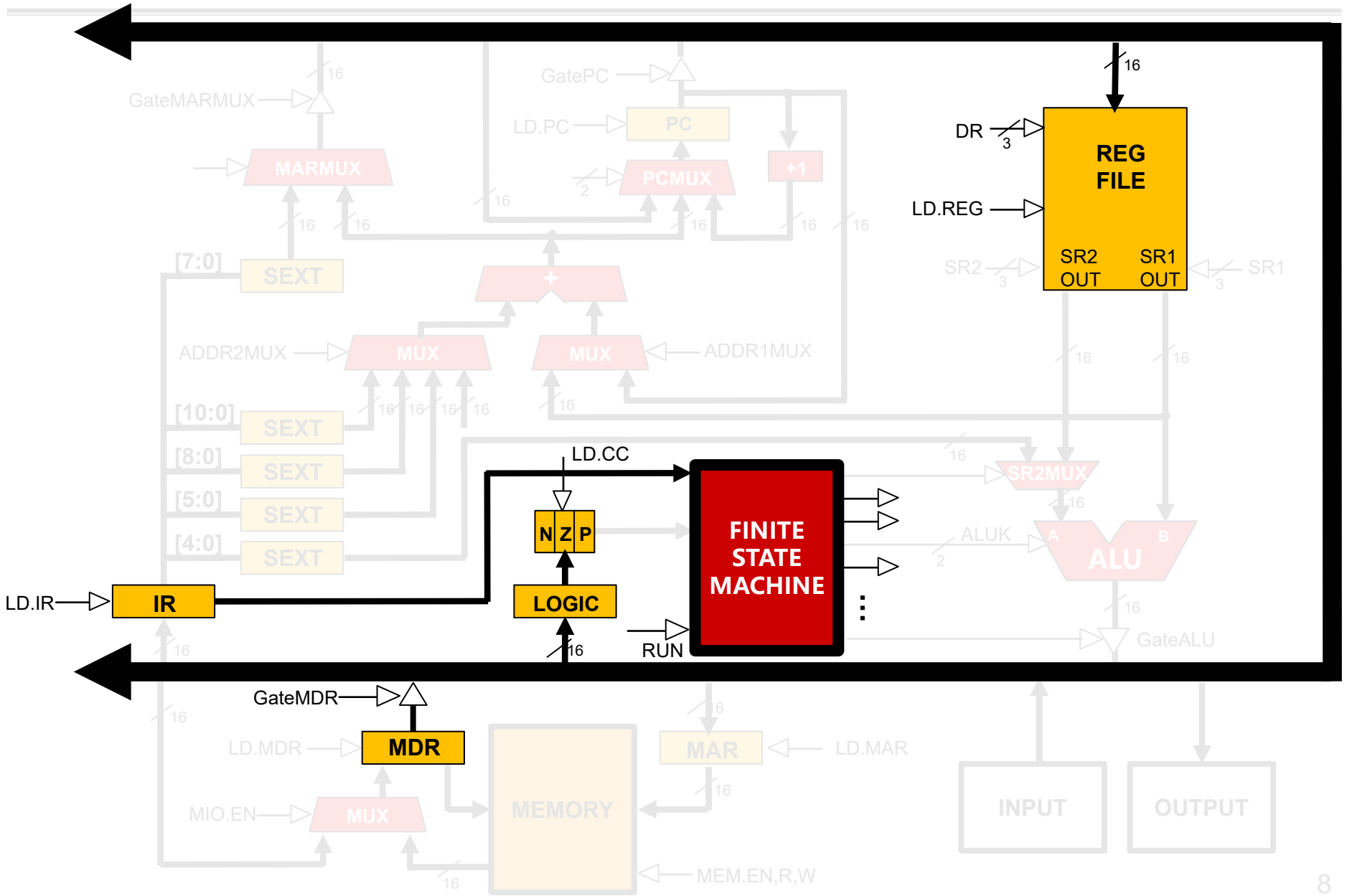
LD (PC-Relative)



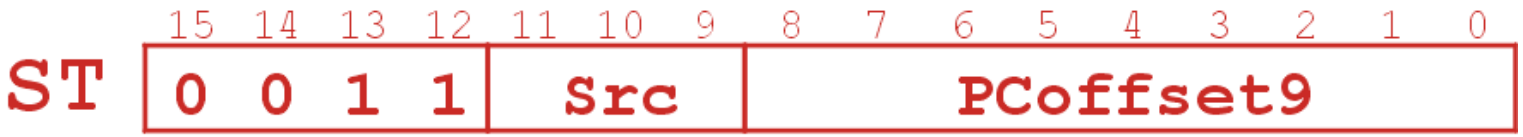


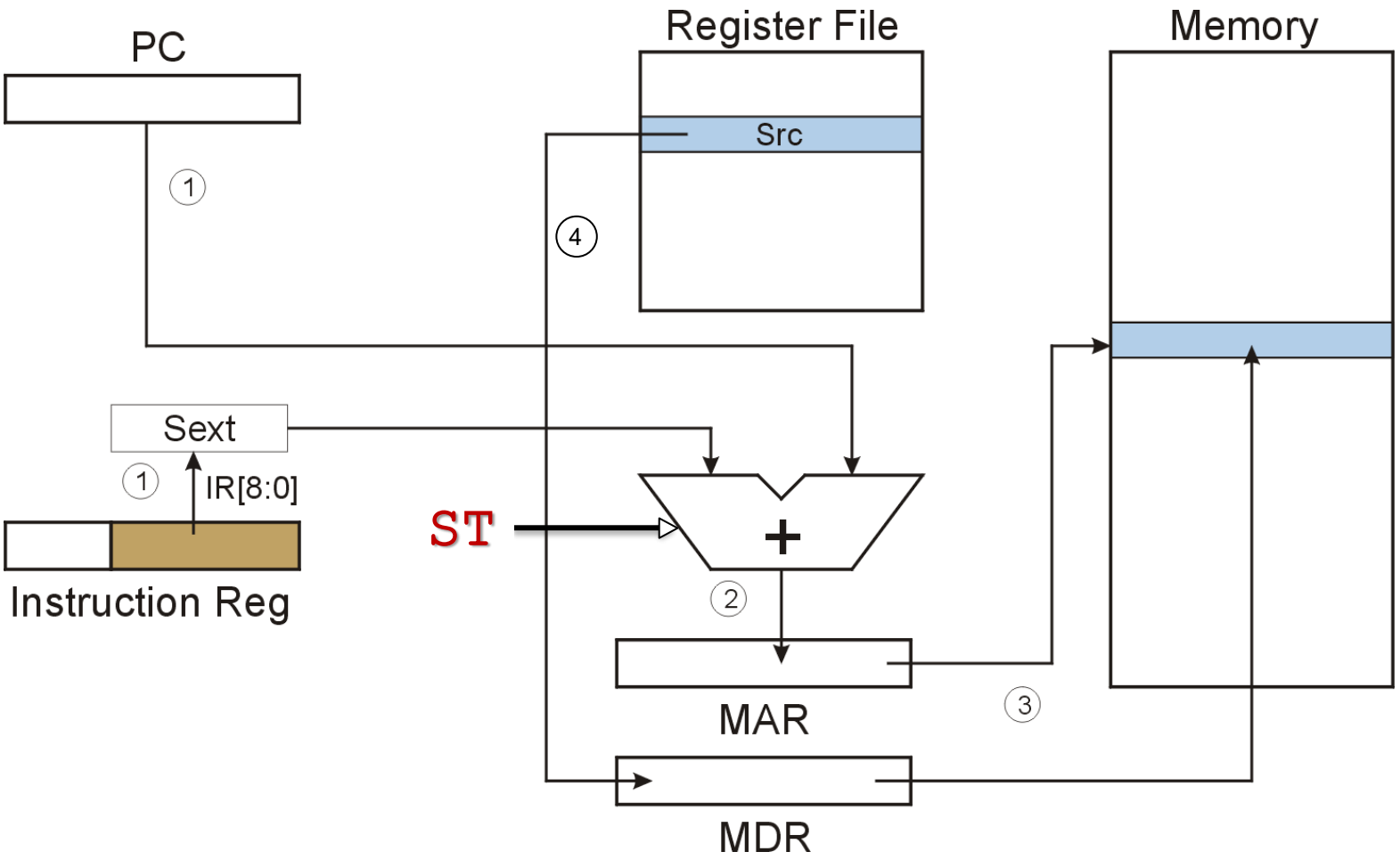
LD (PC-Relative)



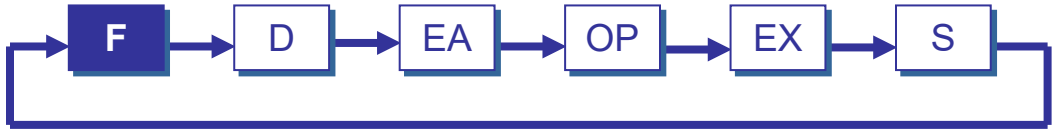


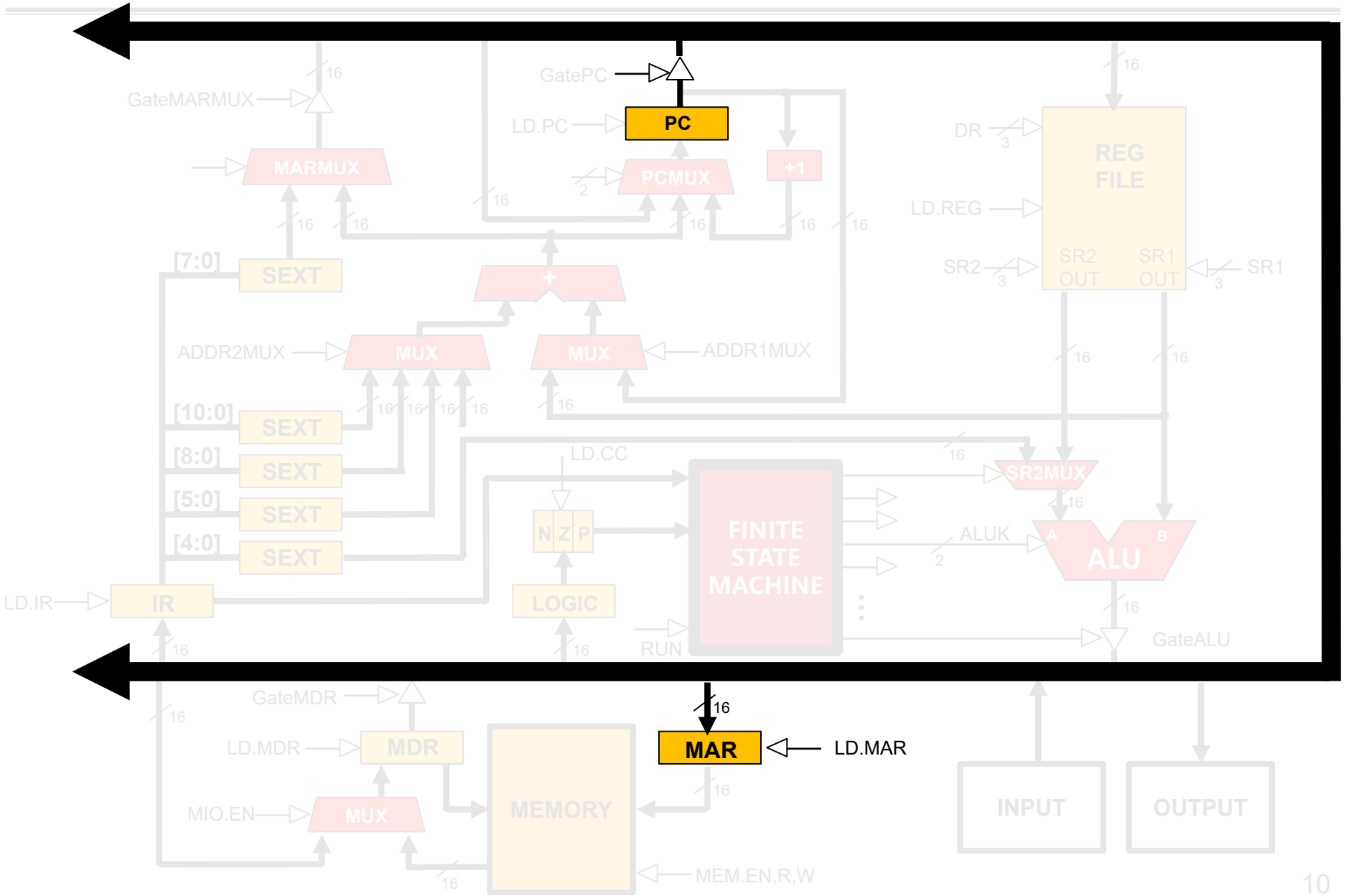
ST (PC-Relative) ST SR, PCoffset9



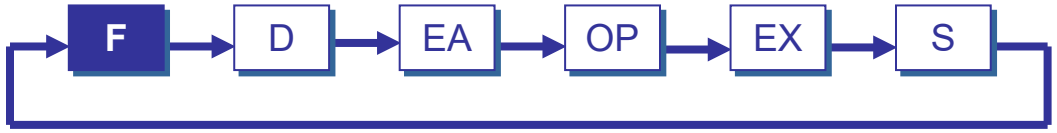


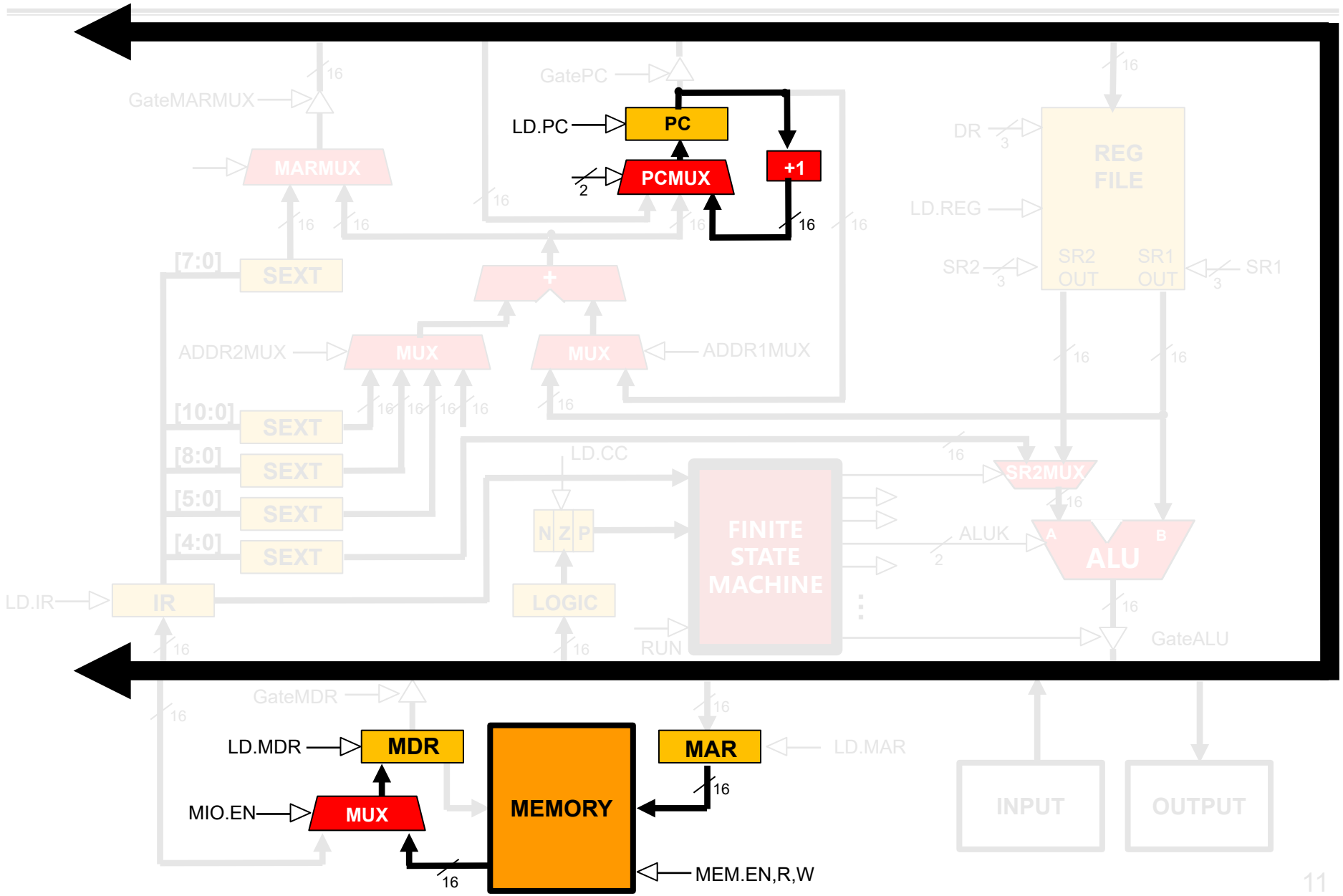
ST (PC-Relative)



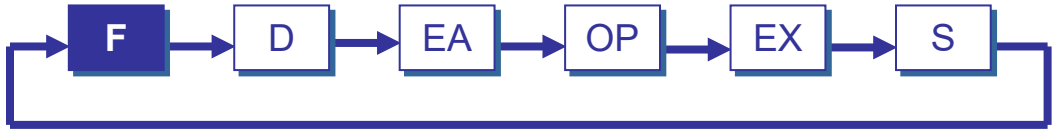


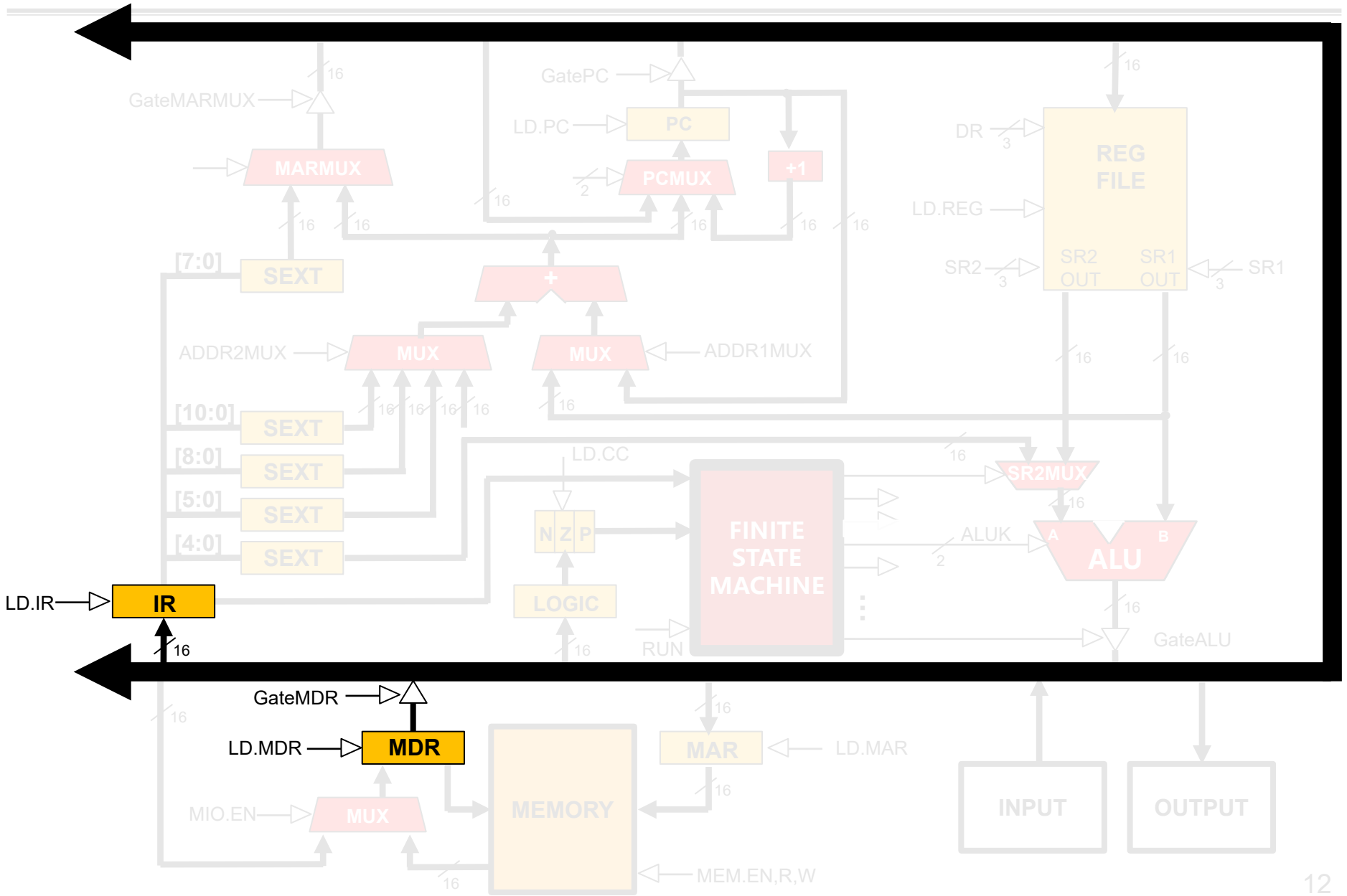
ST (PC-Relative)



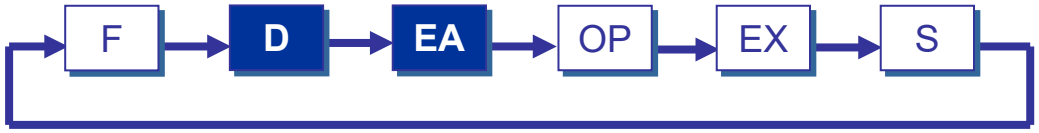


ST (PC-Relative)



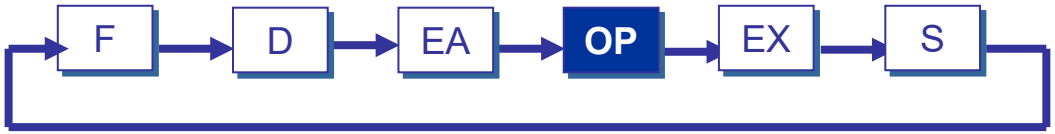


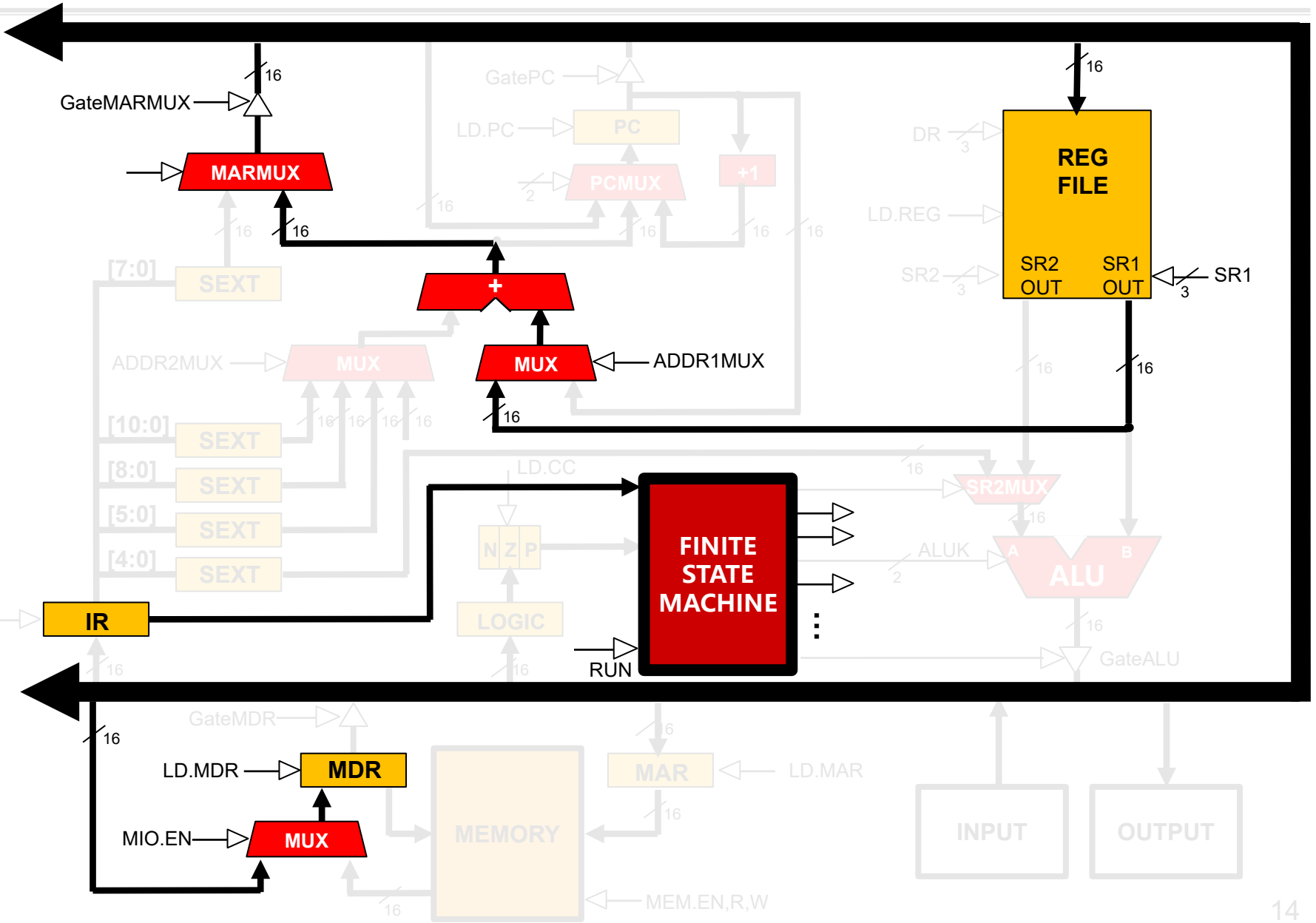
ST (PC-Relative)



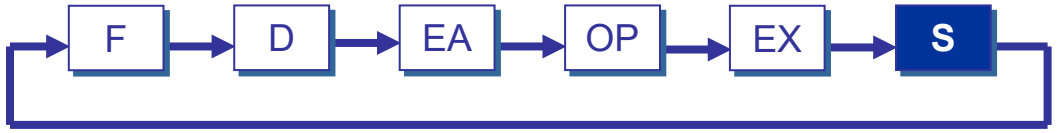
GateMARMUX
MARMUX
16
PC
LD.PC
GatePC
PCMUX
+1
+
ADDR2MUX
MUX
MUX
ADDR1MUX
[7:0]
SEXT
[10:0]
SEXT
[8:0]
SEXT
[5:0]
SEXT
[4:0]
SEXT
LD.IR
IR
LD.CC
N Z P
LOGIC
FINITE STATE MACHINE
RUN
DR
REG FILE
LD.REG
SR2
SR2 OUT
SR1 OUT
SR1
SR2MUX
ALUK
A
B
ALU
GateALU
GateMDR
LD.MDR
MDR
MIO.EN
MUX
MEMORY
MEM.EN,R,W
MAR
LD.MAR
INPUT
OUTPUT

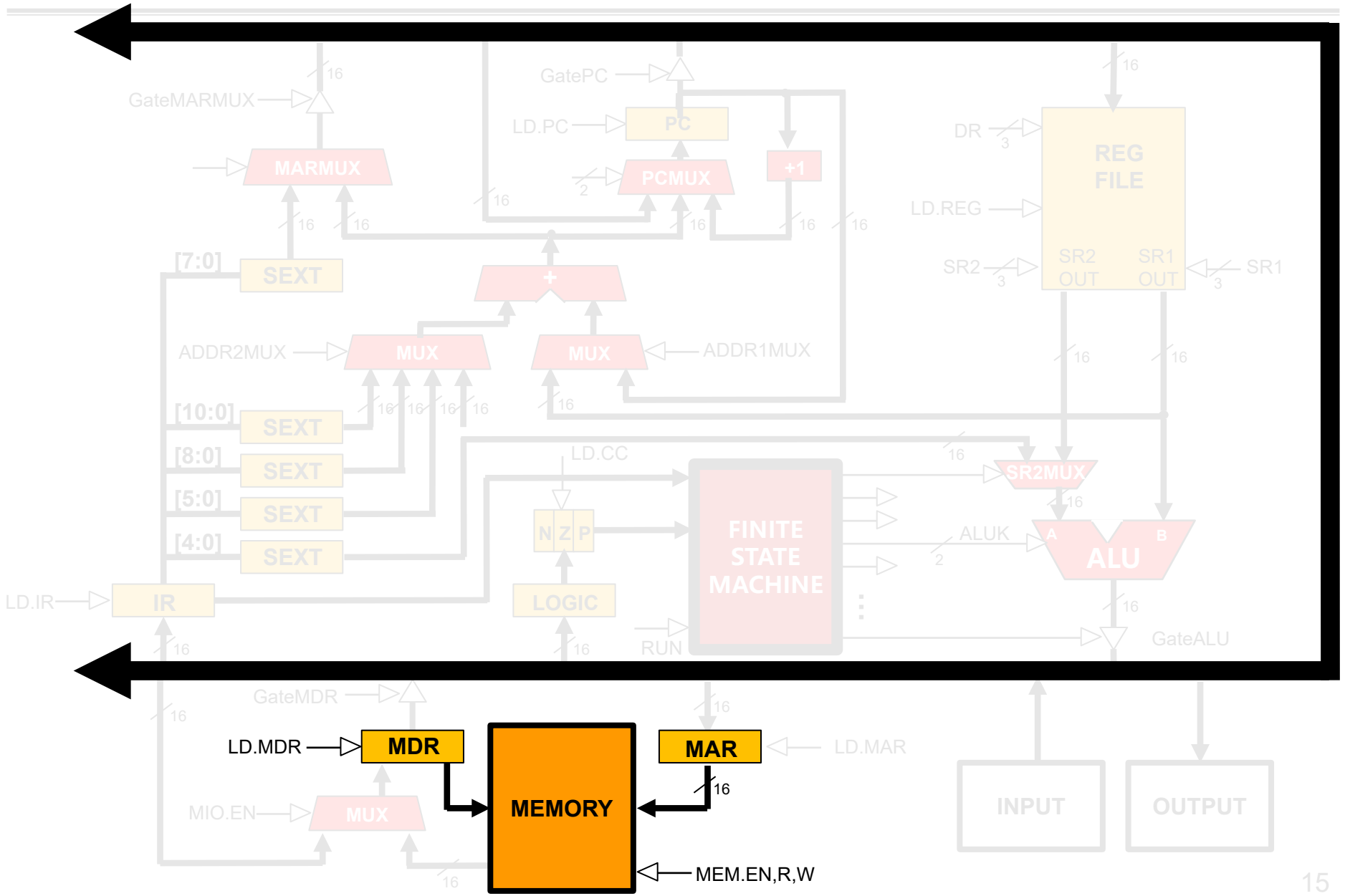
ST (PC-Relative)



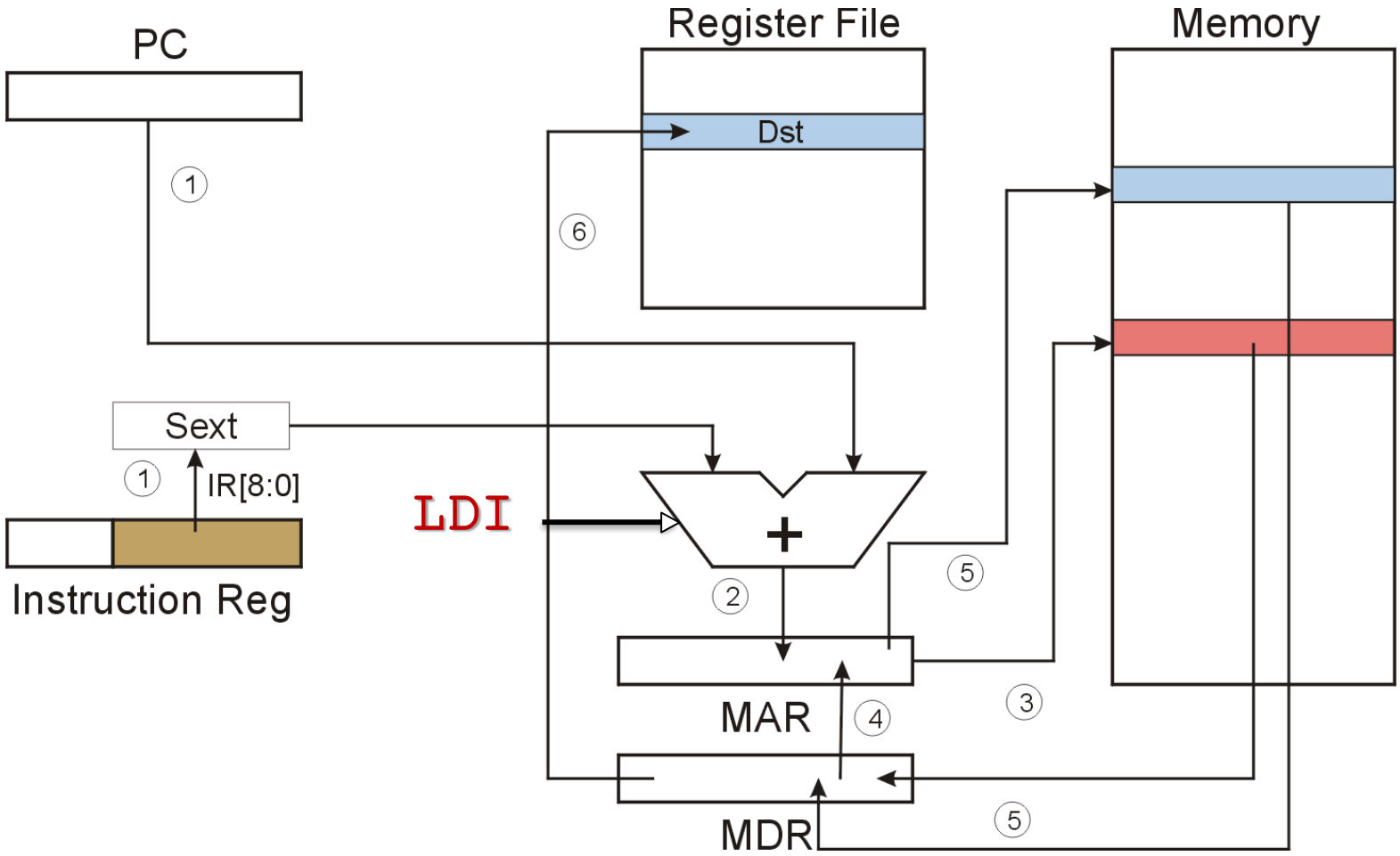


ST (PC-Relative)

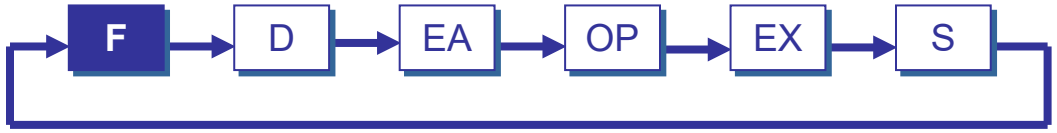


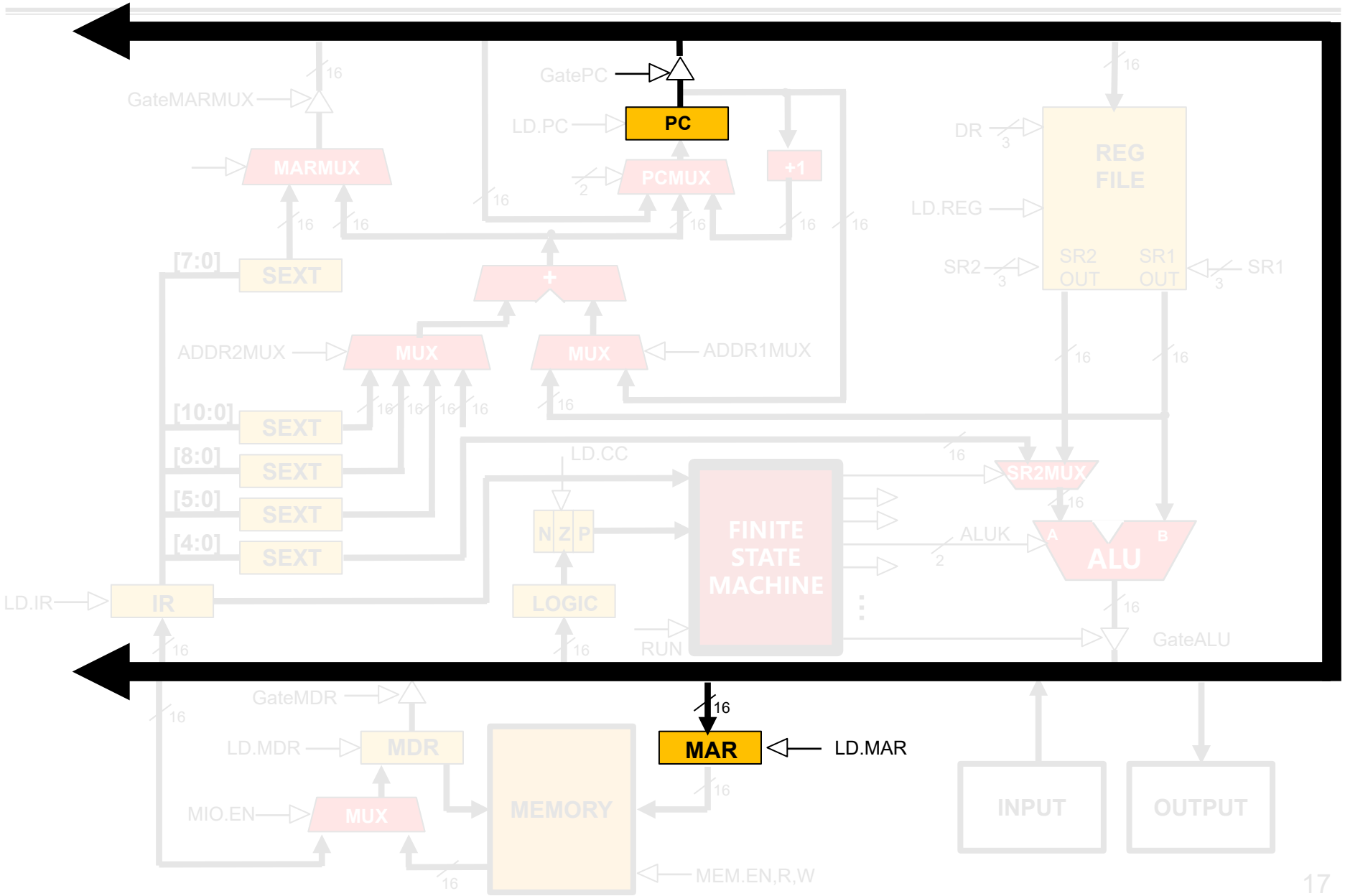


LDI (Indirect) LDI DR, PCoffset9

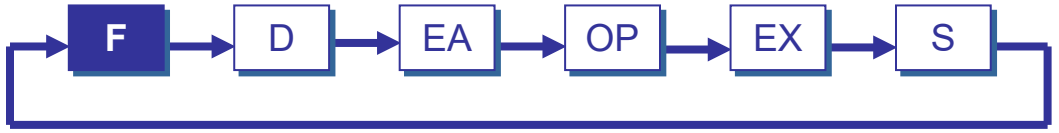


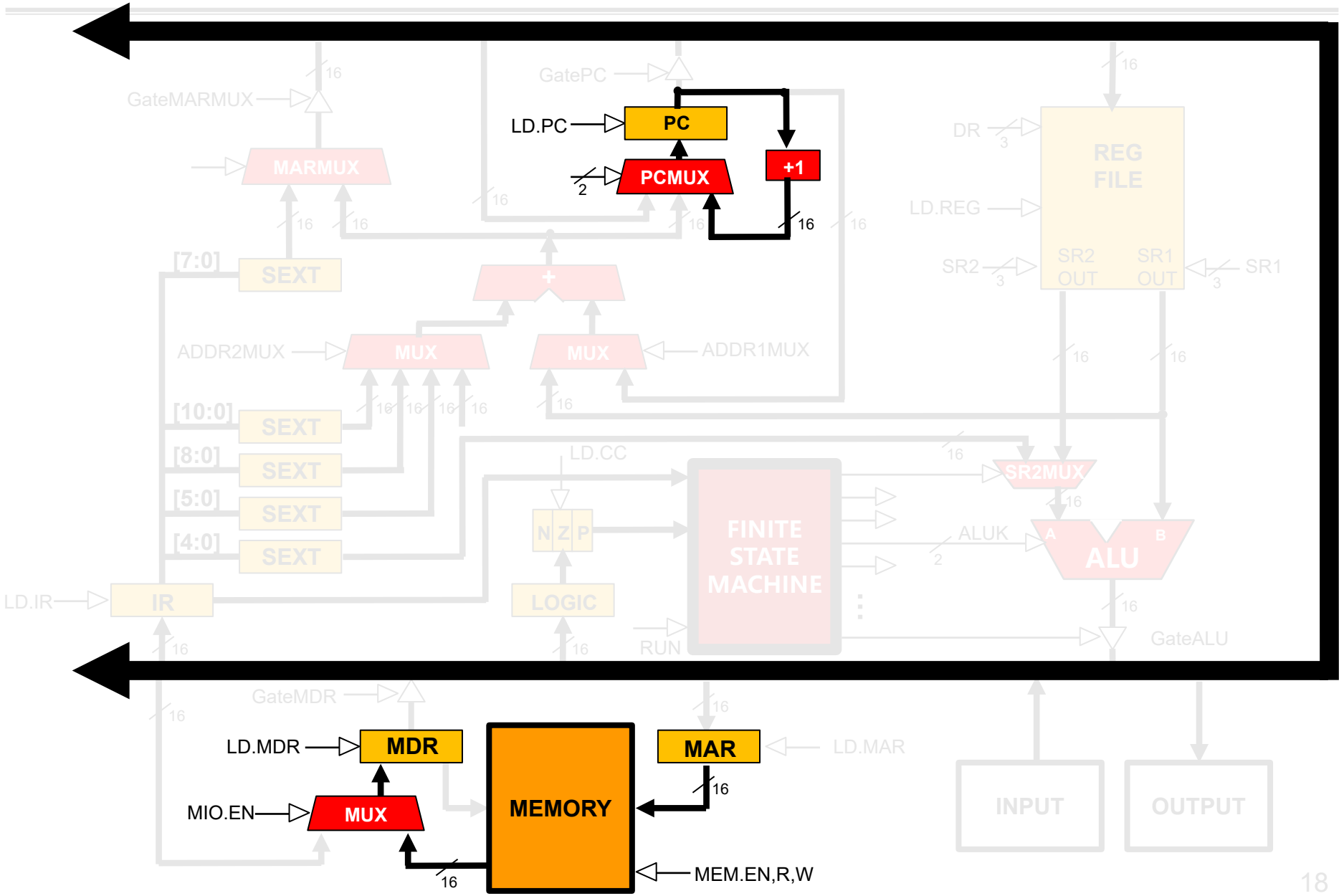
LDI (Indirect)



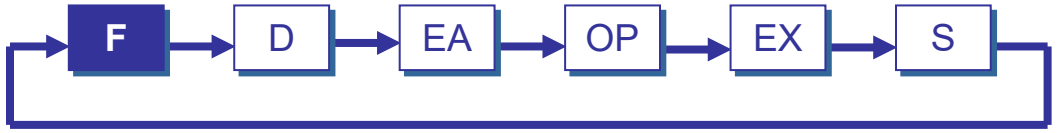


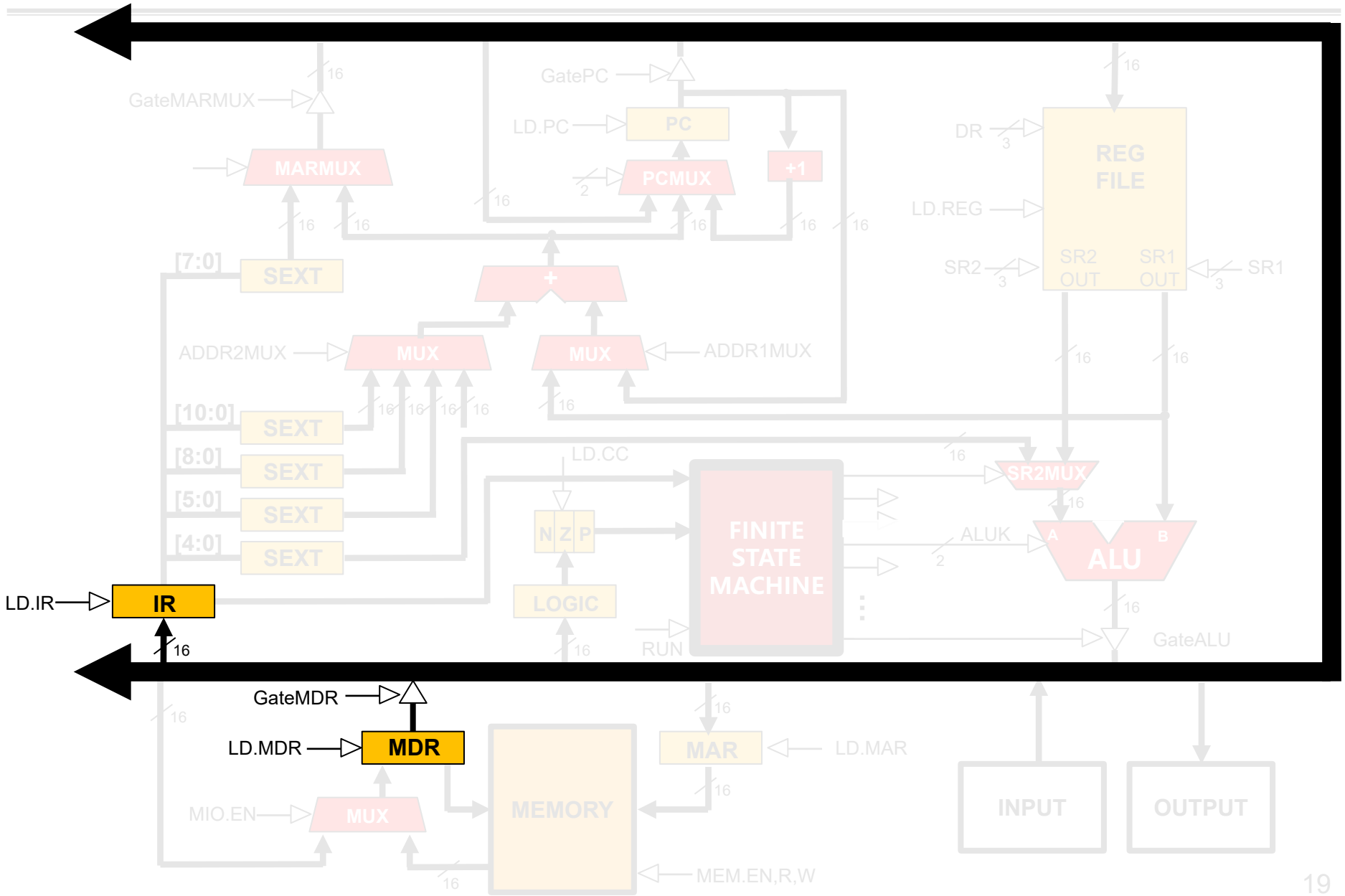
LDI (Indirect)



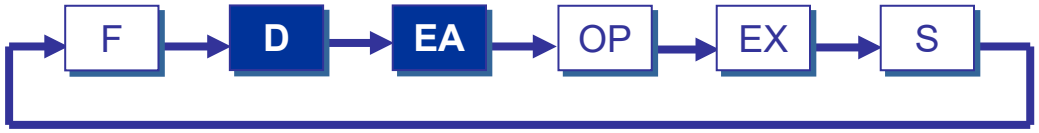


LDI (Indirect)



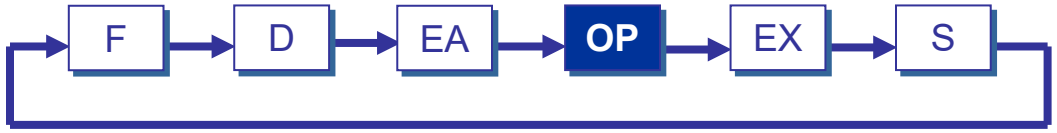


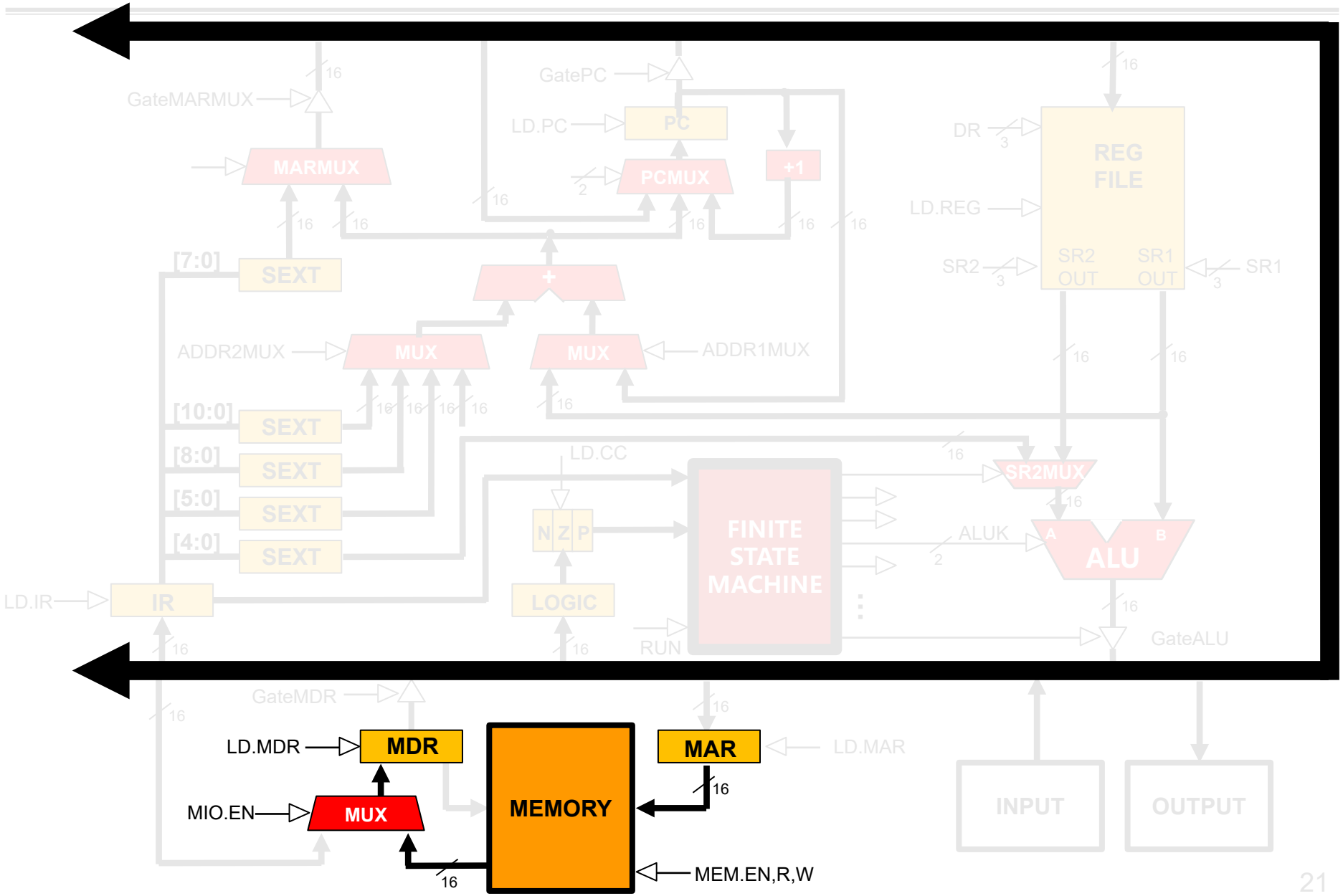
LDI (Indirect)



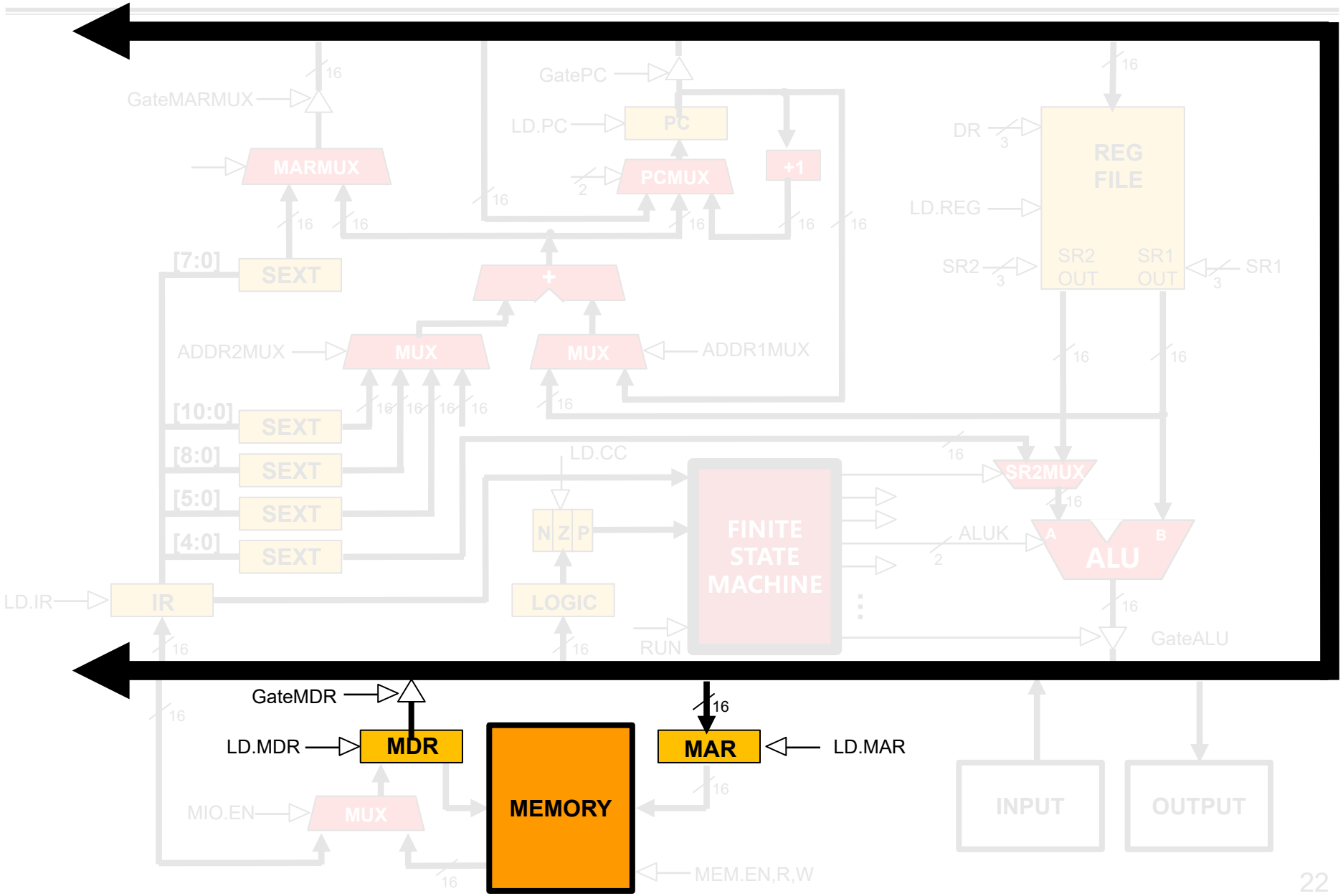
GateMARMUX
MARMUX
16
GatePC
LD.PC
PC
PCMUX
+1
+
ADDR2MUX
MUX
MUX
ADDR1MUX
[7:0]
SEXT
[10:0]
SEXT
[8:0]
SEXT
[5:0]
SEXT
[4:0]
SEXT
LD.CC
N Z P
LOGIC
FINITE STATE MACHINE
RUN
LD.IR
IR
DR
REG FILE
LD.REG
SR2
SR2 OUT
SR1 OUT
SR1
SR2MUX
ALUK
A
B
ALU
GateALU
GateMDR
LD.MDR
MDR
MIO.EN
MUX
MEMORY
MEM.EN,R,W
MAR
LD.MAR
INPUT
OUTPUT

LDI (Indirect)

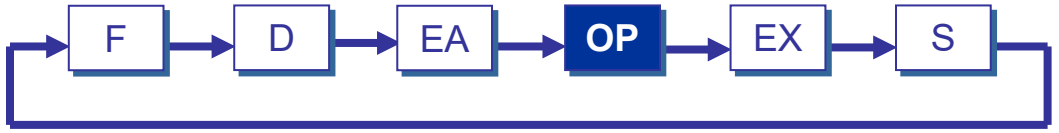


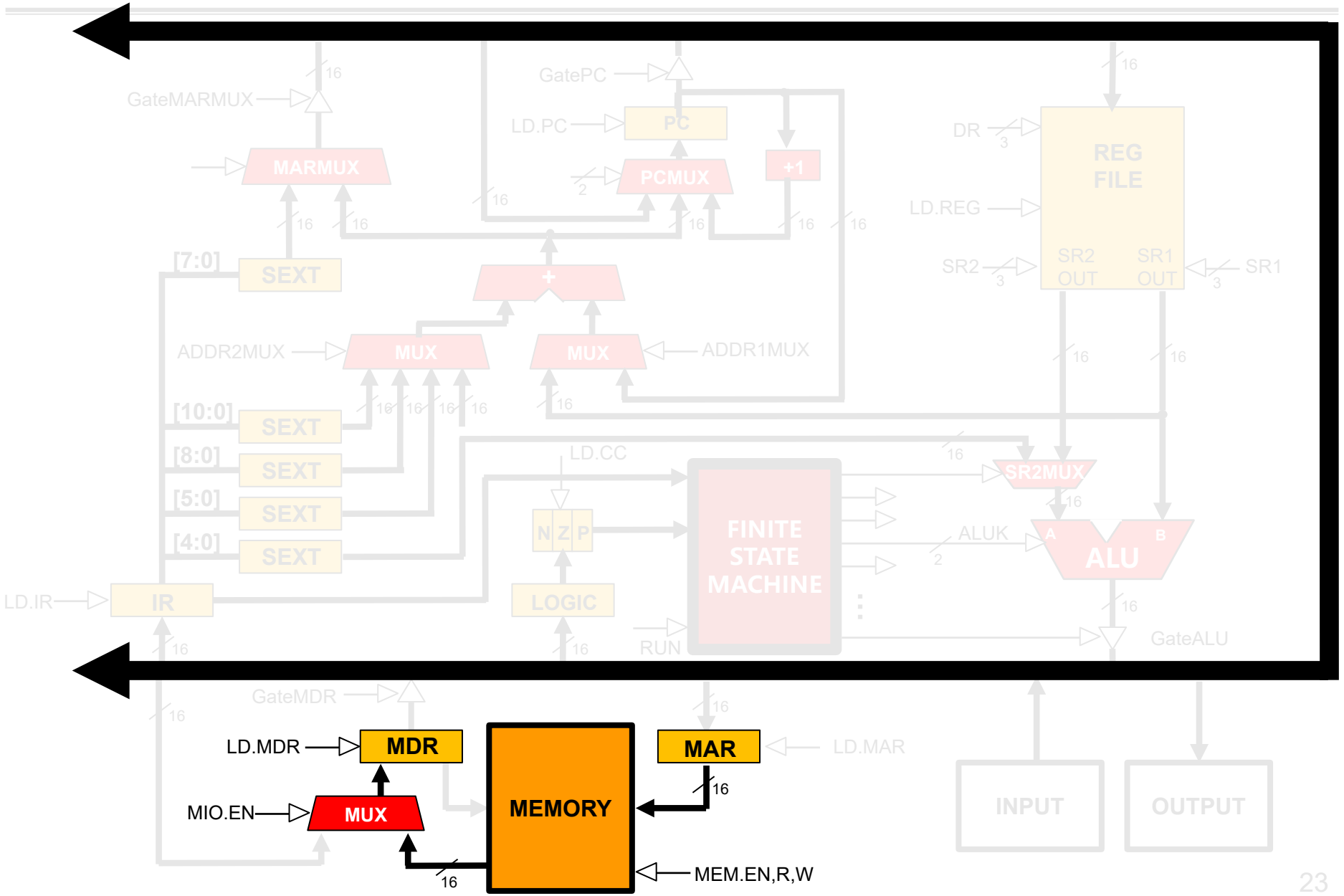


LDI (Indirect)

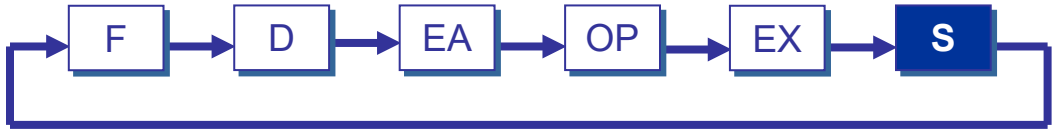


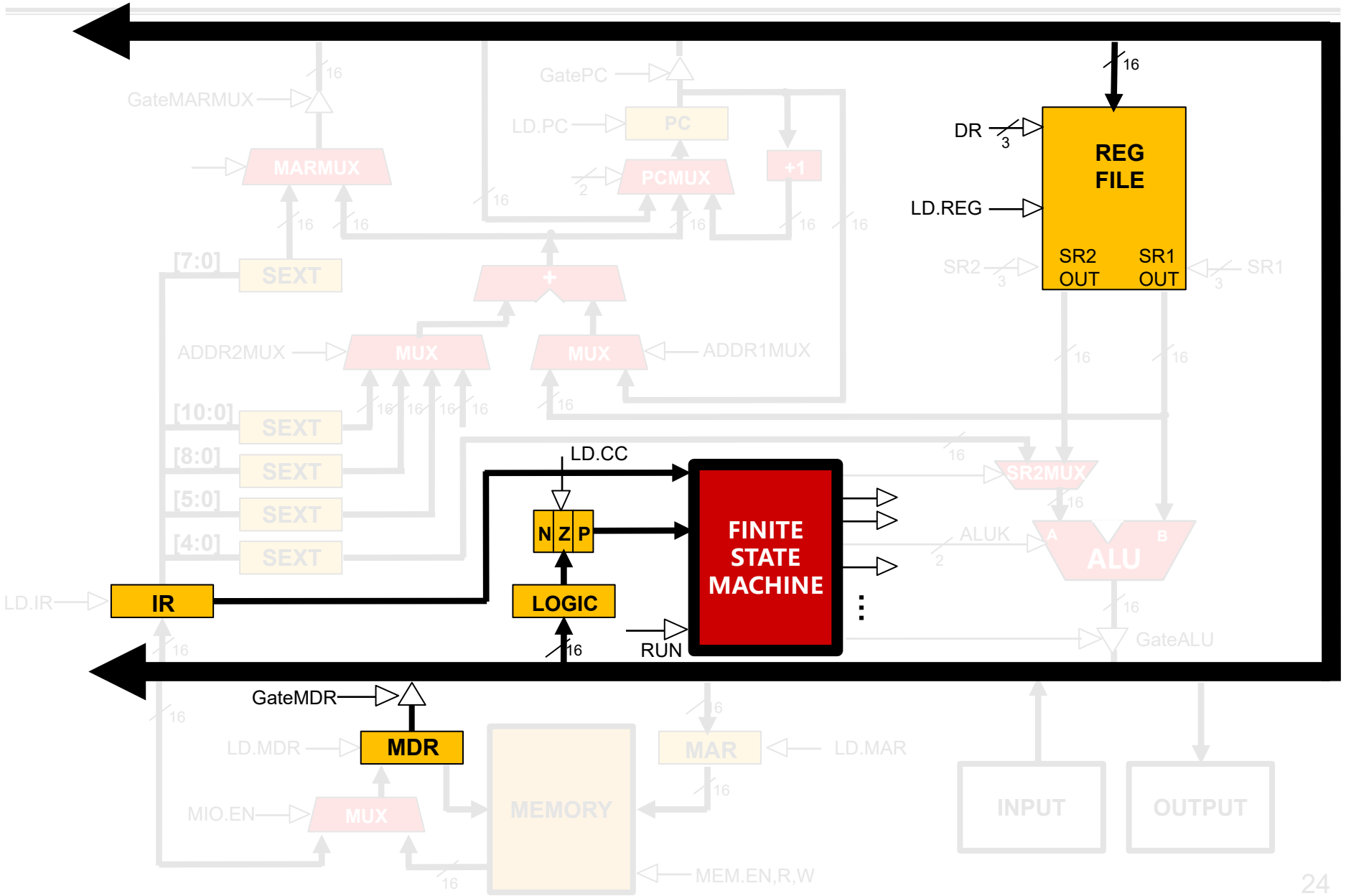
LDI (Indirect)



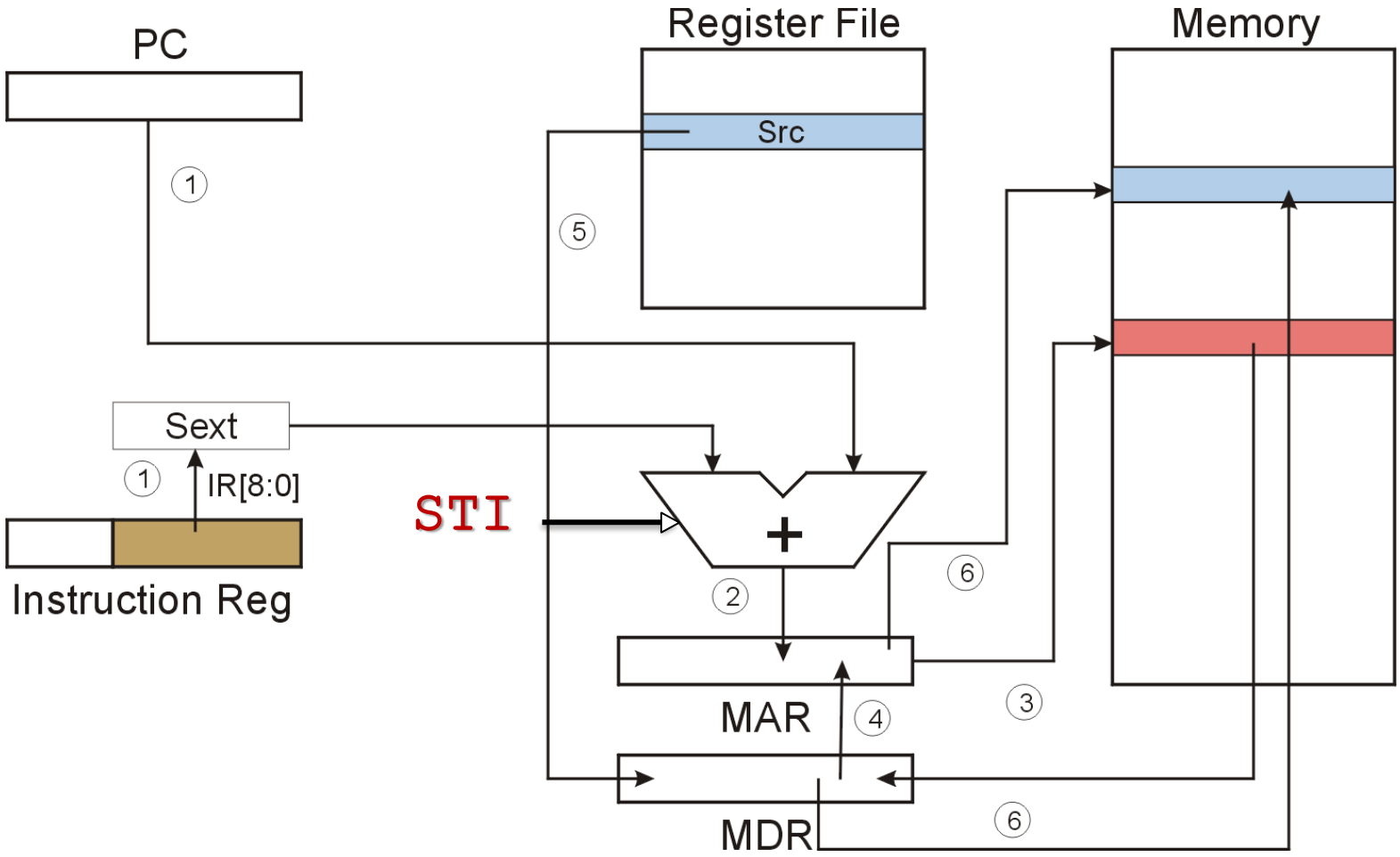


LDI (Indirect)

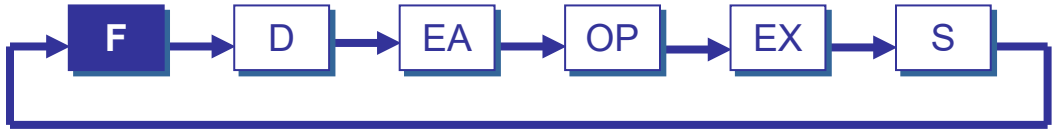


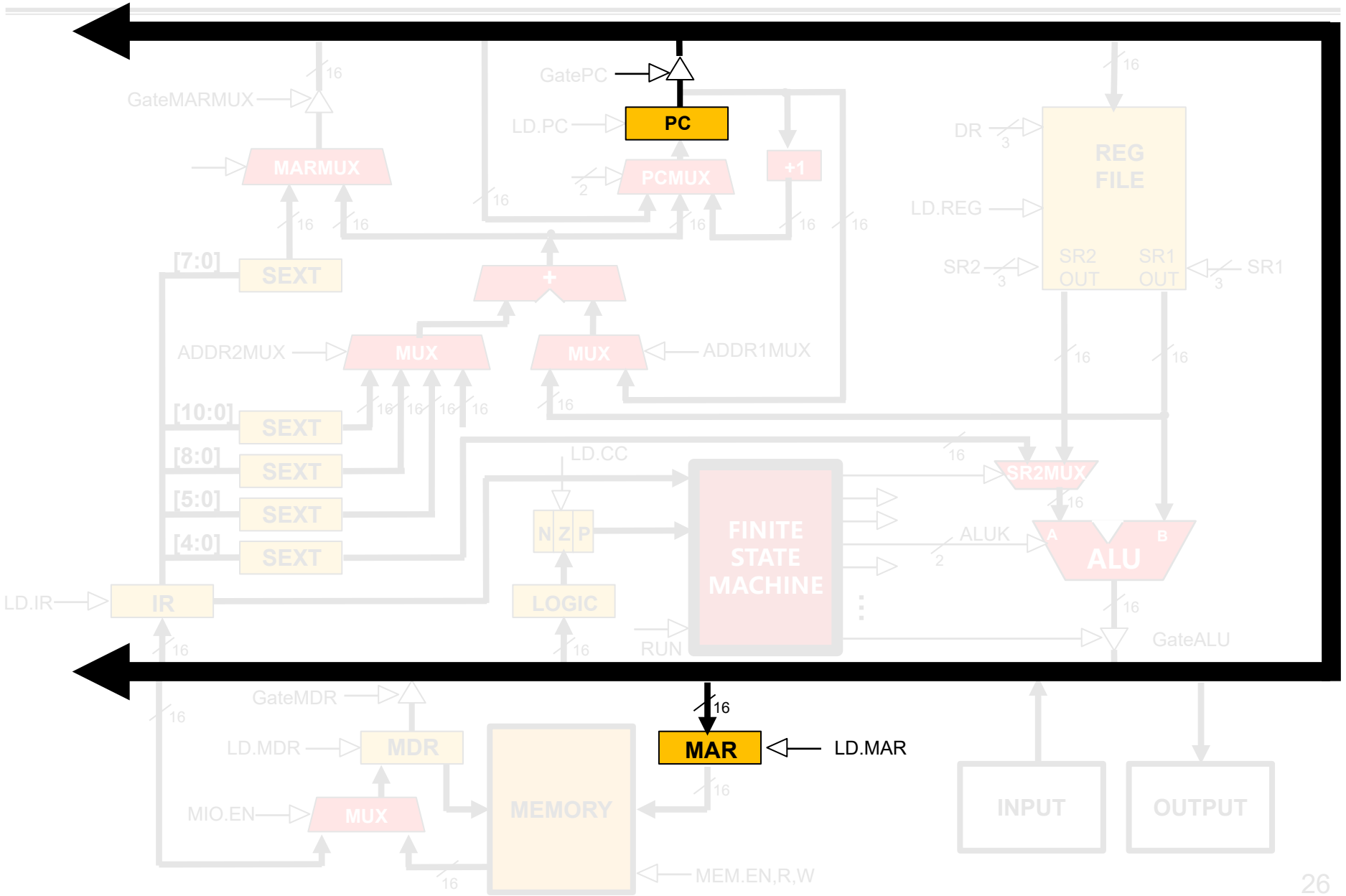


STI (Indirect) STI SR, PCoffset9

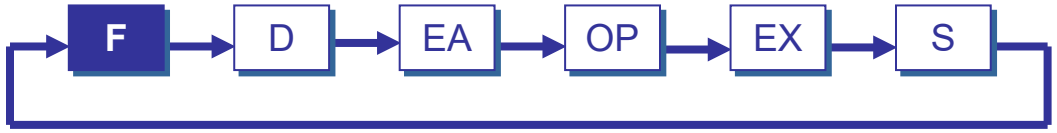


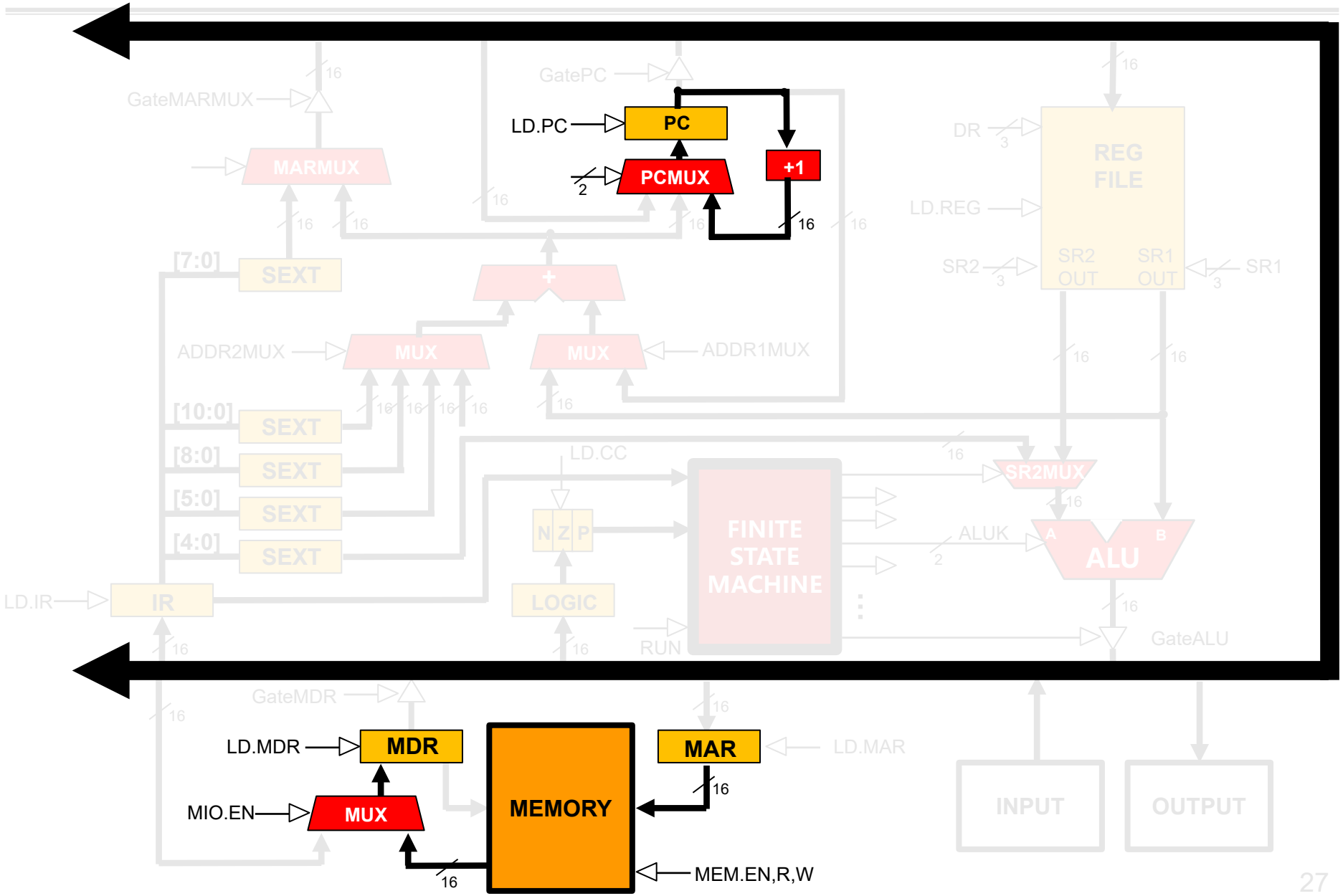
STI (Indirect)



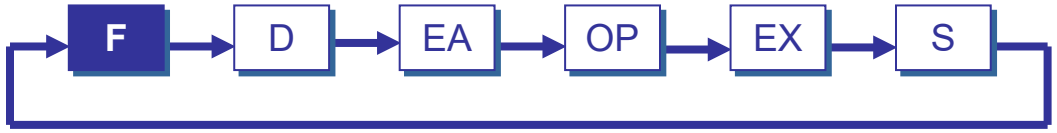


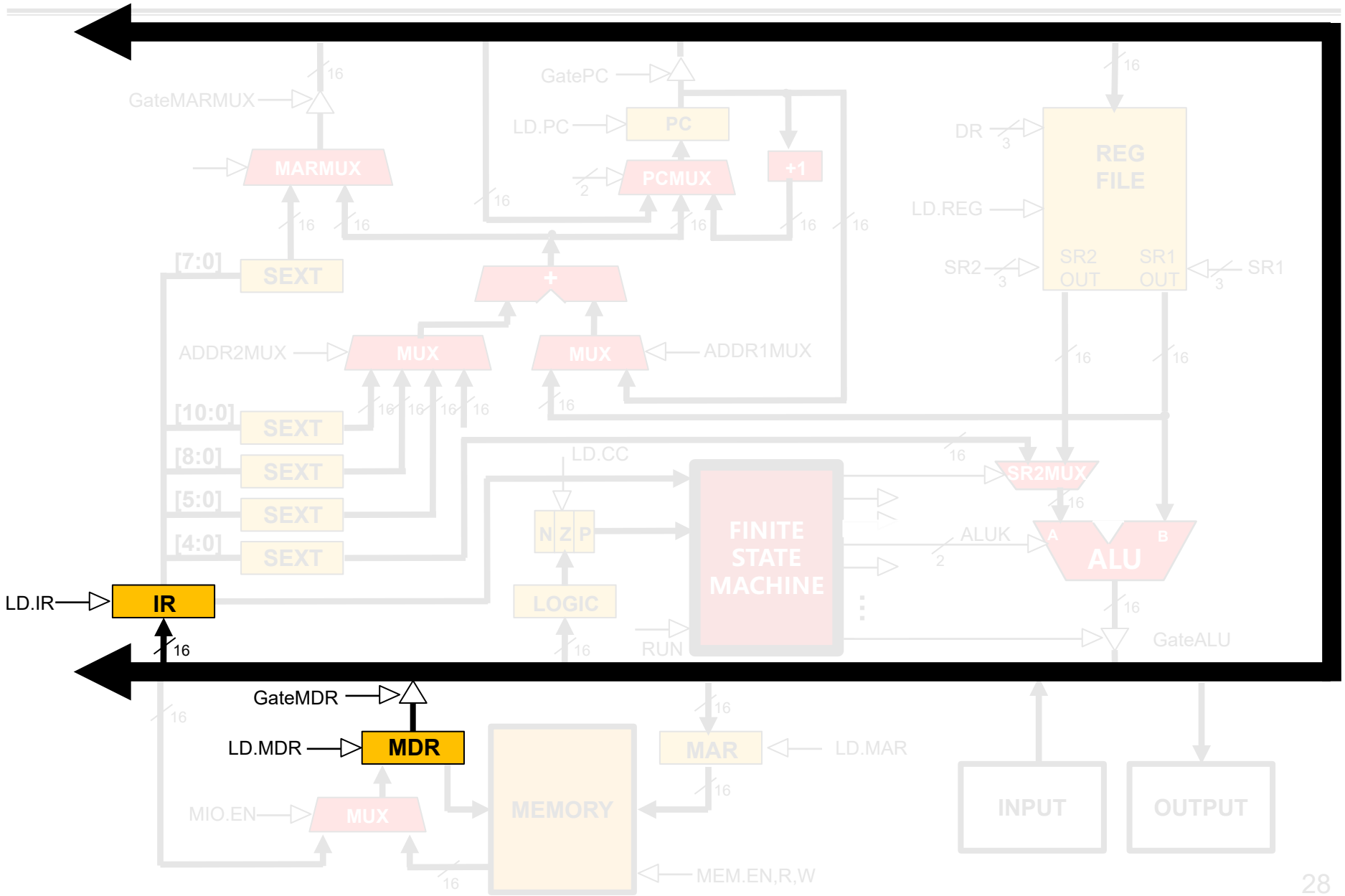
STI (Indirect)



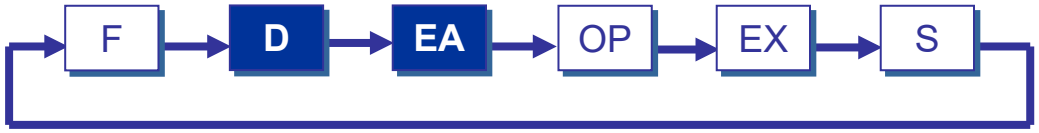


STI (Indirect)



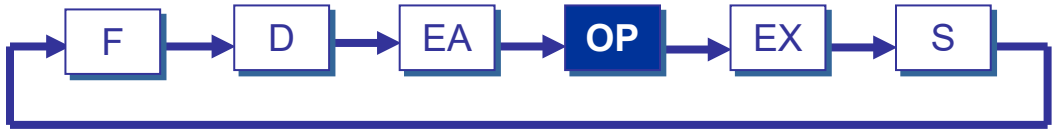


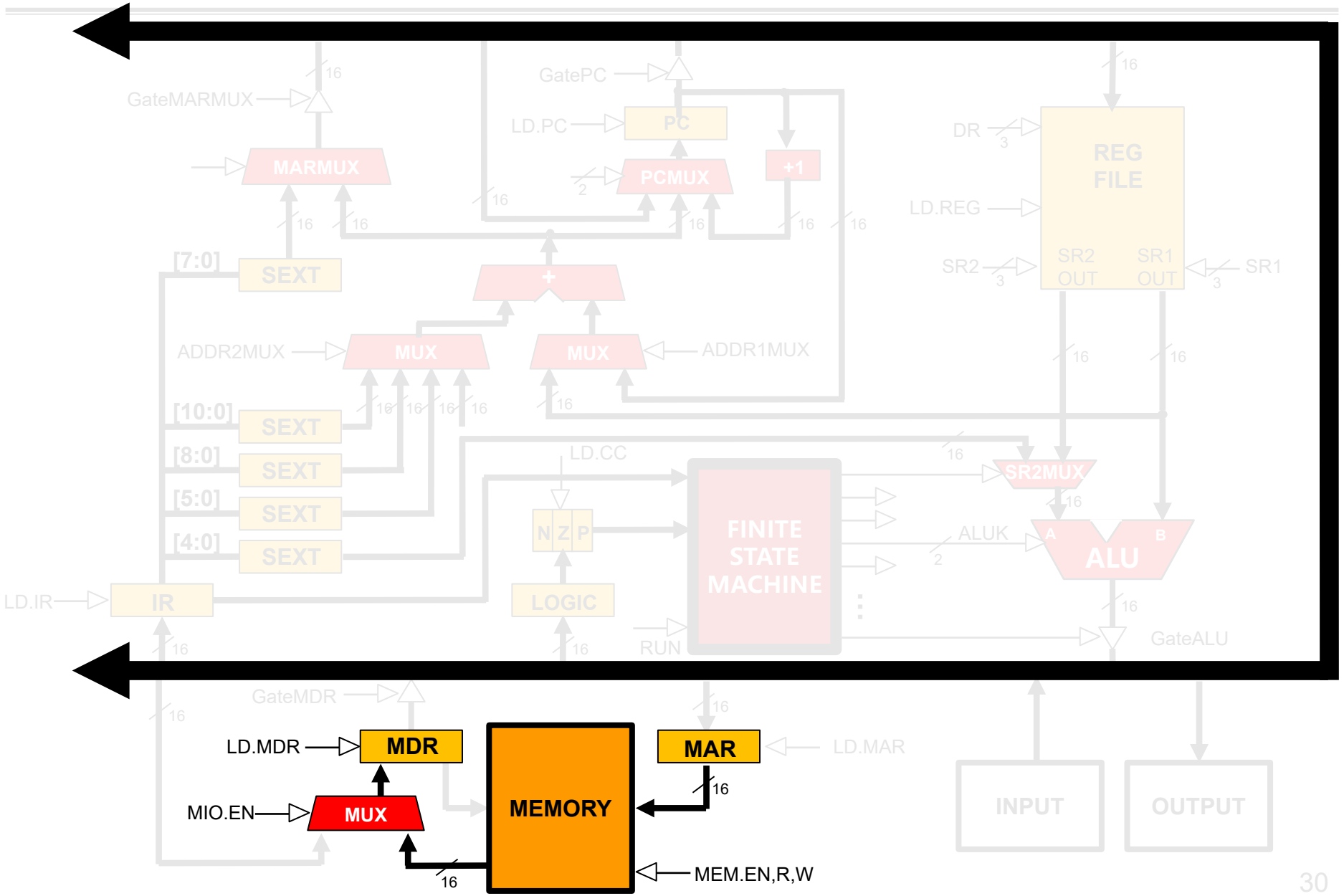
STI (Indirect)



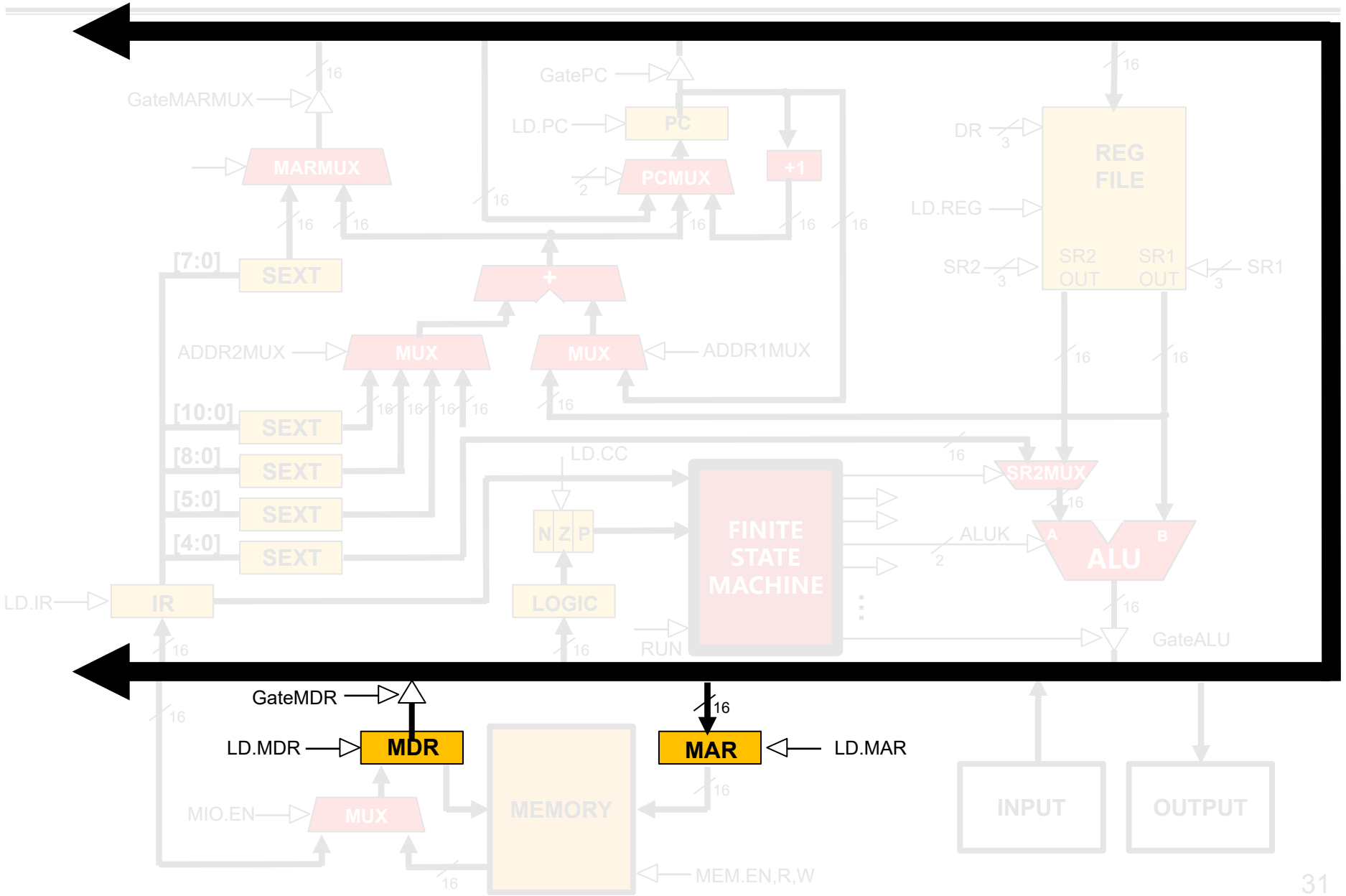
GateMARMUX
MARMUX
16
GatePC
LD.PC
PC
PCMUX
+1
+
ADDR2MUX
MUX
MUX
ADDR1MUX
[7:0]
SEXT
[10:0]
SEXT
[8:0]
SEXT
[5:0]
SEXT
[4:0]
SEXT
LD.IR
IR
LD.CC
N Z P
LOGIC
FINITE STATE MACHINE
RUN
DR
REG FILE
LD.REG
SR2
SR2 OUT
SR1 OUT
SR1
SR2MUX
ALUK
A
B
ALU
GateALU
GateMDR
LD.MDR
MDR
MIO.EN
MUX
MEMORY
MEM.EN,R,W
MAR
LD.MAR
INPUT
OUTPUT

STI (Indirect)

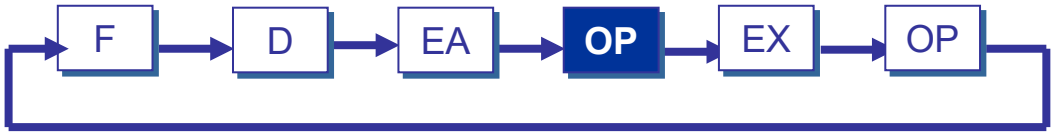




STI (Indirect)

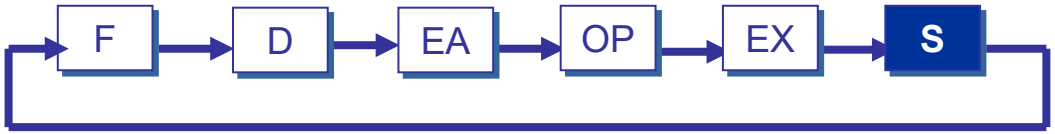


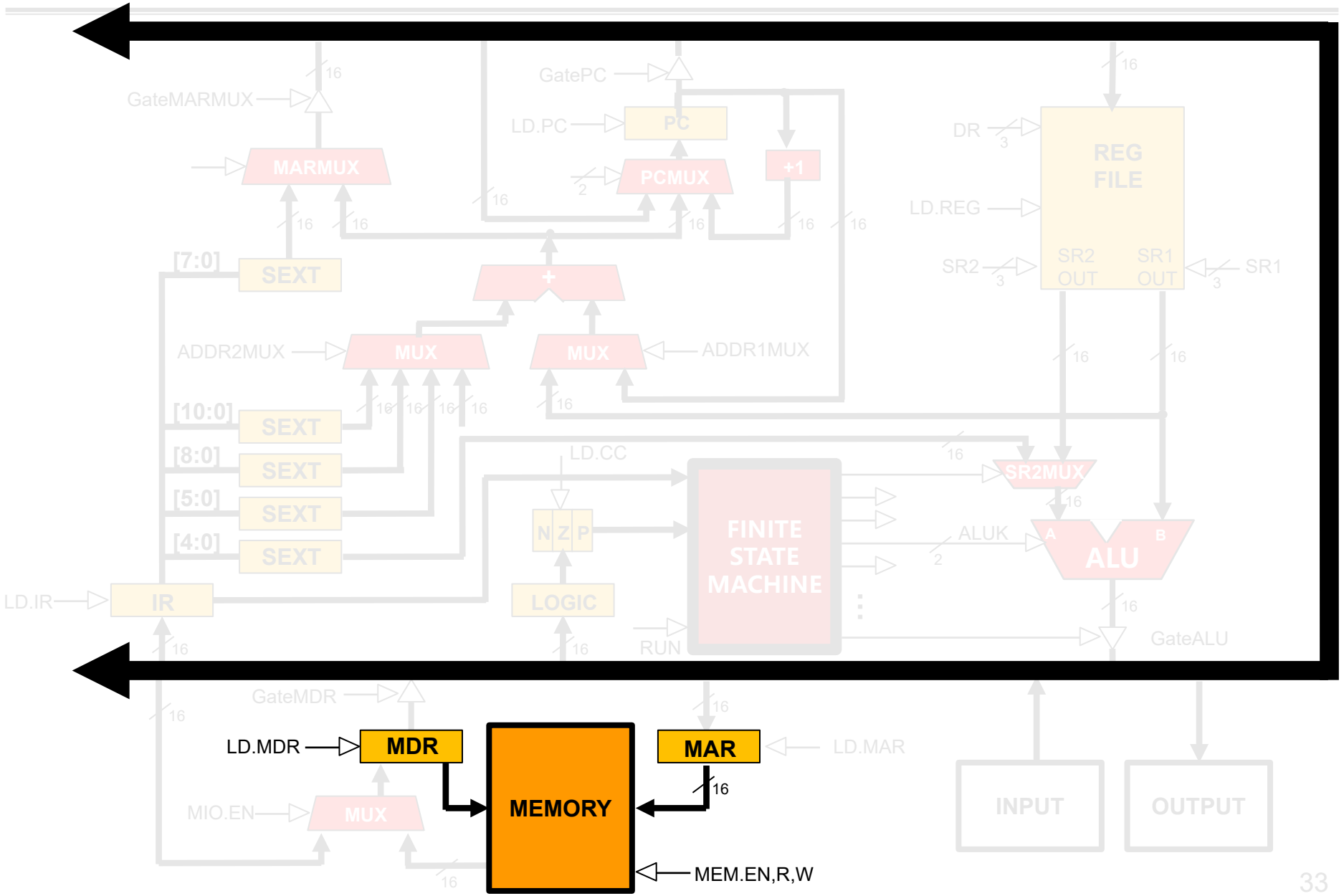
STI (Indirect)



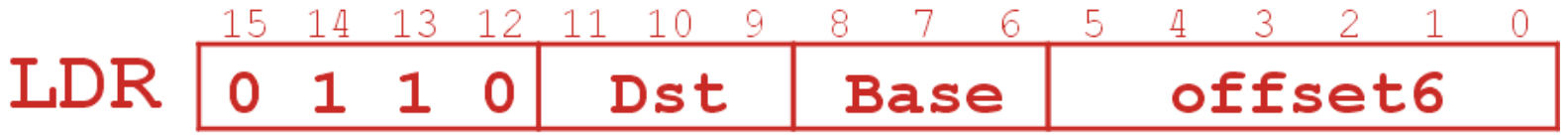
GateMARMUX
MARMUX
16
+
MUX
ADDR1MUX
SEXT
[7:0]
[10:0]
[8:0]
[5:0]
[4:0]
ADDR2MUX
REG FILE
DR
SR1
SR2
LD.REG
SR2 OUT
SR1 OUT
LD.CC
N Z P
FINITE STATE MACHINE
RUN
SR2MUX
ALUK
ALU
GateALU
LD.IR
IR
LOGIC
GatePC
LD.PC
PC
PCMUX
+1
GateMDR
LD.MDR
MDR
MIO.EN
MUX
MEMORY
MAR
LD.MAR
MEM.EN,R,W
INPUT
OUTPUT

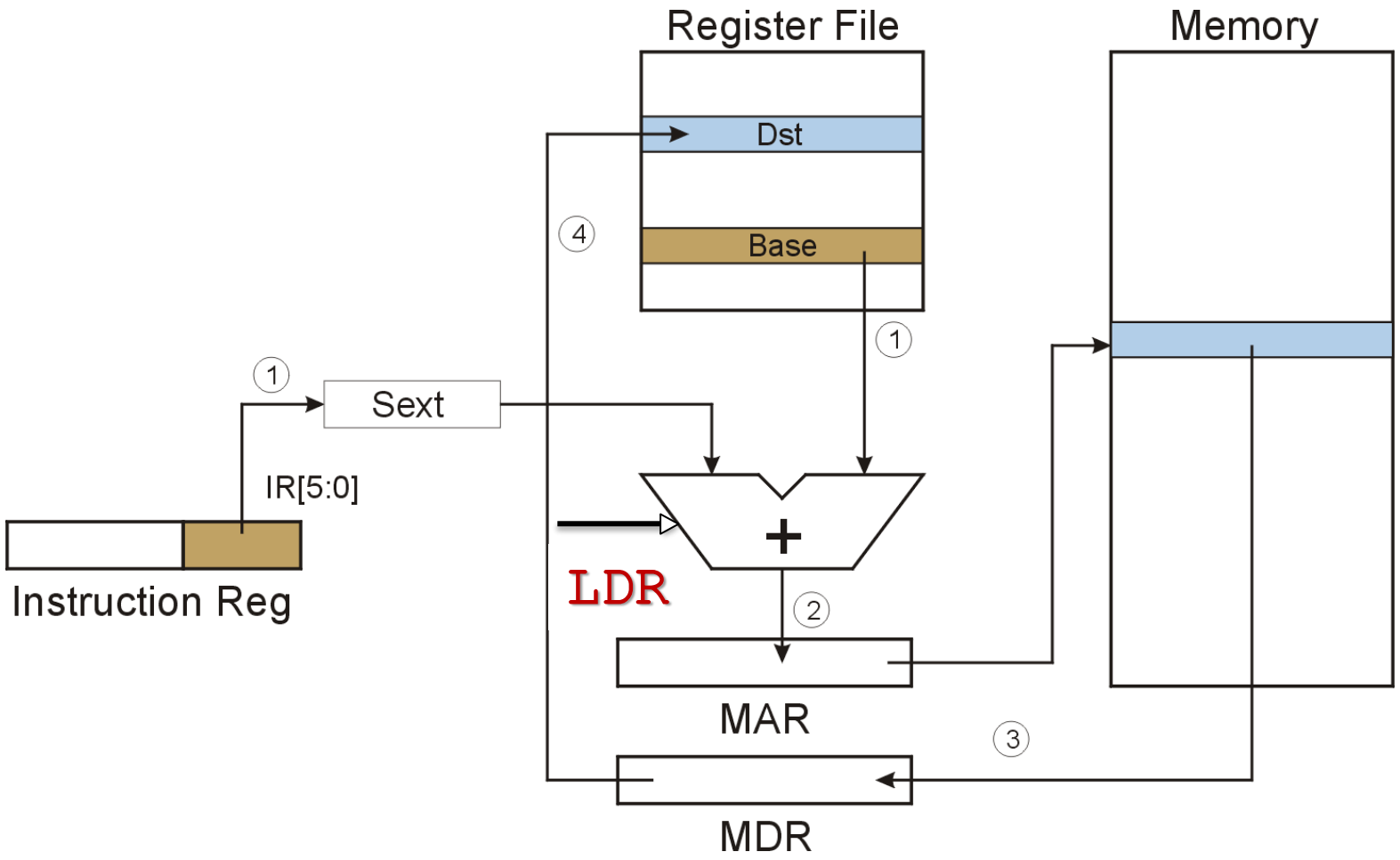
STI (Indirect)



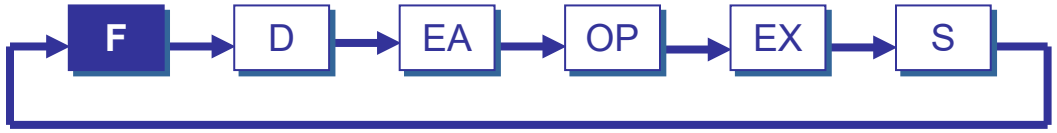


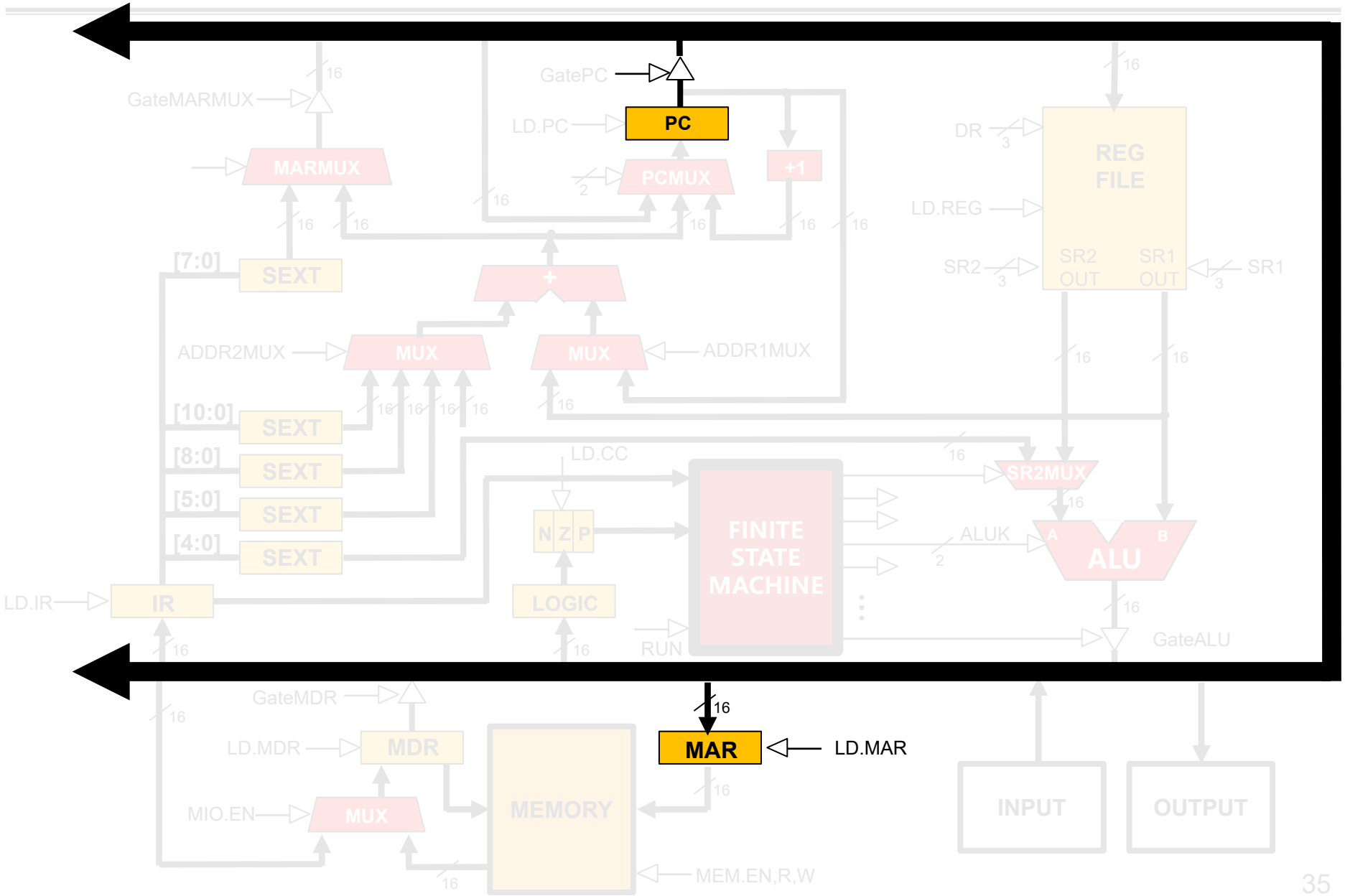
LDR (Base+Offset) LDR DR, BaseR, offset6



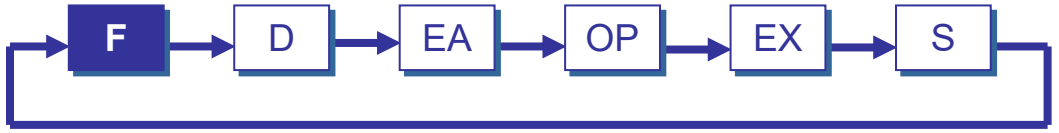


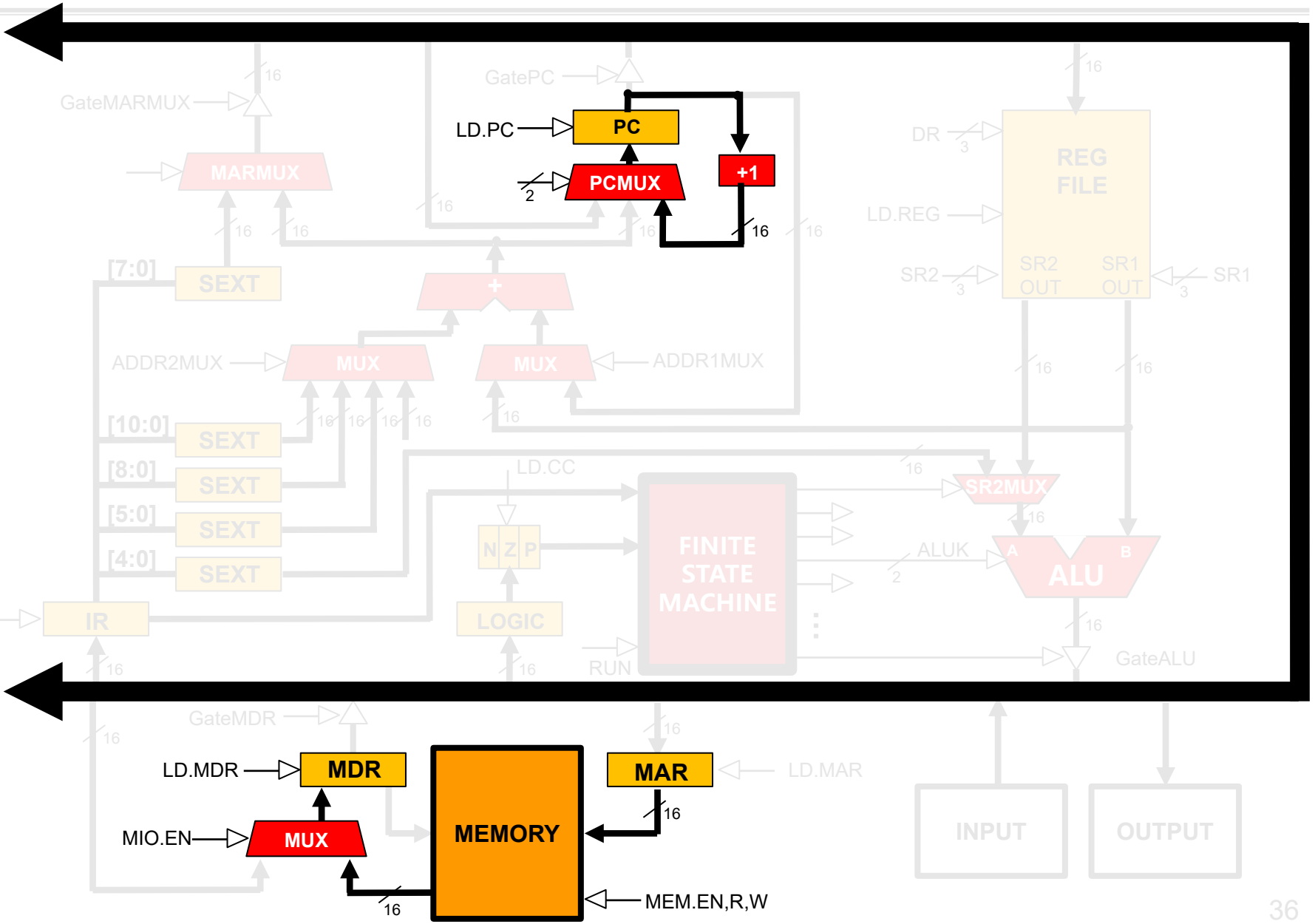
LDR (Base+Offset)



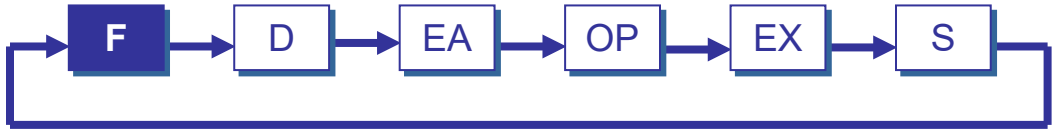


LDR (Base+Offset)

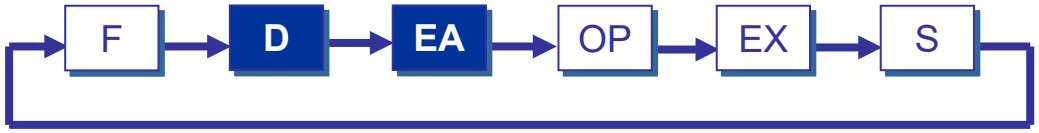




LDR (Base+Offset)

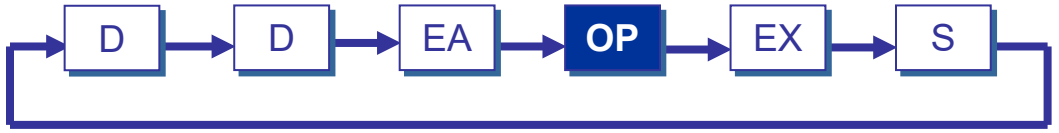


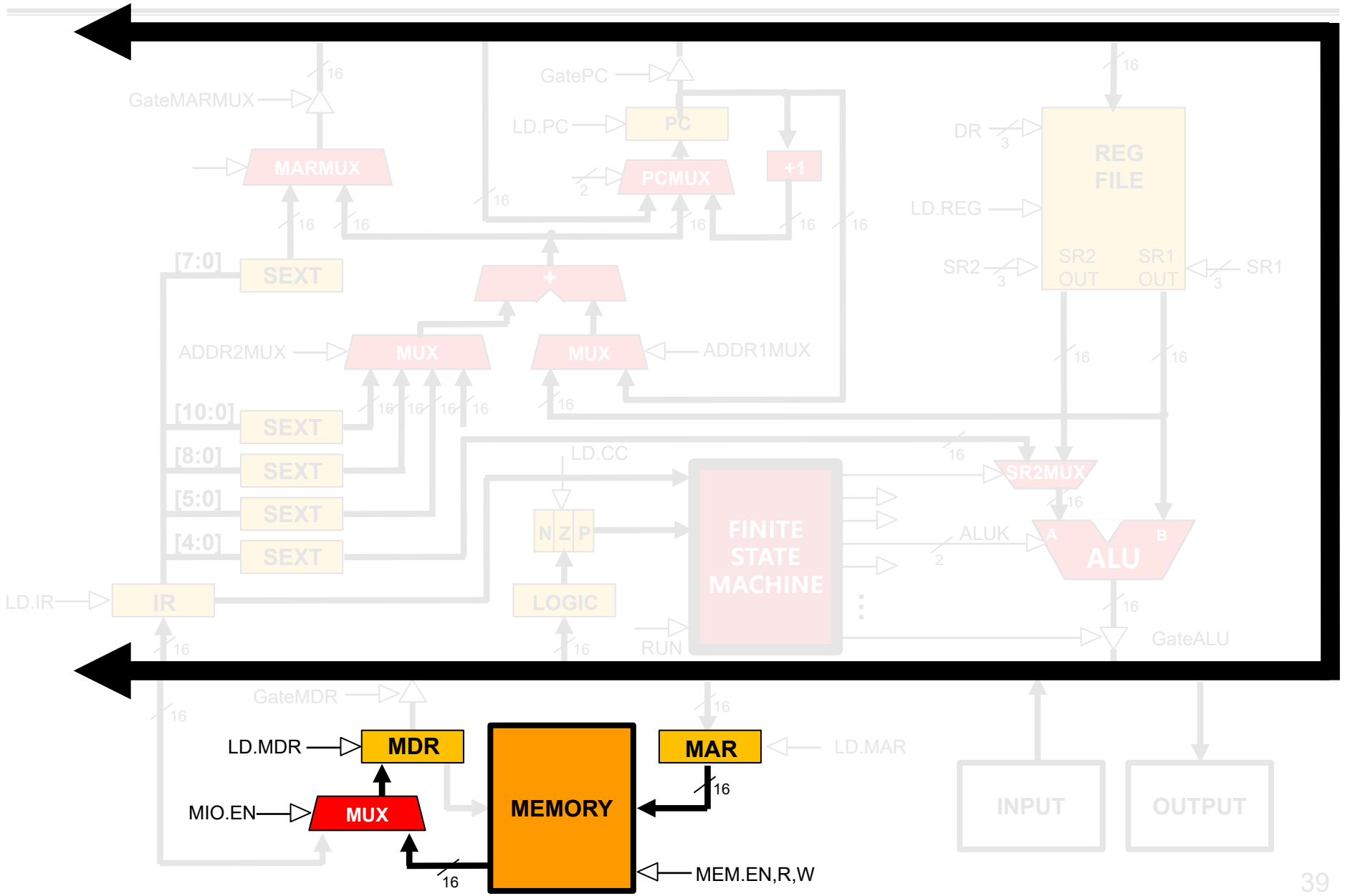
LDR (Base+Offset)



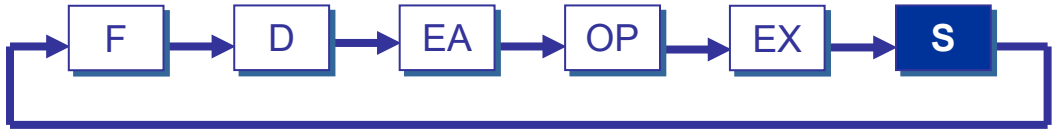
GateMARMUX
16
MARMUX
16
+
ADDR2MUX
MUX
MUX
ADDR1MUX
16
16
[7:0]
SEXT
[10:0]
SEXT
[8:0]
SEXT
[5:0]
SEXT
[4:0]
SEXT
LD.IR
IR
GatePC
LD.PC
PC
PCMUX
+1
2
16
DR
3
REG FILE
LD.REG
SR2
3
SR2 OUT
SR1 OUT
SR1
3
16
16
16
LD.CC
N Z P
LOGIC
FINITE STATE MACHINE
RUN
SR2MUX
ALUK
2
A
B
ALU
GateALU
GateMDR
LD.MDR
MDR
MIO.EN
MUX
MEMORY
MEM.EN,R,W
16
MAR
LD.MAR
INPUT
OUTPUT

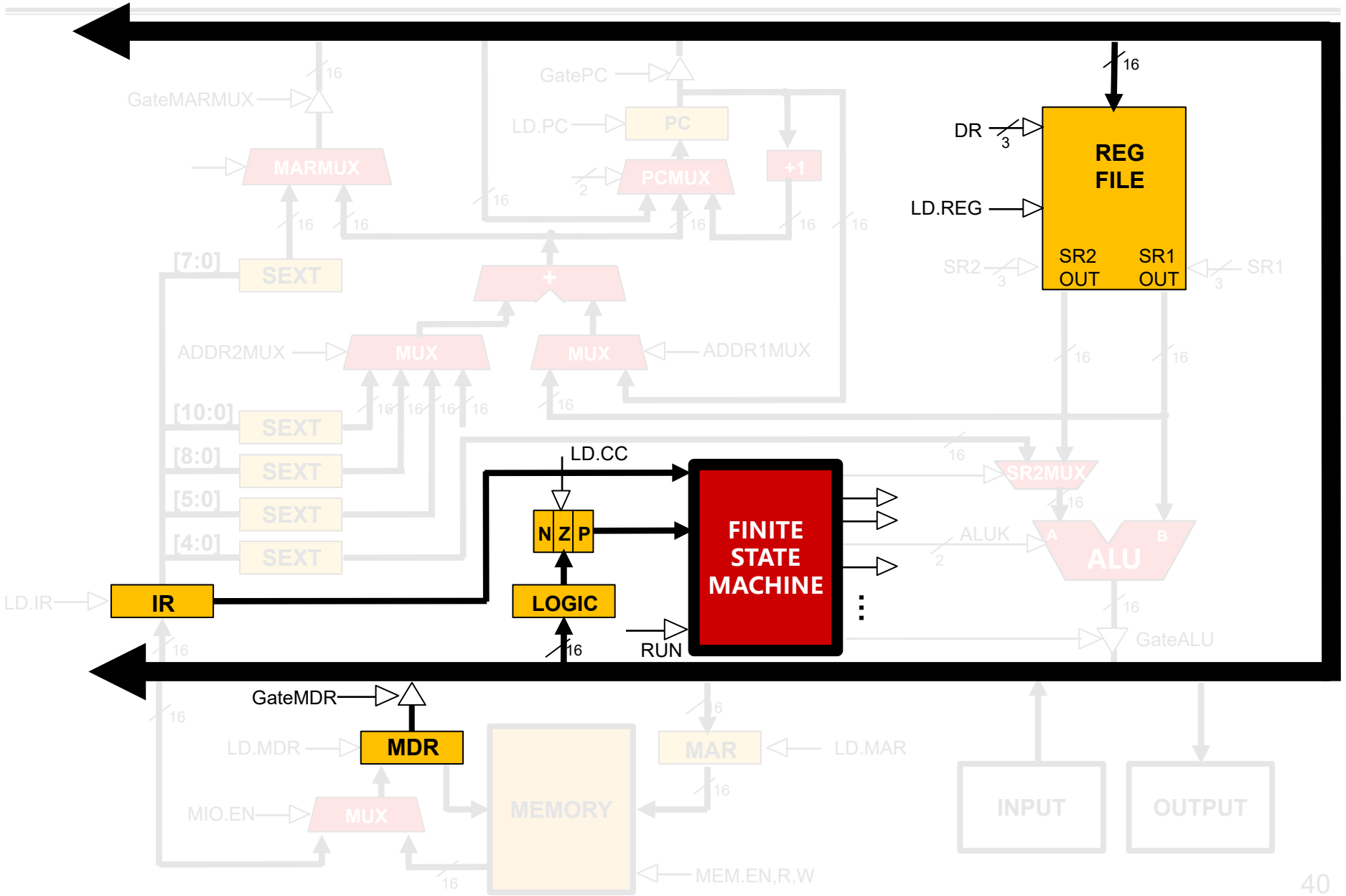
LDR (Base+Offset)



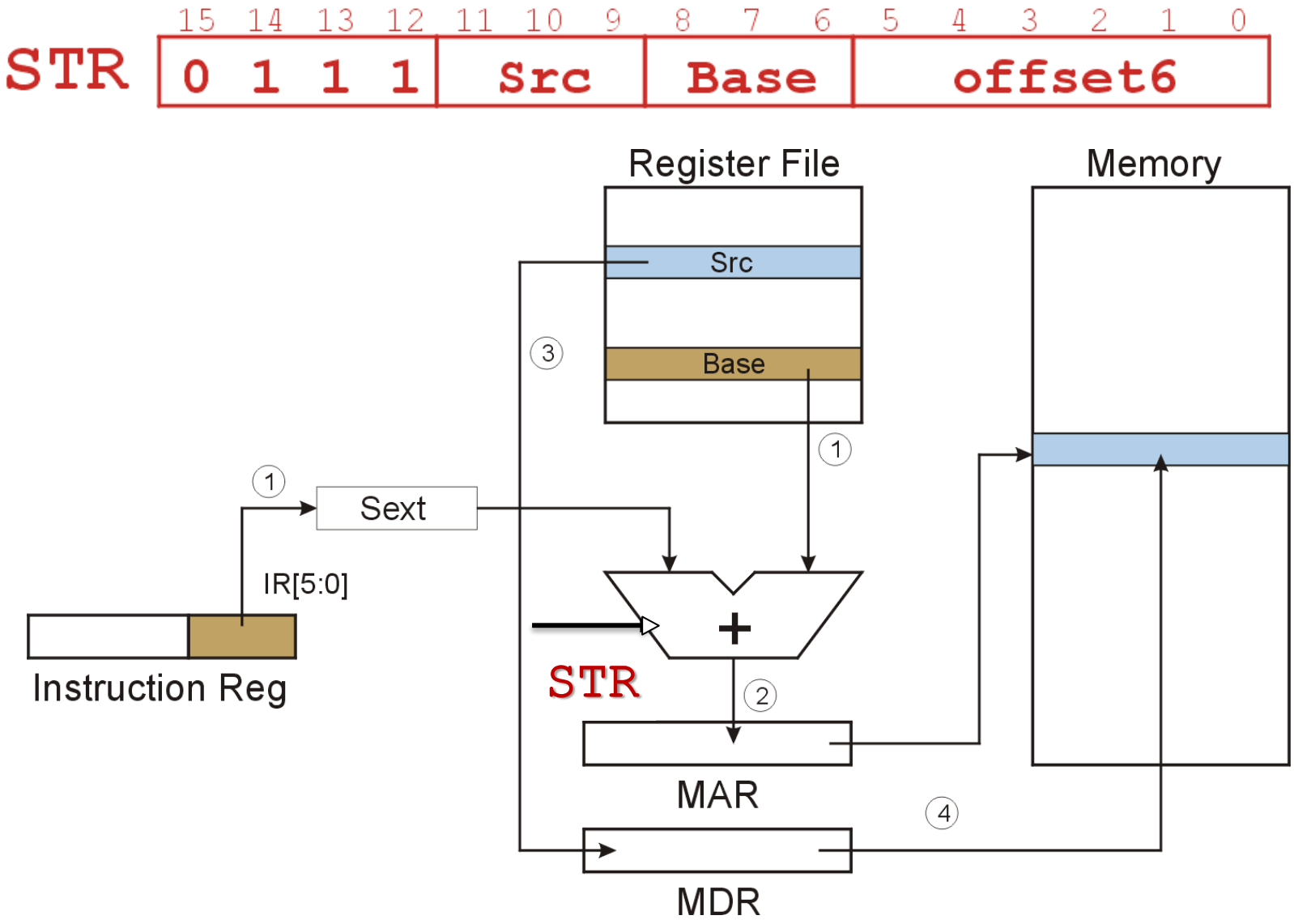


LDR (Base+Offset)

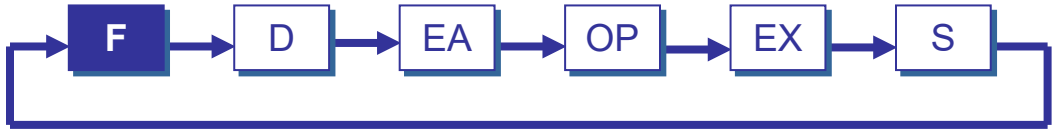


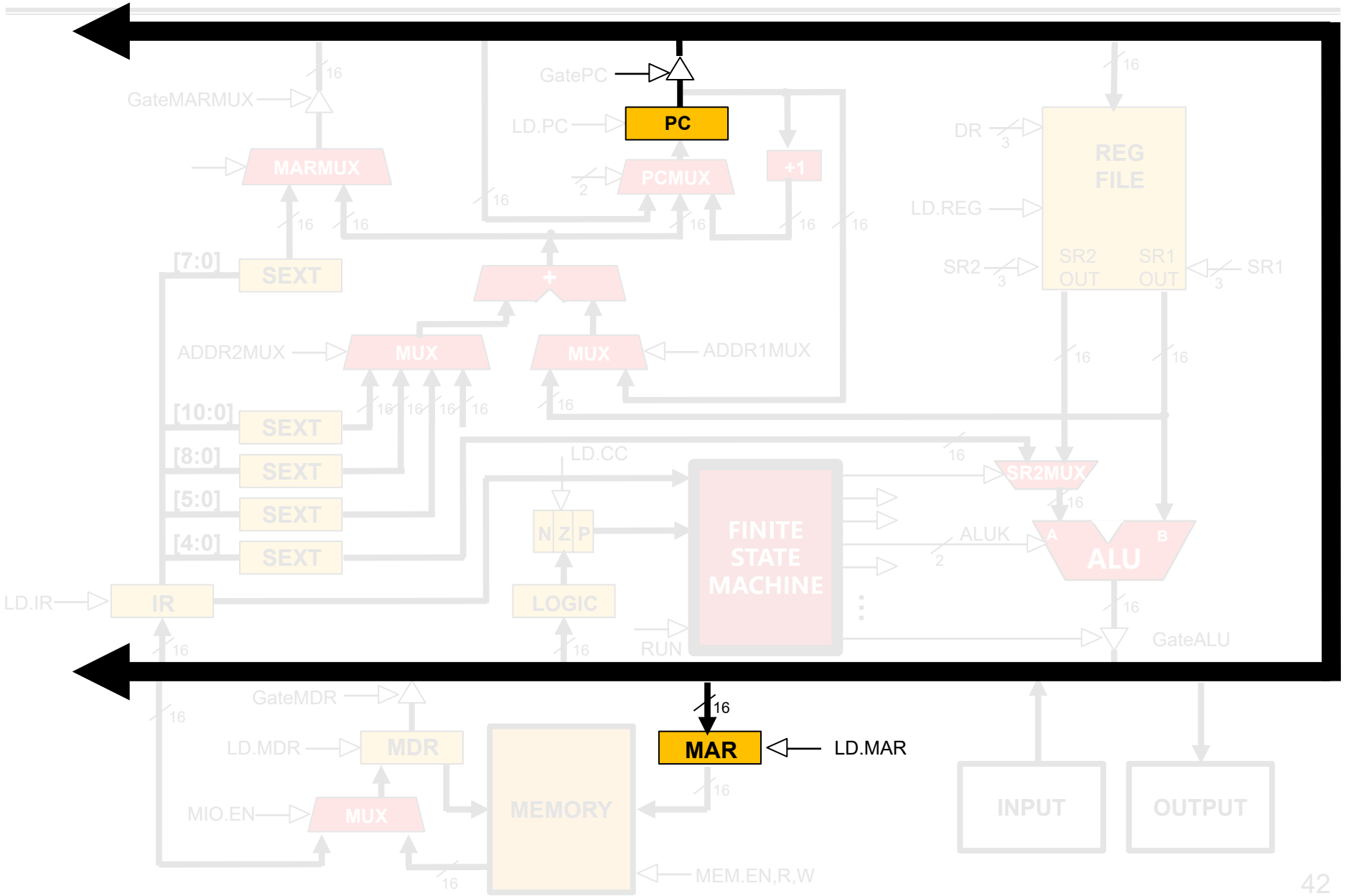


STR (Base+Offset) STR SR, BaseR, offset6

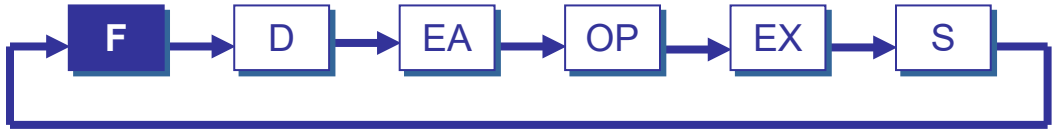


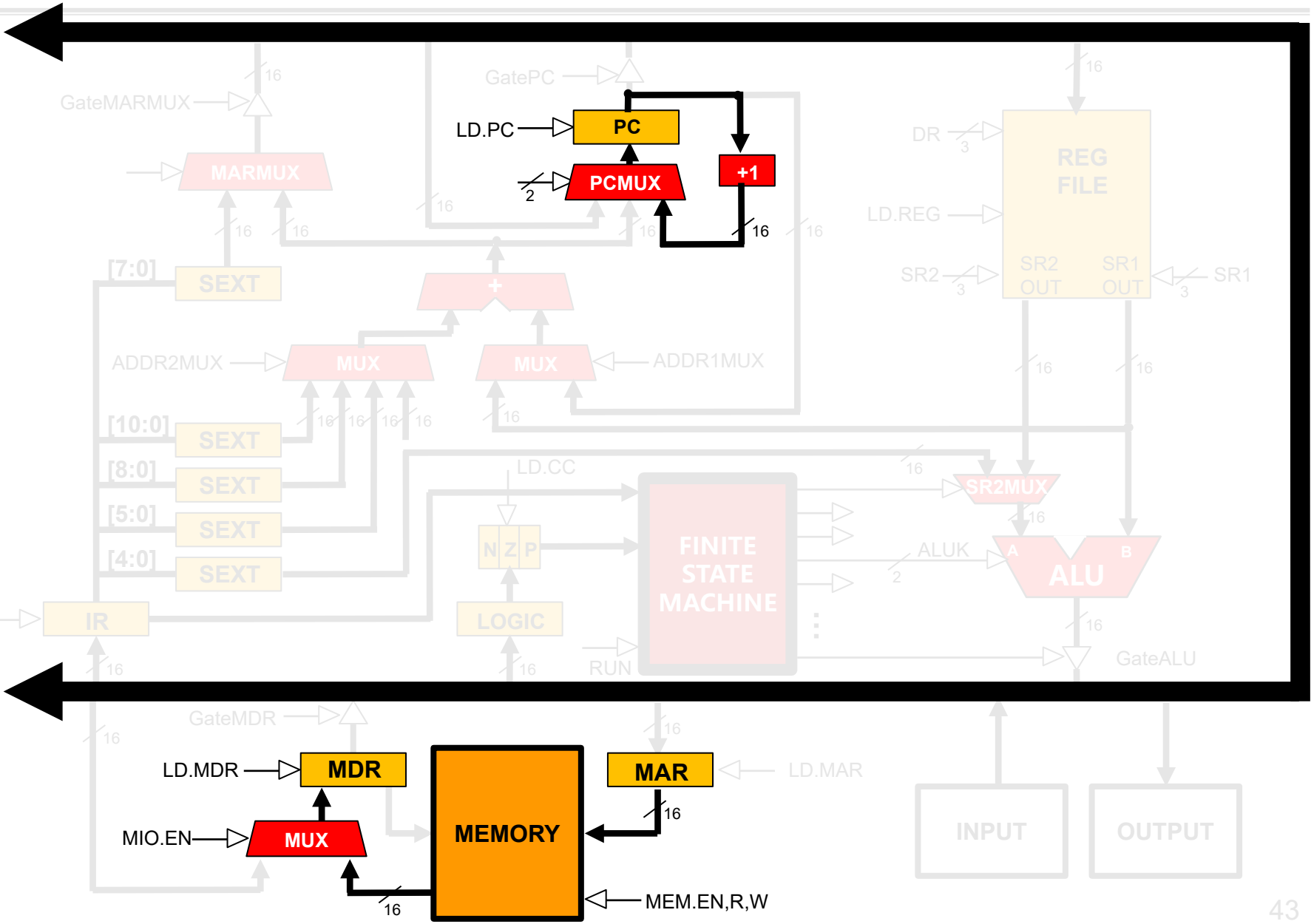
STR (Base+Offset)



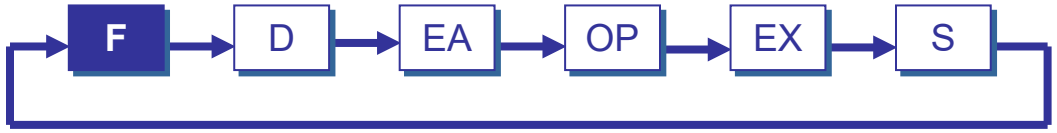


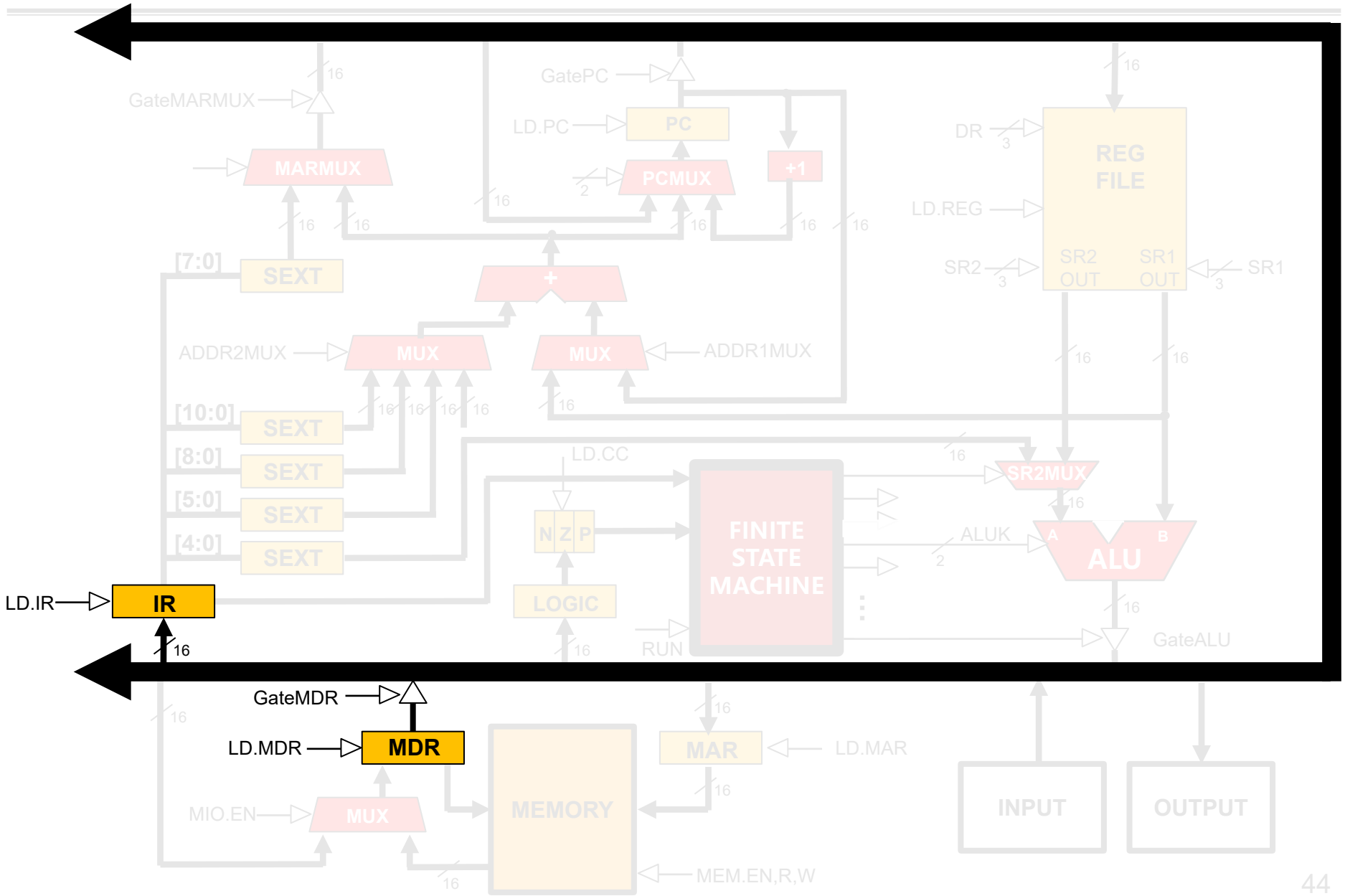
STR (Base+Offset)



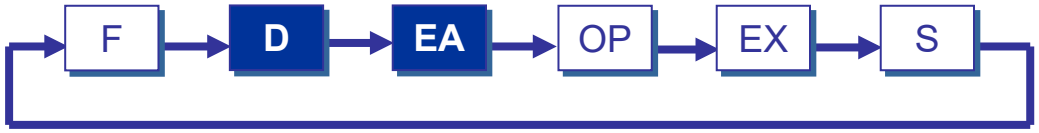


STR (Base+Offset)



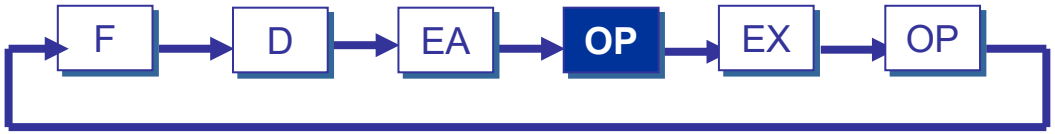


STR (Base+Offset)



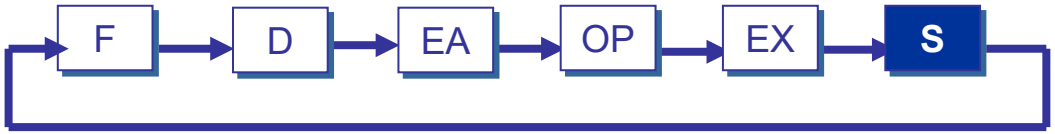
GateMARMUX
16
MARMUX
16
+
ADDR2MUX
MUX
MUX
ADDR1MUX
16
16
[7:0]
SEXT
[10:0]
SEXT
[8:0]
SEXT
[5:0]
SEXT
[4:0]
SEXT
LD.IR
IR
LD.CC
N Z P
LOGIC
FINITE STATE MACHINE
RUN
REG FILE
DR
3
LD.REG
SR2
3
SR2 OUT
SR1 OUT
SR1
3
16
SR2MUX
ALUK
2
A
B
ALU
GateALU
GatePC
LD.PC
PC
PCMUX
2
+1
GateMDR
LD.MDR
MDR
MIO.EN
MUX
MEMORY
MEM.EN,R,W
16
MAR
LD.MAR
INPUT
OUTPUT

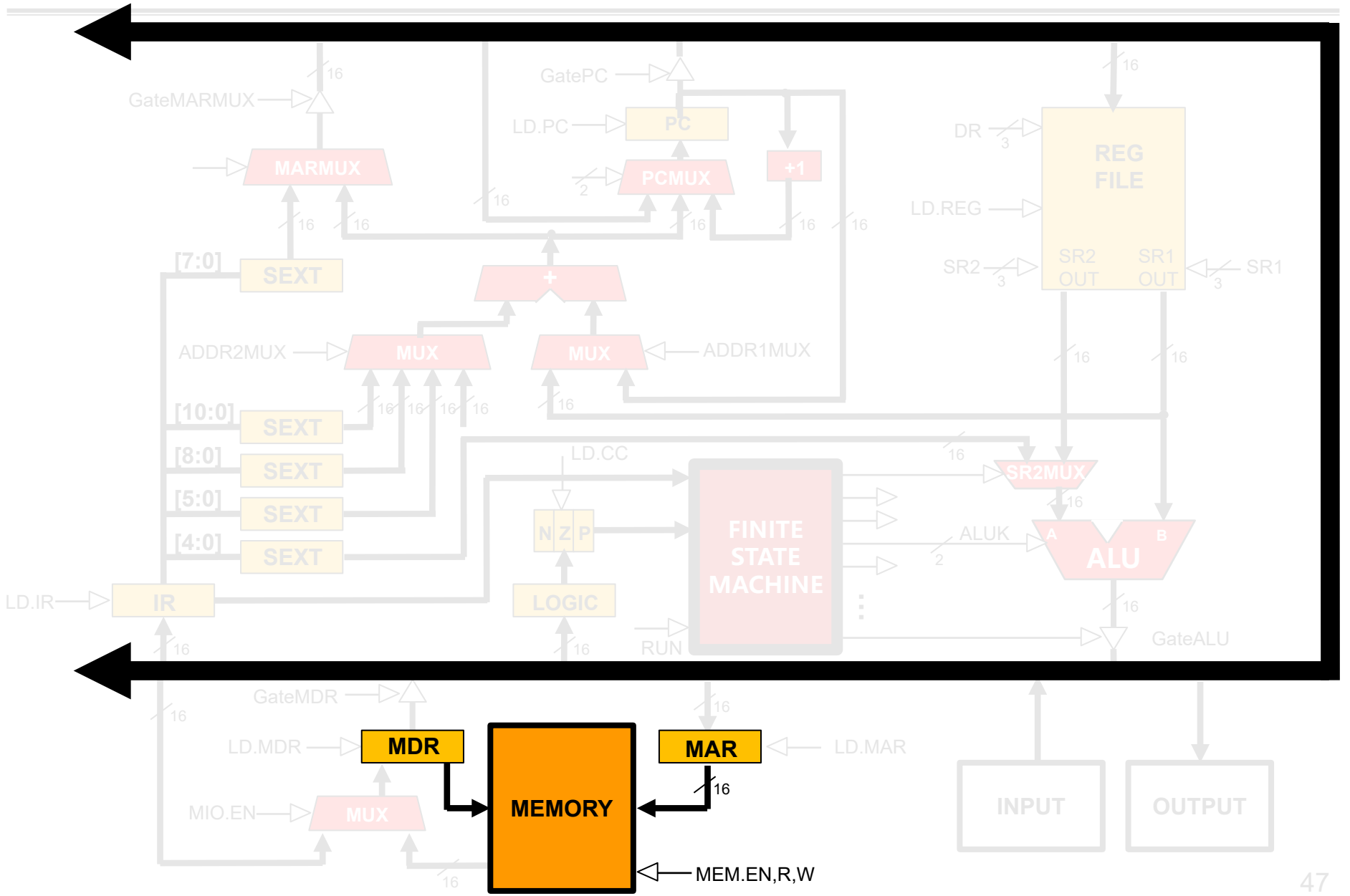
STR (Base+Offset)



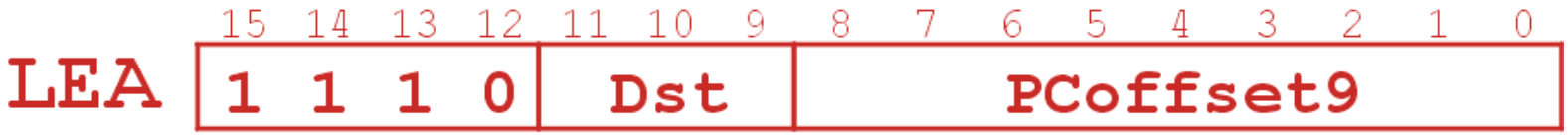
GateMARMUX
16
MARMUX
16
16
[7:0]
SEXT
+
ADDR2MUX
MUX
MUX
ADDR1MUX
16
16
[10:0]
SEXT
[8:0]
SEXT
[5:0]
SEXT
[4:0]
SEXT
LD.IR
IR
LD.CC
N Z P
LOGIC
FINITE STATE MACHINE
RUN
GatePC
LD.PC
PC
PCMUX
+1
2
DR
3
REG FILE
LD.REG
SR2
3
SR2 OUT
SR1 OUT
SR1
3
16
SR2MUX
ALUK
2
A
B
ALU
GateALU
GateMDR
LD.MDR
MDR
MIO.EN
MUX
MEMORY
MAR
LD.MAR
INPUT
OUTPUT
MEM.EN,R,W

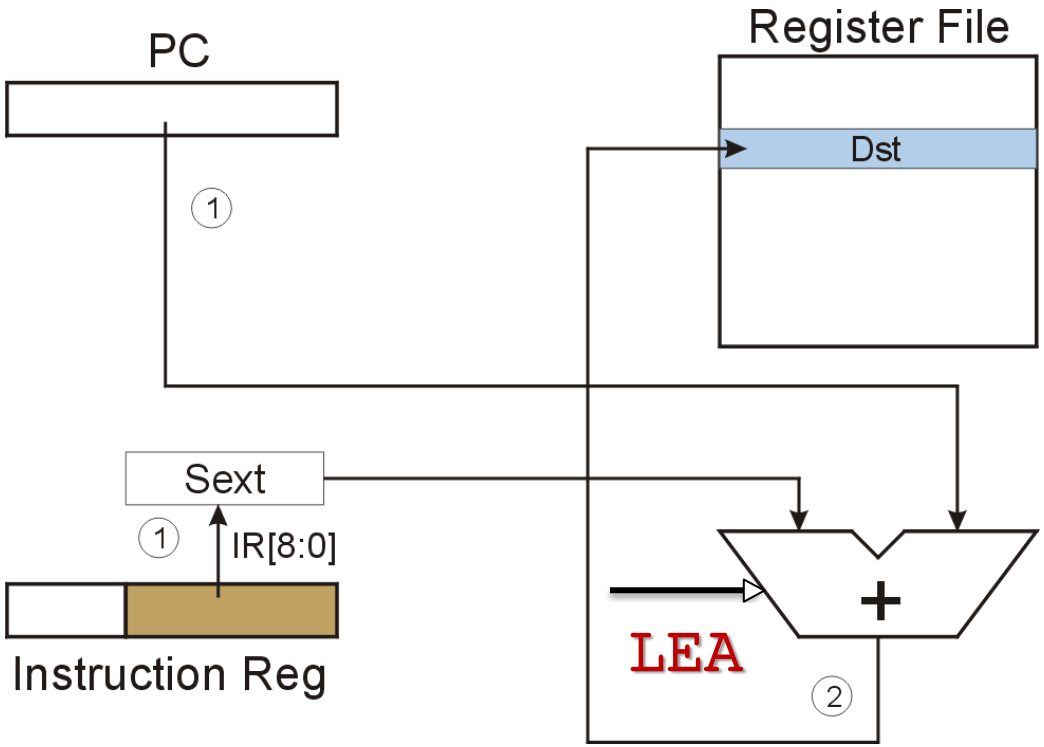
STR (Base+Offset)



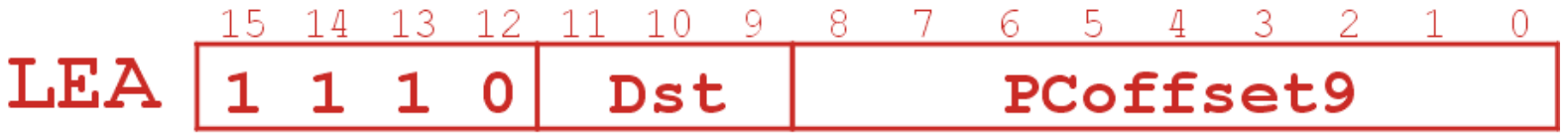


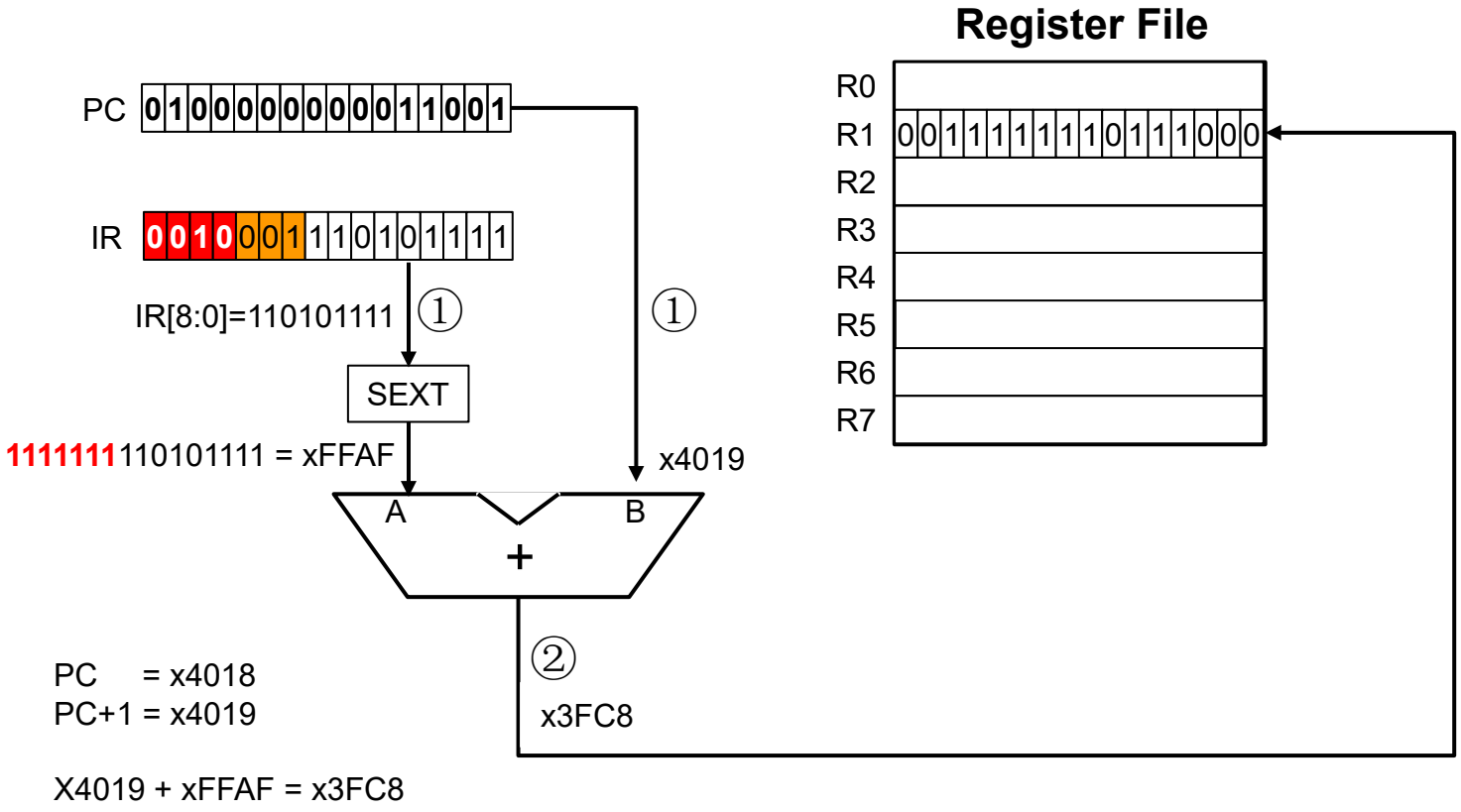
LEA (Immediate) LD DR, PCoffset9



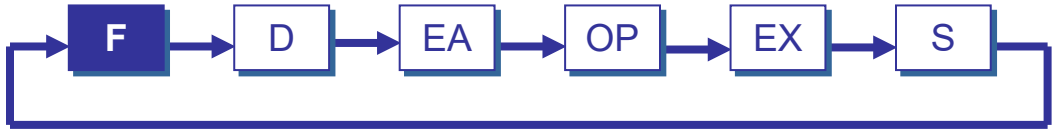


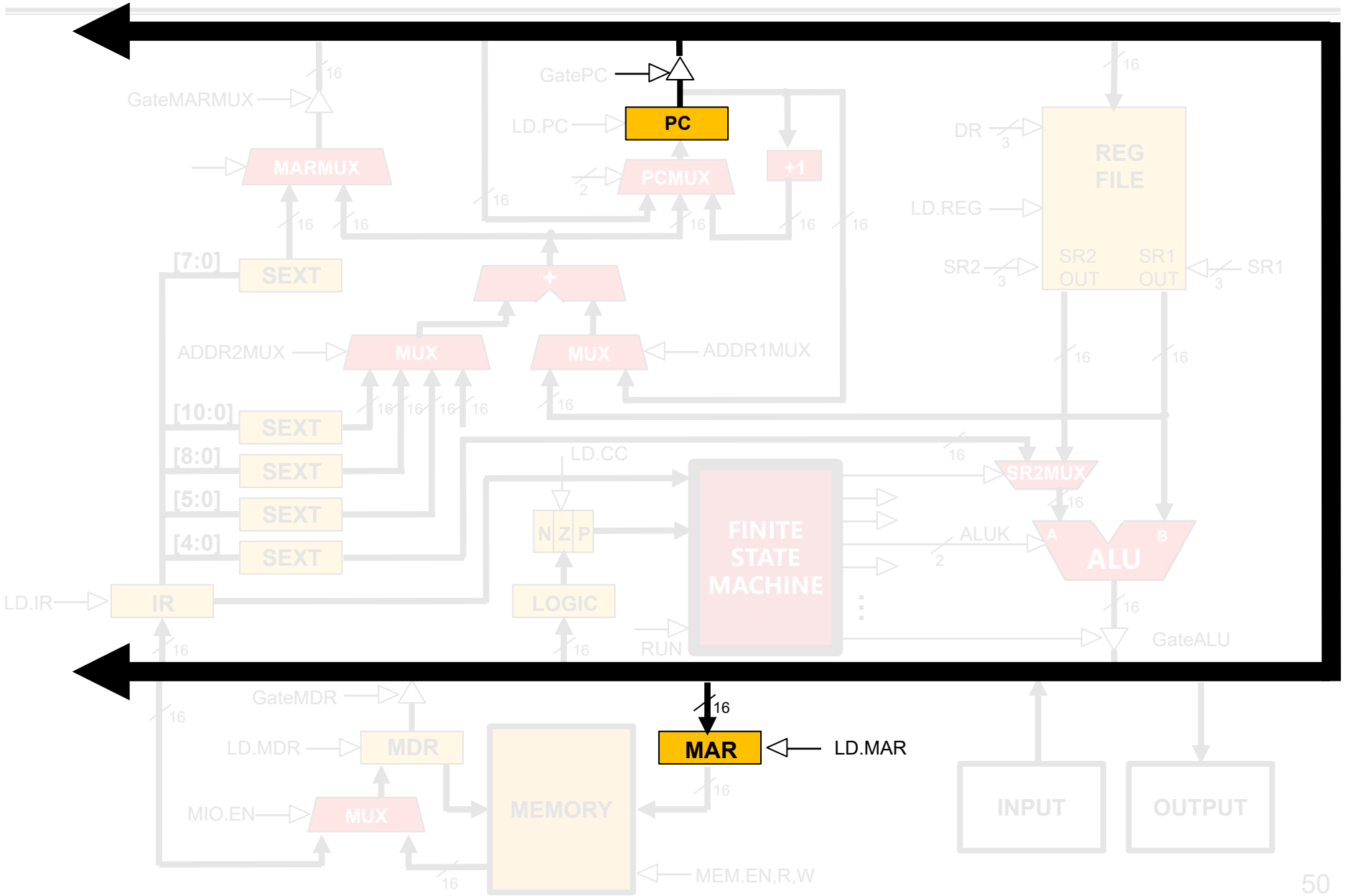
LEA (Immediate): LEA R1, x1AF



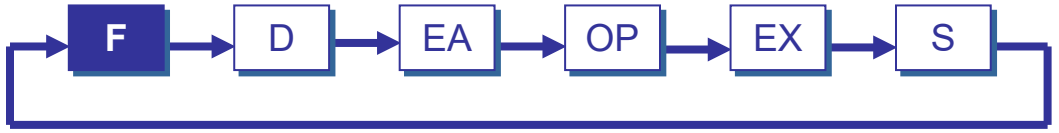


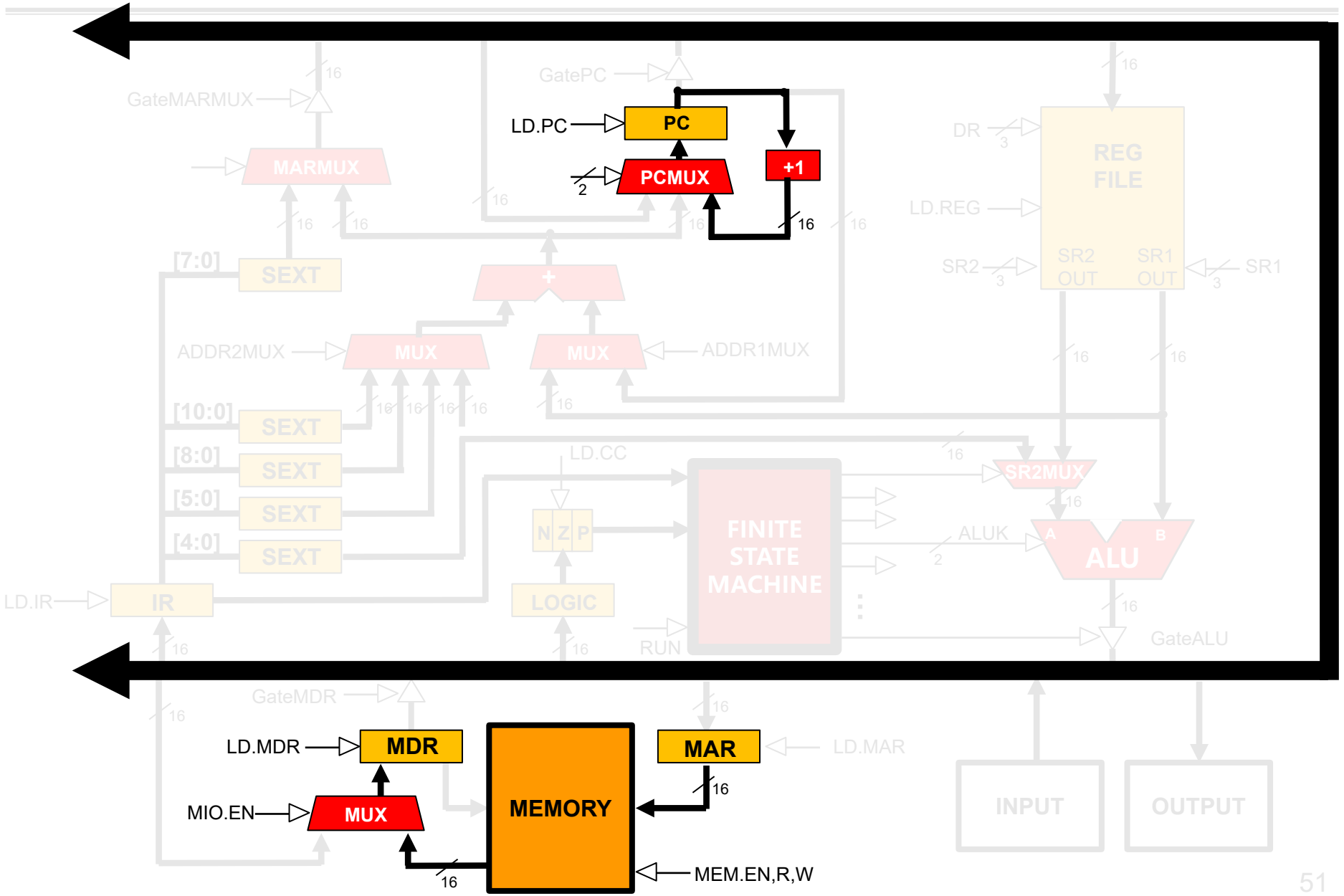
LEA (Immediate)



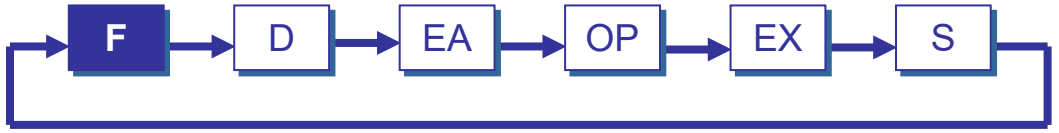


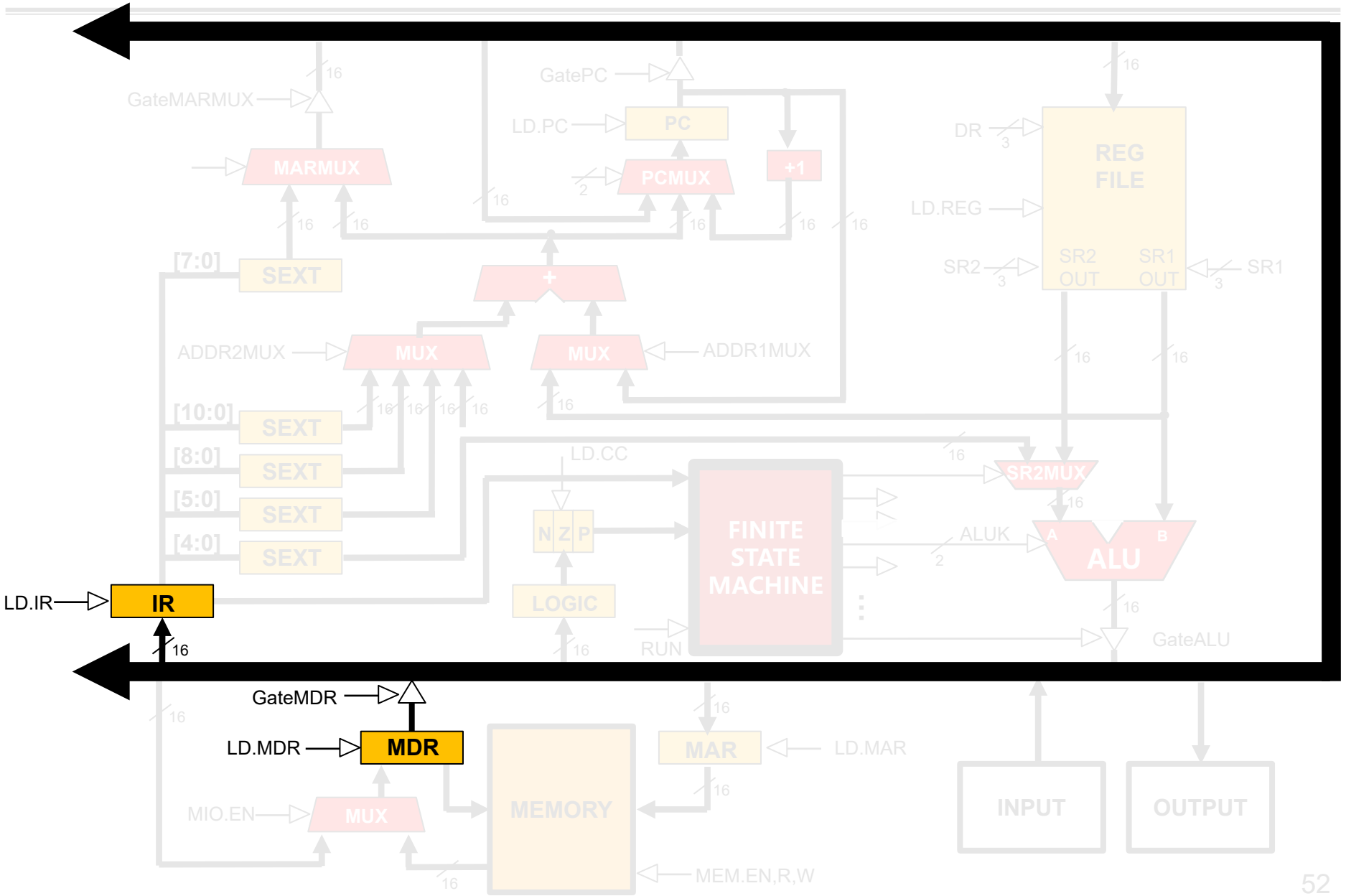
LEA (Immediate)



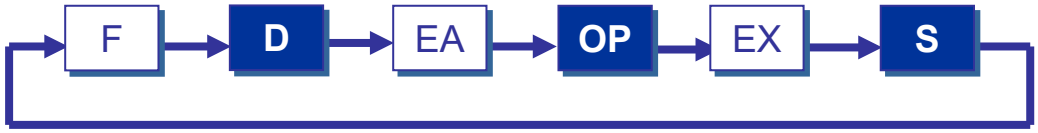


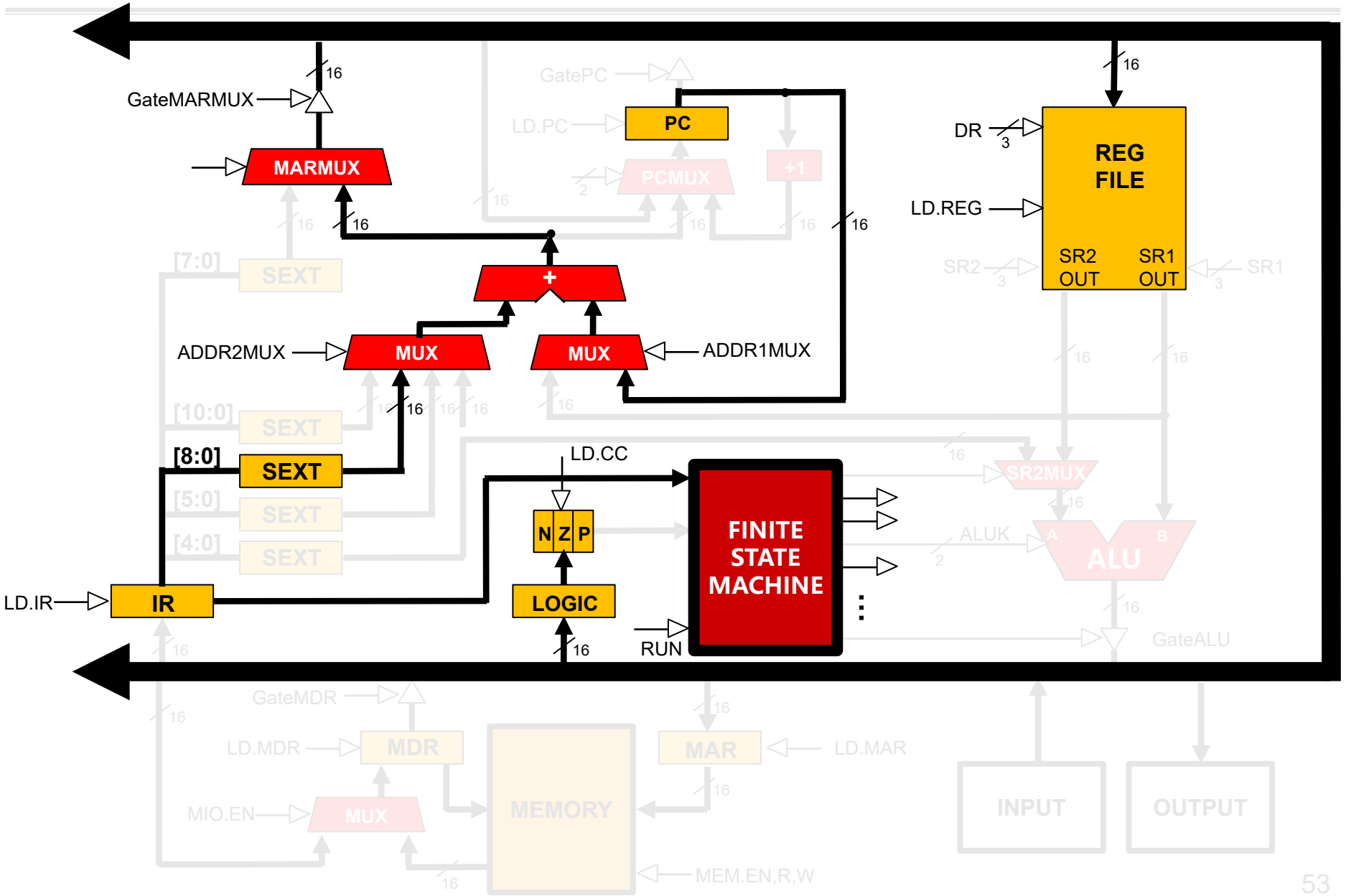
LEA (Immediate)



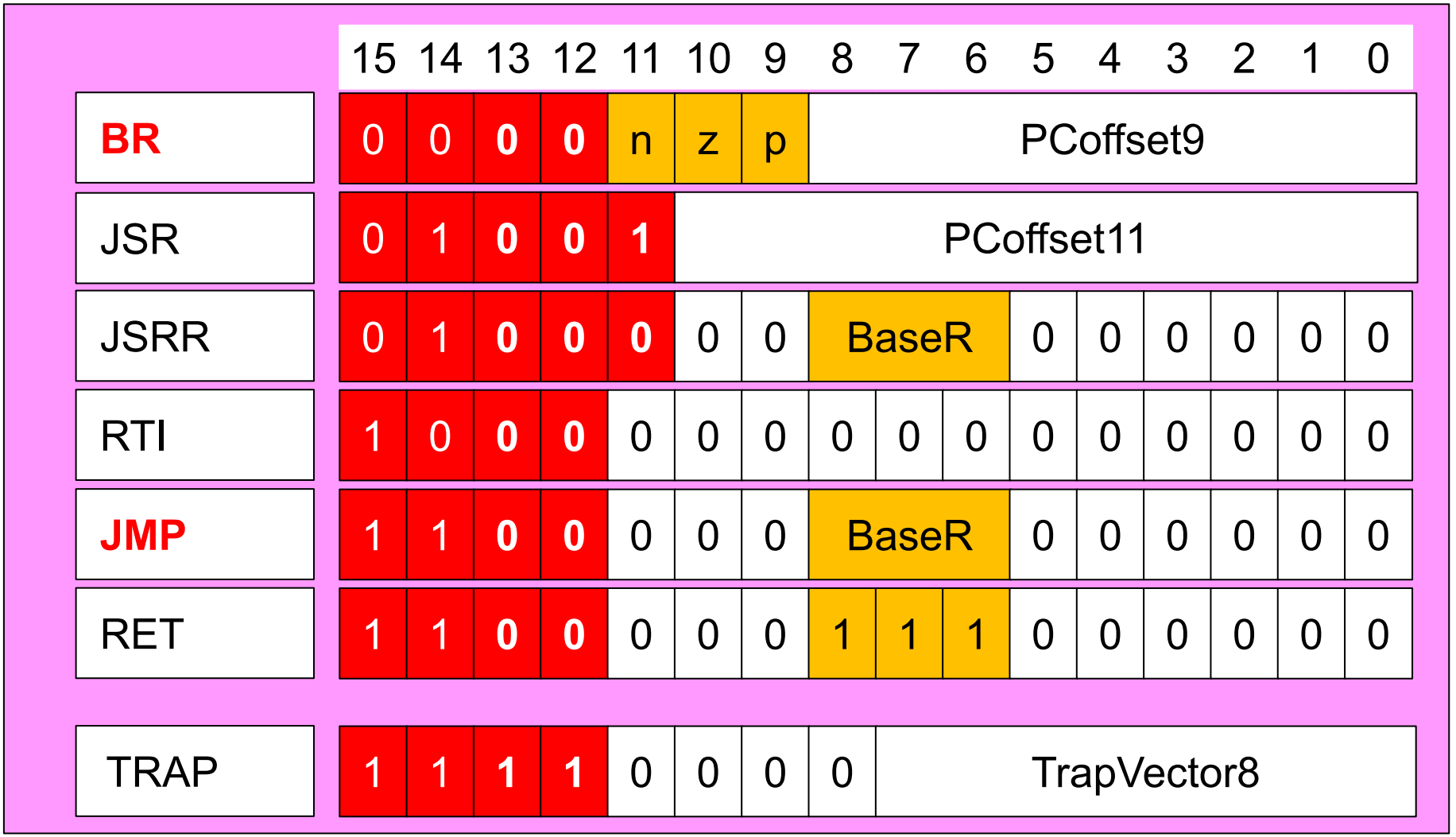


LEA (Immediate)





Control Instructions



Conditional Branch Instruction

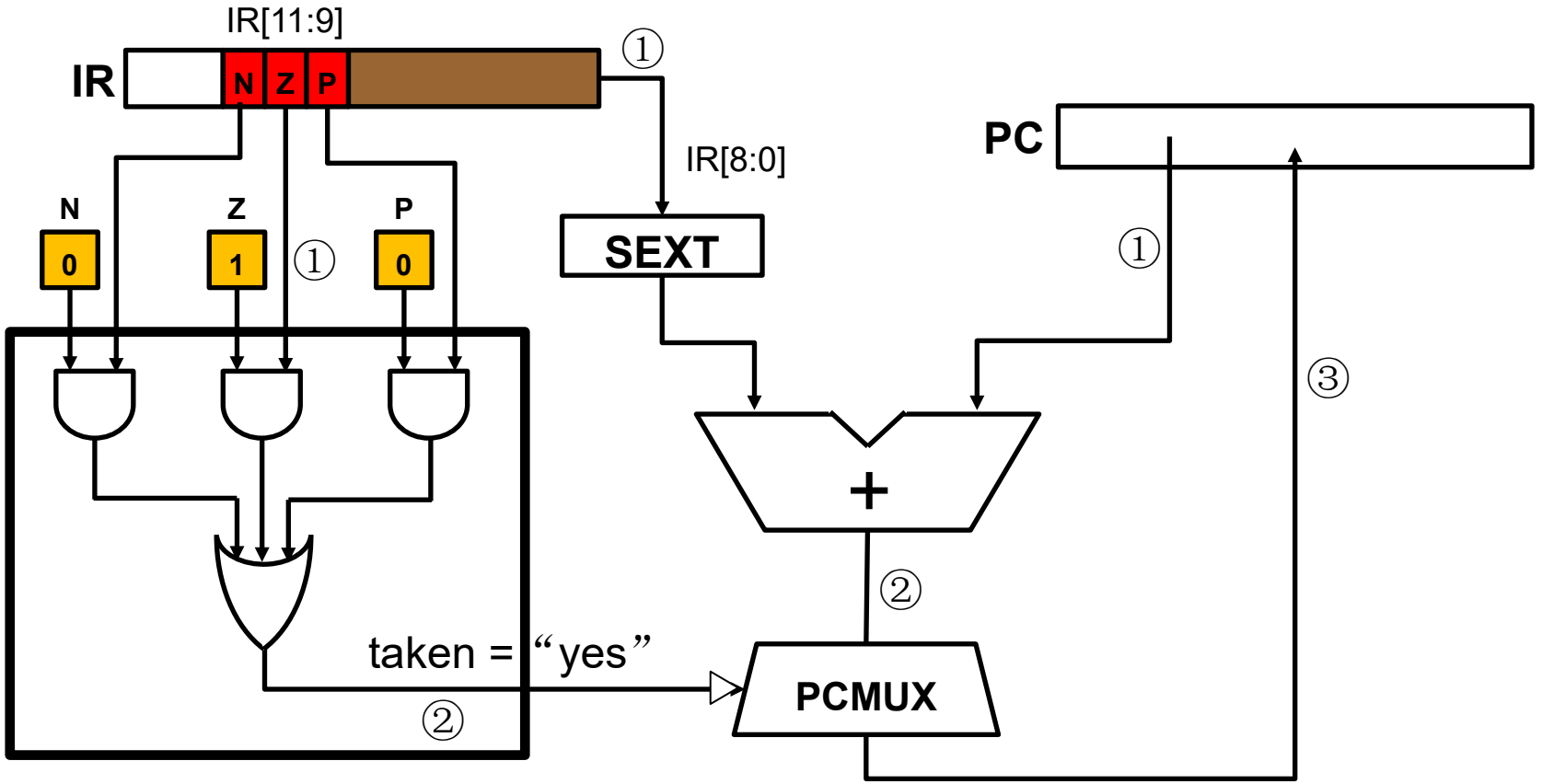
Branch specifies one or more condition codes.

If the specified bit is set, the branch is taken.

- **PC-relative addressing:**
  target address is made by adding signed offset (IR[8:0])
  to current PC.
- **Note: PC has already been incremented by FETCH stage.**
- **Note: Target must be within 256 words of BR instruction.**

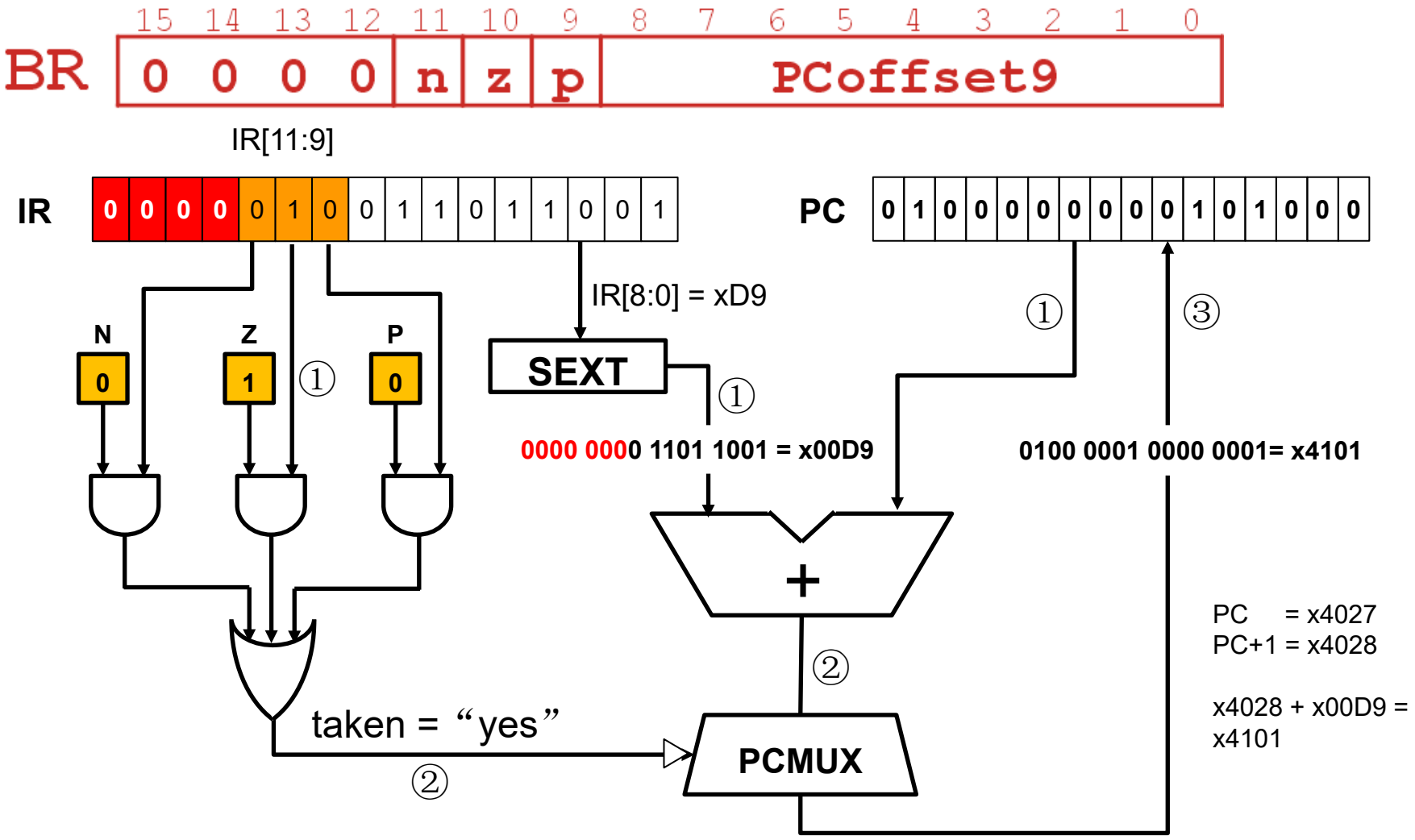
If the branch is not taken, the next sequential instruction is executed.

BR (PC-Relative)



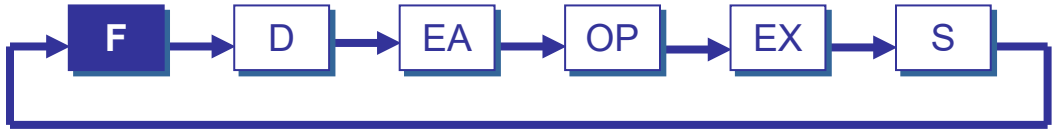
What happens if bits [11:9] are all zero?
What happens if bits [11:9] are all one?

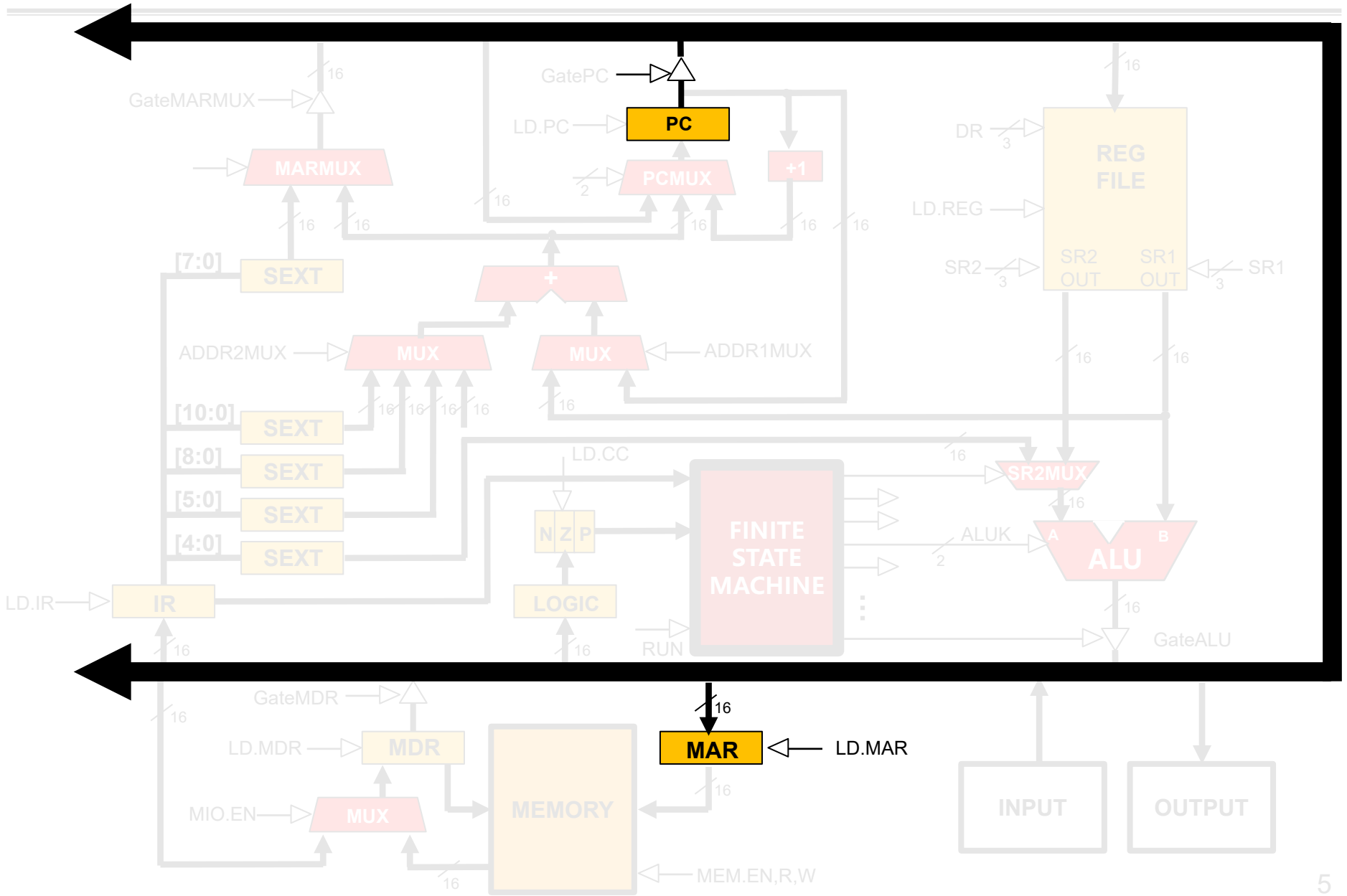
BR (PC-Relative): BR_z x4101



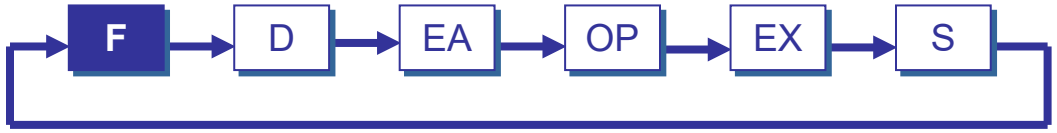
What happens if bits [11:9] are all zero?
What happens if bits [11:9] are all one?

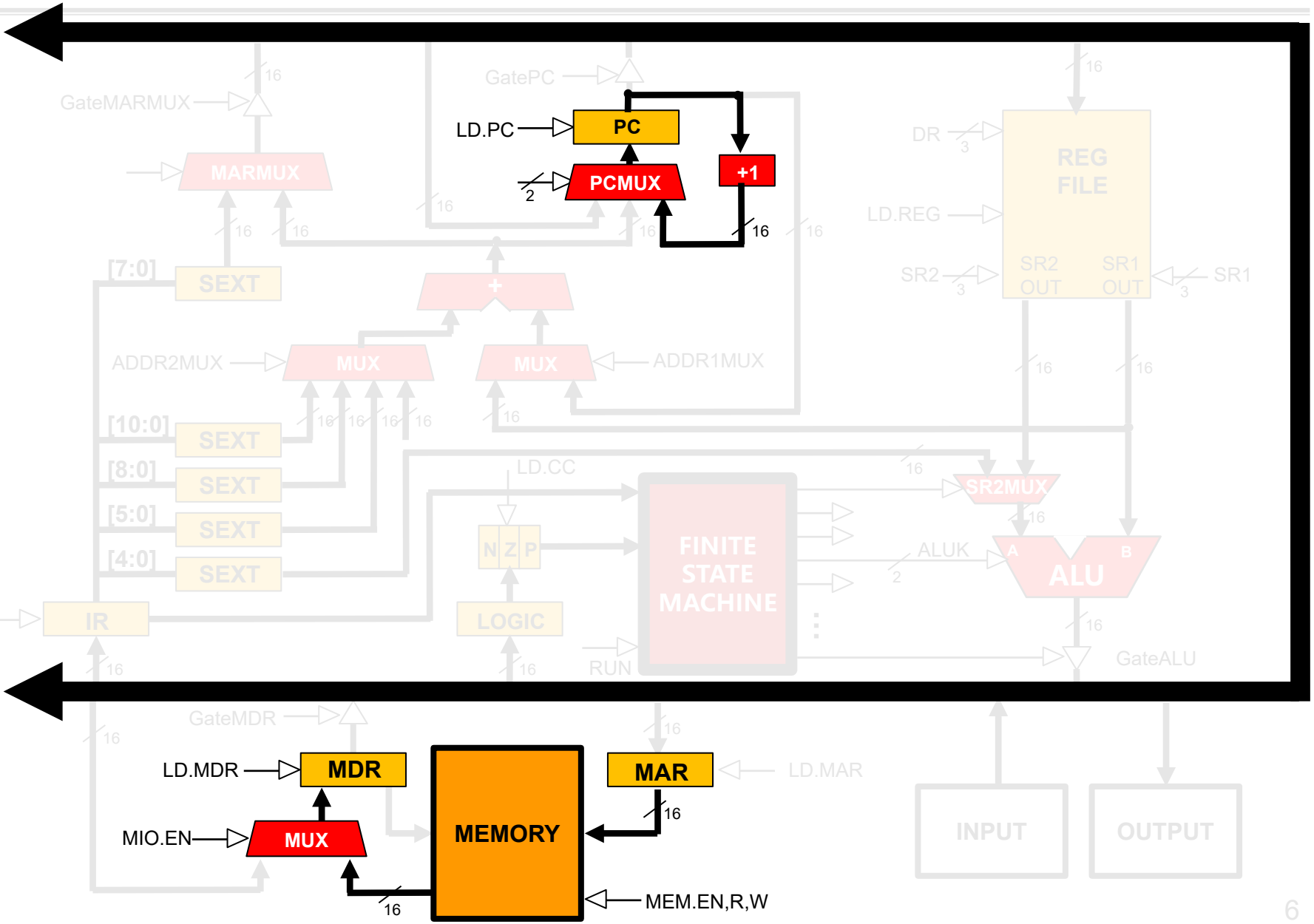
BR (PC-Relative)



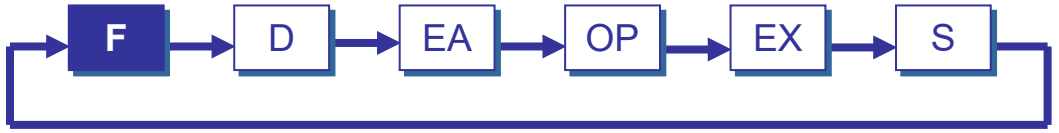


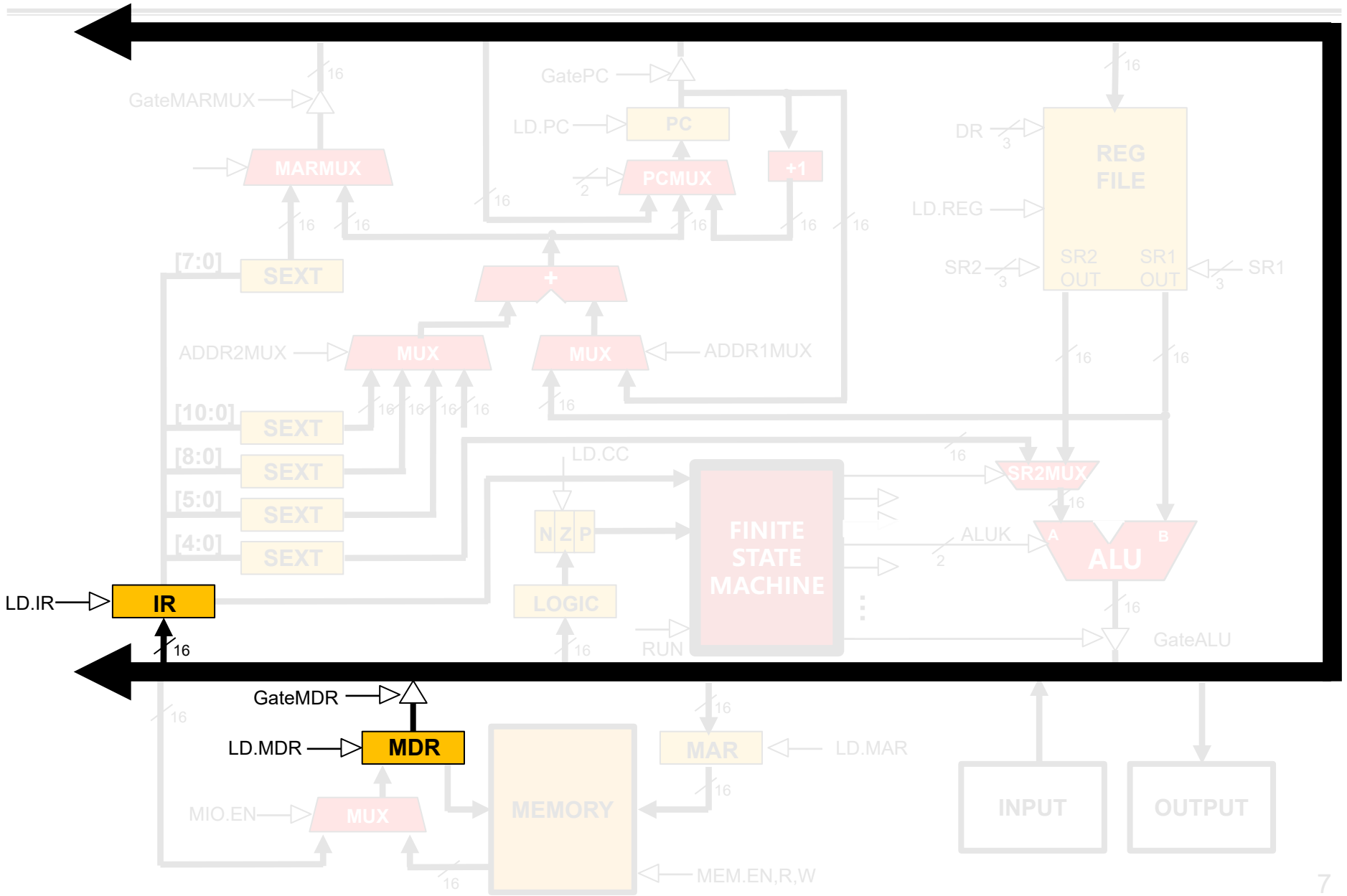
BR (PC-Relative)



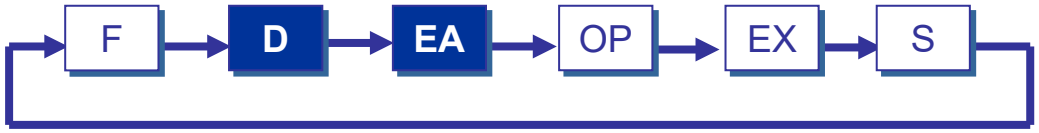


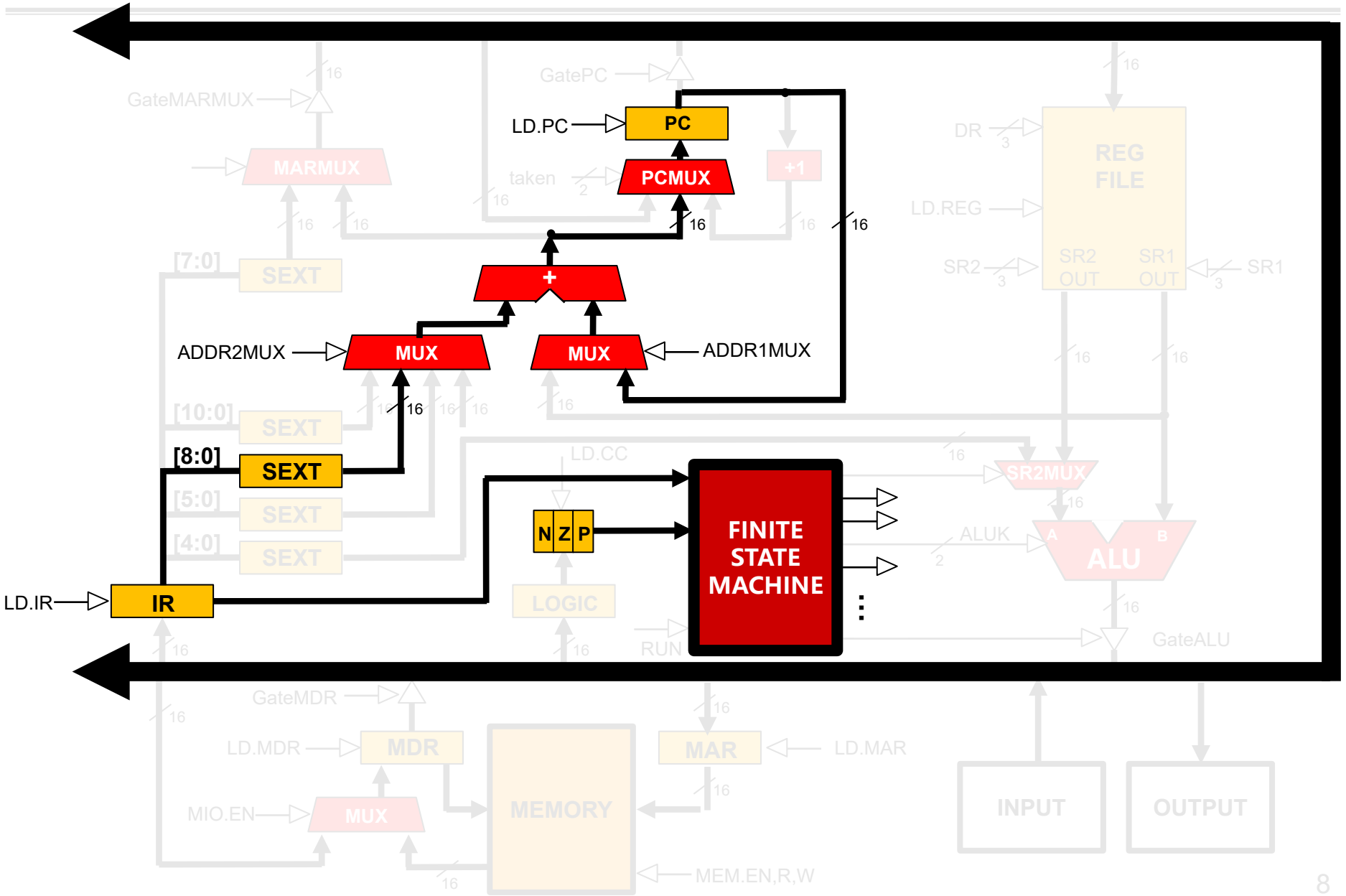
BR (PC-Relative)



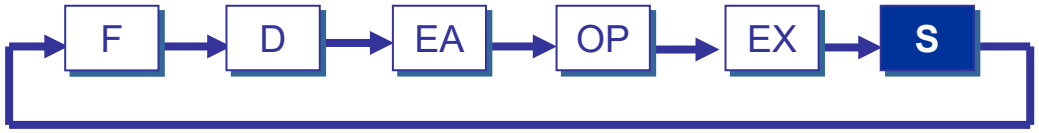


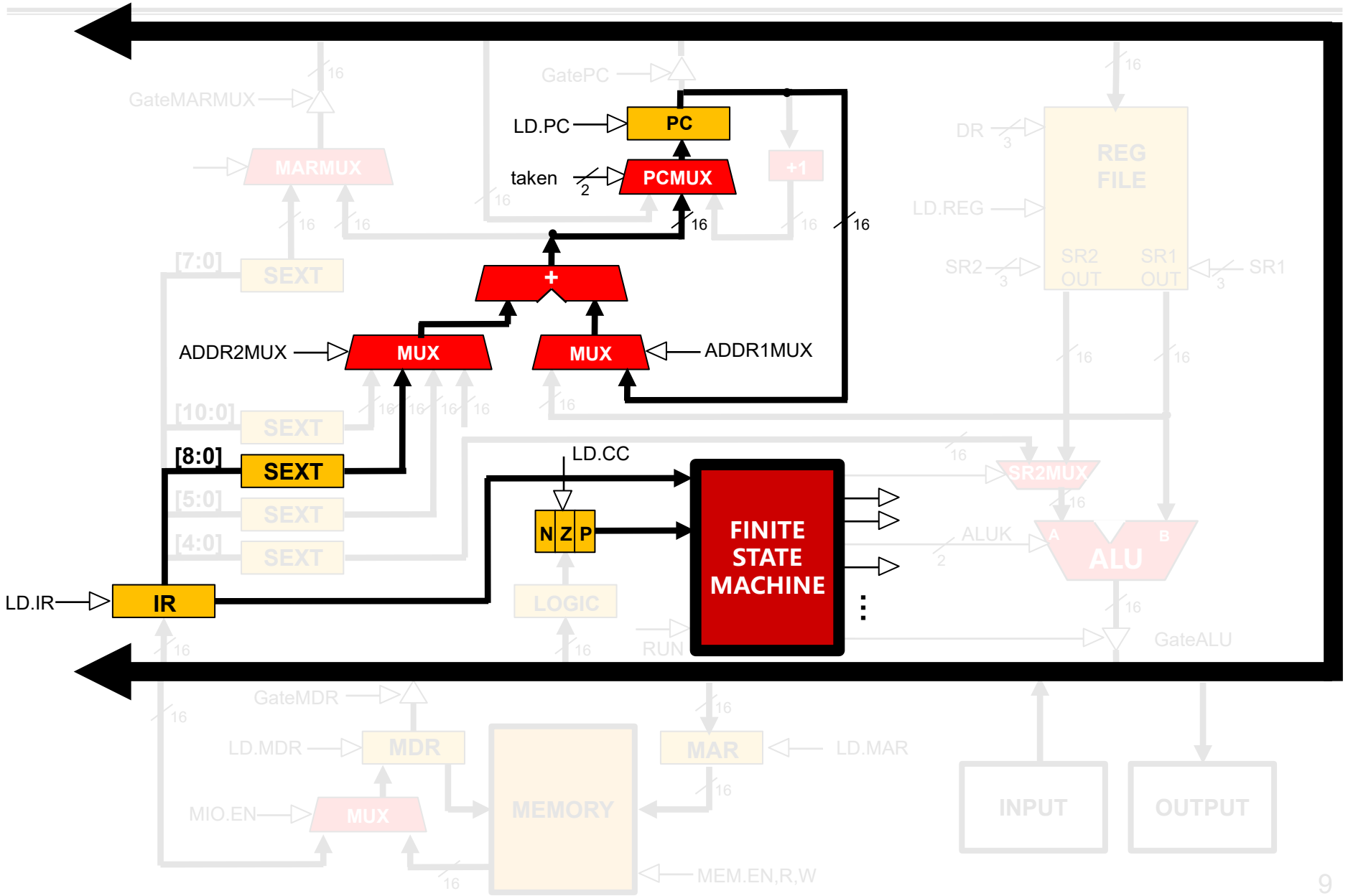
BR (PC-Relative)





BR (PC-Relative)





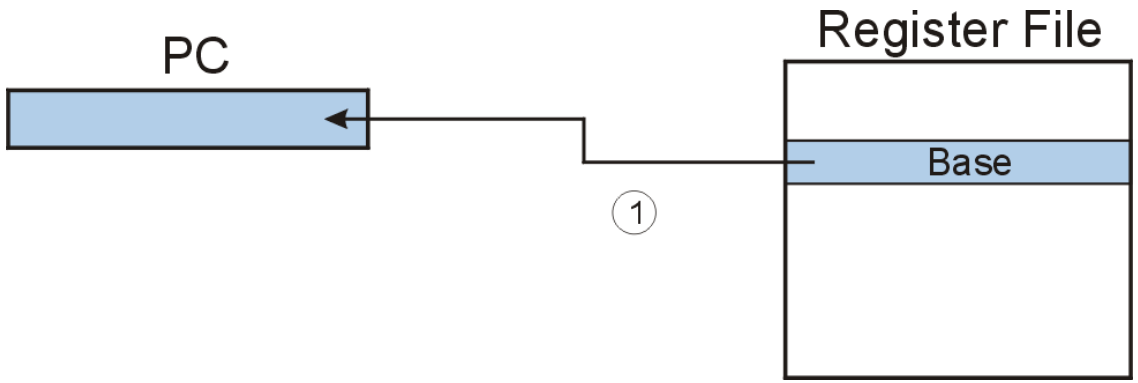
BR (PC-Relative)

- **Check**
  - BR_{nzp} x4101 ; if (n=1 or z=1 or p=1)，JMP x4101
  - BR_{n} x4101 ; if (n=1)
  - BR_{z} x4101 ; if (z=1)
  - BR_{p} x4101 ; if (p=1)
  - BR_{nz} x4101 ; if (n=1 or z=1)
  - BR_{np} x4101 ; if (n=1 or p=1)
  - BR_{zp} x4101 ; if (z=1 or p=1)
  - BR x4101 ; PC=PC+1
- **Set**
  - If DR < 0, set N=1 and Z=0 and P=0
  - If DR = 0, set N=0 and Z=1 and P=0
  - If DR > 0, set N=0 and Z=0 and P=1

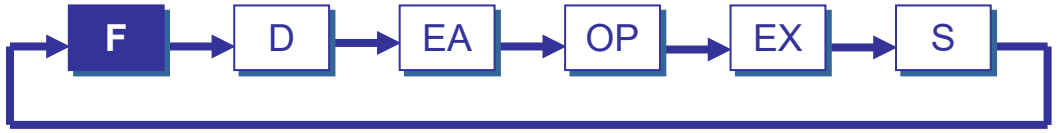
JMP (Register)

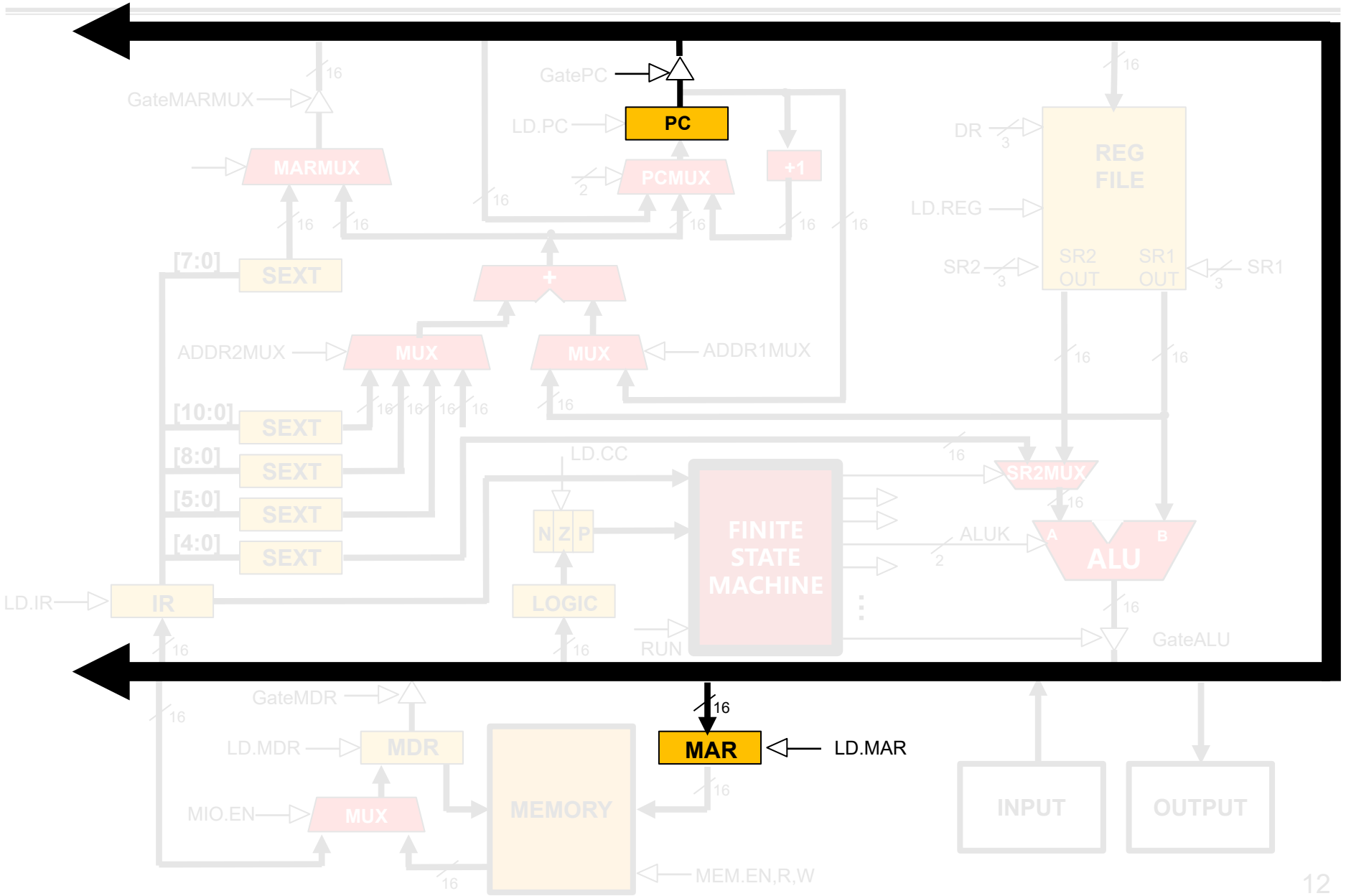
Jump is an unconditional branch -- *always* taken.

- **Target address is the contents of a register.**
- **Allows any target address.**

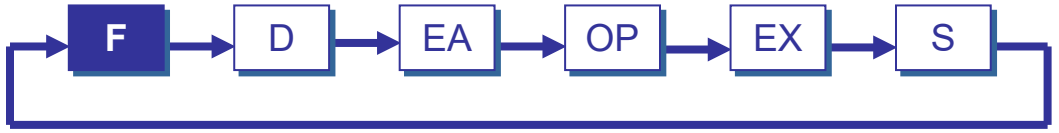


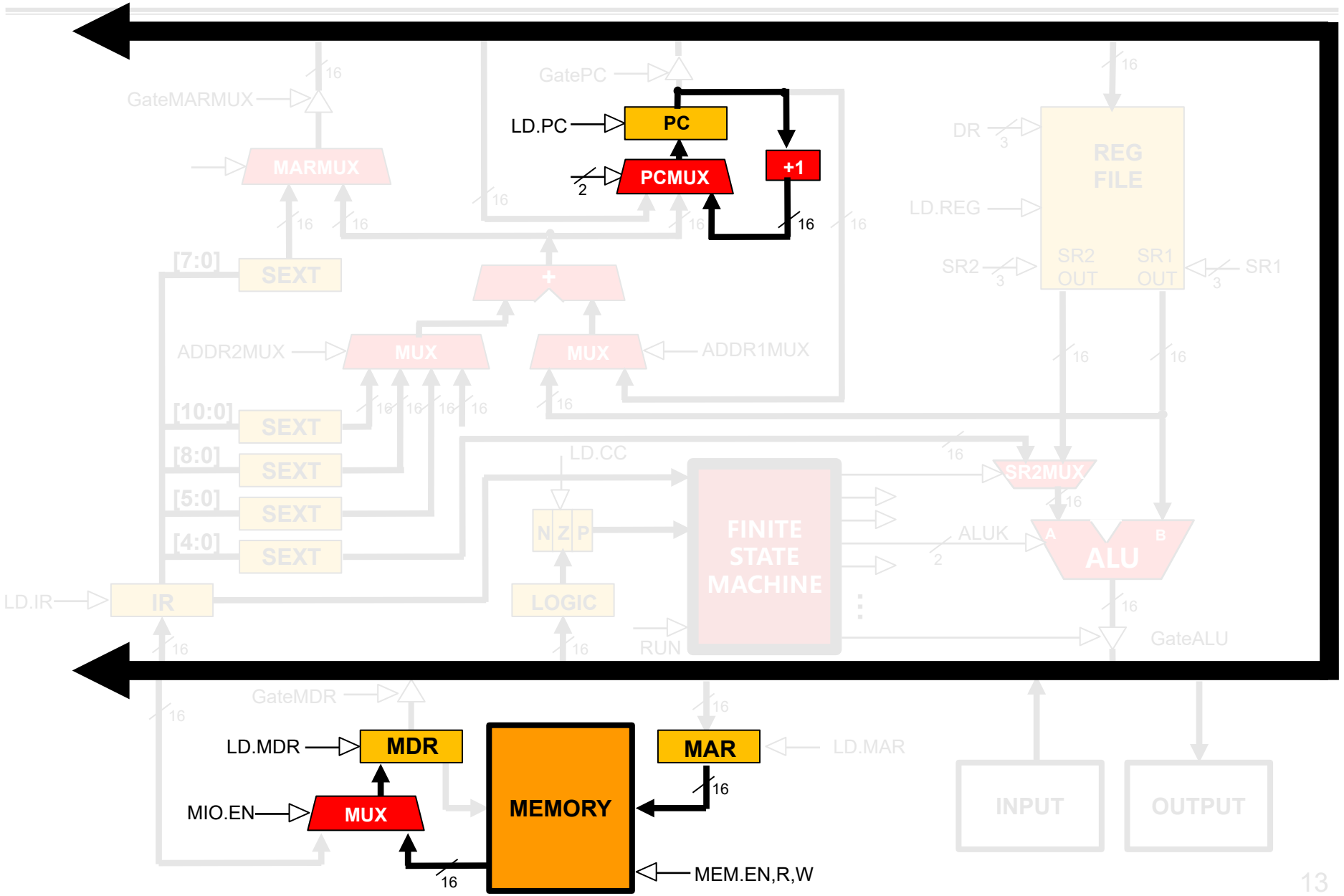
JMP (Register)



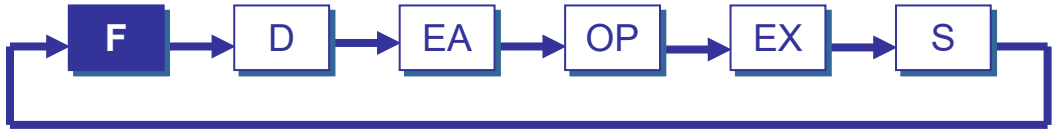


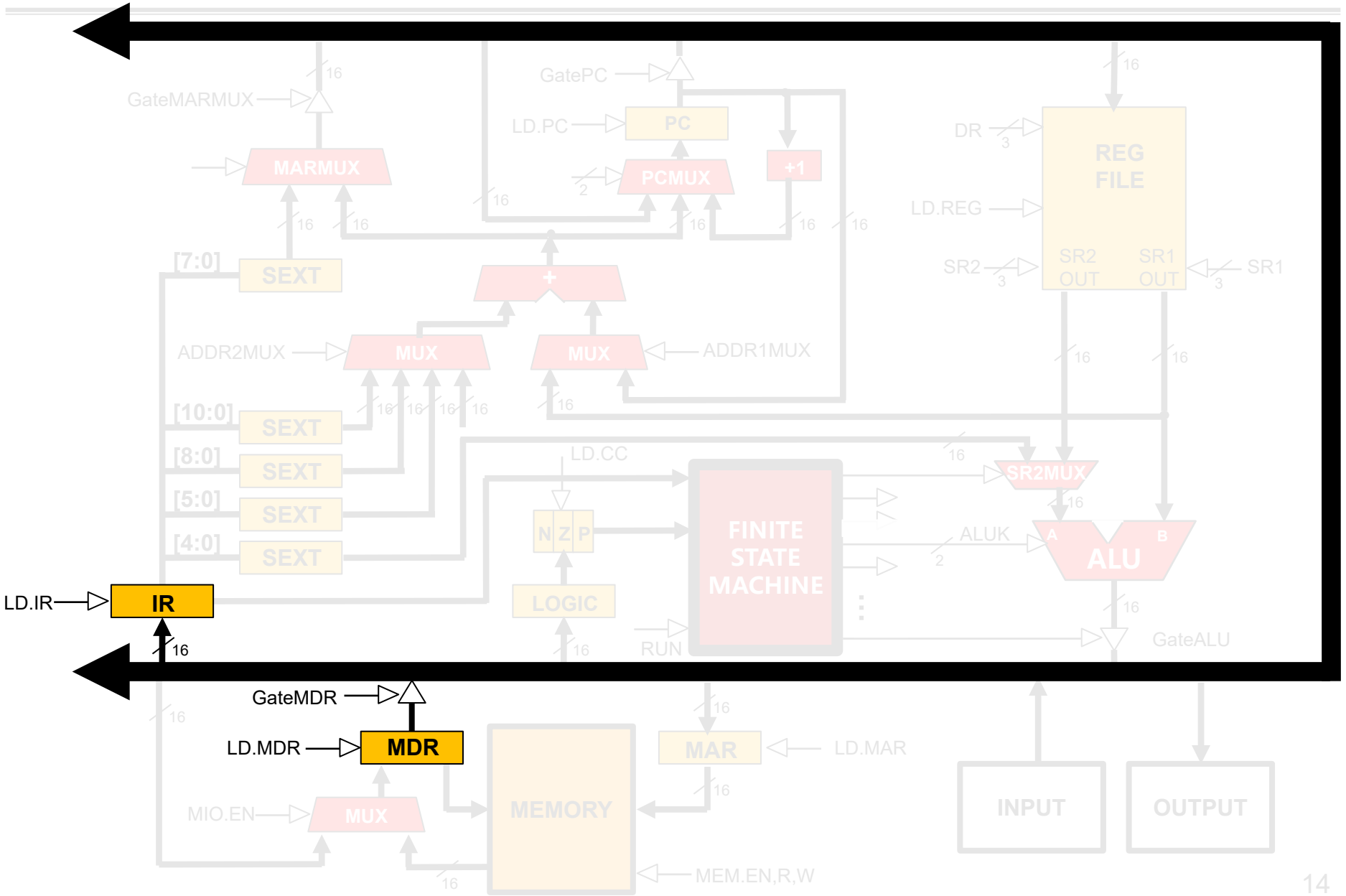
JMP (Register)



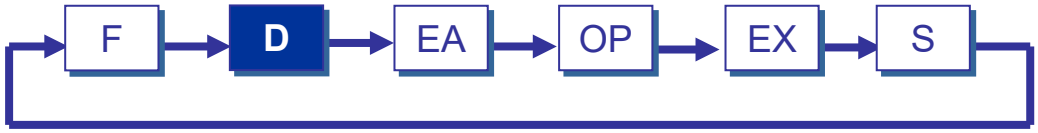


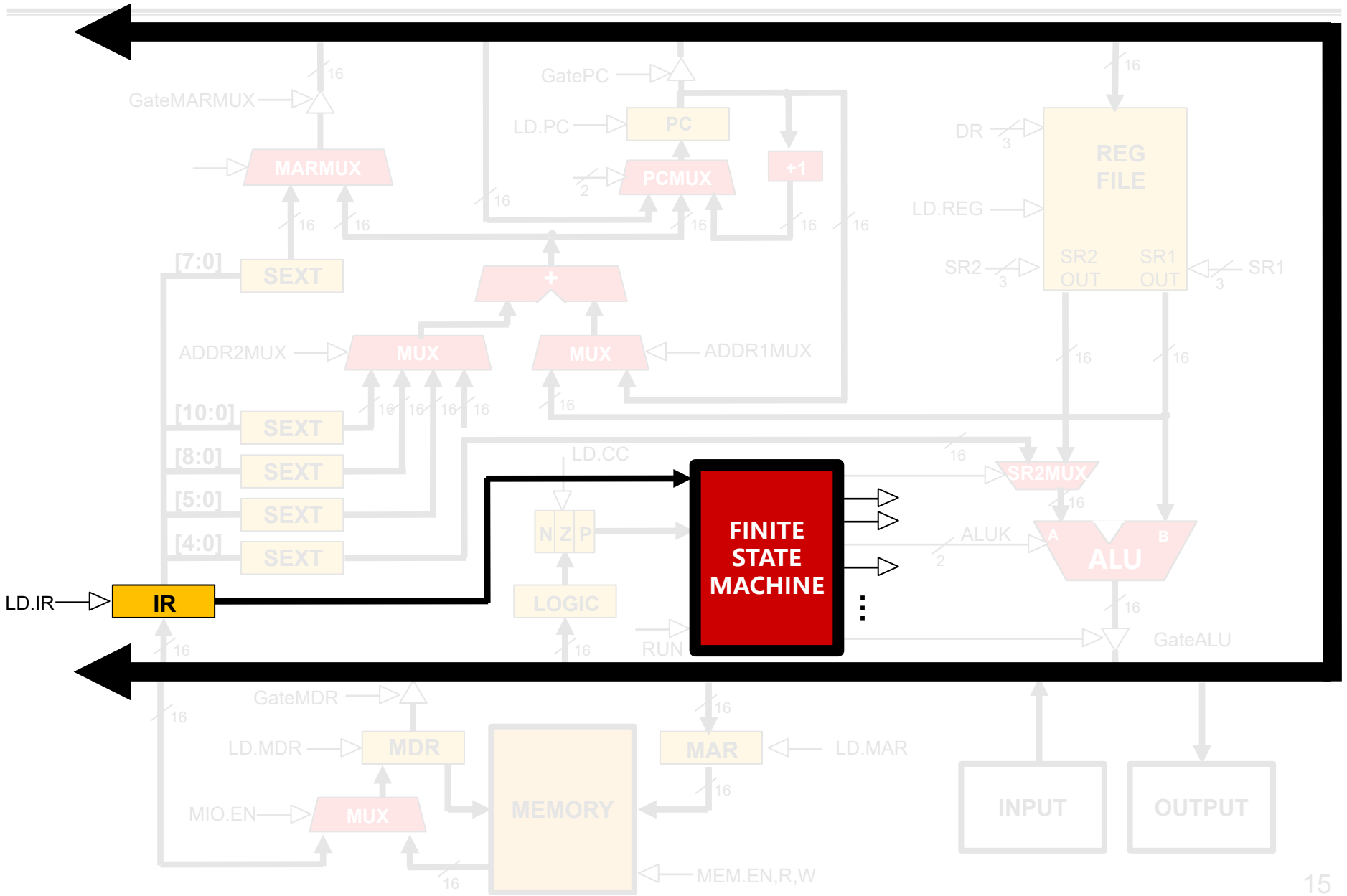
JMP (Register)



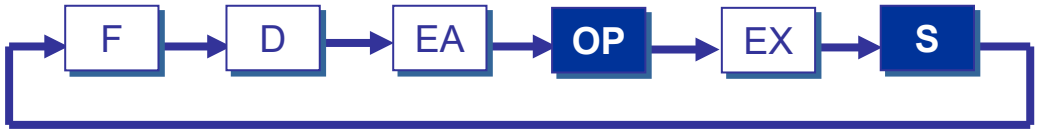


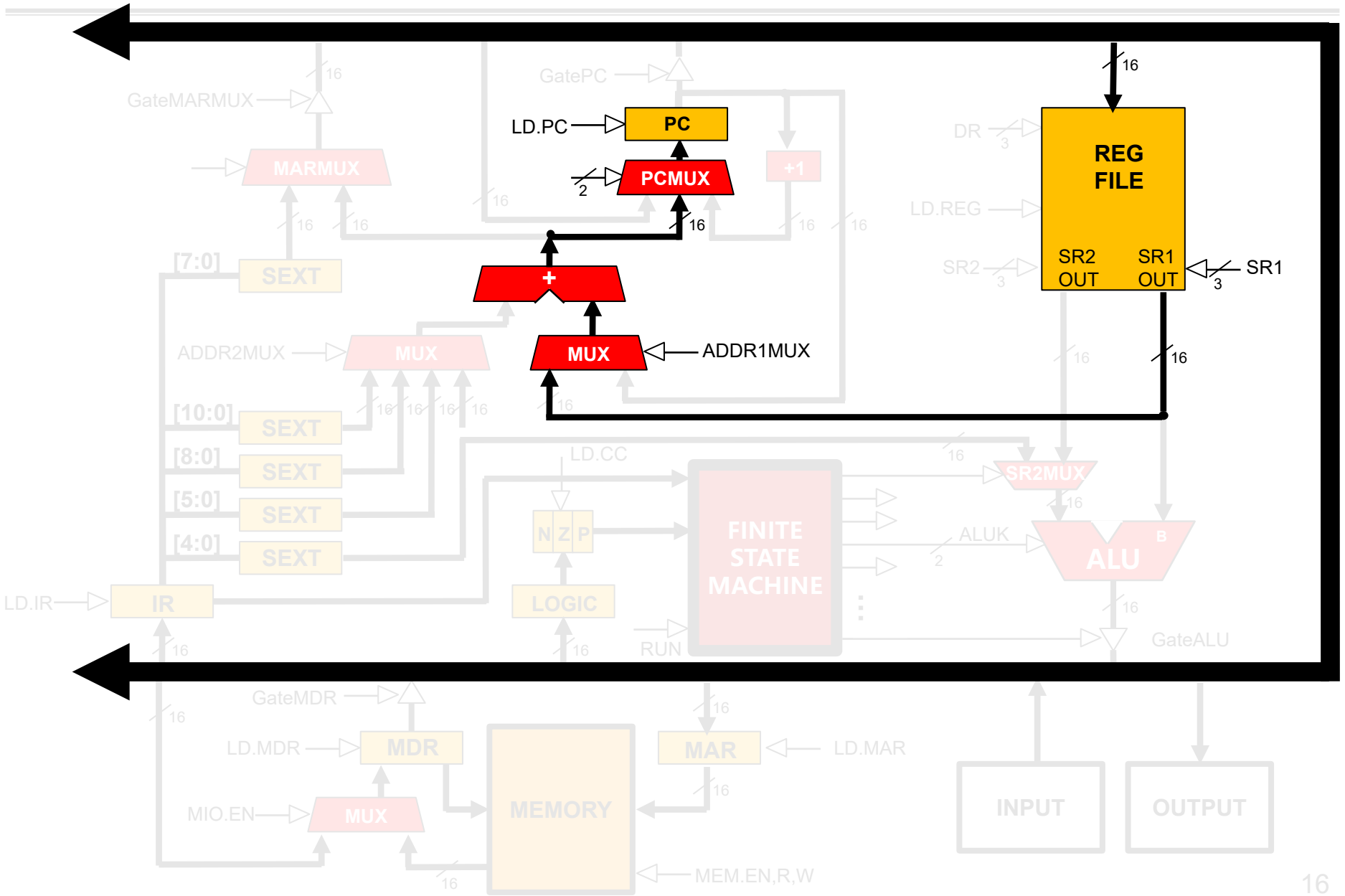
JMP R7(Register)





JMP R7(Register)





TRAP

Calls a service routine, identified by 8-bit "trap vector."

vector	*routine*
x23	input a character from the keyboard
x21	output a character to the monitor
x25	halt the program

Example:

TRAP x23

; Directs the operating system to execute the IN system call.

; The starting address of this system call is contained in memory location x0023.

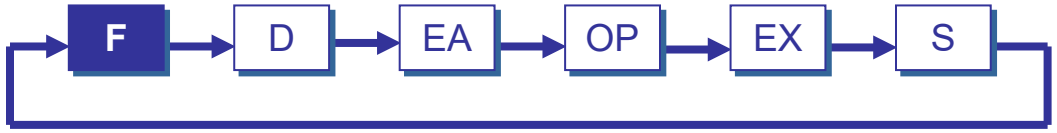
TRAP

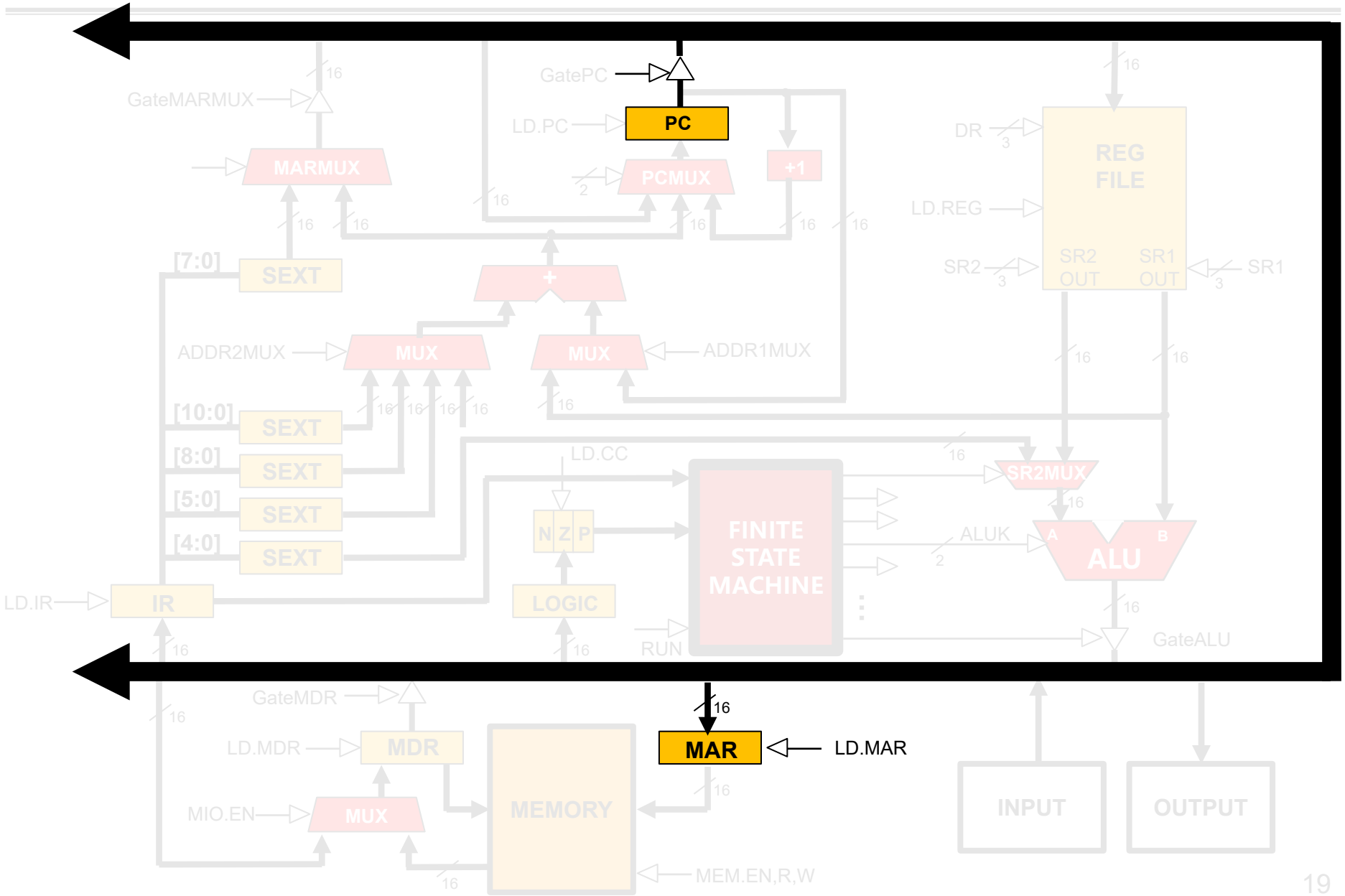
Calls a service routine, identified by 8-bit "trap vector."

vector	*routine*
x23	input a character from the keyboard
x21	output a character to the monitor
x25	halt the program

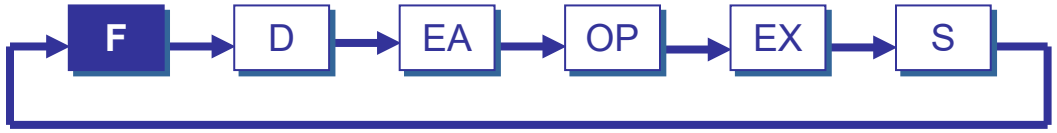
When routine is done,
PC is set to the instruction following TRAP.
(We'll talk about how this works later.)

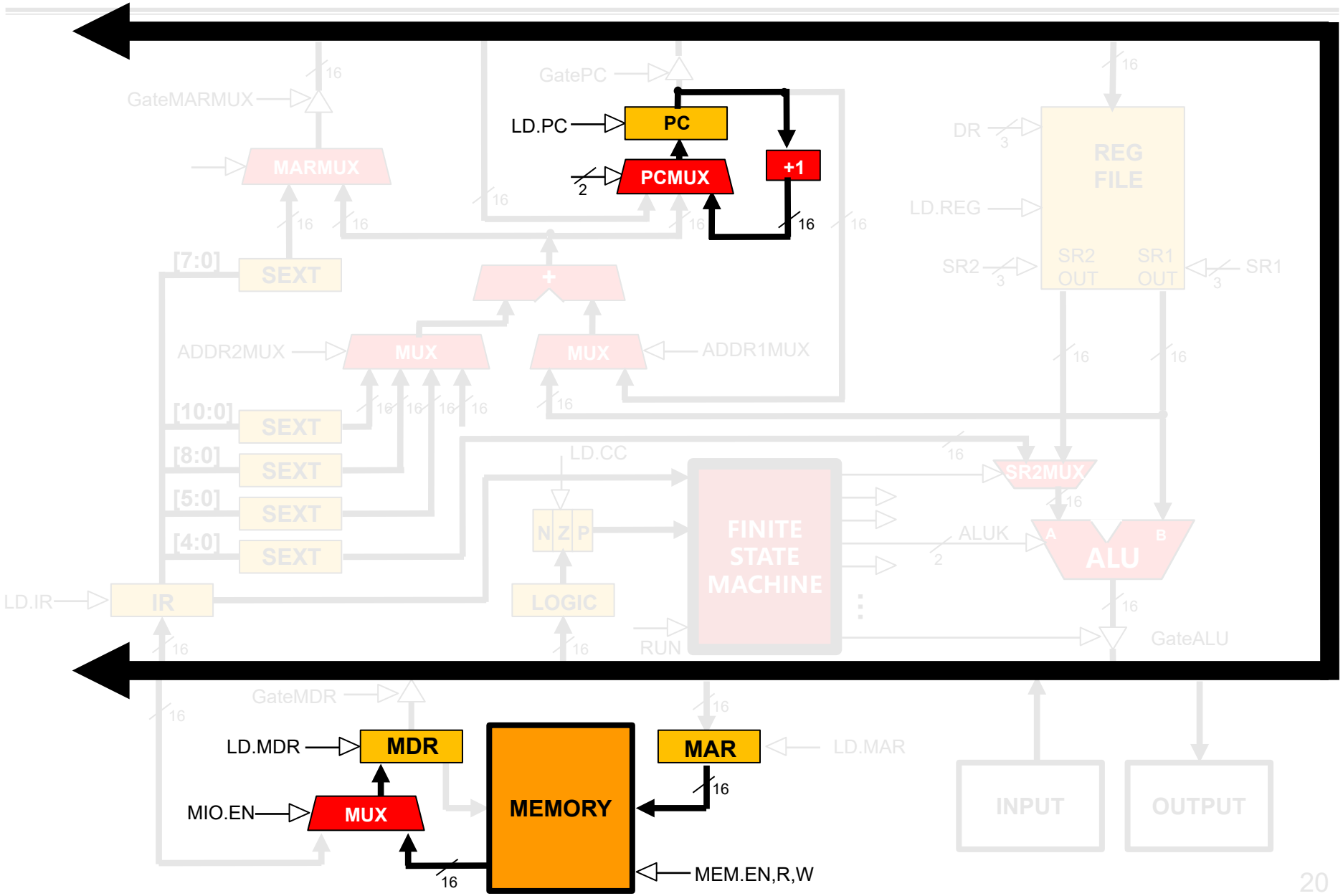
TRAP


F
D
EA
OP
EX
S

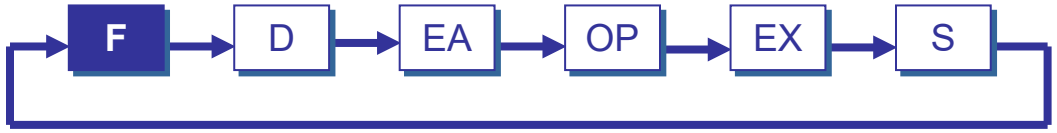

GatePC
LD.PC
PC
GateMARMUX
MARMUX
PCMUX
+1
+
ADDR2MUX
MUX
MUX
ADDR1MUX
[7:0]
SEXT
[10:0]
SEXT
[8:0]
SEXT
[5:0]
SEXT
[4:0]
SEXT
LD.IR
IR
LD.CC
N Z P
LOGIC
RUN
FINITE STATE MACHINE
DR
REG FILE
LD.REG
SR2
SR2 OUT
SR1 OUT
SR1
SR2MUX
ALUK
A
B
ALU
GateALU
GateMDR
LD.MDR
MDR
MIO.EN
MUX
MEMORY
MEM.EN,R,W
MAR
LD.MAR
INPUT
OUTPUT
16
2
3

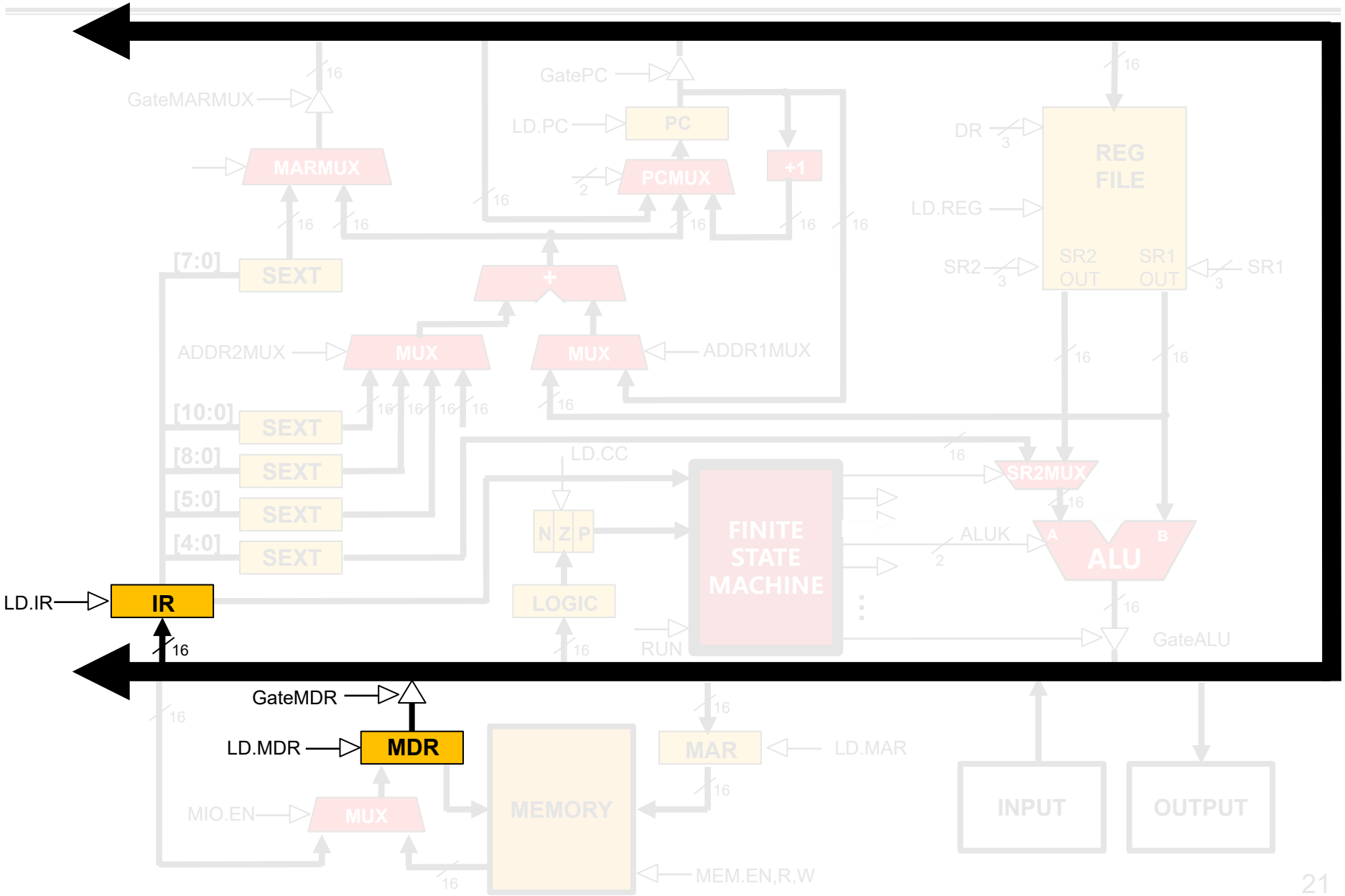
TRAP


F
D
EA
OP
EX
S

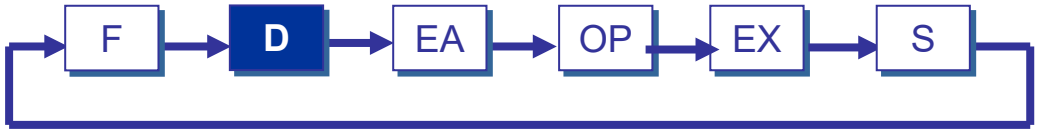

GatePC
LD.PC
PC
PCMUX
+1
16
2
GateMARMUX
MARMUX
16
[7:0]
SEXT
+
ADDR2MUX
MUX
MUX
ADDR1MUX
[10:0]
SEXT
[8:0]
SEXT
[5:0]
SEXT
[4:0]
SEXT
LD.CC
N Z P
FINITE STATE MACHINE
LD.IR
IR
LOGIC
RUN
DR
REG FILE
LD.REG
SR2
SR2 OUT
SR1 OUT
SR1
3
SR2MUX
ALUK
A
B
ALU
GateALU
GateMDR
LD.MDR
MDR
MIO.EN
MUX
MEMORY
MAR
LD.MAR
MEM.EN,R,W
INPUT
OUTPUT

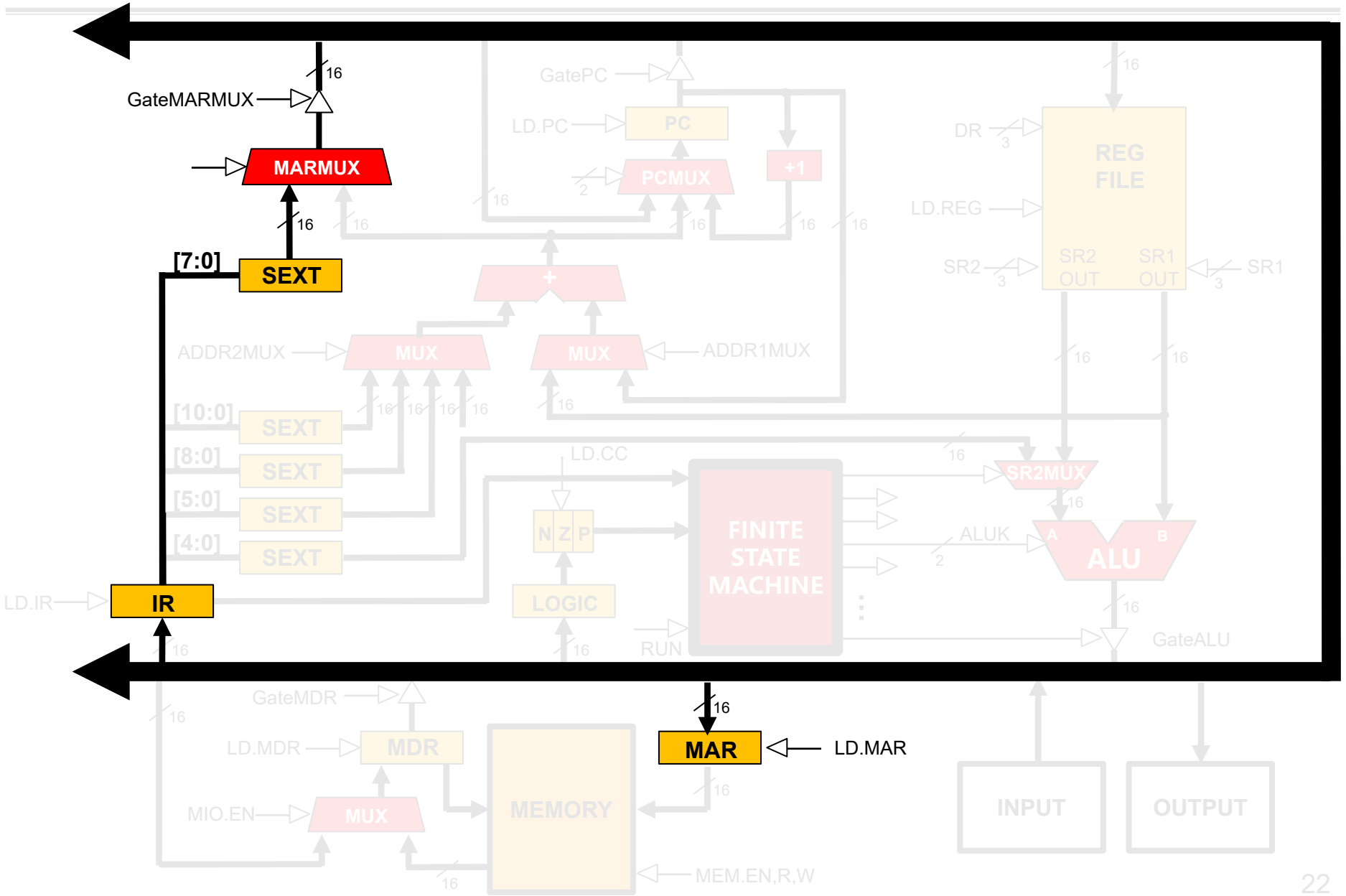
TRAP


F
D
EA
OP
EX
S

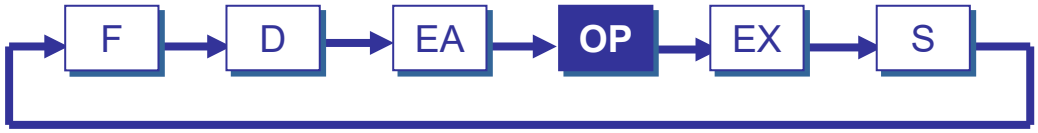

GateMARMUX
GatePC
LD.PC
PC
MARMUX
PCMUX
+1
DR
REG FILE
LD.REG
SR2
SR2 OUT
SR1 OUT
SR1
[7:0]
SEXT
+
ADDR2MUX
MUX
MUX
ADDR1MUX
[10:0]
[8:0]
[5:0]
[4:0]
LD.CC
N Z P
FINITE STATE MACHINE
SR2MUX
ALUK
A
B
ALU
LD.IR
IR
LOGIC
RUN
GateALU
GateMDR
LD.MDR
MDR
MAR
LD.MAR
MIO.EN
MEMORY
INPUT
OUTPUT
MEM.EN,R,W

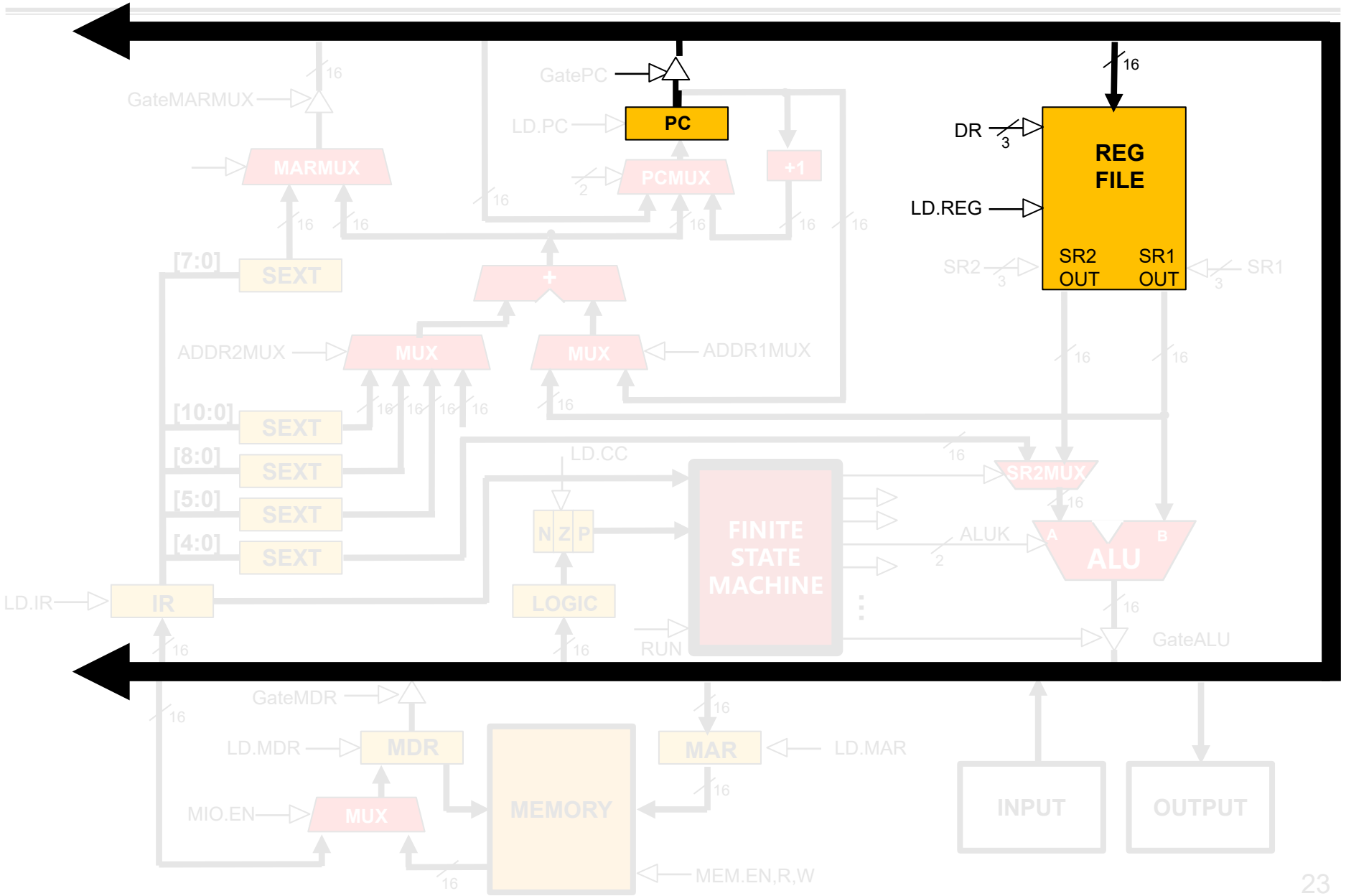
TRAP


F
D
EA
OP
EX
S

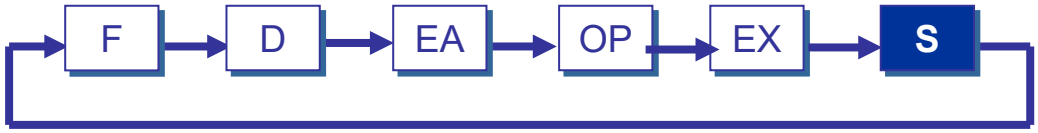

16
GateMARMUX
MARMUX
16
[7:0]
SEXT
GatePC
LD.PC
PC
PCMUX
+1
2
16
+
ADDR2MUX
MUX
MUX
ADDR1MUX
[10:0]
SEXT
[8:0]
SEXT
[5:0]
SEXT
[4:0]
SEXT
LD.CC
N Z P
FINITE STATE MACHINE
LOGIC
RUN
LD.IR
IR
DR
3
REG FILE
LD.REG
SR2
SR2 OUT
SR1 OUT
SR1
SR2MUX
ALUK
2
A
B
ALU
GateALU
GateMDR
LD.MDR
MDR
MIO.EN
MUX
MEMORY
MEM.EN,R,W
MAR
LD.MAR
INPUT
OUTPUT

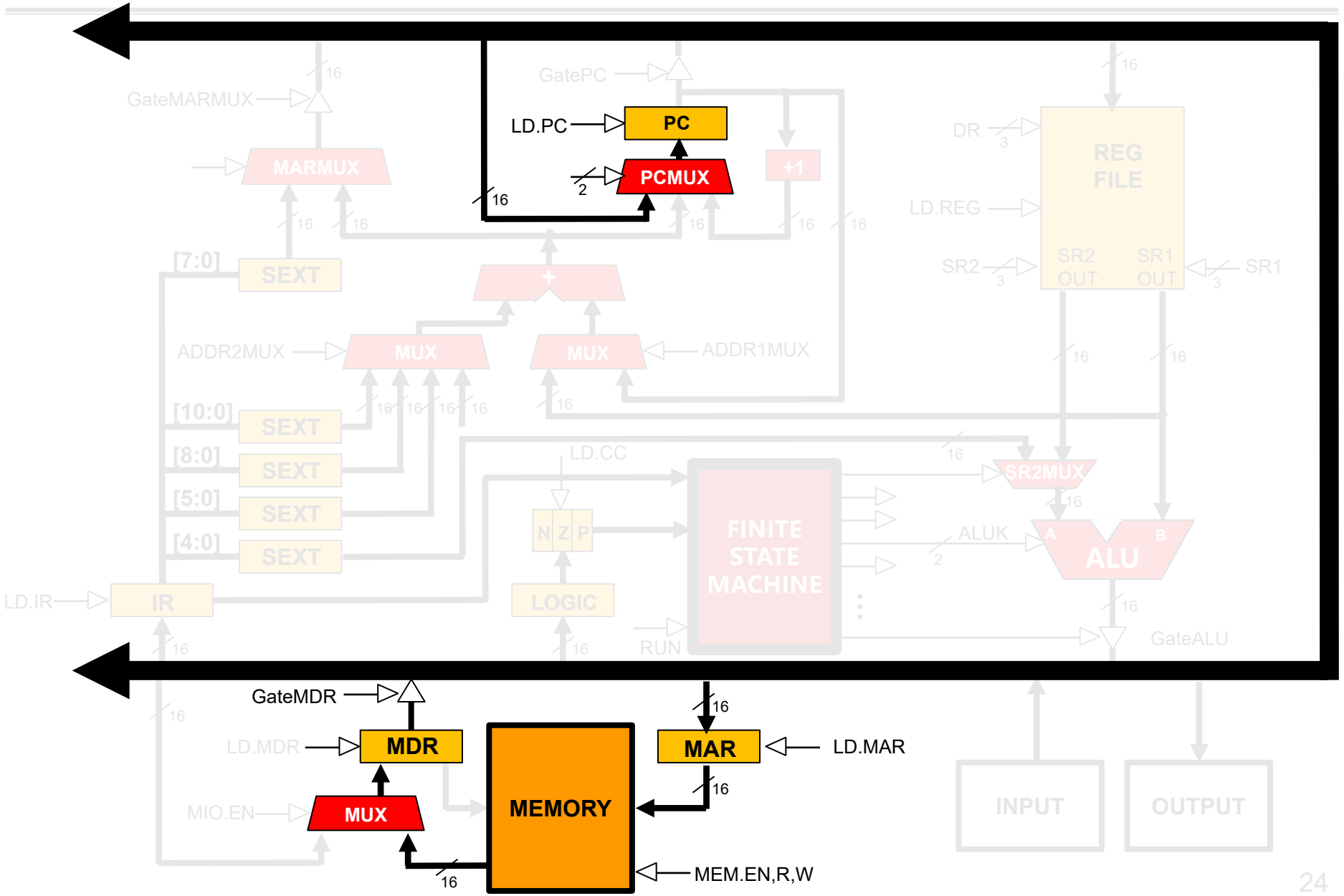
TRAP


F
D
EA
OP
EX
S

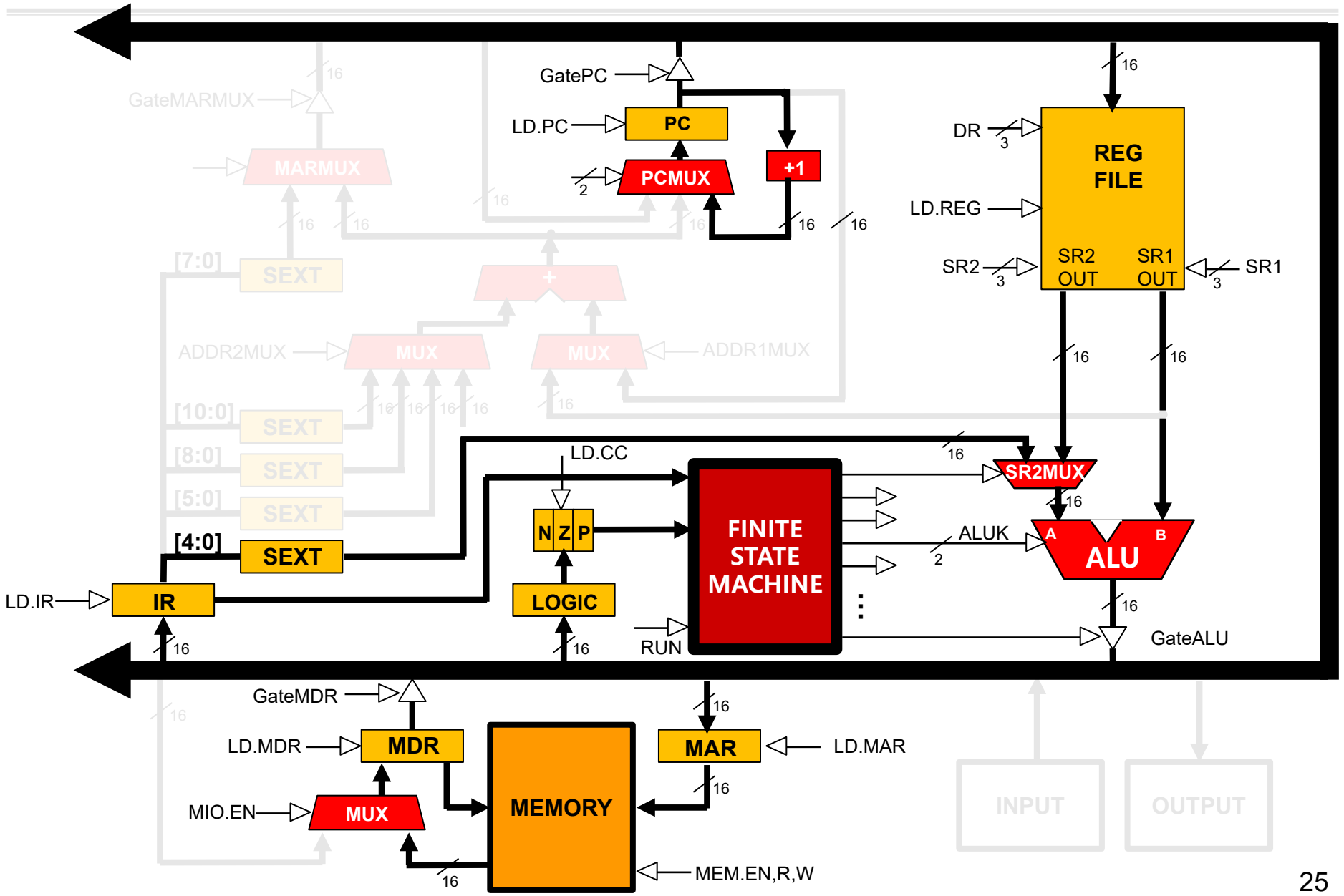

GatePC
LD.PC
PC
PCMUX
+1
GateMARMUX
MARMUX
16
2
3
DR
REG FILE
LD.REG
SR2 OUT
SR1 OUT
SR2
SR1
[7:0]
SEXT
+
ADDR2MUX
MUX
ADDR1MUX
[10:0]
[8:0]
[5:0]
[4:0]
LD.CC
N Z P
FINITE STATE MACHINE
SR2MUX
ALUK
A
B
ALU
LD.IR
IR
LOGIC
RUN
GateALU
GateMDR
LD.MDR
MDR
MIO.EN
MEMORY
MAR
LD.MAR
MEM.EN,R,W
INPUT
OUTPUT

TRAP

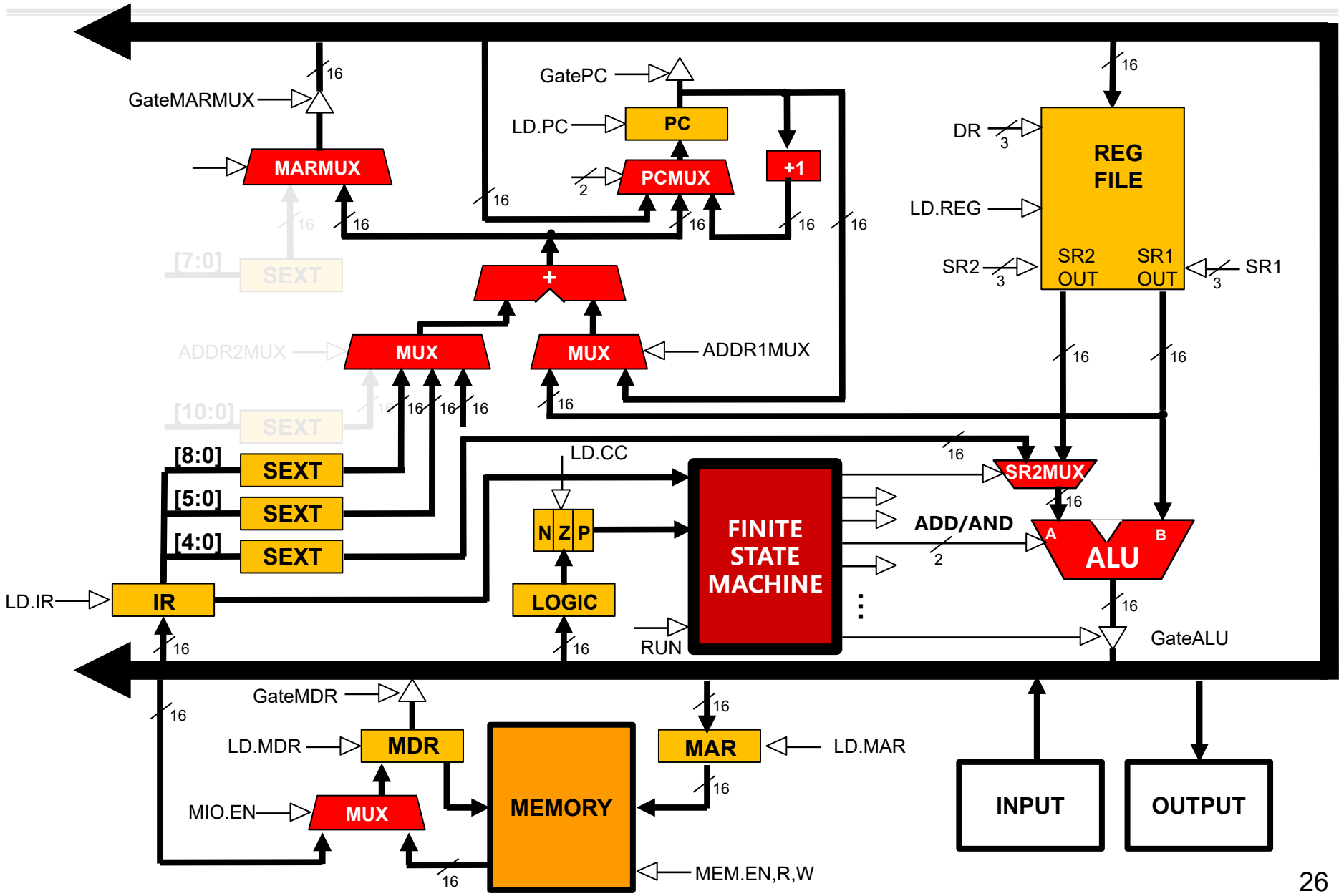

F
D
EA
OP
EX
S


GatePC
LD.PC
PC
PCMUX
+1
16
2
GateMARMUX
MARMUX
DR
REG FILE
LD.REG
SR2
SR2 OUT
SR1 OUT
SR1
3
[7:0]
SEXT
+
ADDR2MUX
MUX
ADDR1MUX
[10:0]
[8:0]
[5:0]
[4:0]
LD.CC
N Z P
FINITE STATE MACHINE
SR2MUX
ALUK
A
B
ALU
LD.IR
IR
LOGIC
RUN
GateALU
GateMDR
LD.MDR
MDR
MIO.EN
MUX
MEMORY
MAR
LD.MAR
MEM.EN,R,W
INPUT
OUTPUT

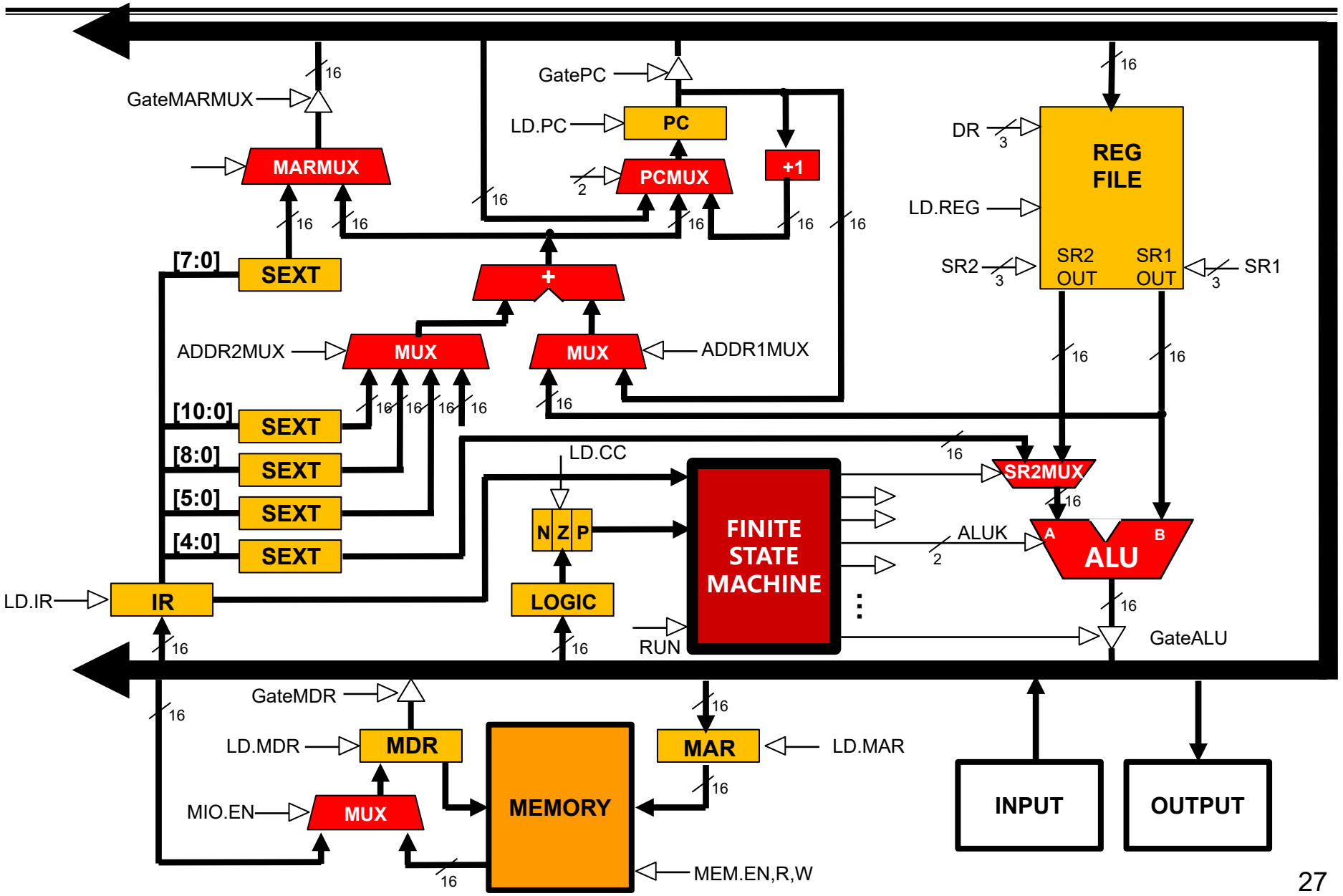
LC-3 Data Path After Operate Instruction



LC-3 Data Path After Load/Store Instruction



LC-3 Data Path After Control Instruction



LC-3 Data Path