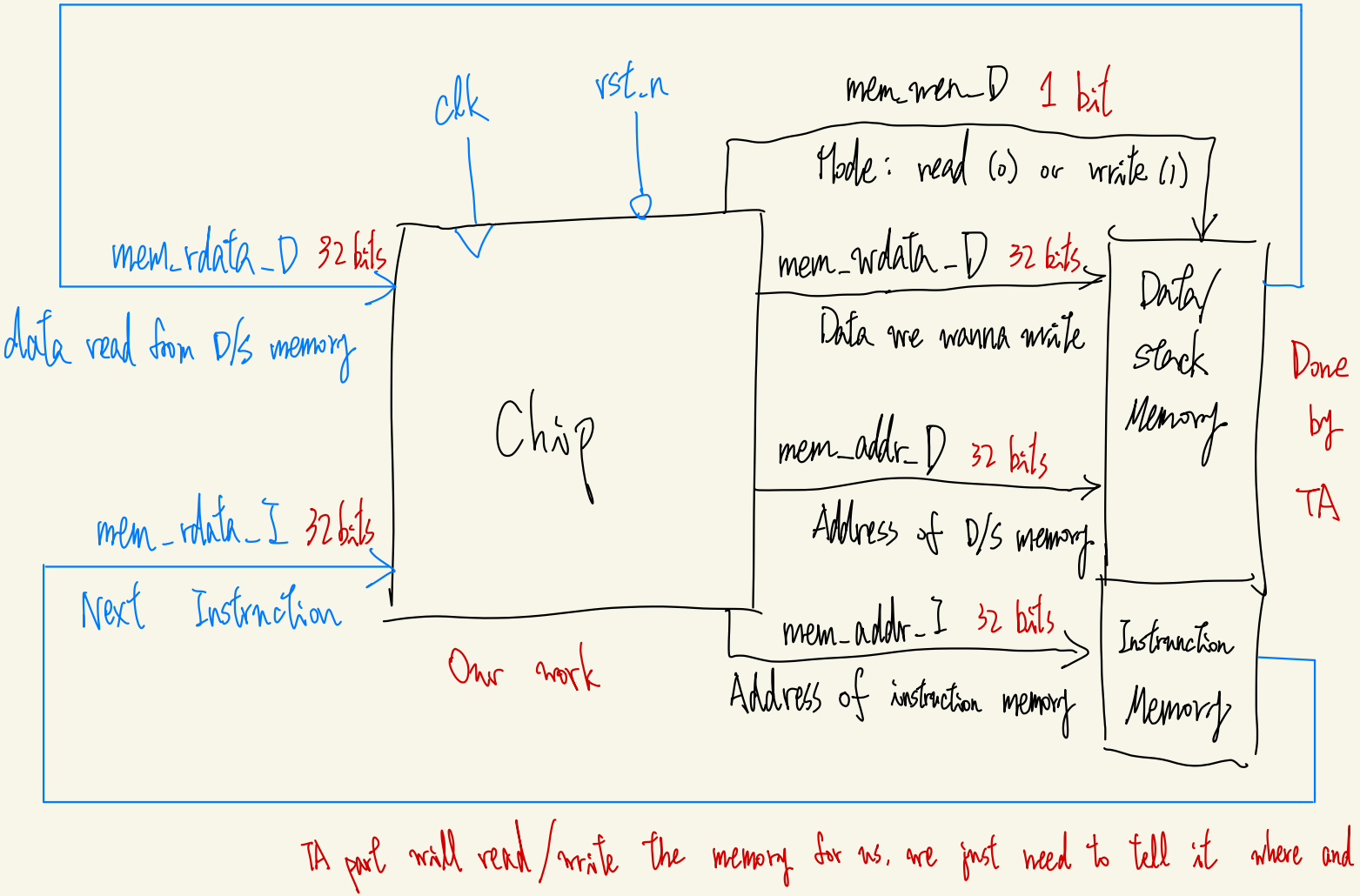
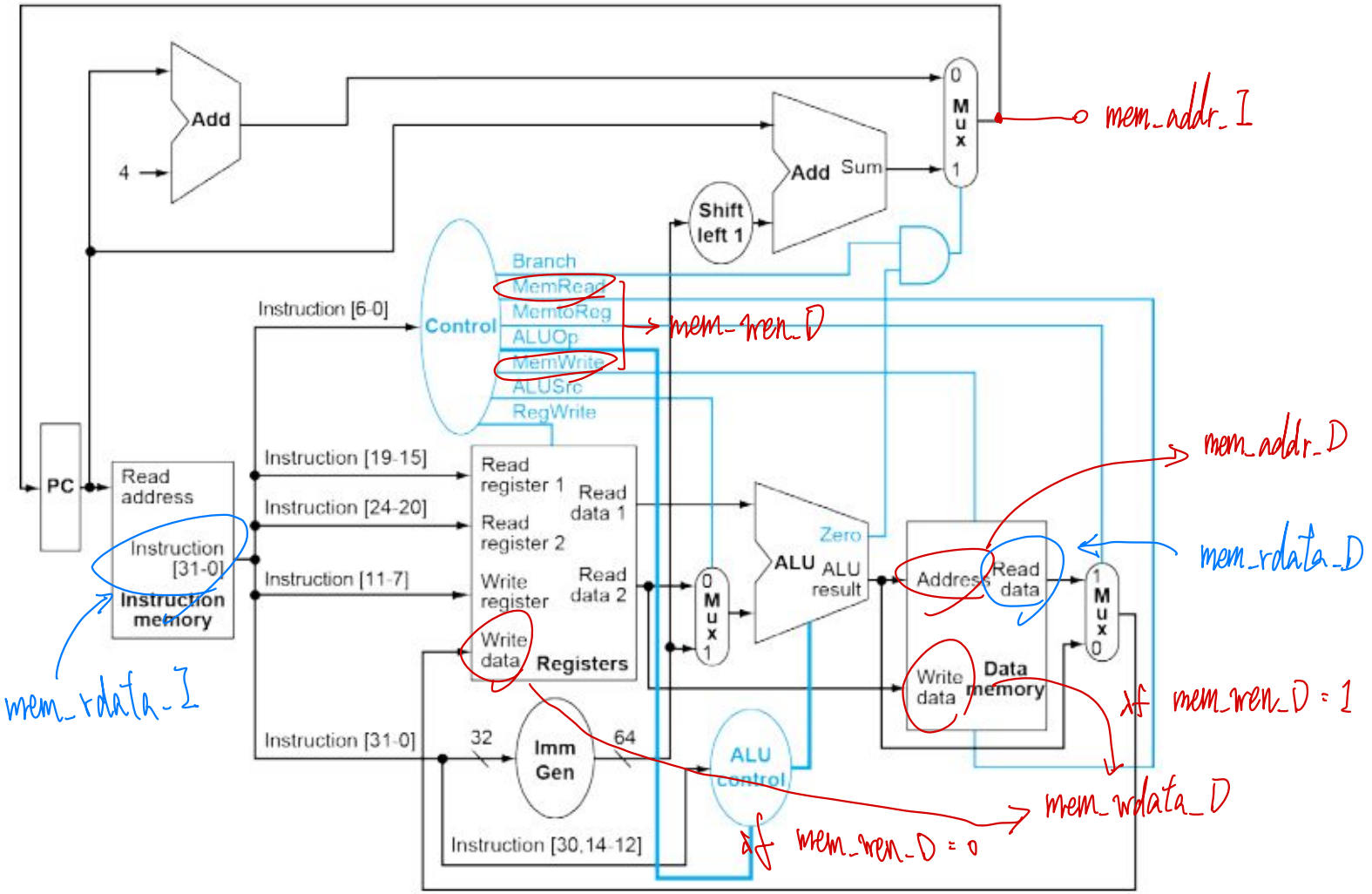


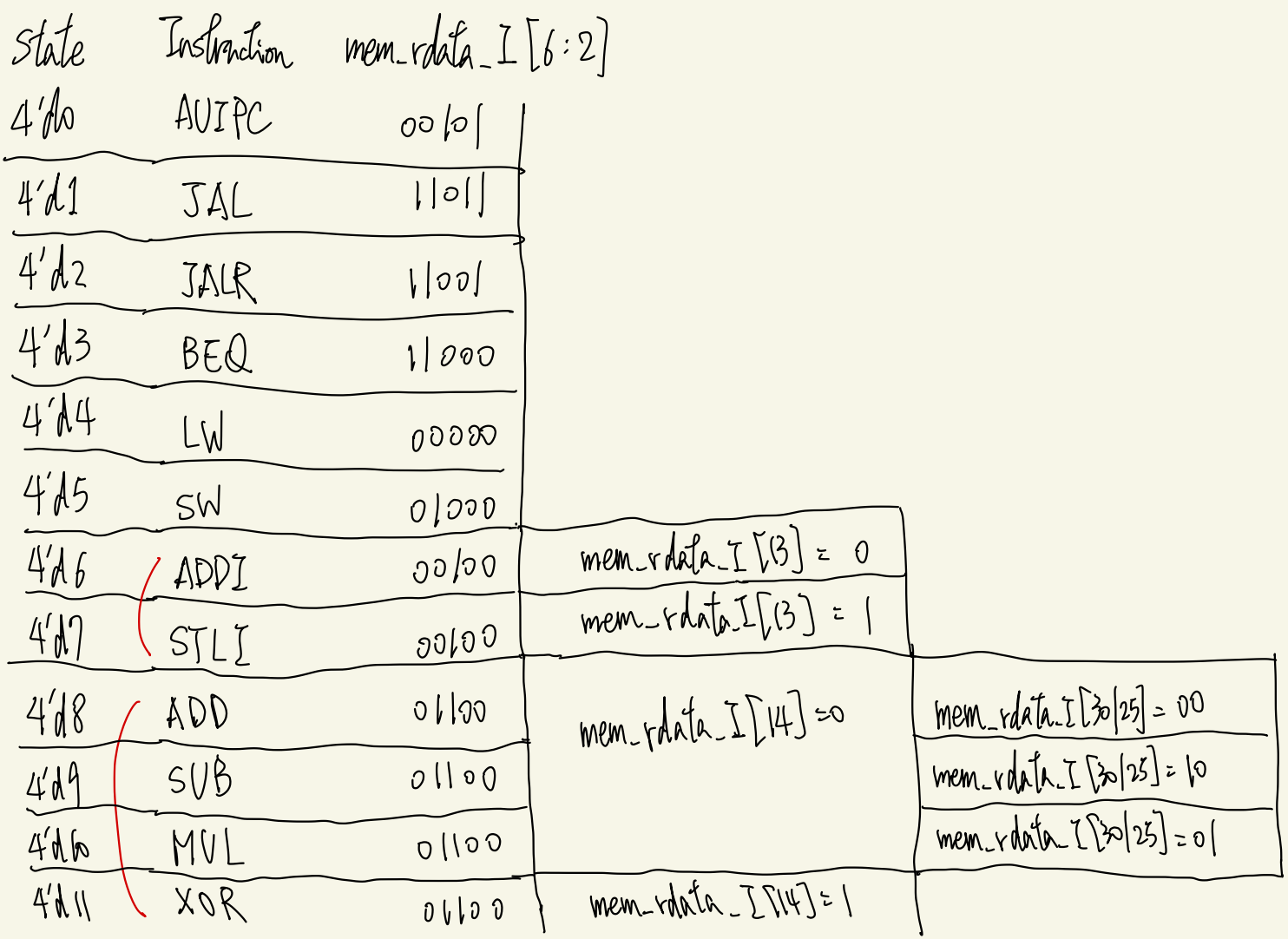
TA part will read/write the memory for us, we just need to tell it where and what.

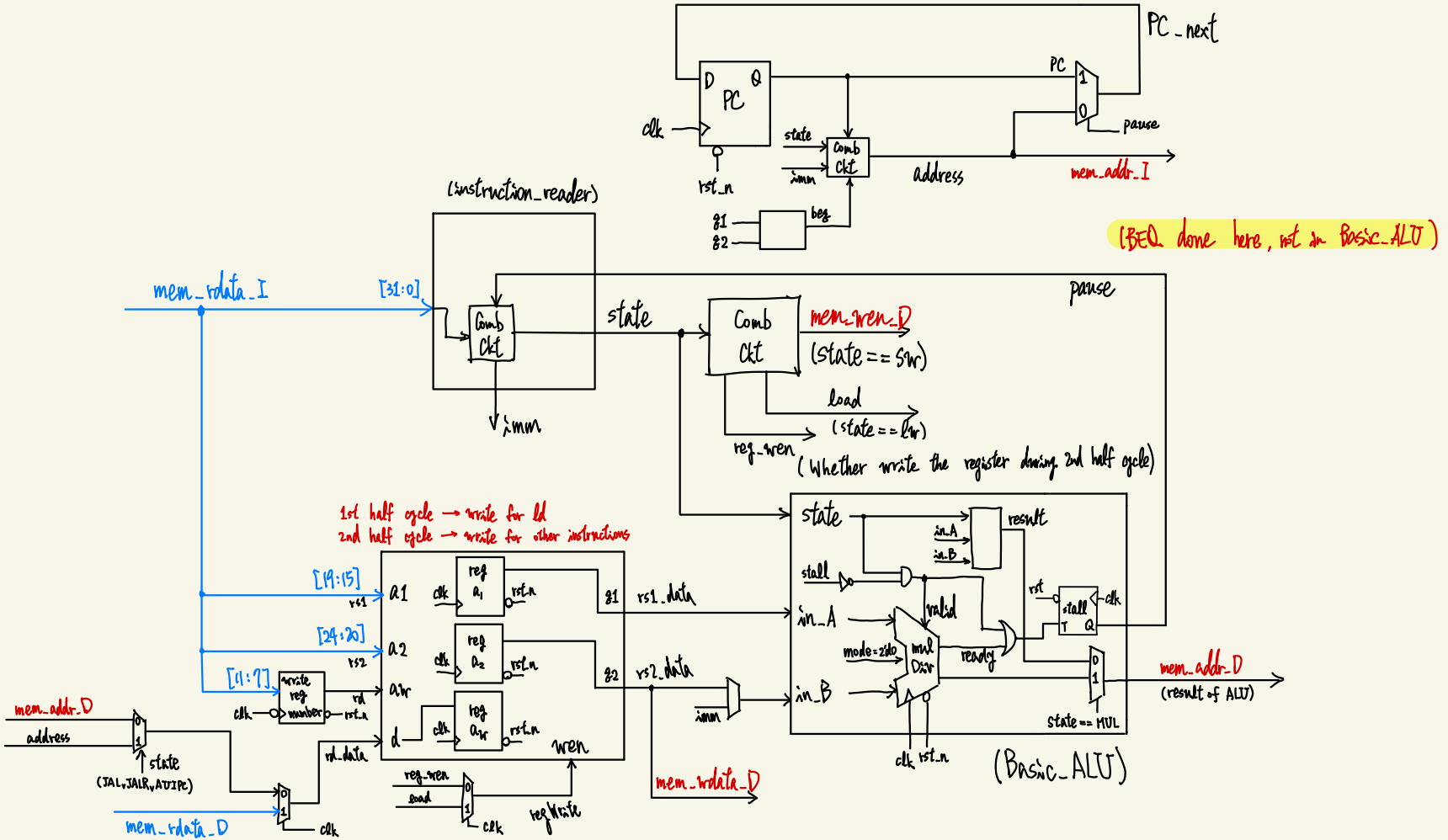
Architecture

- Not complete (does not include jal, jalr, …)



mem_wen_D


State
Instruction
mem_rdata_I[6:2]
4'd0
AUIPC
00101
4'd1
JAL
11011
4'd2
JALR
11001
4'd3
BEQ
11000
4'd4
LW
00000
4'd5
SW
01000
4'd6
ADDI
00100
4'd7
STLI
00100
4'd8
ADD
01100
4'd9
SUB
01100
4'd10
MUL
01100
4'd11
XOR
01100
mem_rdata_I[3] = 0
mem_rdata_I[3] = 1
mem_rdata_I[14] = 0
mem_rdata_I[14] = 1
mem_rdata_I[30|25] = 00
mem_rdata_I[30|25] = 10
mem_rdata_I[30|25] = 01


PC_next
PC
D
Q
clk
rst_n
state
imm
Comb Ckt
address
mem_addr_I
pause
beq
g1
g2
(BEQ done here, rst in Basic_ALU)
(instruction_reader)
mem_rdata_I
[31:0]
Comb Ckt
state
imm
Comb Ckt
mem_wen_D
(state == SW)
load
(state == LW)
reg_wen
(Whether write the register during 2nd half cycle)
1st half cycle → write for ld
2nd half cycle → write for other instructions
[19:15]
rs1
[24:20]
rs2
[11:7]
write reg number
rd
a1
a2
a3r
d
reg a1
reg a2
reg a3r
rst_n
g1
g2
rs1_data
rs2_data
wen
reg_wen
load
regWrite
mem_addr_D
address
state
(JAL,JALR,AUIPC)
rd_data
mem_rdata_D
clk
mem_wdata_D
state
in_A
in_B
result
stall
valid
rst
mode=2'b0
mul
Div
ready
mem_addr_D
(result of ALU)
state == MUL
clk
rst_n
(Basic_ALU)