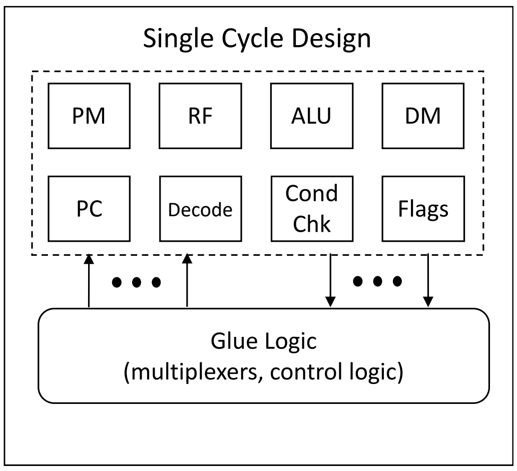
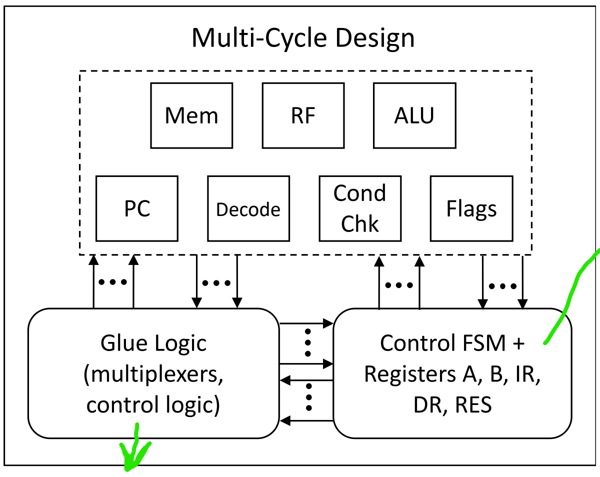
COL216 Computer Architecture

Lab Assignment 2, Stage 3: Multi-cycle processor design

This stage involves modifying the design of stage 2 to make it multi-cycle design. The set of instructions to be executed and the variants to be supported remain same.

The design may be based on slide 29 of Lec10. The set of components designed previously will be reused, except for the program memory. Data memory will take care of storing instructions as well as data. You may double its size. The main change will be introduction of Control FSM as part of the main body. This FSM will have 9 states as shown in slide 38 of Lec10 and will be implemented as a clock triggered process. Registers A, B, IR, DR and RES can be introduced as separate components, but it will be easier to declare these as local signals in the main body and include assignments to these in the control FSM process in appropriate states. Figures below show how the overall structure of multi-cycle design would differ from the single cycle design of stage 2.




Multi-Cycle Design
Mem
RF
ALU
PC
Decode
Cond Chk
Flags
Glue Logic (multiplexers, control logic)
Control FSM + Registers A, B, IR, DR, RES

this could be a process which is clocked bcoz registers are also clocked

glue logic will be combinational circuit with concurrent assignments

Two changes will be required in the component containing PC.

i. Logic for addition and sign extension is no longer required. ALU will provide the next address that goes into PC.
ii. A write enable signal will be required since PC will not clock in every cycle.

Is it required to modify ALU to perform PC + byte offset +8? Not necessary. First note that PC would have already incremented by 4 in the fetch state, so the constant to be added should be 4, not 8. Second, if we give word addresses and offsets, rather than byte addresses and offsets, to ALU, we neeed to do PC(31 downto 2) + word offset + 1. This can be done by ALU, specifying the operation as adc (add with carry) and making carry input as '1'. Even PC + 4 can be done in the same way, that is PC(31 downto 2) + 0 + 1. Word offset is bits 23 downto 0 of the instructions and will need sign extensionby 8 bits. PC(31 downto 2) needs to be extended by two zeros. The wordd addresses produced by ALU would need to be converted into byte addresses (logical left shift by 2 positions) before writing into PC.

This design requires more multiplexing. The glue logic would need to take care of that.

for this we can do adc
put 1 as carry_in in alu