1.) Take the single cycle processor developed in class, and add the following instructions. Note that each instruction will require a separate datapath and control logic that extends the one in class – do not do one processor that does all of the following in one machine. Instead, you will have 4 separate processors, which can do ALL the instructions from class plus one of the instructions below. Your machines should be as simple as possible. If you need new hardware units and/or wires you may add them, or change the existing units, but only as needed. Your machine must complete each of these instructions in a single clock cycle.

   a.) Show the changes to the single-cycle datapath needed to support this instruction, as well as all the ones in class.

   b.) Show the control signal settings needed for this new instruction.

**CMOVZ Rd, Rn, Rm:** conditional move if zero. The instruction copies the value from the Rn register to the Rd register if the contents of the Rm register are 0. If the contents of the Rm register are non-zero, the rd register is left as-is.

**ADD3 Rd, Rn, ALU_Imm12:** add together the values of register Rd, register Rn, and the imm12 (interpreted as a 2’s comp number), and store the result in register Rd.

**BL BrAddr26:** Same as B, but also stores the value of PC+4 (where PC is the address of this instruction) into register X30.

**SWAP Rd, Rn:** An R-type instruction. Takes the values from Rd and Rn and puts them into registers Rn and Rd respectively, thus swapping their contents.

2.) Create a single-cycle CPU that can perform the following two instructions ONLY. Your CPU should be as simple, and as fast, as possible. If anything is not required for this CPU, do not include it. Show both the datapath and the control.

   CBNZ Rd, CondAddr19  if (Reg[Rd]!=0) PC = PC + SE(CondAddr19)<<2;

   STUR Rd, [Rn, DAddr9]  Mem[Reg[Rn]+SE(DAddr9)] = Reg[Rd];