A simple circuit: Count (cont.)

- This simple circuit is actually a state machine. Once “reset”, RA and RB have 0’s to start, IR to 0’s, and PC is at 0 to start. Then, the state of the circuit transit between “fetch” an instruction and “execute” the instruction.
  - At the “fetch” state, IR will read the instruction at the address stored in PC, PC’s value will be incremented by 1, and the current state of the state machine will transit to “execute.”
  - At the “execute” state, execution logic will execute the instruction stored in IR, and update the current state of the state machine to “fetch.”

- A program/set of instructions stored in program memory can be:

  I0: S1S0 = 01 C0 = 0 00 = 0 : RA <= RA + 1: Increment RA
  I1: S1S0 = 10 C0 = 0 00 = 1 : RB <= RA + 0 : Move RA to RB
  I2: S1S0 = 10 C0 = 1 00 = 0 : RA <= 0 : Move 0 to RA
  I3: S1S0 = 11 C0 = 1 00 = 0 : RA <= -1 : Move -1 to RA
  I4: S1S0 = 10 C0 = 1 00 = 0 : RA <= 0 : Move 0 to RA
  I5: S1S0 = 11 C0 = 1 00 = 0 : RA <= -1 : Move -1 to RA
  I6: S1S0 = 10 C0 = 1 00 = 0 : RA <= 0 : Move 0 to RA
  I7: S1S0 = 00 C0 = 0 00 = 0 : RA <= RB+RA : Add RA to RB and put it in RA

- You can consider I0 ~ I7 as the addresses of these instructions and the binary sequence for S1S0C0O0 as the instructions. In a real circuit, the instructions will enable the execution logic
to work. However, if you use VHDL to simulate the circuit, you will need to implement the operation corresponding to each instruction.

- A computer can do four things:
  - Store data
  - Move data
  - Manipulate data
  - Adjust control flow based on data

- This simple count can do the first three things but not the last one.
- How do we modify this machine to be able to do the last thing?