**CPU MODULE DESCRIPTION**

**Various segments of code**:

1.) 2 10-bit registers (pc and mar), 3 16-bit registers (ir,ac,dr) and 2 1-bit register (s and f).

All these registers have load input. Register value gets updated at the positive edge of clock only when load input is high. I have written separate expressions for load of each register.

2.) 1 2-bit synchronous counter with synchronous clear(clr).

Counter is used to generate different time variables t0,t1,t2,t3. These variables along with value of f are used to completely describe flow of instructions. Using the clear variable clr counter resets if clr is high at the positive edge of clock.

3.) Control variables are derived from opcode using simple behavioural logic.

4.) 16-bit Bus is used to collect and distribute data from memory and various registers. Every register is connected with this bus and when their load goes high, they stores the information. Output of all the registers are supplied to a combinational circuit whose output is connected to bus.

**Approach**: I have divided the process in fetch (f=0) and execute cycle(f=1). In each cycle counter starts with count 0.

Fetch cycle: When count is 0 (t0=1) , address stored in pc is transferred to memory addressing register(mar). This is done by giving bus the value of pc and load of mar is kept high. mar gets updated at next positive edge of clock. When count is 1 (t1=1) ,data out from memory is transferred to bus and load of ir is kept one. Along with this incpc is also set to 1 to increment the value of pc at next clock tick. As soon as ir value updates, opcode and mode also changes. Now the count is 2(t2=1). In this interval we analyse the opcode and mode. Variables c1 and c2 are assigned an expressions which represent the instruction given in opcode 20,21 and 22. If c2 is 1 than it means nothing to be carried out and we will simply clear the counter and again start with new fetch cycle. But if it is zero than we consider values of c1 and mode from which we have four combinations :

1. mode=1 : as mode one implies indirect addressing, again load of mar is made 1 and bus is given value of ir. At next count (t3=1) , mar will have indirect address of the data to be fetched.

* if c1 is 1 : This implies that pc should be given the data out from the memory. So we make load of pc high and supply bus the memory output. Also to restart fetch cycle we clr of counter is made high.
* if c1 is 0 : In this case load of mar is again kept high to get address of the data and bus is given value of memory dataout. Also load of f is done high to change f to 1 and clr is also kept high.

2. mode = 0 :

* if c1 is 1 : this implies that pc should be given the data out from the memory. So we make load of pc high and supply bus the memory output. Also to restart fetch cycle we clr of counter is made high.
* if c1 =0: In this case we will load mar with ir using bus. Also load of f is made high to change f to 1 and clr of counter is also kept high.

Execute Cycle: In this cycle f is high and counter starts counting again from 0.

If opcode is 19 then read input is memory is made 0 (implies writing to memory) and bus (which is input data to memory) is given value of ac. Also load of f and clr are made high to start fetch cycle again.

If opcode <=18,then load of dr(data register) is made high and bus is given value of data output from memory.

In next count (t1) , if opcode is 18 ,then we have to put data stored in dr to ac through bus.

If opcode <18 ,then ac is given the output of CalC (aluout) through bus.

Load of f and clr of counter are also made high at t1.