Report Assignment 9

Our program primarily consist of following funtions:

```
void IF(vector<string> v, int &PC, int *regs, int *memo);
void ID(string inst, int &PC, int *regs, int *memo);
void EX(string op, int src1, int src2, int src3, int &PC, int *regs, int *memo);
void MEM(string op,int addr, int src, int &PC, int *regs, int *memo);
void WB(int wvalue, int wreg, int &PC, int *regs, int *memo);
```

The main funtion creates an register array aka regs[32] and memory array aka memo[4000]. We initialize the both the arrays with 0 and put the sample values in some registers and memory location to simulate/check the processor.

Process:

> Call the funtion fillInstructions to fill the vector of instructions in string form.

> Call the function IF to fetch the instruction from vector v (defined outside the main).

> Call the function ID to decode the instruction. We keep a 2 element array to keep track of the previous and present destination registers. Then we check if present instruction contains the operand which is being modified in last instruction. Stall the process(stall = “true”) if we find any save(sw) instruction and turn on the forward signal(forwrd= “true”) if we find any arithemetic or shift instruction.

We created a bool signal stall, a string signal prevProcess to facilitate the stall process.

We created a bool signal forwrd, an int signal fwdvalue to store forward value, an int signal fwdreg to store the forwarding register to facilitate forward process.

> Call the funtion EX to execute the instruction of arithemetics (add, sub), shifting(sll,srl) and moving to the WB; calculating addresses for jump instructions(j, jal, jr) finish them and calling next instruction; calculated condition for branch instructions(bne, beq, blez, bgtz) and move to the MEM; calculating address for (sw, lw) and moving to MEM.

In this process we check if the forward contion is on and calculate the results accordingly with the help of fwdvalue and fwdreg. We turn off the stall process due to previous load condition if any and turn on the forward condition and again call the same function because the reult has been loading to facilitate the forward condition.

> Call the function MEM to check and branch the branching instruction with correct PC, store the reg value in memory and moving on the next instruction for sw, loading the data from memory addr for load instruction lw and moving to WB to store the data in the register.

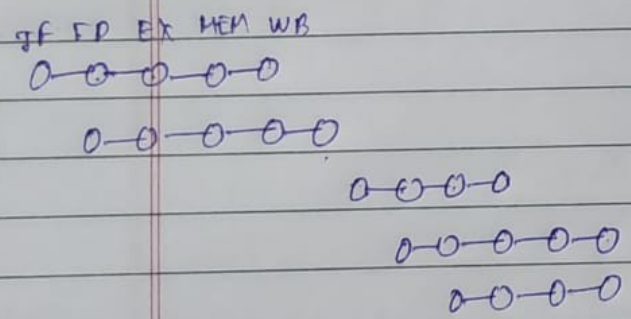
In this function we check if the forward condition for save(sw) is on and the save the data accordingly through fwdvalue.> Call the funtion WB to write back the value calculates and load to the regs for (add,
sub, sll, srl) and lw respectively.

** Comparition for processors of ass8 and ass9 with clock cycle comparition for one test case. A significant change in cycles value can be seen.

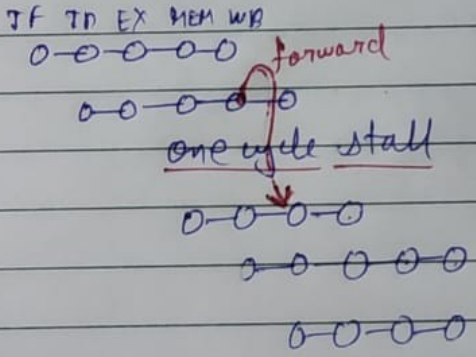
	ass 8	ass 9
t0:	$23/28$ Inst/cycle	$23/25$ Inst/cycle
t1:	$5/12$ Inst/cycle	$5/9$ Inst/cycle
t2:	$7/13$ Inst/cycle	$7/12$ Inst/cycle

e.g. t1 comparision

ass 8



ass 9



Total cycles = 12 9