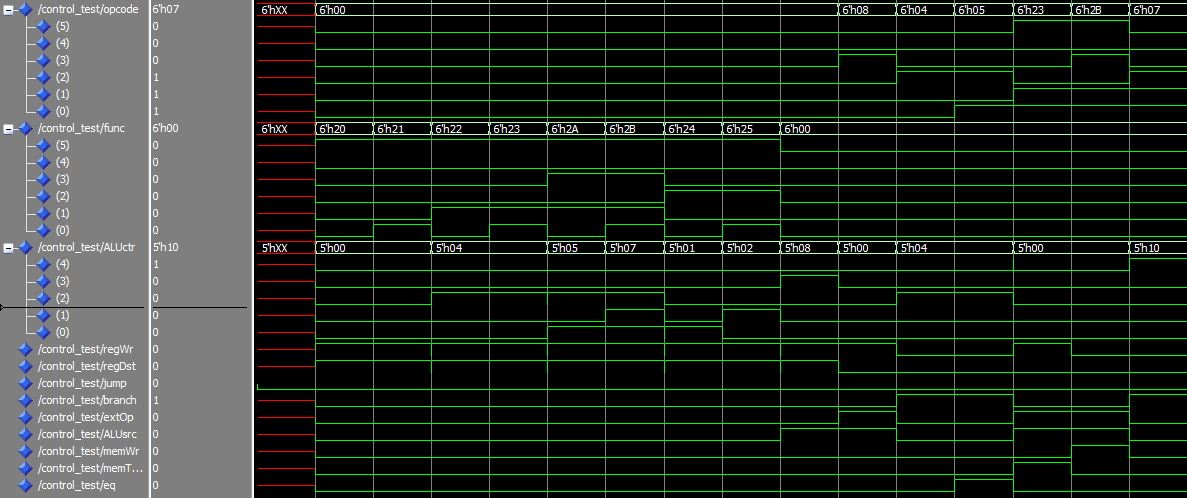
To prove the correctness of our project, we wrote three test files for control, instruction and datapath.

Firstly, the ALU designed from last individual project need to be modified and applied in this single cycle process as a core component, to implement that, a binary connection relationship between ALU opcode and MIPS opcode need to be established, which was showed in table 1:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ALU opcode | | MIPS opcode | | Func |
| add | 00000 | addi | 001000 | N/A |
| and | 00001 | add | 000000 | 100000 |
| or | 00010 | addw | 000000 | 100001 |
| xor | 00011 | sub | 000000 | 100010 |
| sub | 00100 | subu | 000000 | 100011 |
| slt | 00101 | slt | 000000 | 101010 |
| sltu | 00110 | sltu | 000000 | 101011 |
| sll | 00111 | beq | 000100 | N/A |
| srl | 01000 | bne | 000101 | N/A |
| bqtu | 01001 | lw | 100011 |  |
|  |  | sw | 101011 |  |

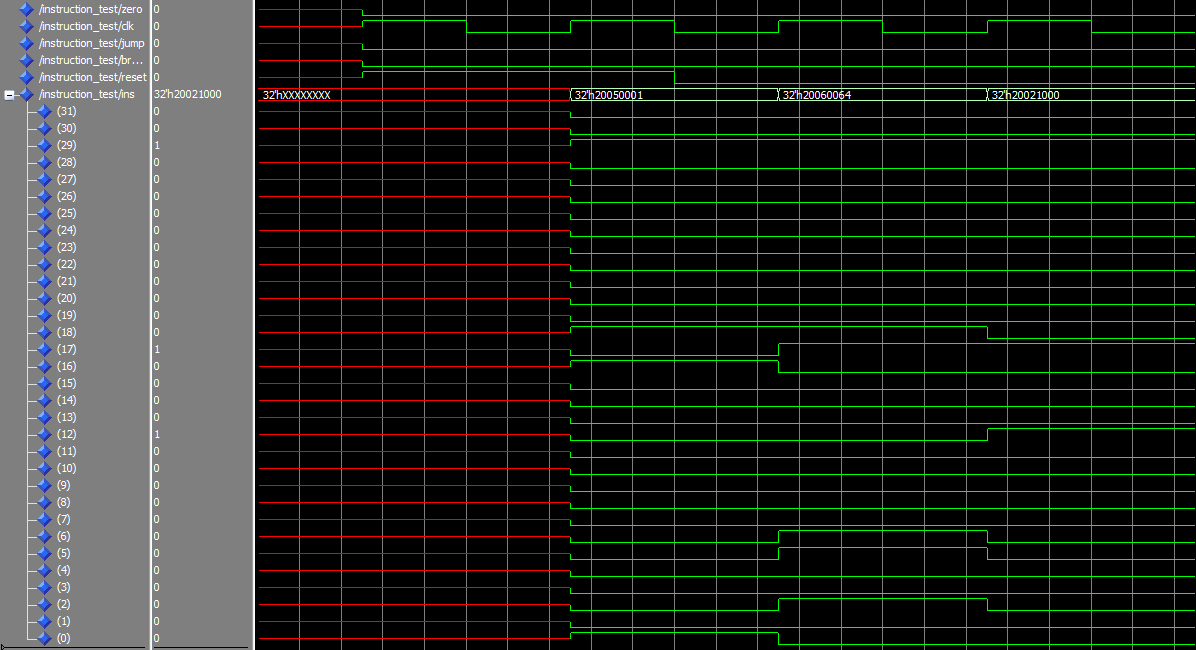
**Table 1: The opcode for ALU and processor and func**



**Figure 1: Control test simulation**

Figure 1 gives a test result for those three opcode relationship, MIPS opcode and func were given as input signal, and ALU opcode was calculated out as an output signal, it could be seen that for each signal state, ALU counter gives a correct responsive signal as we designed in Table 1. Besides that, if we read the control signal state, we will find it as same as the table given in “Control signal summarize.doc”. Therefore, the above simulation result proves that our control part is successful.

Furthermore, for the instruction part, it will read instruction from "bills\_branch.dat" and output it on table as illustrated in Figure 2, in the test file, all inputs have been initialized and the only change is clock signal, it could be seen that at each rising edge of the clock signal, output signal will read the next instruction from data file, which has been checked as correct machine code.



**Figure 2: Instruction test simulation**

Finally, it moves to the datapath test part. To test the correctness of this part, the "bills\_branch.dat" file has been translated into C code to help understanding:

int x = [10, 9, 8, 700, 5, 6, 400, 1, 2 , 3];

int r5 = 1, r6 = 64;

r2 = x; // (0x1000 0000) Address of x

r7 = r2 + 10;

do{

r3 = \*r2;

if ( r6 < r3) r4 =1;

else r4 = 0;

if (r5 != r4){

r6 = r6 – r3;

\*r2 = 0;

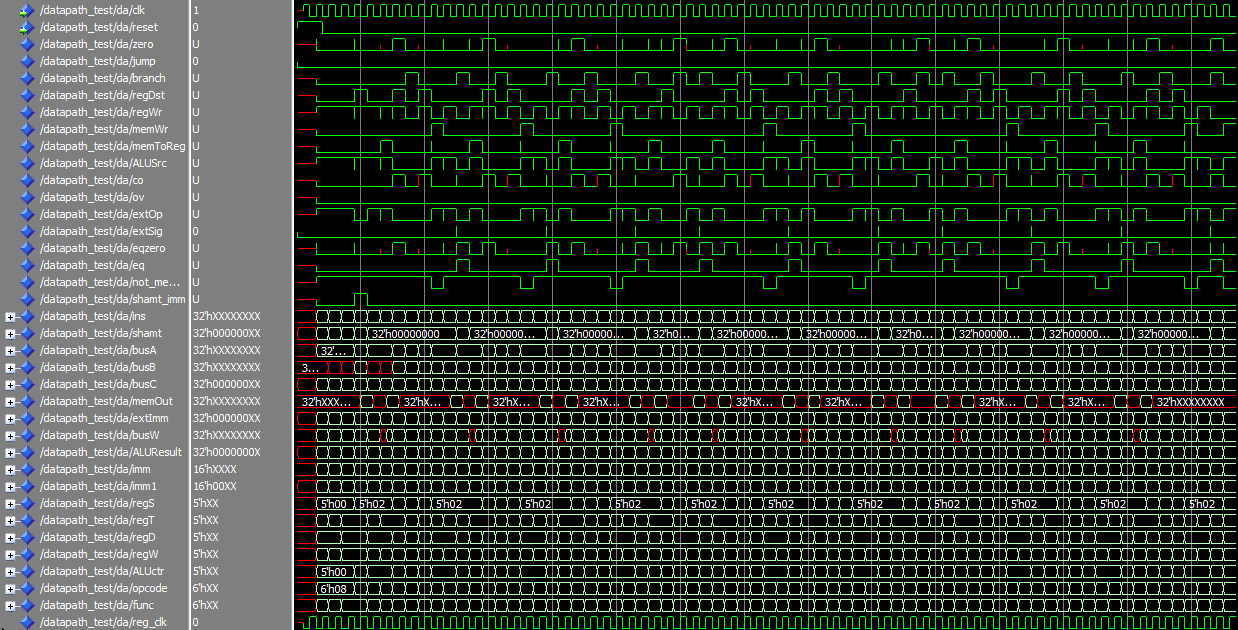
}

r2 ++; //next address of array

}while(r2!=r7);

X[10] = r6;

The above code asked to add a value r6 to array x, and r6 has been initialized and calculated by while loop. Figure 3 gives all control signals and datapath. Every time a new instruction was read, the control signal will change and perform corresponding functions. Due to the page limit, it only shows the state of control signals. But we have checked the fist five instructions and the datapath has been verified correctly.



**Figure 3: Datapath test simulation**

(我看还给了另两个.dat，那两个还需要分析吗？)