Jonah Boe
Calvin Passmore

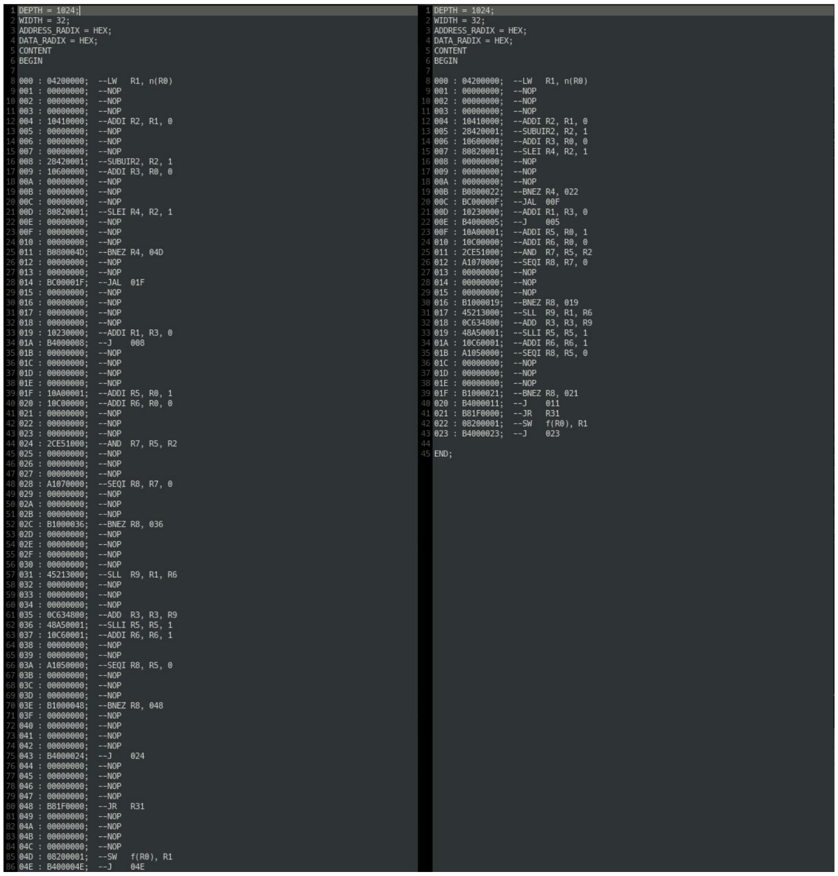
Project 07 – DLX Hazards

Procedure

The goal of this project was to handle all of the data and control hazards. This would allow our code to run without the inclusion of no-ops. In order to handle data hazards we implemented a fast forwarding mechanism in our processor to push results from both the memory stage and the write-back stage directly back into the ALU when required by a sequence of instructions.

In order to handle control hazards, we implemented a mechanism which always assumes the branch will not be taken, then the pipeline can be flushed if that assumption proves wrong. We took this approach because the instructions following the branching instructions enter the pipeline already without any modification to our system. Assuming that branches are always taken would have required a more substantial change to our system.

We were not able to completely debug our system, and our factorial program still requires a few no-ops, but it has been greatly reduced as a result of our work. The before and after comparison of our program is shown in Figure 1.



```
DEPTH = 1024;
WIDTH = 32;
ADDRESS_RADIX = HEX;
DATA_RADIX = HEX;
CONTENT
BEGIN

000 : 04200000;  --LW   R1, n(R0)
001 : 00000000;  --NOP
002 : 00000000;  --NOP
003 : 00000000;  --NOP
004 : 10410000;  --ADDI R2, R1, 0
005 : 00000000;  --NOP
006 : 00000000;  --NOP
007 : 00000000;  --NOP
008 : 28420001;  --SUBUIR2, R2, 1
009 : 10600000;  --ADDI R3, R0, 0
00A : 00000000;  --NOP
00B : 00000000;  --NOP
00C : 00000000;  --NOP
00D : 80820001;  --SLEI R4, R2, 1
00E : 00000000;  --NOP
00F : 00000000;  --NOP
010 : 00000000;  --NOP
011 : B080004D;  --BNEZ R4, 04D
012 : 00000000;  --NOP
013 : 00000000;  --NOP
014 : BC00001F;  --JAL  01F
015 : 00000000;  --NOP
016 : 00000000;  --NOP
017 : 00000000;  --NOP
018 : 00000000;  --NOP
019 : 10230000;  --ADDI R1, R3, 0
01A : B4000008;  --J    008
01B : 00000000;  --NOP
01C : 00000000;  --NOP
01D : 00000000;  --NOP
01E : 00000000;  --NOP
01F : 10A00001;  --ADDI R5, R0, 1
020 : 10C00000;  --ADDI R6, R0, 0
021 : 00000000;  --NOP
022 : 00000000;  --NOP
023 : 00000000;  --NOP
024 : 2CE51000;  --AND  R7, R5, R2
025 : 00000000;  --NOP
026 : 00000000;  --NOP
027 : 00000000;  --NOP
028 : A1070000;  --SEQI R8, R7, 0
029 : 00000000;  --NOP
02A : 00000000;  --NOP
02B : 00000000;  --NOP
02C : B1000036;  --BNEZ R8, 036
02D : 00000000;  --NOP
02E : 00000000;  --NOP
02F : 00000000;  --NOP
030 : 00000000;  --NOP
031 : 45213000;  --SLL  R9, R1, R6
032 : 00000000;  --NOP
033 : 00000000;  --NOP
034 : 00000000;  --NOP
035 : 0C634800;  --ADD  R3, R3, R9
036 : 48A50001;  --SLLI R5, R5, 1
037 : 10C60001;  --ADDI R6, R6, 1
038 : 00000000;  --NOP
039 : 00000000;  --NOP
03A : A1050000;  --SEQI R8, R5, 0
03B : 00000000;  --NOP
03C : 00000000;  --NOP
03D : 00000000;  --NOP
03E : B1000048;  --BNEZ R8, 048
03F : 00000000;  --NOP
040 : 00000000;  --NOP
041 : 00000000;  --NOP
042 : 00000000;  --NOP
043 : B4000024;  --J    024
044 : 00000000;  --NOP
045 : 00000000;  --NOP
046 : 00000000;  --NOP
047 : 00000000;  --NOP
048 : B81F0000;  --JR   R31
049 : 00000000;  --NOP
04A : 00000000;  --NOP
04B : 00000000;  --NOP
04C : 00000000;  --NOP
04D : 08200001;  --SW   f(R0), R1
04E : B400004E;  --J    04E
```

```
DEPTH = 1024;
WIDTH = 32;
ADDRESS_RADIX = HEX;
DATA_RADIX = HEX;
CONTENT
BEGIN

000 : 04200000;  --LW   R1, n(R0)
001 : 00000000;  --NOP
002 : 00000000;  --NOP
003 : 00000000;  --NOP
004 : 10410000;  --ADDI R2, R1, 0
005 : 28420001;  --SUBUIR2, R2, 1
006 : 10600000;  --ADDI R3, R0, 0
007 : 80820001;  --SLEI R4, R2, 1
008 : 00000000;  --NOP
009 : 00000000;  --NOP
00A : 00000000;  --NOP
00B : B0800022;  --BNEZ R4, 022
00C : BC00000F;  --JAL  00F
00D : 10230000;  --ADDI R1, R3, 0
00E : B4000005;  --J    005
00F : 10A00001;  --ADDI R5, R0, 1
010 : 10C00000;  --ADDI R6, R0, 0
011 : 2CE51000;  --AND  R7, R5, R2
012 : A1070000;  --SEQI R8, R7, 0
013 : 00000000;  --NOP
014 : 00000000;  --NOP
015 : 00000000;  --NOP
016 : B1000019;  --BNEZ R8, 019
017 : 45213000;  --SLL  R9, R1, R6
018 : 0C634800;  --ADD  R3, R3, R9
019 : 48A50001;  --SLLI R5, R5, 1
01A : 10C60001;  --ADDI R6, R6, 1
01B : A1050000;  --SEQI R8, R5, 0
01C : 00000000;  --NOP
01D : 00000000;  --NOP
01E : 00000000;  --NOP
01F : B1000021;  --BNEZ R8, 021
020 : B4000011;  --J    011
021 : B81F0000;  --JR   R31
022 : 08200001;  --SW   f(R0), R1
023 : B4000023;  --J    023

END;
```

Figure 1: Final Factorial Code Reduction

As a result, our program execution greatly improved as well. We saw a 30% improvement in execution time according to ModelSim. The before and after execution results are shown in Figures 2 and 3. The flag marker in both figures shows the end of execution before entering the final "done" loop.

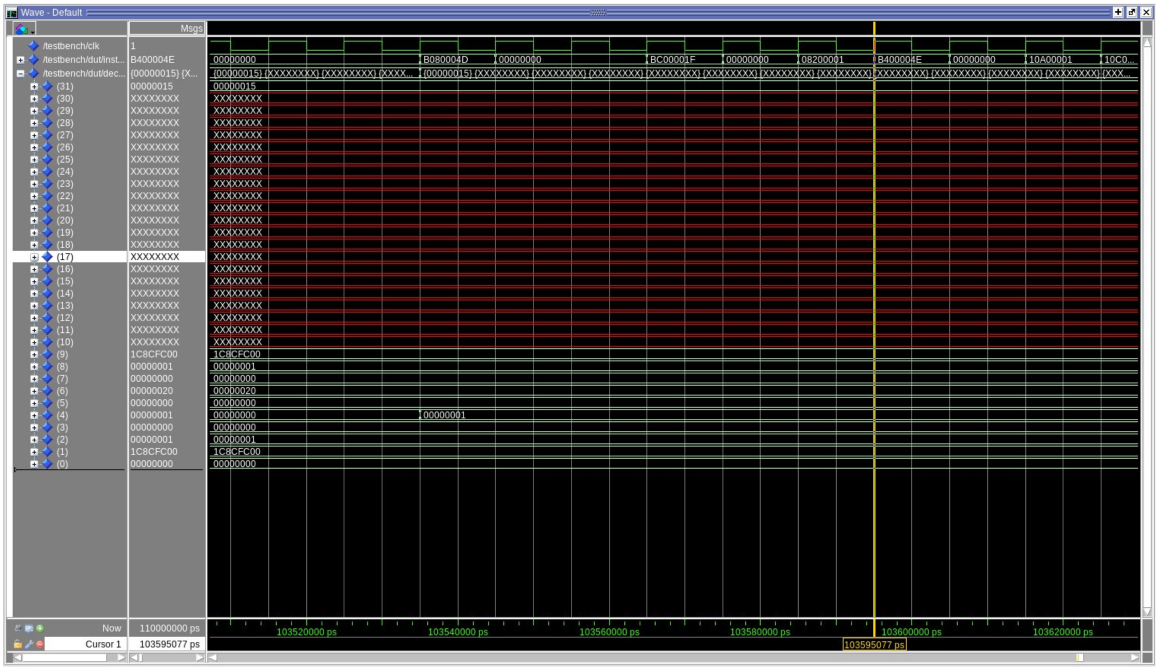


Figure 2: Initial Execution

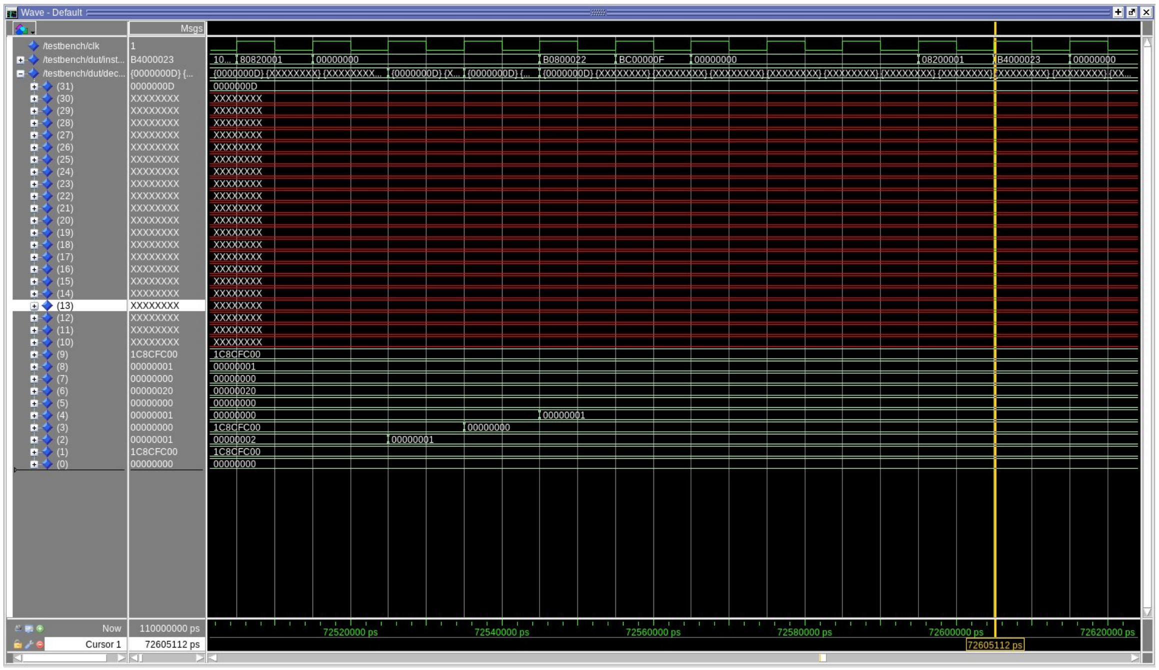


Figure 3: Final Execution

Results

We were mostly successful in eliminating the data and control hazards in our DLX processor, improving the efficiency greatly.

Conclusions

While this stage of design was not as successful as we had hoped, we definitely have everything in place to finish later this week. The majority of hazards are handled successfully, and the few that are not should be a matter of minor debugging. We just ran out of time.