1 Truth Tables and Boolean Equations

Derive a truth table and sum-of-products representation for a function:

- Inputs: consist of 3 values – A, B, C – that may be either True or False
- Output: a single value – X – that is True when two and only two adjacent inputs are true

Full instructions can be viewed in the problem set 1 document.

1.1 Truth Table

A	B	C	X
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	0

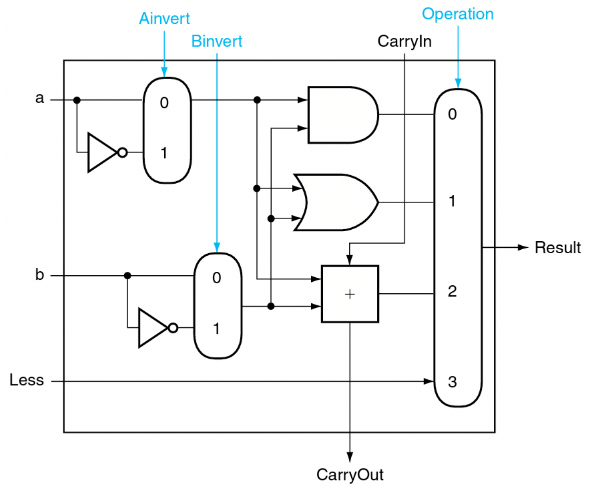
1.2 Sum-of-Products Representation

$$X = (!A\&\&B\&\&C)||(A\&\&B\&\&!C)$$

2 Truth Tables and Boolean Equations 2

Derive a truth table and sum-of-products representation for the function describing the operation of the 4-channel multiplexor shown in P&H Fig B.5.10 (below).

Full instructions can be found here.



2.1 Truth Table

Ainvert	Binvert	Less	S0	S1	S2	S3	Result
F	F	F	F	F	F	F	F
F	F	F	F	F	F	T	T
F	F	F	F	F	T	F	F
F	F	F	F	F	T	T	T
F	F	F	F	T	F	F	F
F	F	F	F	T	F	T	T
F	F	F	F	T	T	F	F
F	F	F	F	T	T	T	T
F	F	F	T	F	F	F	F
F	F	F	T	F	F	T	T
F	F	F	T	F	T	F	F
F	F	F	T	F	T	T	T
F	F	F	T	T	F	F	F
F	F	F	T	T	F	T	T
F	F	F	T	T	T	F	F
F	F	F	T	T	T	T	T
T	T	T	T	T	T	T	T

2.2 Sum-of-Products Representation

$$
\begin{aligned}
\text{Result} = &(!Ainvert \&\& !Binvert \&\& !Less \&\& !S0 \&\& !S1 \&\& !S2 \&\& S3) \\
&|| (!Ainvert \&\& !Binvert \&\& !Less \&\& !S0 \&\& !S1 \&\& S2 \&\& !S3) \\
&|| (!Ainvert \&\& !Binvert \&\& !Less \&\& !S0 \&\& !S1 \&\& S2 \&\& S3) \\
&|| (!Ainvert \&\& !Binvert \&\& !Less \&\& !S0 \&\& S1 \&\& !S2 \&\& !S3) \\
&|| (!Ainvert \&\& !Binvert \&\& !Less \&\& !S0 \&\& S1 \&\& !S2 \&\& S3) \\
&|| (!Ainvert \&\& !Binvert \&\& !Less \&\& !S0 \&\& S1 \&\& S2 \&\& !S3) \\
&|| (!Ainvert \&\& !Binvert \&\& !Less \&\& !S0 \&\& S1 \&\& S2 \&\& S3) \\
&|| (!Ainvert \&\& !Binvert \&\& !Less \&\& S0 \&\& !S1 \&\& !S2 \&\& !S3) \\
&|| (!Ainvert \&\& !Binvert \&\& !Less \&\& S0 \&\& !S1 \&\& !S2 \&\& S3) \\
&|| (!Ainvert \&\& !Binvert \&\& !Less \&\& S0 \&\& !S1 \&\& S2 \&\& !S3) \\
&|| (!Ainvert \&\& !Binvert \&\& !Less \&\& S0 \&\& !S1 \&\& S2 \&\& S3) \\
&|| (!Ainvert \&\& !Binvert \&\& !Less \&\& S0 \&\& S1 \&\& !S2 \&\& !S3) \\
&|| (!Ainvert \&\& !Binvert \&\& !Less \&\& S0 \&\& S1 \&\& !S2 \&\& S3) \\
&|| (!Ainvert \&\& !Binvert \&\& !Less \&\& S0 \&\& S1 \&\& S2 \&\& !S3) \\
&|| (!Ainvert \&\& !Binvert \&\& !Less \&\& S0 \&\& S1 \&\& S2 \&\& S3) \\
&|| \ldots \\
&|| (Ainvert \&\& Binvert \&\& Less \&\& S0 \&\& S1 \&\& S2 \&\& S3)
\end{aligned}
$$

3 Processor Organization, Datapath and Control Pathways

This homework consists of 4 questions (below), each of which is a multi-part answer. Full instructions can be found here.

3.1 Question 1

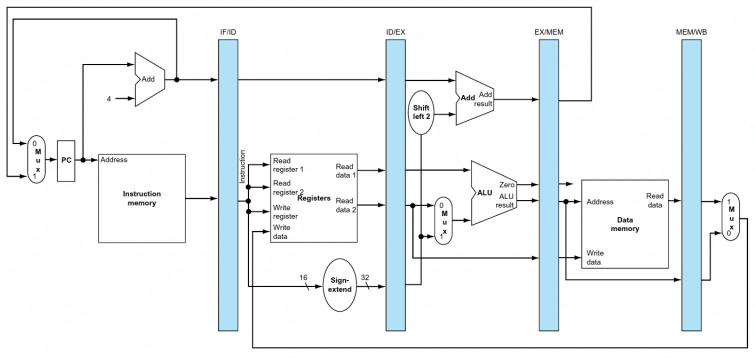


FIGURE 4.35 The pipelined version of the datapath in Figure 4.33. The pipeline registers, in color, separate each pipeline stage. They are labeled by the stages that they separate; for example, the first is labeled *IF/ID* because it separates the instruction fetch and instruction decode stages. The registers must be wide enough to store all the data corresponding to the lines that go through them. For example, the IF/ID register must be 64 bits wide, because it must hold both the 32-bit instruction fetched from memory and the incremented 32-bit PC address. We will expand these registers over the course of this chapter, but for now the other three pipeline registers contain 128, 97, and 64 bits, respectively.

1. Why is the width of the ID/EX buffer 128 bits? What values does it hold?

   This buffer is 128 bits because it holds the 32-bit values of the two registers read during the ID stage, the 32-bit sign-extended immediate value, the 6-bit funct field, the 5-bit register destination number, and a 26-bit control field for a total of 128 bits. [P&H 292p2, 296p2]

2. Why is the width of the EX/MEM buffer 97 bits? What values does it hold?

   This buffer is 97 bits because it contains the 32-bit ALU result, the 32-bit value of the second register read during the ID stage (needed for store instructions), the 5-bit register destination number, the 26-bit control field, and a 2-bit field indicating the branch outcome for a total of 97 bits. [P&H 292p3,4]

3. Why is the width of the EX/MEM buffer 97 bits? What values does it hold?

   This buffer is 97 bits because it contains the 32-bit data read from memory or the ALU result, the 5-bit register destination number, and a 27-bit control field for a total of 64 bits. [P&H 296p2]

3.2 Question 2

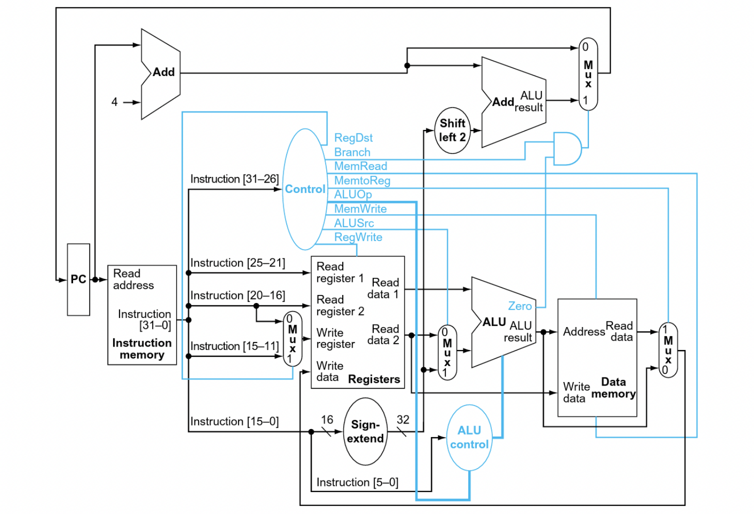


FIGURE 4.17 The simple datapath with the control unit. The input to the control unit is the 6-bit opcode field from the instruction. The outputs of the control unit consist of three 1-bit signals that are used to control multiplexors (RegDst, ALUSrc, and MemtoReg), three signals for controlling reads and writes in the register file and data memory (RegWrite, MemRead, and MemWrite), a 1-bit signal used in determining whether to possibly branch (Branch), and a 2-bit control signal for the ALU (ALUOp). An AND gate is used to combine the branch control signal and the Zero output from the ALU; the AND gate output controls the selection of the next PC. Notice that PCSrc is now a derived signal, rather than one coming directly from the control unit. Thus, we drop the signal name in subsequent figures.

1. What are the values of the 9 control signals generated by the control in Fig 4.17 for this instruction?

RegDst	1
ALUSrc	0
MemtoReg	0
RegWrite	1
MemRead	0
MemWrite	0
Branch	0
ALUOp1	1
ALUOp0	0

[P&H 266]

2. What resources (logic blocks) perform a useful function that contributes to the final answer for this instruction?

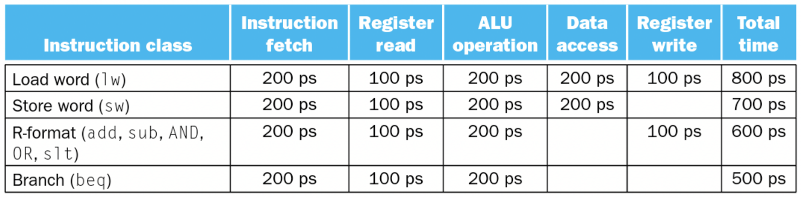
   Instruction memory, PC, Adder (PC + 4), Registers, ALU, ALU Control, Multiplexer. [P&H Fig. 4.1]

3. Which resources (logic blocks) produce outputs, but their outputs are not used for this instruction?

   Sign-extend, Multiplexer, Data Memory, Shift left 2, Adder, Multiplexer. [P&H 254]

3.3 Question 3

Assume the following instructions and time requirements for each resource stage associated with each instruction:



Instruction class	Instruction fetch	Register read	ALU operation	Data access	Register write	Total time
Load word (lw)	200 ps	100 ps	200 ps	200 ps	100 ps	800 ps
Store word (sw)	200 ps	100 ps	200 ps	200 ps		700 ps
R-format (add, sub, AND, OR, slt)	200 ps	100 ps	200 ps		100 ps	600 ps
Branch (beq)	200 ps	100 ps	200 ps			500 ps

1. What is the clock cycle time required for a non-pipelined implementation that will accommodate all the instructions shown in the figure?

lw	800 ps
sw	700 ps
R-Format	600 ps
Branch	500 ps

[P&H 275]

2. What is the clock cycle time for a pipelined implementation that will accommodate all the instructions shown in the figure?

Data access	200 ps
ALU operation	200 ps
Register read	100 ps
Register write	100 ps
Instruction fetch	200 ps

[P&H Fig. 4.26]

3. What is the total latency of the LW instruction in a non-pipelined implementation?

   800 ps

4. What is the total latency of the LW instruction in a pipelined implementation?

   1000 ps

3.4 NERSC Account Data

1. On what date and time did you submit your NERSC account request?

   Sept. 19th

2. What was your requested NERSC username?

   uzylol

4 Performance Modeling

1a 1,000,000 (N)

1b 0

1c 5,000,000,000,000 clock cycles

1d 5 CPI

1e 0.0025s

2a 1,000,000 (N)

2b 1,000,000 (N)

2c 105,000,000,000,000 clock cycles

2d 105 CPI

2e 0.0525s

3a 1,000,000 (N)

3b 2,000,000 (2N)

3c 205,000,000,000,000 clock cycles

3d 205 CPI

3e 0.1025s