Processing Unit Design

Dr. Suman Kumar Maji

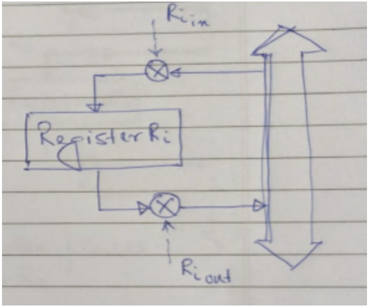
Faculty of CSE Department
Indian Institute of Technology Patna

August 11, 2023

Example: ADD R1, R2

So consider we want to do the ADD R1, R2 operation. Contents from R1 have to be brought to the ALU, and similarly for R2. After the operation is done in the ALU, the result has to be brought back to R2.

A register typically has two control signals: Rin and Rout. Data available in the data bus will be put into Ri through the control signal Rin. Rout will place the value from Ri into the bus when required.



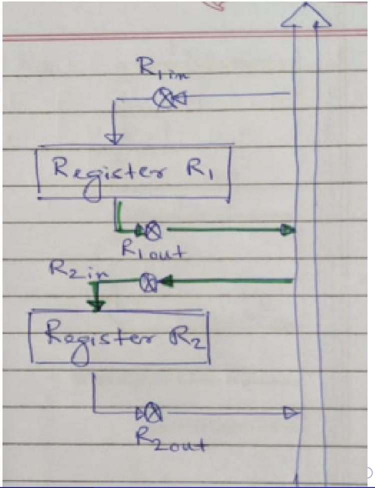
Register Transfer

Example 1: Register Transfer

MOVE R1, R2 // R2 ← R1.

So, the contents of R1 will be moved into R2. The control signals will be:

- Enable the output of R1 by setting Rout = 1.
- Enable the input of R2 by setting R2in = 1.
- Make the data in R1 available to the bus and place the data available in the bus into R2.



Register Transfer (Contd.)

Now all operations are performed in synchronism with the processor clock. The control signals are asserted at the start of the clock cycle. We can write it as:

Step	Micro Operation	Control
t0	R2 $\leftarrow R1$	R1out, R2in
		Select R1 as source (R1out) Select R2 as destination (R2in)

After data transfer, the control signals will return to '0'.

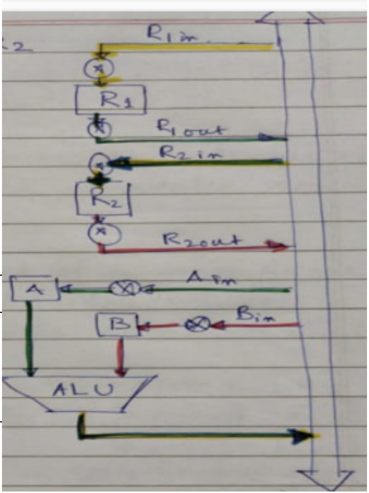
So, these are called control signals, and t0 is the particular time step where these particular control signals are activated.

Example 2: ADD R1, R2; R2 = R1 + R2

Example 2: ADD R1, R2; R2 = R1 + R2

So, bring the two operands (R1) and (R2) to the two inputs of the ALU, one through A (R1) and another through B (R2) via the local bus. Results will be stored in R2 from the ALU via the bus. Since this is a one-bus organization, the procedure will be completed in three steps.

Step	Micro-operation	Control
t0	A ← (R1)	R1out, Ain –
t1	B ← (R2)	R2out, Bin –
t2	R2 ← (A) + (B)	Add, R2in –



Control Signal Generation Approaches

So, till now we have seen the various internal bus architectures and also seen how the various instructions are executed. Now we will look into the approaches required for the generation of control signals.
Broadly, there are two types of approaches:

- hardwired control unit design
- microprogrammed control unit design

Hardwired Control Signal Generation

In hardwired control, direct implementation is accomplished using logic circuits. For each control line, one must find the Boolean expression in terms of the input to the control signal generator. Let's see how we can do it.

Assume that the instruction set of a machine has 2 instructions: `Inst-x` and `Inst-y`, where `Inst-x` is `MOV R1, R2` and `Inst-y` is `ADD R1, R2`.

Control Signal Generation Table

Step	**Move R1, R2**	**ADD R1, R2**
t0	(R2 ← R1) R1out, R2in	A ← R1: R1out, A in
t1		B ← R2: R2out, Bin
t2		R2 ← (A) + (B): ADD R2in

Control Signal Generation for R1out and R2in

Now let's generate the control signals for R1out and R2in.
For R1out, we have it for time step t0 for both instructions: MOV and ADD. So, R1out is activated for MOV at t0 and for ADD at t0. We can write the Boolean expression for R1out as:

R1out = MOV_t0 + ADD_t0 = (MOV + ADD) $\cdot t0$

Similarly, for R2in, the Boolean expression will be:

R2in = MOV.t0 + ADD.t2

Now we have derived the Boolean expressions for R1out and R2in based on the control signal generation table.

Next Steps
With these Boolean expressions, we can now proceed to implement the logic circuits for the control signal generator.

Gate Representation

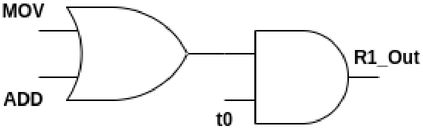


Figure: R1_out

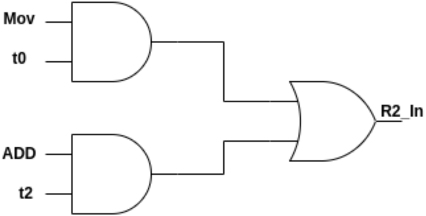


Figure: R2_in

Boolean Expressions for Control Lines

Draw the circuit for executing the following 3 instructions

Step	Inst-x	Inst-y	Inst-z
t0	D,B,E	F,H,G	E,H
t1	C,A,H	G	D,A,C
t2	G,C	B,C	

Boolean expression for the control lines or control signals for (A,,,H) will be

$$\begin{aligned} A &= \text{Inst-x} \cdot t_1 + \text{Inst-z} \cdot t_1 \\ &= (\text{Inst-x} + \text{Inst-z}) \cdot t_1 \end{aligned}$$

$$B = \text{Inst-x} \cdot t_0 + \text{Inst-y} \cdot t_2$$

$$\begin{aligned} C &= \text{Inst-x} \cdot t_1 + \text{Inst-x} \cdot t_2 + \text{Inst-y} \cdot t_2 + \text{Inst-z} \cdot t_1 \\ &= (\text{Inst-x} + \text{Inst-z}) \cdot t_1 + (\text{Inst-x} + \text{Inst-y}) \cdot t_2 \end{aligned}$$

Microprogrammed Control Unit

Microprogrammed Control Unit

Microprogramming was motivated to reduce the complexity of hardwired control.

An instruction like ADD R1, R2 is implemented using a set of microoperations.

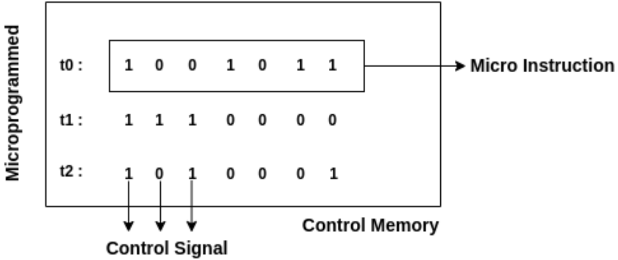
For each microoperation, a set of control signals must be activated.

Microprogrammed control stores the control signals associated with a certain instruction (i.e., inst-x) as a microprogram into control memory (CM).

- A microprogram consists of a sequence of micro instructions, where each bit represents a control signal.
- A micro-program consists of a sequence of micro. instructions.
- A microinstruction is a vector of bits.
- Each bit is a control signal.

Controll Signal Diagram

So for a given instruction execution say inst-x, my controll memory will something like this:



t0 :	1	0	0	1	0	1	1
t1 :	1	1	1	0	0	0	0
t2 :	1	0	1	0	0	0	1

Figure: R2_in

Control Signal Coding

Micro Instruction	...	R1_Out	Ain	R2_Out	Bin	ADD	R2_in	...
1	0	1	1	0	0	0	0	0
2	0	0	0	1	1	0	0	0
3	0	0	0	0	0	1	1	0

Let's say we have many control signals in addition to the signals for the ADD R_1, R_2 instruction. In this example, we'll focus on two control signals: $R1_{\text{out}}$ and A_{in} for step 1.

The above microprogram instruction represents the ADD R_1, R_2 operation. If we have 100 control signals, we would need 100 control bits.

Encoding types

Broadly there are 2 alternate schemes to code the control signals (or control bits) here.

Horizontal Micro-instruction Encoding:

- Each control signal is represented by a bit.
- Each control signal can be long.

Vertical Micro-instruction Encoding:

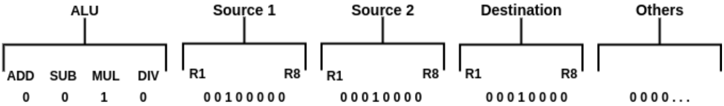
1. Control signals are coded into specific fields in a microinstruction.
2. Decoders are required to map a field of k bits to 2^k possible outcomes.For example, a 3-bit field in a microinstruction could be used to specify any one of 8 possible control signals.
3. For example, a 3-bit field in a microinstruction could be used to specify any one of 8 possible control signals.

Control Signal Generation Approaches (Contd.)

Let's consider a process that supports 4 instructions (ADD, SUB, MUL, DIV) and has 8 registers (R1-R8). Implement MUL R3, R4 and show the control memory.

Control Memory for MUL R3, R4, R4←R3*R4

Consider the three bus architecture , where there are 3buses and consider you have a single i/o line from register to a bus.



Select R3 on Out Bus1, Select R4 on Out Bus2, Select R4 on In-Bus, Select ALU function MUL.

Vertical Micro-programmed

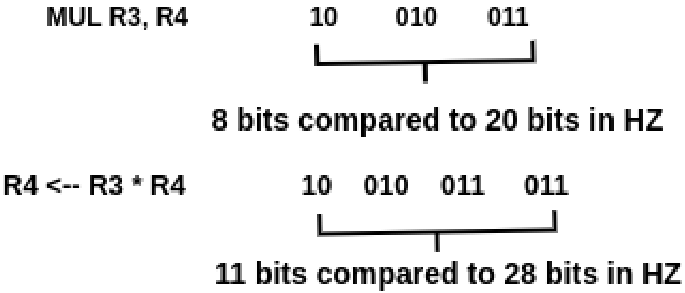
Let's consider the previous example using vertical micro-instruction to implement MUL R3 ,R4 There are 4 instructions ADD, SUB, MUL, DIV. How bits can have 4 combinations? 2 bits. Let us suppose

- ADD → 00
- SUB → 01
- MUL → 10
- DIV → 11

Same way, 8 registers can be represent using 3 bits

- R1 → 000
- R2 → 001
- R3 → 010
- R4 → 011
- R5 → 100
- R6 → 101
- R7 → 110
- R8 → 111

Vertical Micro-programmed



MUL R3, R4 10 010 011

8 bits compared to 20 bits in HZ

R4 <-- R3 * R4 10 010 011 011

11 bits compared to 28 bits in HZ