CSA FINAL ASSIGNMENT

Submitted to: Sir Sohaib

SUBMITTED BY:

MUHAMMAD REHAN ASGHAR

BSSE 15126

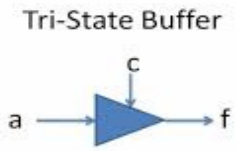
Topic no 1:

Common bus using 3-state buffer

3-state device is a device that can be active low or high. It is similar to a buffer, but in its ads an additional "enable" input that will Control whether the primary input is passed to output or not. If the enable inputs Signal is true then tri state buffer behaves like a normal buffer. If the enable inputs Signal is false then 3-state buffer passed a high impedance signal which fortunately disconnects its output from the circuit.

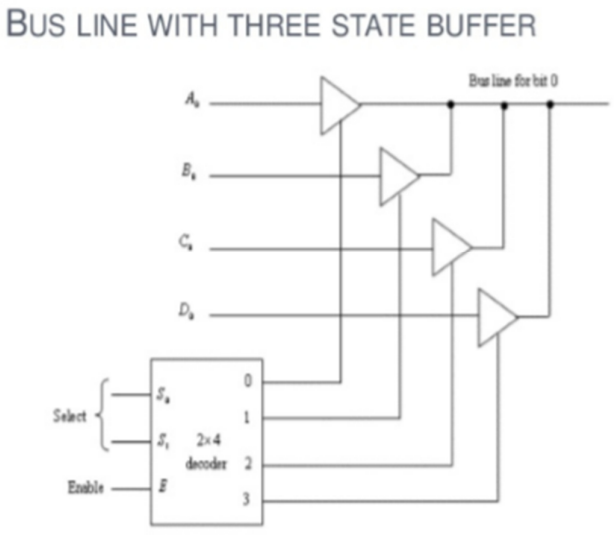
It can be called controlled unit, where “a” is input signal, “c” is input control signal and “f” is output signal

3-state outputs are implemented in many registers, bus drivers and flip-flops in the7400 and 4000 Series as well as in other types, but also internally in many integrated circuits. Many devices are controlled. By an active-low input called output Enable, which decide whether the outputs should be held in a high-impedance state or drive their respective loads (either 0 or 1 level).



c	a	f
0	0	Z
0	1	Z
1	0	0
1	1	1

Block Diagram



Topic no 2:

Pipelining

It a form of computer organization in which several steps of command sequence are fulfill in turn by sequence of modules able to operate right away, so that the another command can be begun before the previous one is finished.

Types of Pipelining:

There are three types of pipelining, which are as given below:

Arithmetic pipeline:

Arithmetic logic units of a computer can be one of the parts into which something naturally separates for pipeline operations in various data formats.

Instruction pipelining:

The execution of stream of instruction can be pipelined by overlapping the execution of current instruction with fetch, decode and operand fetch of the other instruction.

Processor pipeline:

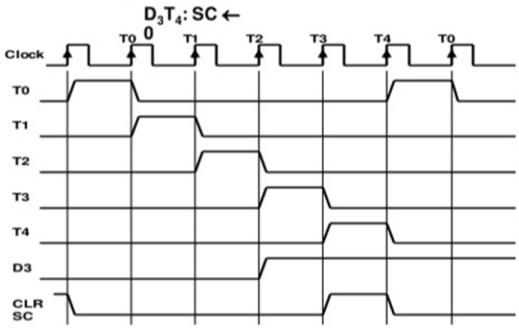
The pipeline processing of the same data stream by a cascade of processors, each with specific task. The data passes form the 1st processor with the results stored in the memory block which is also accessible by the second processor. The second processor repeat this process and the refined result to the third and so on.

Topic no 3:
Control timing signal (Figure 5.7)

It control the timing of signals.

the end 3 waveform show us that how sequence counter is cleared when D3T4 equal to 1.the output D3 from the operation decoder becomes active at the end of T2.when T4 actives, the output of AND gate is implements the control function

D3T4 active then the signals applied to the CLR input of sequence counter. On the one marked T4 in the diagram the counter is cleared to 0.In this case the T0 will active instead of T5 that would have been active, if sequence counter incremented instead of cleared. This process will keep going until rise in edge of timing signal.



Topic no 4:
Register transfer for fetch phase (Figure 5.8)

The program Counter is loaded with many address of 1st instruction in the program. The sequence counter is cleared to 0 while giving decoded T0.When clock pulse in the sequence counter incremented by one the timing signals go thru the T0, T1, T2 and so on.

STEP 1:- T0: AR <-- PC

STEP 2:- T1: IR <--M [AR], PC<--PC+1

STEP 3:- T2: D0,....D7 <-- Decode IR (12-14), AR <-- IR (0-11), I <-- IR (15)

The first to steps shows the fetching operation where in the back end the incremented operation were used this operation will continue. In 3rd step decoding operation.

Here the AR is directly linked with the address inputs of memory, it's important to transfer address from PC to AR during the clock process associated with T0.The instruction read from memory is then placed in the instruction register with clock process with T1. AT the same time, PC is incremented is the back end to prepare it for the next address in the program. At T2, the operation code in IR is decoded, in the indirect bit is transferred to flip-flop I, and then address is transferred to AR.

To provide the data path for the transfer of PC to AR we must apply T0 to achieve the followings:

1. Place the PC onto the bus by making the bus selection inputs like S0S1S2 =010.

2. To Transfer the content of bus to AR by giving the LD input to AR.

When the T0=1 then it initiates the transfer from PC to AR. To implement the second statement

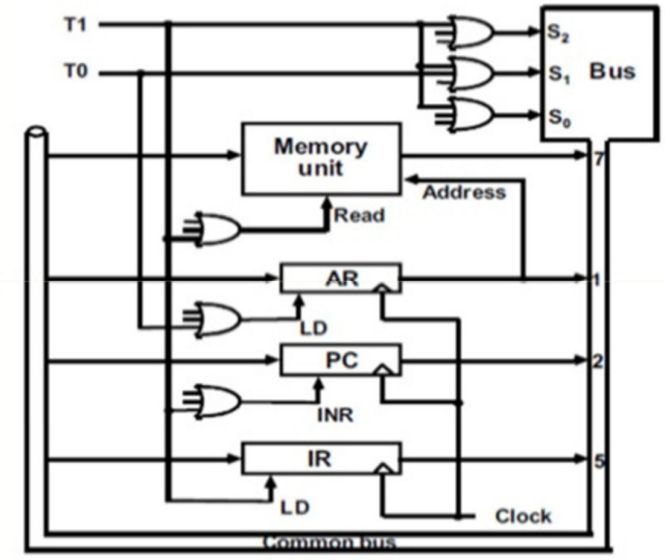
T1: IR <-- M [AR], PC <-- PC +1.

These following connections are used in the bus system
1. Give the read input of memory.

2. Place the content of memory onto the bus by making S0S1S2 = 111.

3. Transfer instruction of the bus to IR By giving the LD input of IR.

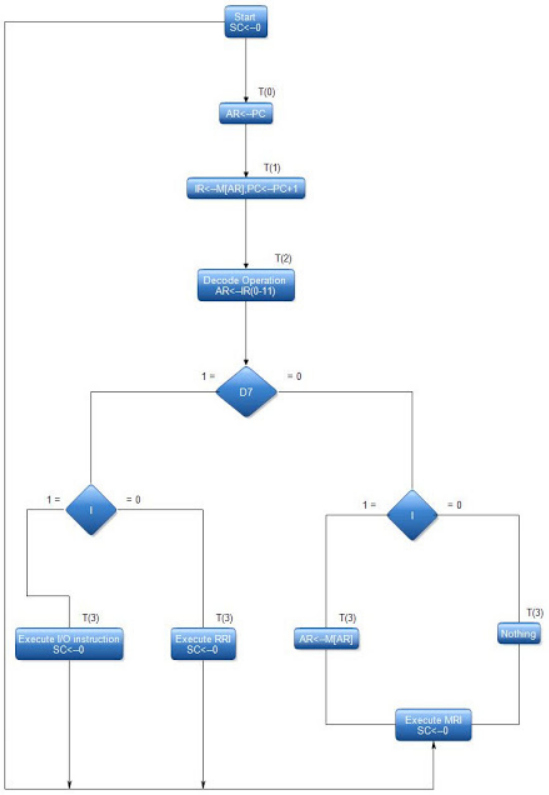
4. Increment PC by giving the INR input of PC.



Flow Chart for Instruction Cycle (Figure 5.9)

Computer receives instructions to implement different tasks and functions. These instructions consists of different cycles which are as follows:

1. Fetch
2. Decode
3. Execute



Fetch:

During the phase of fetching, computer performs different micro – operations to fetch or retrieve the data from different places.

It usually performs this micro-operation during fetching phase:

$$AR \leftarrow PC$$

$$IR \leftarrow M\,[AR]$$

By this micro-operation, the address of data shifts to address register.

Decode:

In decoding phase, computer just takes the data and finds out that either the data is in memory, register or input/output data.

It usually performs this micro-operation during fetching phase:

$$D0 – D7 \leftarrow IR\ (\ 0 – 11\ )$$

Execute:

Execution is the last phase of Instruction cycle. In this phase , computer actually executes or performs the required function or task. After finding out the instruction type, computer executes the required micro-operation.

Topic no 5:

Arithmetic Logic Shift Unit

This tables shows us the eight arithmetic operations, four logic operations and two shift operations.

Each operation that will carried out in this circuit will be selected by using these five variables or inputs S0, S1, S2, S3 and Cin. Where Cin input will be used only, when we will select arithmetic operation.

TABLE 4-8 Function Table for Arithmetic Logic Shift Unit

Operation select						
S_3	S_2	S_1	S_0	C_{in}	Operation	Function
0	0	0	0	0	$F = A$	Transfer A
0	0	0	0	1	$F = A + 1$	Increment A
0	0	0	1	0	$F = A + B$	Addition
0	0	0	1	1	$F = A + B + 1$	Add with carry
0	0	1	0	0	$F = A + \overline{B}$	Subtract with borrow
0	0	1	0	1	$F = A + \overline{B} + 1$	Subtraction
0	0	1	1	0	$F = A - 1$	Decrement A
0	0	1	1	1	$F = A$	Transfer A
0	1	0	0	×	$F = A \wedge B$	AND
0	1	0	1	×	$F = A \vee B$	OR
0	1	1	0	×	$F = A \oplus B$	XOR
0	1	1	1	×	$F = \overline{A}$	Complement A
1	0	×	×	×	$F =$ shr A	Shift right A into F
1	1	×	×	×	$F =$ shl A	Shift left A into F

In this table, 14 Arithmetic Logic unit operations are written. Now we can select them by using Selection inputs. First eight of these operations are Arithmetic operations and we can select them by using S2 and S3 at 0 state. The next four operations are logic operations and we can select them by S2 and S3 at 0 and 1 state respectively. The Carry input does not affects during these operations. Last two of these operations are selected by S3, S2 at 10 and 11 respectively. The other three inputs will not effects during Shift operations.

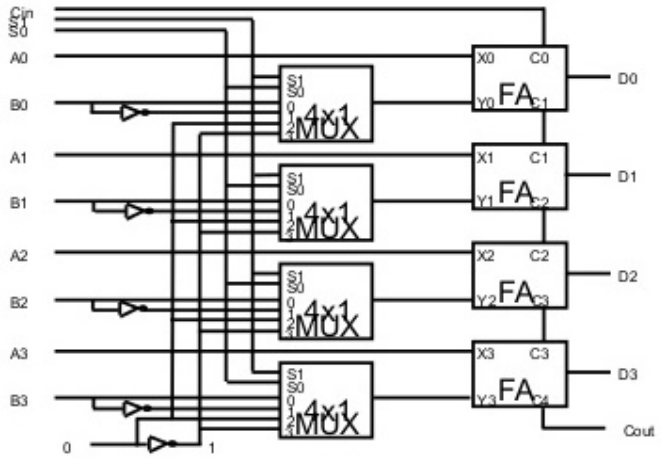


Diagram of Arithmetic Logic Shift Unit from table 4.8