Lecture 17 : March 21, 2007

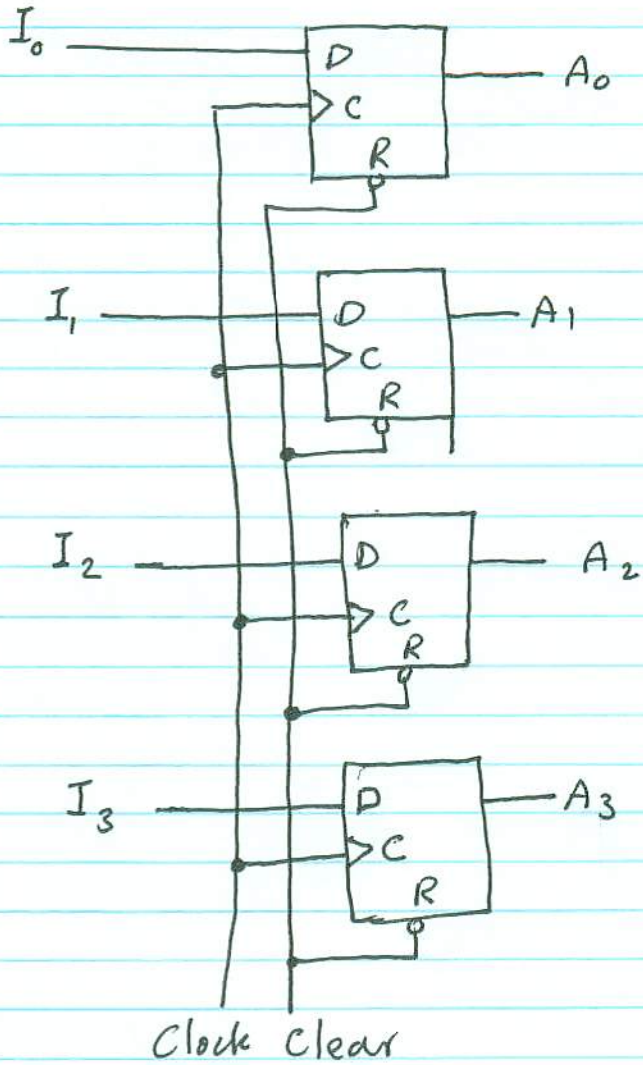
Registers and Counters

In a sequential circuit, a group of flip-flops is used to store information. The content of these flip-flops determine the state of the circuit. Groups of flip-flops may be organized in different ways to perform different functions. Two main configurations are : registers and counters

A register is a group of flip-flops where each flip-flop stores one bit of information.

In addition to flip-flops, a register may have some gates that determine how the content of the register is modified.

The simplest type of registers, is one that only has flip-flops. The circuit diagram below shows a four-bit register with nothing but flip-flops. It is loaded with the value of its inputs. With each clock pulse, the inputs I_0, I_1, I_2, I_3 will be loaded into the register. That is $A_0 = I_0$,

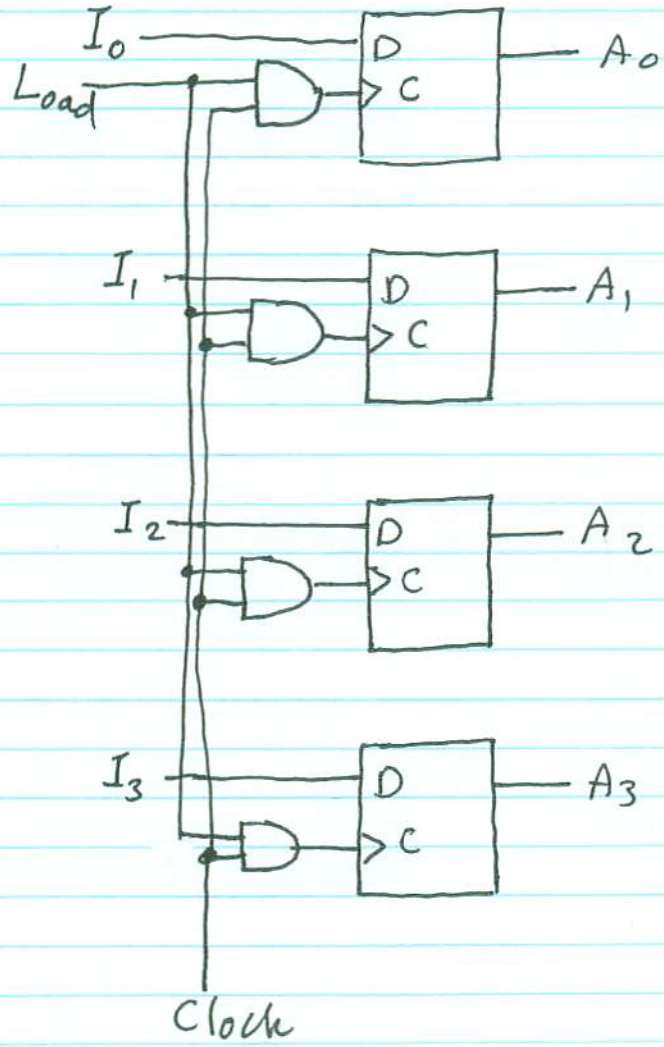


$A_1 = I_1$, $A_2 = I_2$ and $A_3 = I_3$ after the clock.

The circuit has also a reset input that makes the content of the register 0000 when it is low (grounded).

Note that since we have used D-flip-flops, we always load the register when there is a clock pulse. That is, we do not have a means to tell the register not to change on clock occurence.

One way to correct this problem is to gate the clock as shown below.

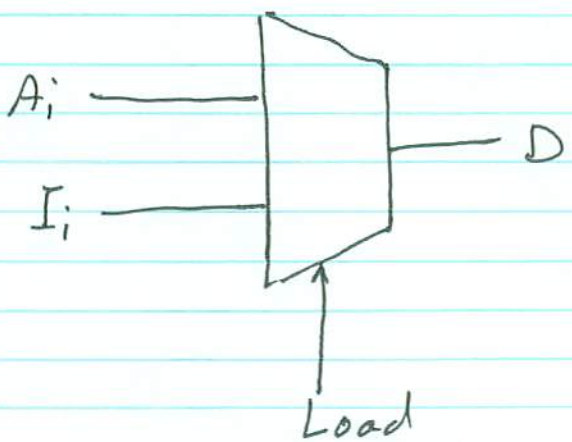


In this circuit, the content of register changes only if Load = 1. The problem with gating the clock is that it creates variable delay between the master clock and the input of the flip-flops in the circuit. A better solution is to use a logic circuit for feeding the inputs that has as input

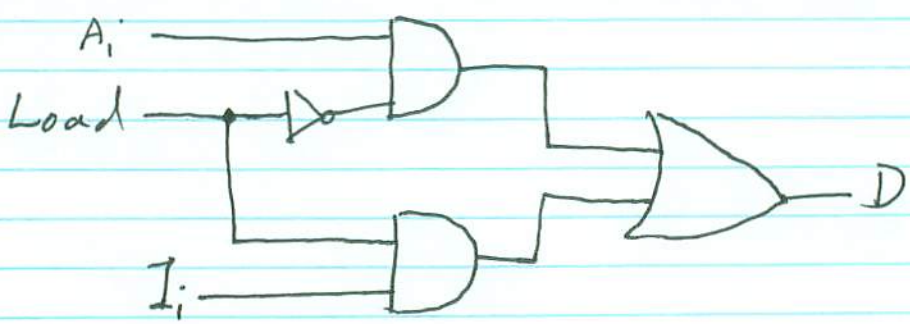
the Load signal, the input I_i and the output A_i of the flip-flop, $i = 0, 1, 2, 3$.

We would like the input be transferred to the output, i.e., $D = I_i$ when Load = 1 and A_i stay unchanged, i.e., $D = A_i$ when Load = 0.

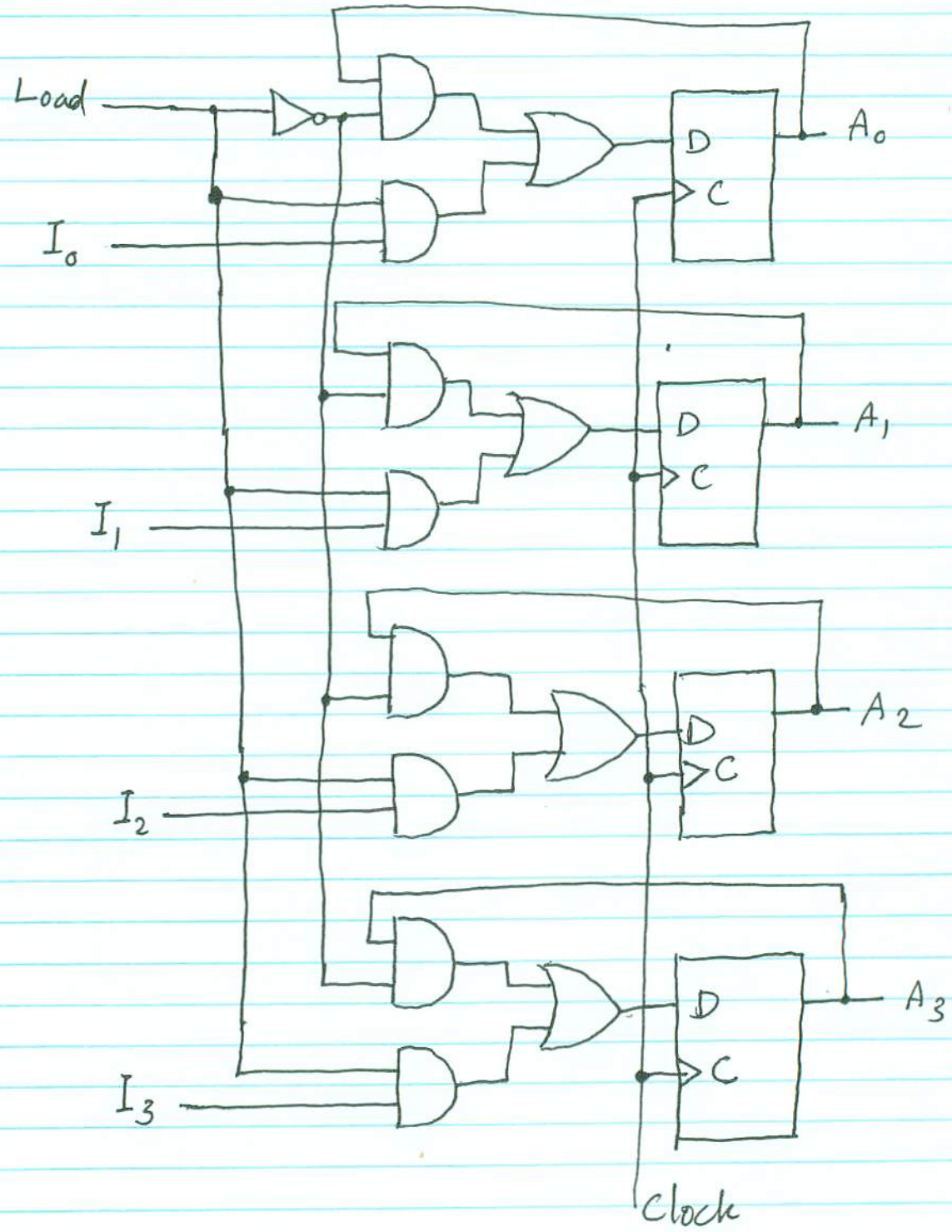
Note that the above describes the function of a 2-to-1 multiplexer with inputs A_i and I_i and the output D and Load acting as the select (S) signal.



The implementation of the multiplexer is:

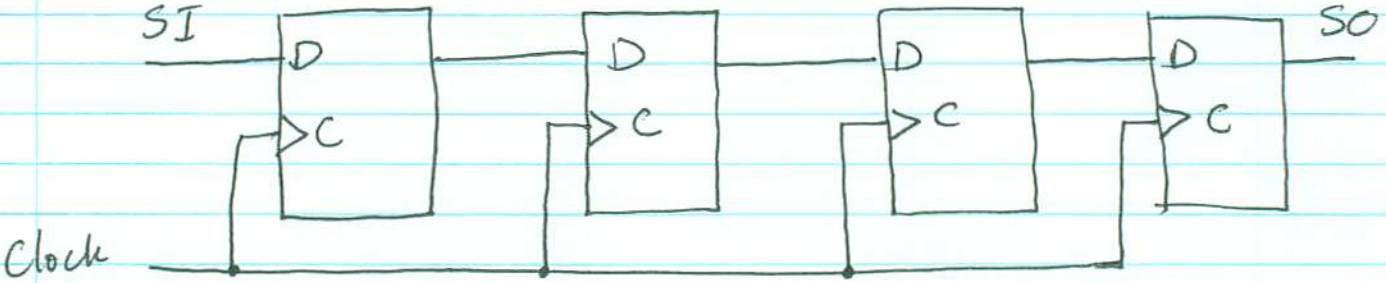


Adding the multiplexer to the 4-bit register, we will have the following circuit with capability to decide whether to load the input or leave the content of the register intact.



Shift Registers

A shift register is a register that can shift bits in the flip-flops to the next flip-flop. A simple shift register using D flip-flops is shown below:



On each clock pulse, the serial input, SI, will be shifted to the left-most cell. The content of the cell will be moved into the second flip-flop and so on. The output of the last (the right-most) cell will be moved out of the register as the serial output, SO.

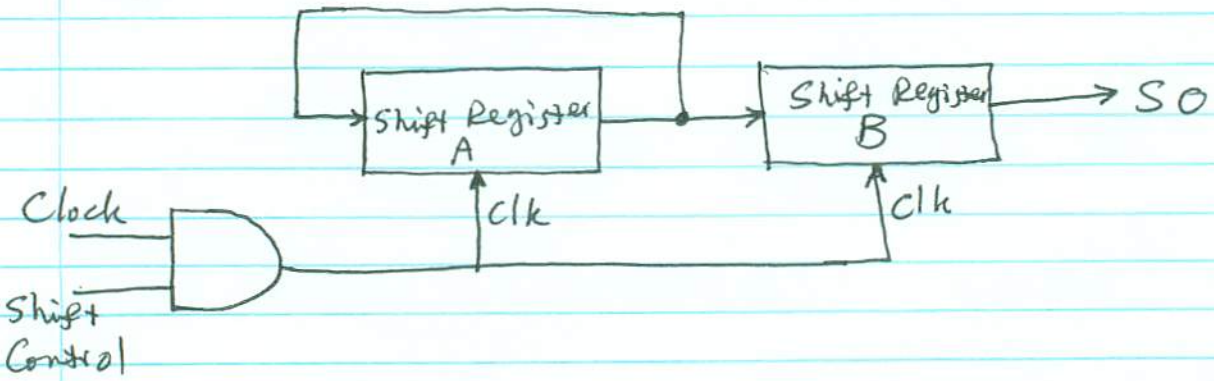
Serial Transfer

Information may be transferred serially, i.e., bit-by-bit from one register to another.

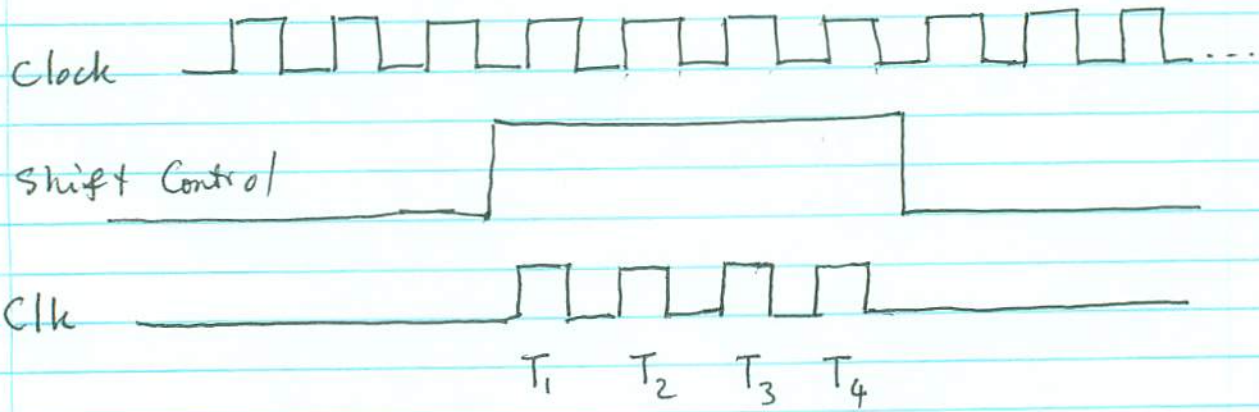
The following diagram shows the transfer of contents of register A into register B. In order

to keep the contents of the register A intact, the serial output of A is connected to its input. So, the bits in register A circulate.

The clock is blocked using an AND gate and clock is only applied when shifting of bits is required.



The clock is always ticking, giving pulses, but the <u>clk</u> signal applied to the registers ticks only when <u>Shift Control</u> is high.



Assume that the shift register A contains 1011 and the shift register B contains 0010. The following table shows the contents of the two shift registers after each clock pulse.

Timing Pulse	Shift Register A	Shift Register B
Initial value	1 0 1 1	0 0 1 0
After T_1	1 1 0 1	1 0 0 1
After T_2	1 1 1 0	1 1 0 0
After T_3	0 1 1 1	0 1 1 0
After T_4	1 0 1 1	1 0 1 1

Serial Addition

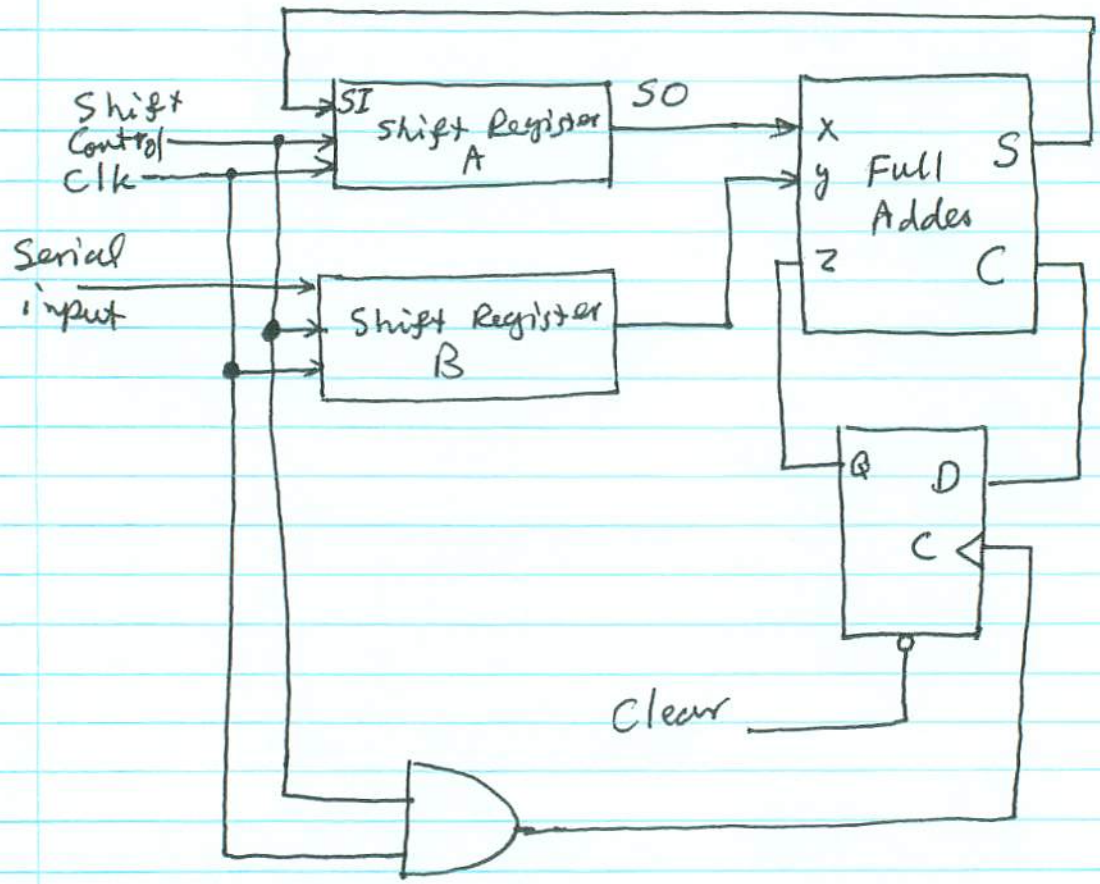
We have already discussed parallel adders. (see Chapter 4 of the text)

In a parallel implementation a full-adder (FA) is needed for each bit. This implies n full-adders for adding two n-bit numbers. Parallel operation while being fast, requires a lot of silicon are, in our example, n full adders.

By doing operations bit-by-bit, i.e., serially, we reduce the number of gates by a factor of n (needing one full adder instead of n), but

but we reduce the speed. In a serial implementation the addition of two n-bit numbers takes n clock pulses. So, the speed is reduced by n.

Following circuit shows an example of serial adder. In this circuit two numbers, one in shift



register A and the other in shift Register B are added and the result is stored in register A. This is usually called an accumulator. The output may, alternatively be stored in a third register.

Implementing the serial adder using JK flip-flop

In this case we follow the procedure we discussed in previous lectures for the design of the sequential circuits, i.e.,

1) We list the different present state, input combinations and specify the next state and the output for each combination.

2) We form the input equations to flip-flops and design circuits to implement them.

Present state	Inputs		Next State	output	FF inputs	
Q	x	y	Q	S	J_Q	K_Q
0	0	0	0	0	0	X
0	0	1	0	1	0	X
0	1	0	0	1	0	X
0	1	1	1	0	1	X
1	0	0	0	1	X	1
1	0	1	1	0	X	0
1	1	0	1	0	X	0
1	1	1	1	1	X	0

we get:

$$J_Q = xy$$

$$K_Q = x'y' = (x+y)'$$

$$S = x \oplus y \oplus Q$$

So, the circuit diagram is :