13-Registers and Counters Part 1

Text: Unit 12

ECEGR/ISSC 201
Digital Operations and Computations
Winter 2011

Overview

- Registers and Transfers
- Shift Registers

Registers

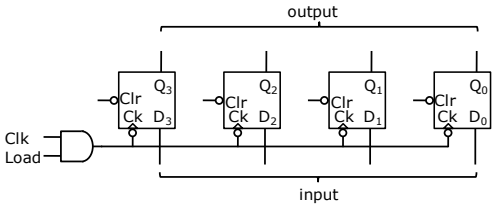
- Register: group of flip-flops (usually D flip-flops) with a common clock input
- Applications:
  - Storing small amounts of binary data
  - Shifting small amounts of binary data

Registers

- Data stored in registers can be accessed very fast
- Efficient use of registers can enhance a program's performance
- Most computers have multiple registers
  - Storing addresses
  - Storing output/input of ALU

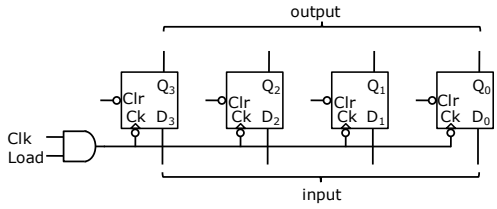
Registers

- Example: 4-bit register (falling edge)
  - Clear (Clr) input sets output of each register to 0
  - Usually from the same source, (ClrN)



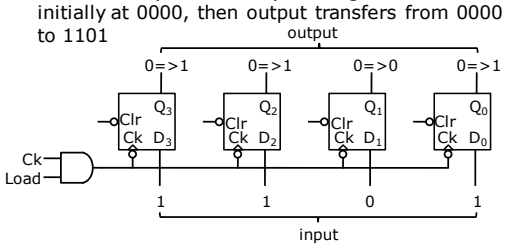
Registers

- Example: 4-bit register
  - Input data is loaded when Clock and Load signals are positive (on falling edge)



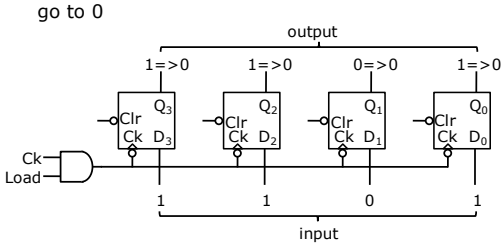
Registers

- If ClrN = 1 (do not clear) and registers are initially at 0000, then output transfers from 0000 to 1101



Registers

- If at <u>any time</u> Clr = 1 (ClrN = 0), then all outputs go to 0

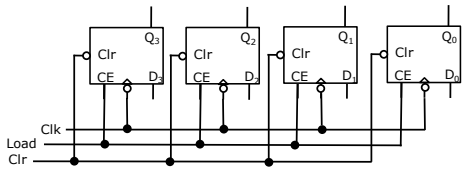


Registers

- Gating the clock signal with another signal can cause timing problems
- Solution: use clock enable (CE) input

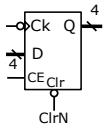
Registers

- Register with Clock Enable
  - Load = 0 clock is disabled, register holds its data
  - Load = 1 clock is enabled, data is transferred to output



Registers

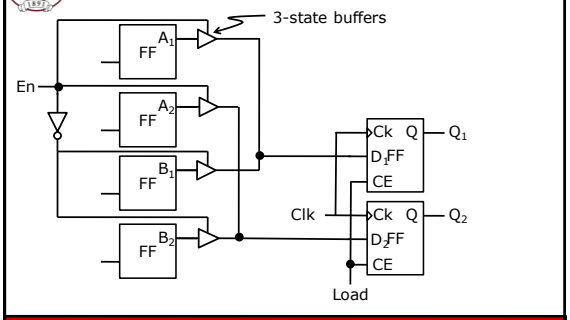
- 4-bit register with clock enable using bus notation



Registers

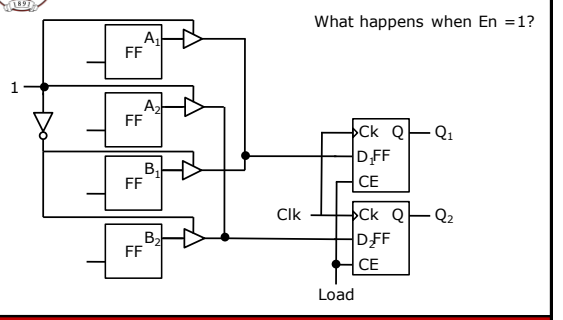
- Often we want to transfer data between registers
- Example:
  - We have a 2-bit register (Q) and we want to load 2-bit data from either the 2-bit register A or the 2-bit register B

Register Transfer



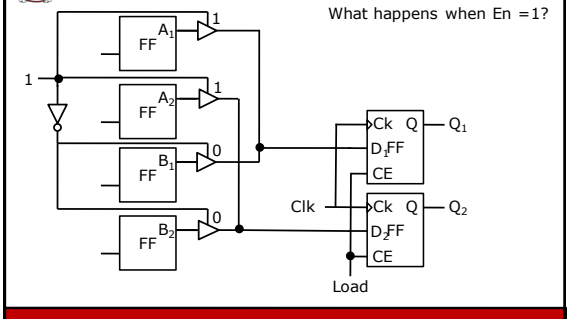
Register Transfer

What happens when En =1?



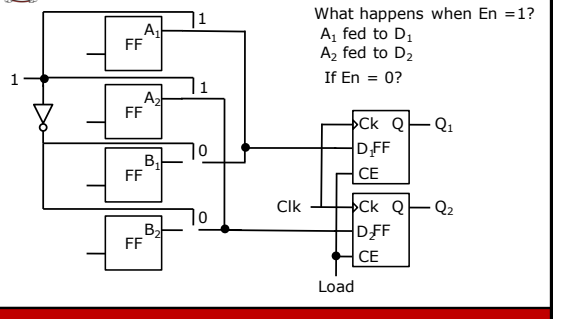
Register Transfer

What happens when En =1?



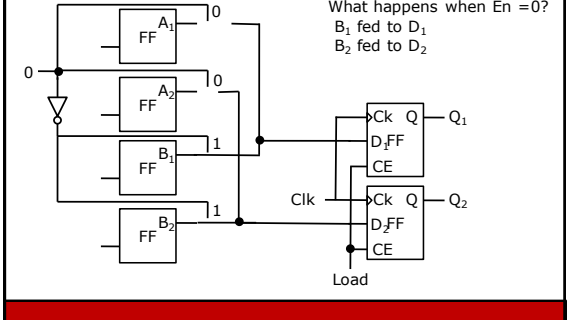
Register Transfer

What happens when En =1?
A_1 fed to D_1
A_2 fed to D_2
If En = 0?

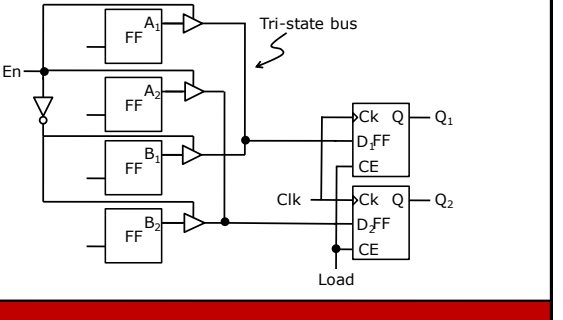


Register Transfer

What happens when En =0?
B_1 fed to D_1
B_2 fed to D_2

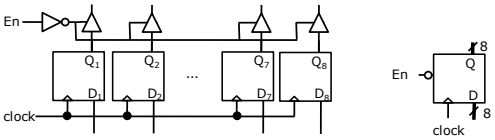


Register Transfer



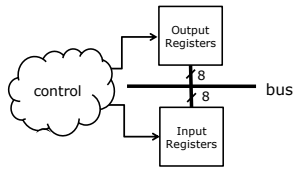
Register Transfer

- An 8-bit register with tri-state output



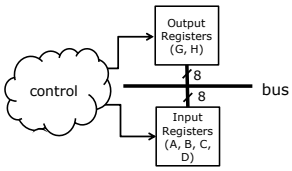
Register Transfer

- Design a digital system that can transfer an 8-bit number from registers A, B, C or D to register G and/or register H



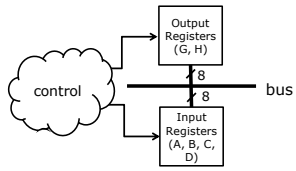
Register Transfer

- Output registers: two 8-bit registers (G, H)
- Input registers: four 8-bit registers (A, B, C, D)
  - Tri-state bus



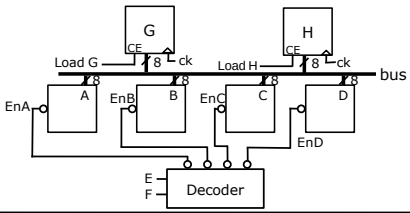
Register Transfer

- Control to Input: select one of four registers to output to the bus
- Control to Output: select which register(s) to load data to



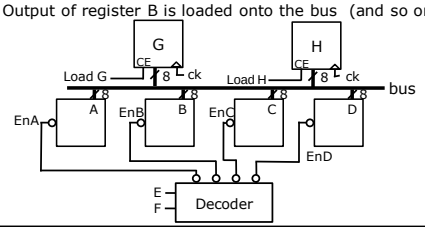
Register Transfer

- Design a digital system that can transfer an 8-bit number from registers A, B, C or D to register G and/or register H



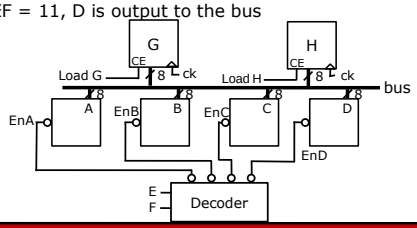
Register Transfer

- EnA = 0;
  - Output of register A is loaded onto the bus
- EnB = 0;
  - Output of register B is loaded onto the bus (and so on)



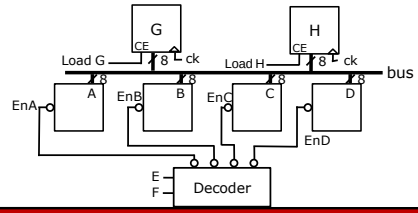
Register Transfer

- If EF = 00, A is output to the bus
- If EF = 01, B is output to the bus
- If EF = 10, C is output to the bus
- If EF = 11, D is output to the bus



Register Transfer

- If LoadG =1, then bus is loaded to G
- If LoadH =1, then bus is loaded to H
- If LoadG = LoadH =1, then bus is loaded to G and H

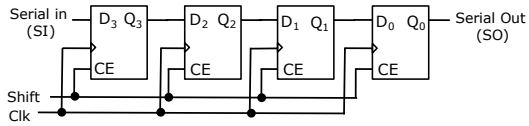


Shift Registers

- Stores binary data, which can be shifted to the left or right when a shift signal is applied
- New bit is shifted in
- Bits shifted out are lost or could be shifted back in

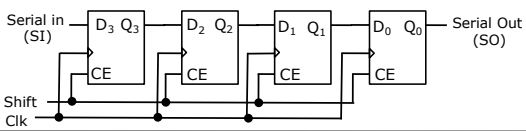
Shift Registers

- When Shift = 0, no shifting occurs



Shift Registers

- When Shift = 1, shifting occurs (on rising edge)
  - Serial input is loaded into D_3
  - Q_3 is loaded into D_2
  - Q_2 is loaded into D_1
  - Q_1 is loaded into D_0
  - D_0 is shifted out (Serial Out)



Shift Registers

- Assume that initially
  - $D_3 = 0$
  - $D_2 = 1$
  - $D_1 = 0$
  - $D_0 = 1$
- The serial input sequence is: 1, 1, 0, 1
- What is the serial output sequence (for four clock pulses)?

Shift Registers

- Registers: 0101
- The serial input sequence is: 1, 1, 0, 1
- What is the serial output sequence (for four clock pulses)?
  - 1 (first pulse)
  - 0
  - 1
  - 0 (fourth pulse)

Shift Registers

- What is the sequence of registers?
  - The serial input sequence is: 1, 1, 0, 1
  - Initially: 0101
  - First clock pulse:
  - Second clock pulse:
  - Third clock pulse:
  - Fourth clock pulse:

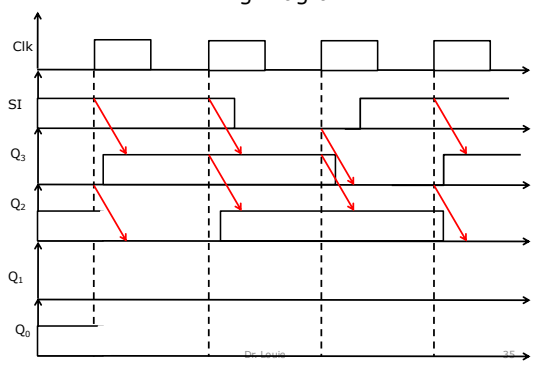
Shift Registers

- What is the sequence of registers?
  - The serial input sequence is: 1, 1, 0, 1
  - Initially: 0101
  - First clock pulse: 1010
  - Second clock pulse: 1101
  - Third clock pulse: 0110
  - Fourth clock pulse: 1011

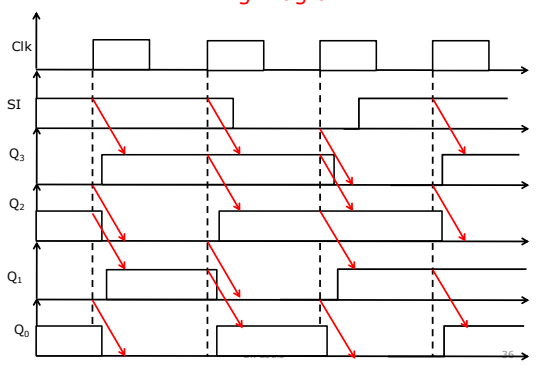
Shift Registers

- Due to propagation delays, the value loaded into each flip-flop is the value before the rising edge

Timing Diagram

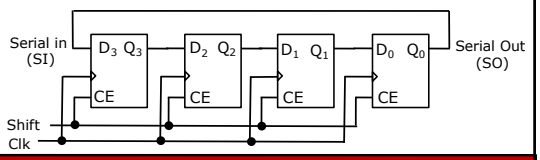


Timing Diagram



Shift Registers

- Consider SI = SO
- Write the next four register values if initially the registers read: 0111

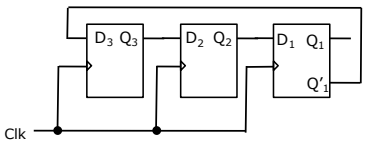


Shift Registers

- Initially: 0111
- First clock pulse: 1011
- Second clock pulse: 1101
- Third clock pulse: 1110
- Fourth clock pulse: 0111

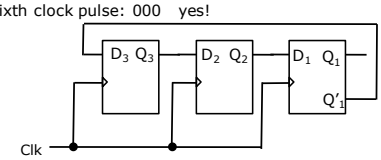
Shift Registers

- Assume the register reads 000
- Does the sequence repeat?



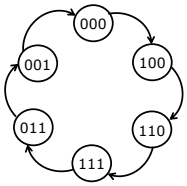
Shift Registers

- Initially: 000
- First clock pulse: 100
- Second clock pulse: 110
- Third clock pulse: 111
- Fourth clock pulse: 011
- Fifth clock pulse: 001
- Sixth clock pulse: 000 yes!



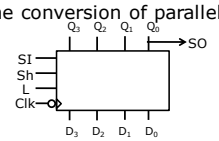
Shift Registers

- The configuration is known as a Johnson Counter
- State graph of Johnson Counter



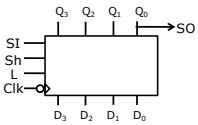
Shift Registers

- What if we want to load data in parallel and out in parallel (not in serial)
  - 4-bit
  - Parallel input, parallel output
  - Shift register
- Allows for the conversion of parallel data to serial data



Shift Registers

- Sh: Shift bits
  - Sh = 1; Clocking causes the bits to shift (SI to Q_3 and so on)
- L: Load input
  - L = 1; Sh = 0; data inputs are loaded in parallel to the flip-flops
  - L = 0; Sh = 0; no change of state



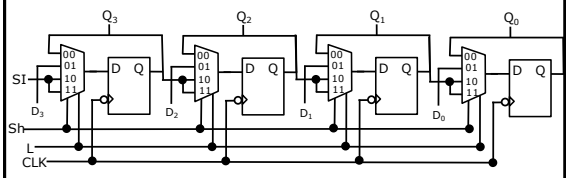
Shift Registers

- Summary of operation

Sh	L	Q_3^+	Q_2^+	Q_1^+	Q_0^+	Action
0	0	Q_3	Q_2	Q_1	Q_0	No change
0	1	D_3	D_2	D_1	D_0	Load
1	X	SI	Q_3	Q_2	Q_1	Shift

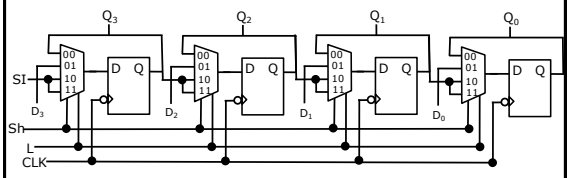
Shift Registers

- Implementation with MUXs and D flip-flops



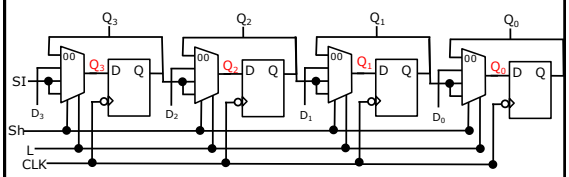
Shift Registers

- Verify no change occurs when Sh = L = 0



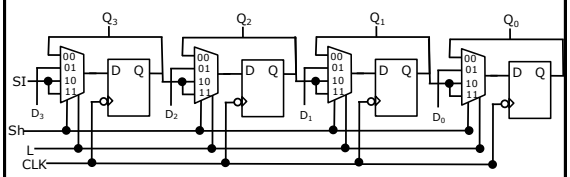
Shift Registers

- Verify no change occurs when Sh = L = 0



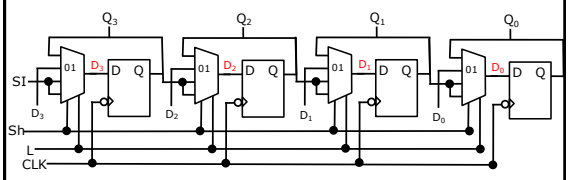
Shift Registers

- Verify load occurs when Sh = 0, L = 1



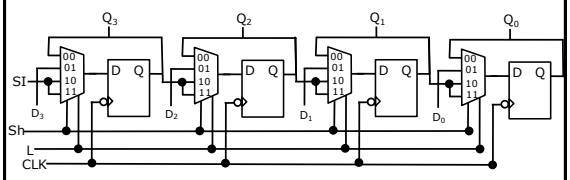
Shift Registers

- Verify load occurs when Sh = 0, L = 1



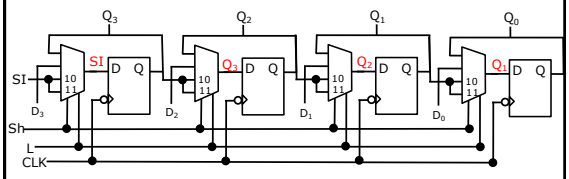
Shift Registers

- Verify shift occurs when Sh = 1, L = X



Shift Registers

- Verify shift occurs when Sh = 1, L = X



Shift Registers

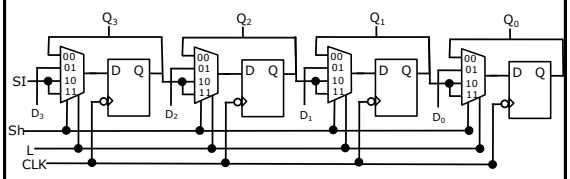
- Characteristic equations
  - $Q_3^+ = Sh'(L')(Q_3) + Sh'(L)(D_3) + Sh(SI)$
  - $Q_2^+ = Sh'(L')(Q_2) + Sh'(L)(D_2) + Sh(Q_3)$
  - $Q_1^+ = Sh'(L')(Q_1) + Sh'(L)(D_1) + Sh(Q_2)$
  - $Q_0^+ = Sh'(L')(Q_0) + Sh'(L)(D_0) + Sh(Q_1)$

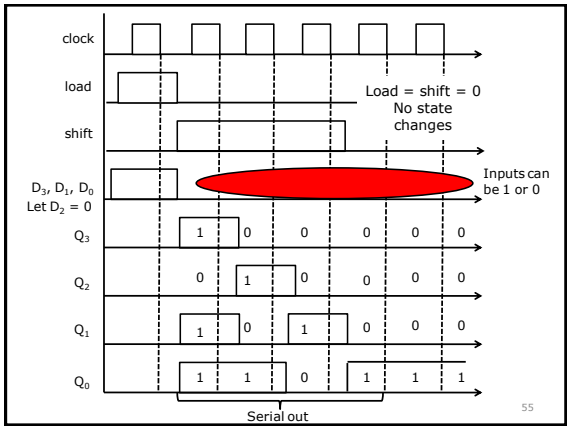
Shift Registers

- Conversion of parallel data to serial data
  - SI = 0 for all pulses
  - Load: first pulse
  - Shift: second through 4 pulses

Shift Registers

- Let SI = 0
- Assume initially $Q_3 = Q_2 = Q_1 = Q_0 = 0$
- Use Q_0 as serial output




clock
load
shift
D_3, D_1, D_0
Let D_2 = 0
Q_3
Q_2
Q_1
Q_0
Load = shift = 0
No state
changes
Inputs can
be 1 or 0
1 0 0 0 0 0
0 1 0 0 0 0
1 0 1 0 0 0
1 1 0 1 1 1
Serial out
55