Synchronous Sequential Circuits II

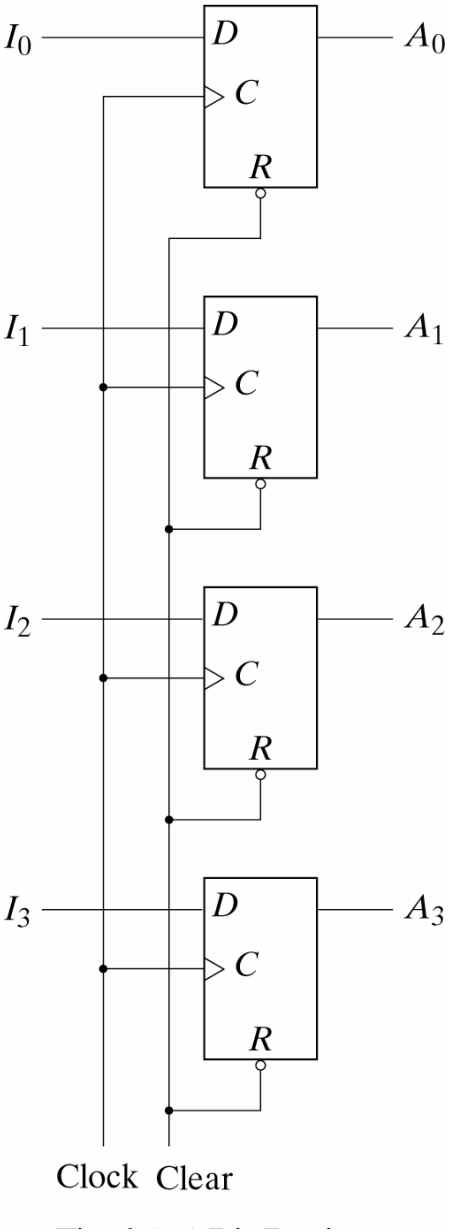
Registers

Register

- A register is a group of flip-flops, each one of which is capable of storing one bit of information.
- A register may have combinational gates that perform certain data-processing tasks.
- In its broadest definition, a register consists of a group of flip-flops together with gates that affect their operation:
  - The flip-flops hold the binary information.
  - The gates determine how the information is transferred into the register.

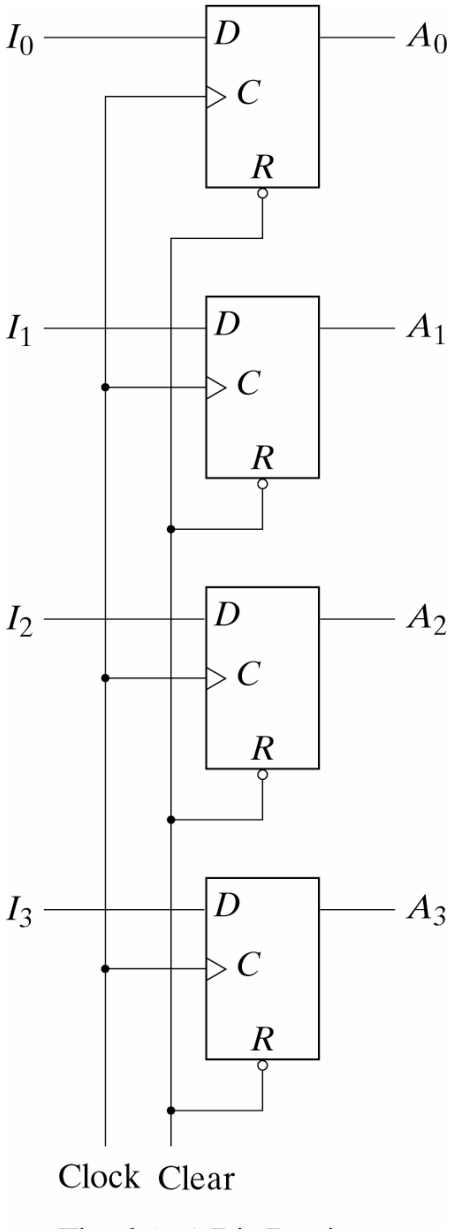
Registers

- The simplest register is one that consists of only flip-flops, without any gates.
- On the right a simple 4-bit register consisting of positive edge triggered D-type flip-flops.
- 4 inputs $\{I_0,I_1,I_2,I_3\}$ and 4 outputs $\{A_0,A_1,A_2,A_3\}$ correspond to the flip-flop inputs and outputs.
- Each flip-flop is triggered by a common (positive edge) clock input.
- Each flip-flop can be reset by applying a common asynchronous reset.



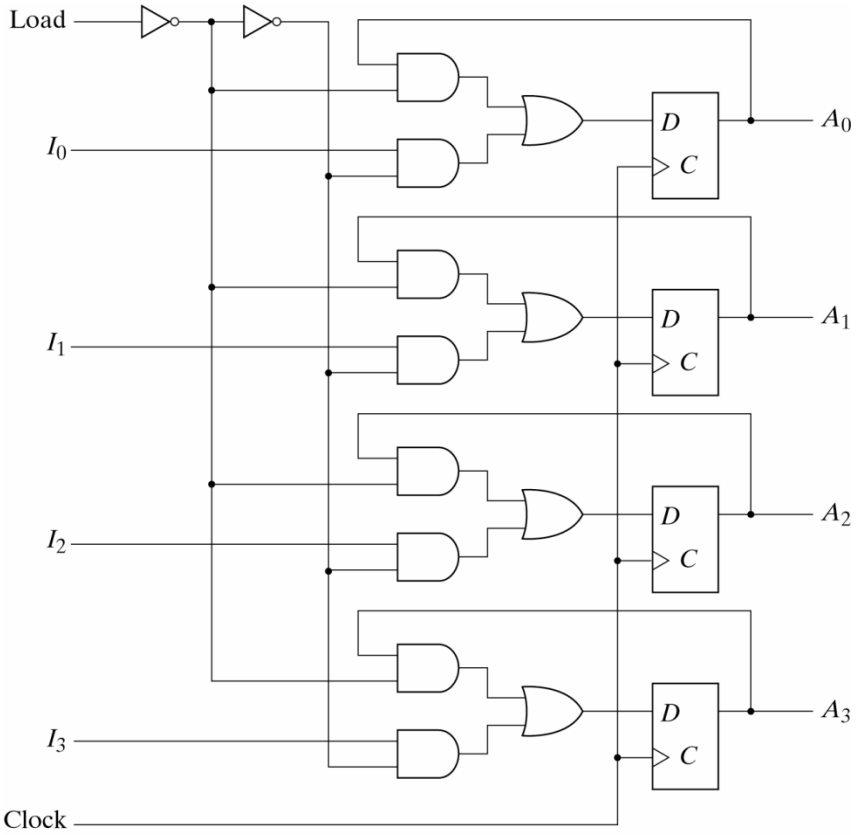
Registers

- When a negative to positive pulse transition occurs the logic signal present at the four inputs is passed to the four outputs and stored there until the next positive edge transition.
- The 4 bit word can be wiped from memory at any time by setting the clear input to logic 0.
- This register is said to have **parallel load**, i.e. the bits are inputted in parallel.



Parallel Load Register

- In the basic implementation of the register data is only stored between positive clock edges.
- In practice a register is equipped with another input which determines whether or not data inputs are loaded into the register or not.



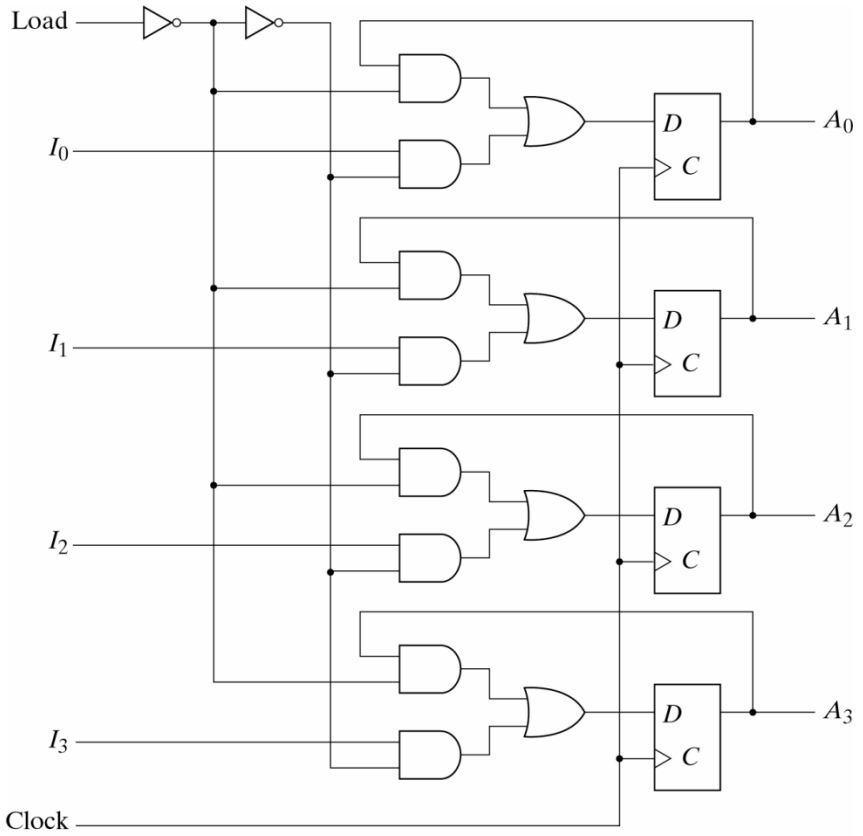
Parallel Load Register

- When the **load** input is set to 1 the data is inputted during the next clock edge.

$$\mathrm{I}.1 + \mathrm{A}.0 \Rightarrow \mathrm{D}$$

- When the load input is set to 0 the data is stored

$$\mathrm{I}.0 + \mathrm{A}.1 \Rightarrow \mathrm{D}$$



Memory Registers

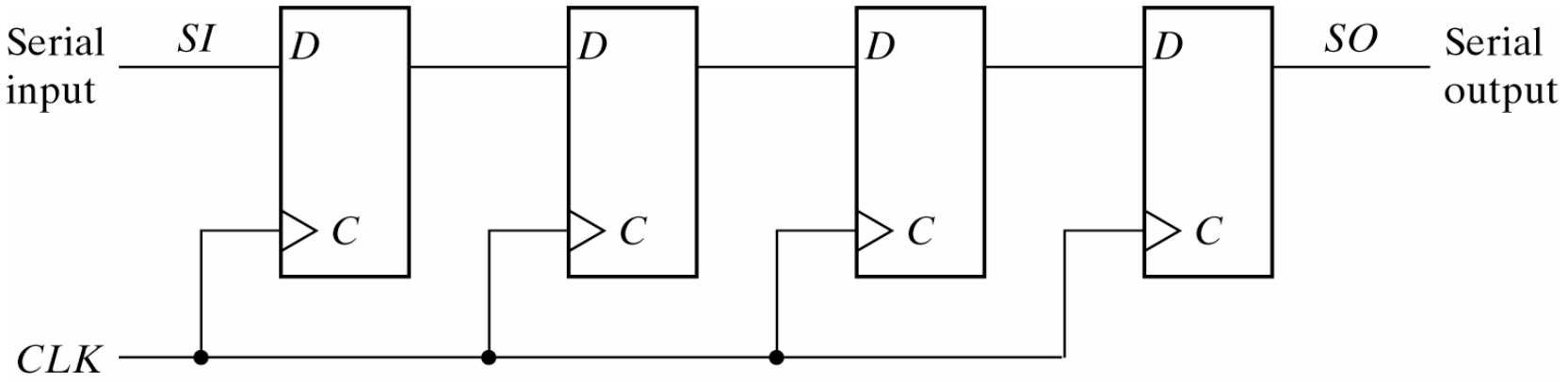
- Typically a register will store 8 bits (a byte) of data.
- In computer architecture each 8-bit register is allocated a hexadecimal address

  0000, 0010, …, 00E0
- Each register contains a byte, here represented by a hexadecimal number.

0000	FF	D8	FF	E1	1D	FE	45	78	69	66	00	00	49	49	2A	00
0010	08	00	00	00	09	00	0F	01	02	00	06	00	00	00	7A	00
0020	00	00	10	01	02	00	14	00	00	00	80	00	00	00	12	01
0030	03	00	01	00	00	00	01	00	00	00	1A	01	05	00	01	00
0040	00	00	A0	00	00	00	1B	01	05	00	01	00	00	00	A8	00
0050	00	00	28	01	03	00	01	00	00	00	02	00	00	00	32	01
0060	02	00	14	00	00	00	B0	00	00	00	13	02	03	00	01	00
0070	00	00	01	00	00	00	69	87	04	00	01	00	00	00	C4	00
0080	00	00	3A	06	00	00	43	61	6E	6F	6E	00	43	61	6E	6F
0090	6E	20	50	6F	77	65	72	53	68	6F	74	20	41	36	30	00
00A0	00	00	00	00	00	00	00	00	00	00	00	00	B4	00	00	00
00B0	01	00	00	00	B4	00	00	00	01	00	00	00	32	30	30	34
00C0	3A	30	36	3A	32	35	20	31	32	3A	33	30	3A	32	35	00
00D0	1F	00	9A	82	05	00	01	00	00	00	86	03	00	00	9D	82
00E0	05	00	01	00	00	00	8E	03	00	00	00	90	07	00	04	00

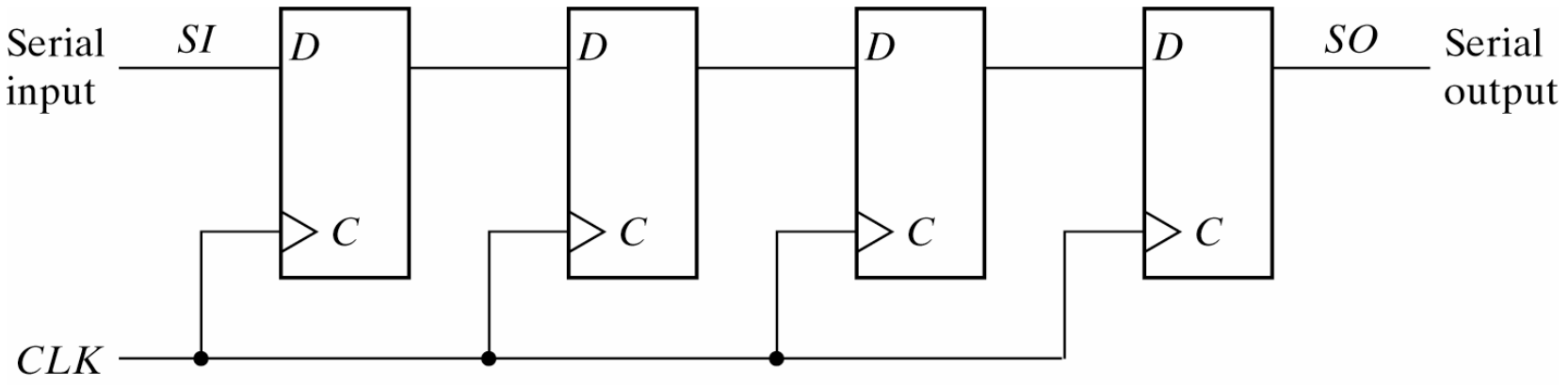
Shift Registers

- In a **shift register** flip-flops are connected in series, with each output connected to the input of another.
- Upon each rising clock edge the contents of each flip-flop is passed to the flip-flop to the right.



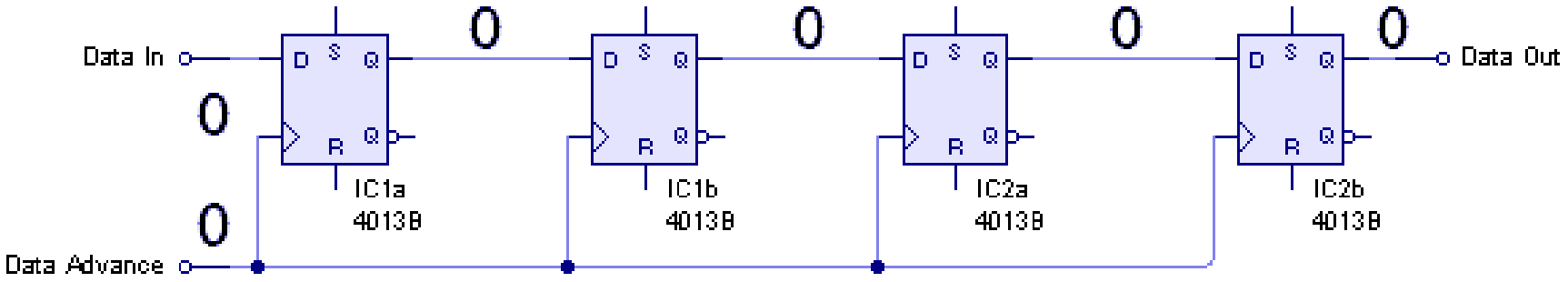
Shift Registers

- So as the clock advances the data is fed into the register via the serial input.
- The data passes into the flip-flop and out again after enough clock cycles.



Shift Registers

- In this animation the register initially stores 0000
- The word 1001 is appears sequentially at the input.
- Upon each positive edge the 1001 passes into the register one bit at a time.
- 1001 is then shifted out of the register one bit at a time.

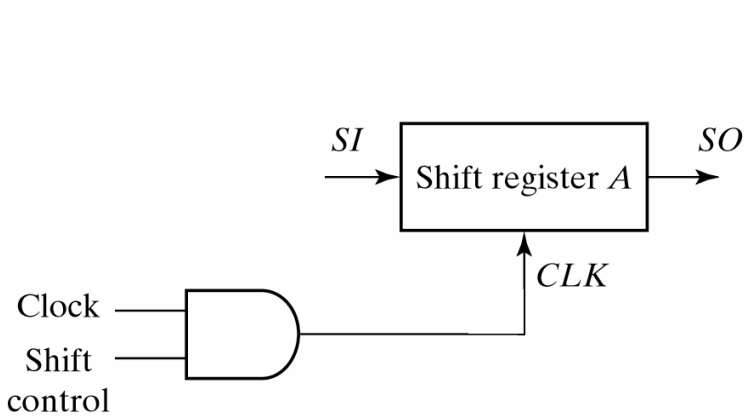


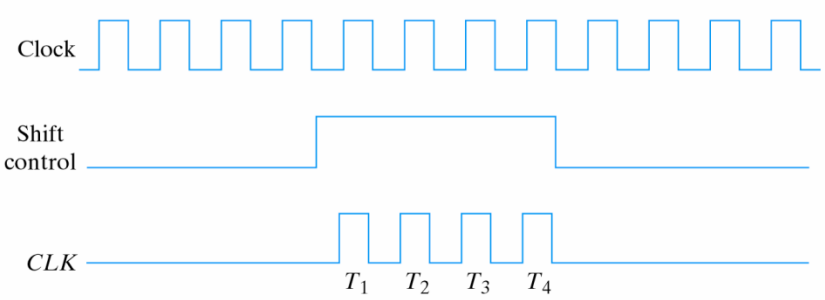
Shift Registers

- Question
  - How can a shift register be used as a sequence detector?

Shift Registers

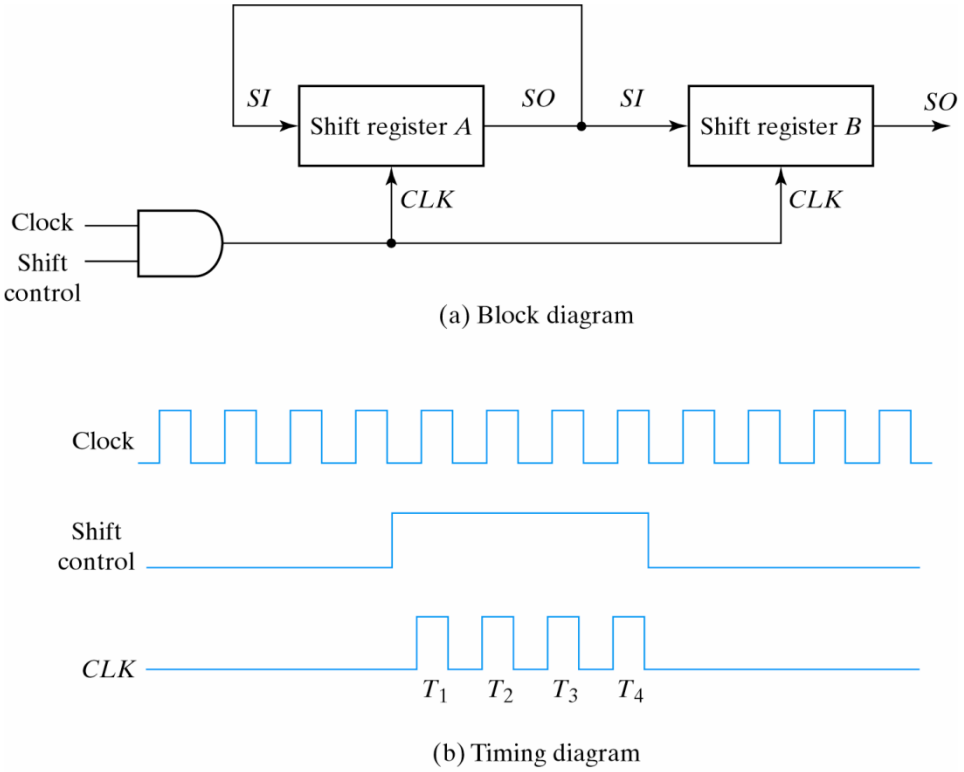
- In the shift register a load control can be used to prevent the clock from affecting the register unnecessarily.
- And as with the parallel register the clock can be prevented from affecting the register with the addition of a **shift** control.





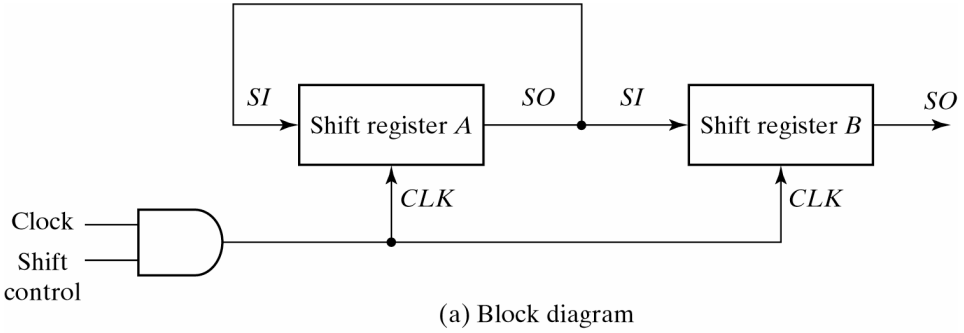
Serial Transfer

- A common operation in digital circuits is the transfer of data.
- Parallel transfer is straight-forward
  - All the bits of the register are transferred at the same time.
- Here, in serial transfer, the transfer of data is performed sequentially between two shift registers.



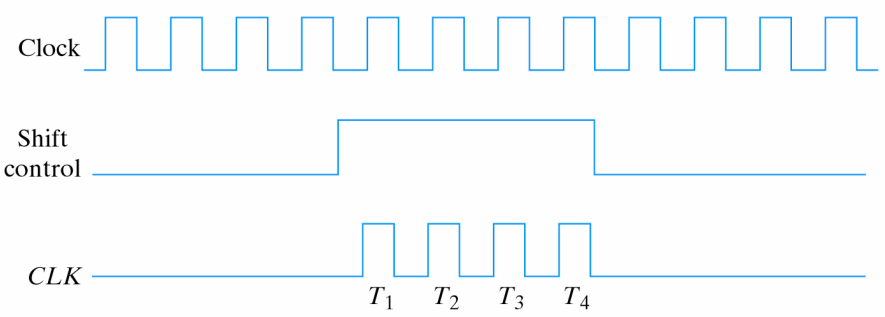
Serial Transfer

- Loss of information is prevented by connecting the output of the first register to its own input.



(a) Block diagram

Register A					Register B			
1	0	1	1		0	0	1	0
1	1	0	1	T_1	1	0	0	1
1	1	1	0	T_2	1	1	0	0
0	1	1	1	T_3	0	1	1	0
1	0	1	1	T_4	1	0	1	1



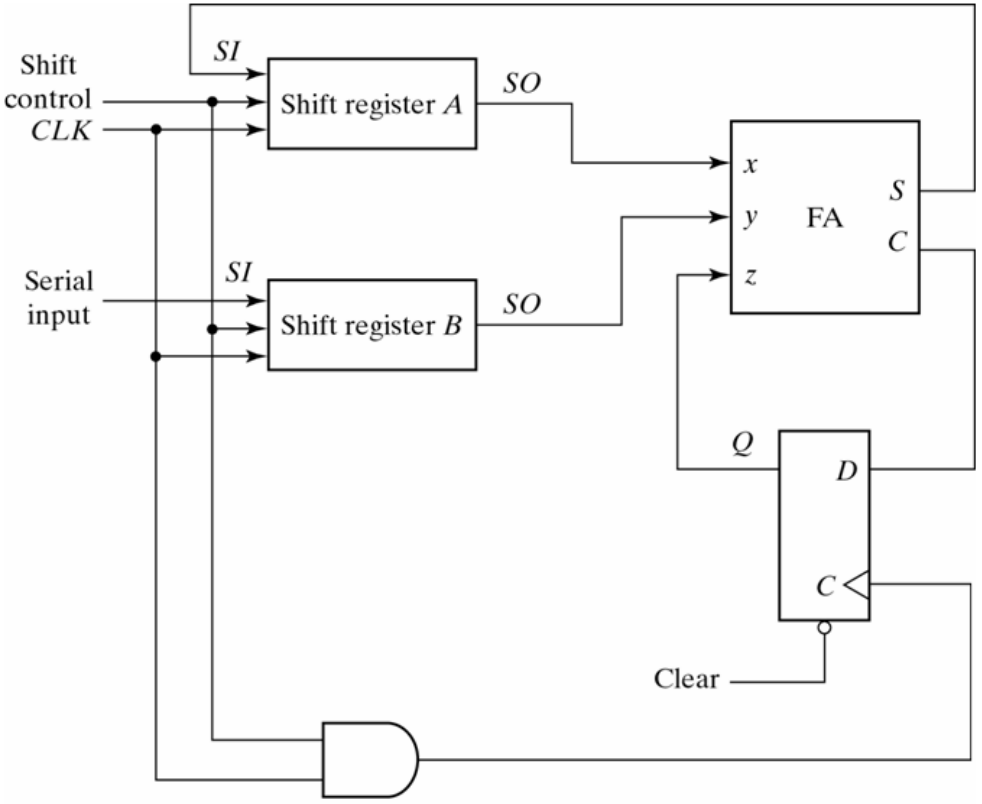
(b) Timing diagram

Serial Addition

- The contents of two registers may be added in a serial manner.

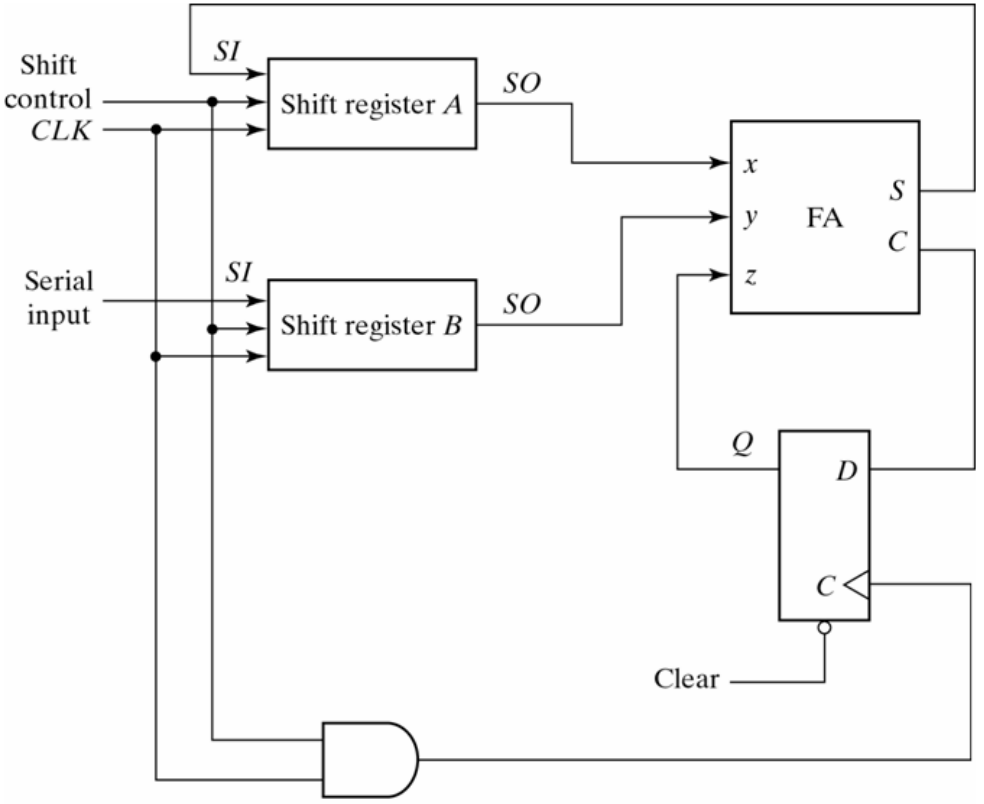
$$\begin{array}{rr} & 101011 \\ + & 011011 \\ \hline & 1000110 \end{array}$$

- Upon each clock edge two bits from the register are shifted out and inputted into the full adder.



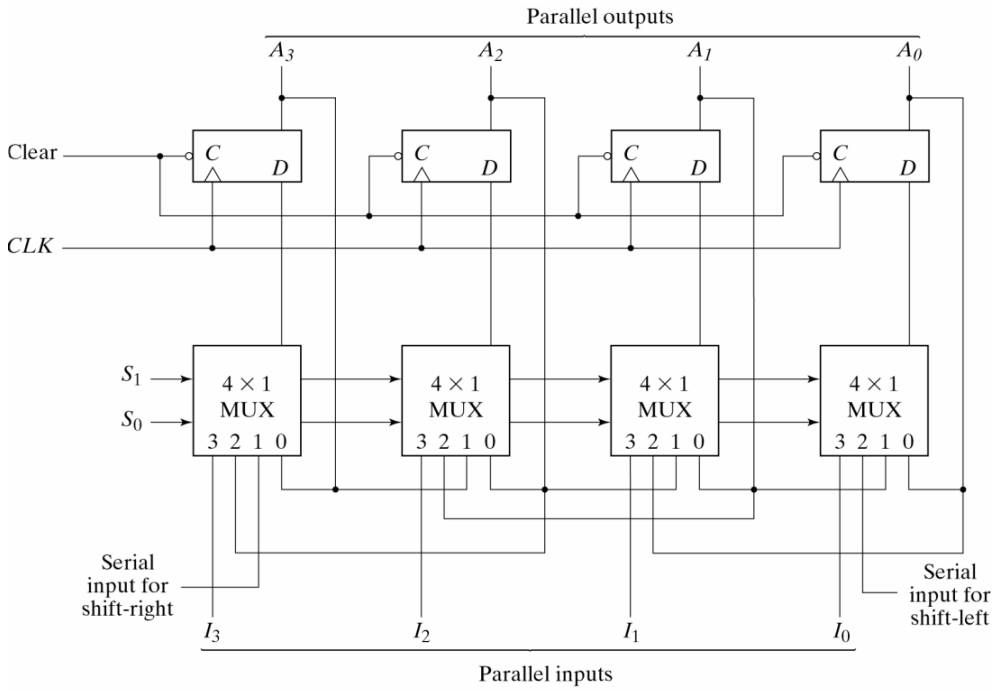
Serial Addition

- The carry output is stored in a flip-flop so it may be included in the summation on the next edge.
- Here the sum output is fed back into register A. It could also be fed to a third register somewhere else.



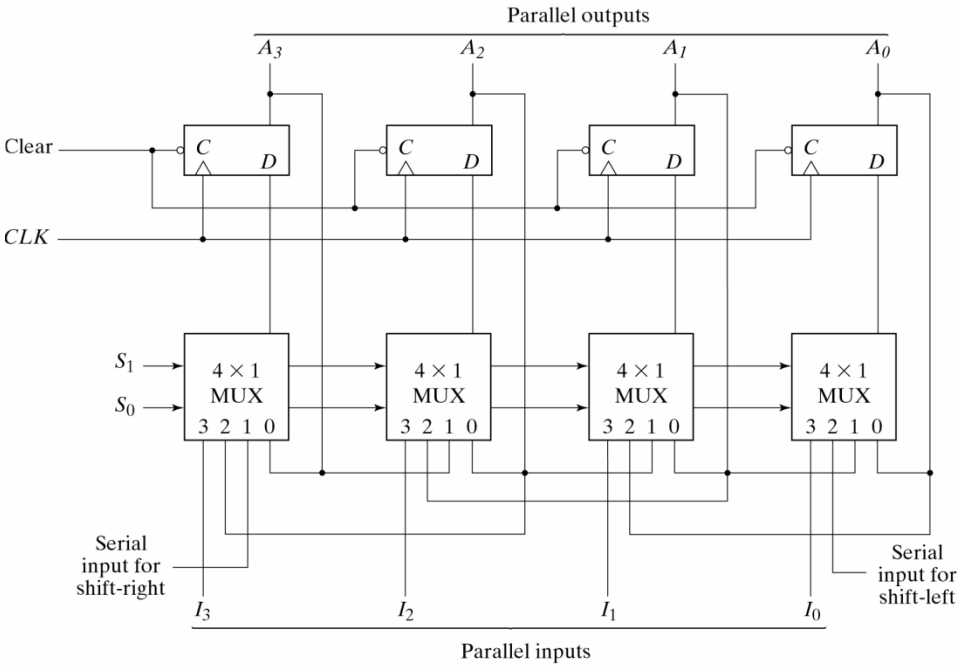
Universal Shift Register

- We have seen registers with parallel inputs, serial inputs, parallel outputs and serial outputs.
- In general a register may have all of these capabilities.



Universal Shift Register

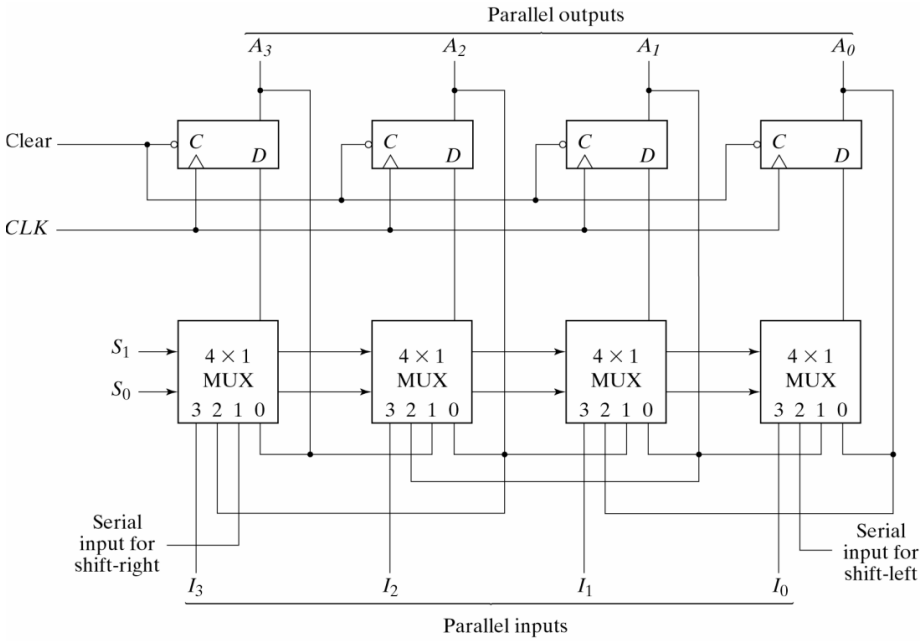
- Here we have a 4 bit register which may
  - **clear** data
  - synchronize to a **clock**
  - **control** the application the clock signal
  - **shift** data **right**
  - **shift** data **left**
  - accept **serial** or **parallel inputs**
  - accept **serial** or **parallel outputs**



Universal Shift Register

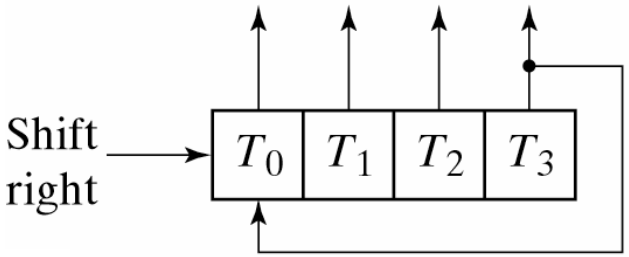
- To switch between these operations four multiplexers are used.

S_1	S_0	MUX	
0	0	0	No change - the outputs are fed back into the flip-flops
0	1	1	shift right – the outputs are fed to the inputs on the right
1	0	2	shift left – the outputs are fed to the inputs on the left
1	1	3	parallel load is fed into flip-flops

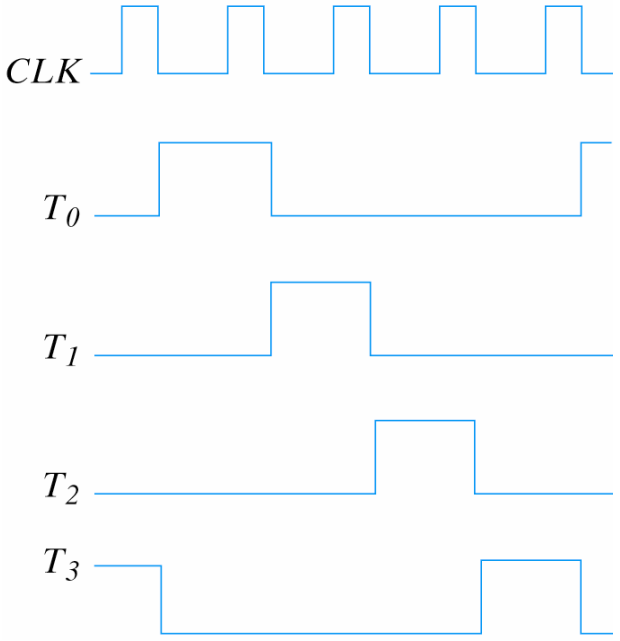


Ring Counter

- A ring counter is a **circular shift register** with only one flip-flop being set at any particular time; all others are cleared.
- The last output is connected to the first input.
- The single bit is shifted from one flip-flop to the next to produce the sequence of timing signals.



Ring-counter (initial value = 1000)

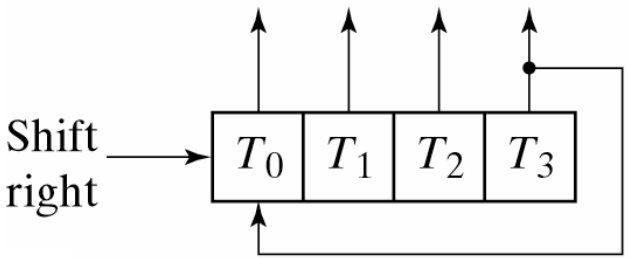


Ring Counter

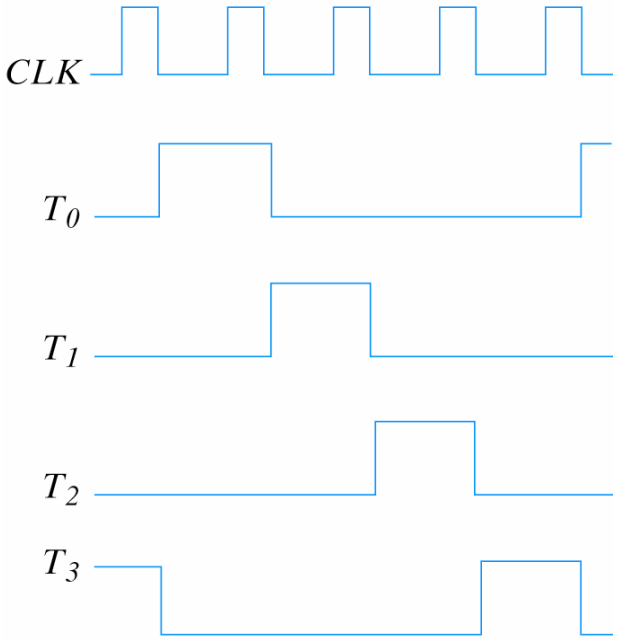
- The initial value of the register is 1000.
- Upon each negative edge the one is shifted to the right

1000>0100>0010>0001>1000

- To generate 2^n timing signals, we need a shift register with 2^n flip-flops.

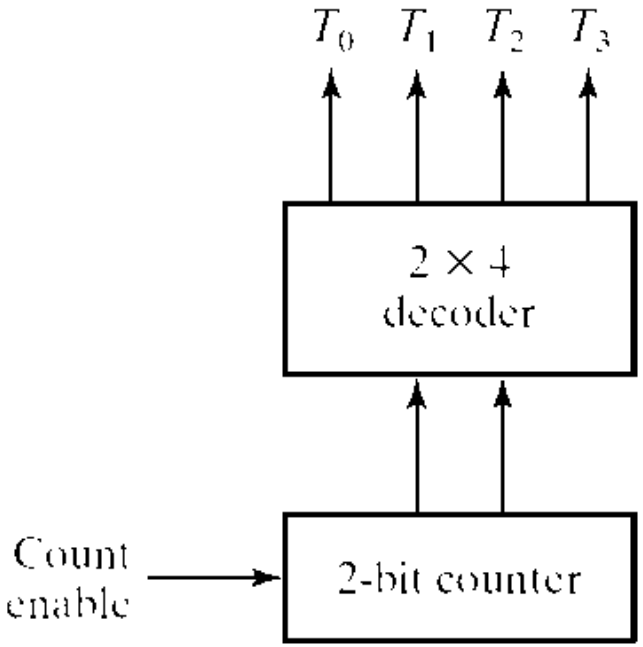


Ring-counter (initial value = 1000)



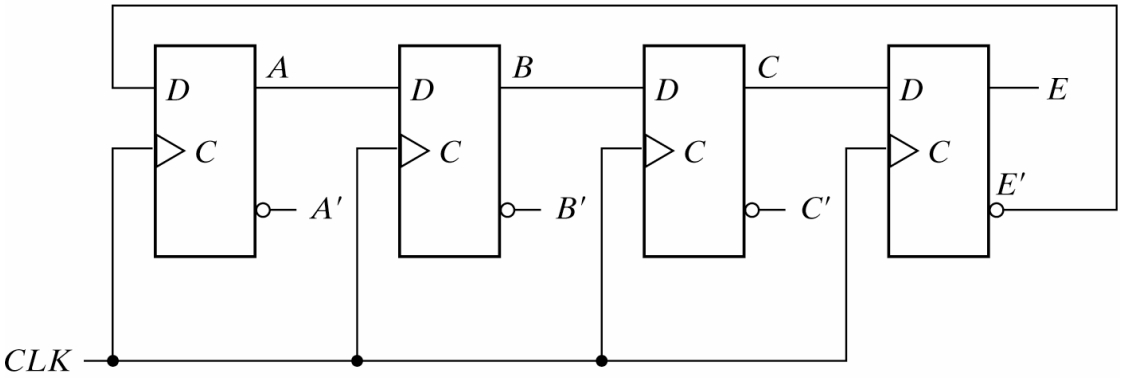
Ring Counter

- For an alternative design, the **2^n** timing signals can be generated by an **n**-bit binary counter together with an **n-to-2^n**-line decoder.



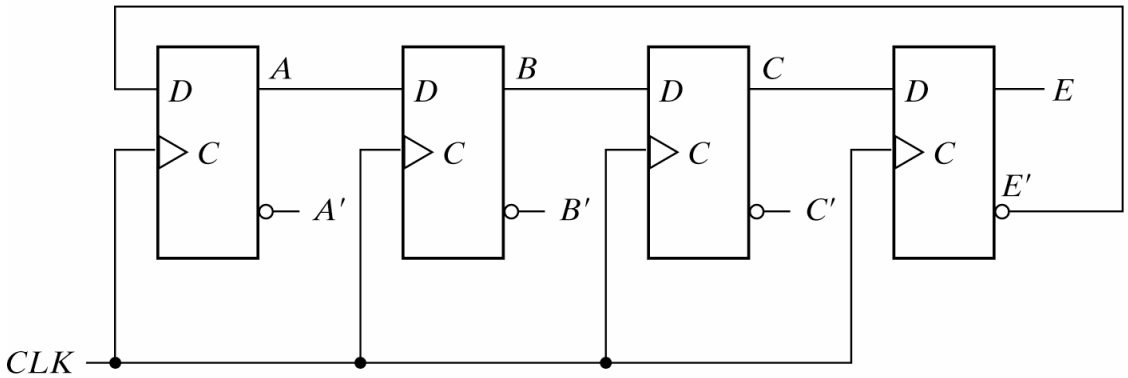
Johnson Counter

- There are *k* distinguishable states in a *k*-bit ring counter.
- This number of distinguishable states can be **doubled** if the shift register is connected as a **switch-tail ring counter:** the **complement** output of the last flip-flop is connected to the input of the first.
- These distinguishable states can be decoded by an decoder.
- A combination of a shift register and decoder is known as a **Johnson counter.**



Johnson Counter

- The counter begins at the state 0000
- The complement of the E output is passed to the first input resulting in next state 1000.
- On the next edge the A output is passed to B resulting in the state 1100.



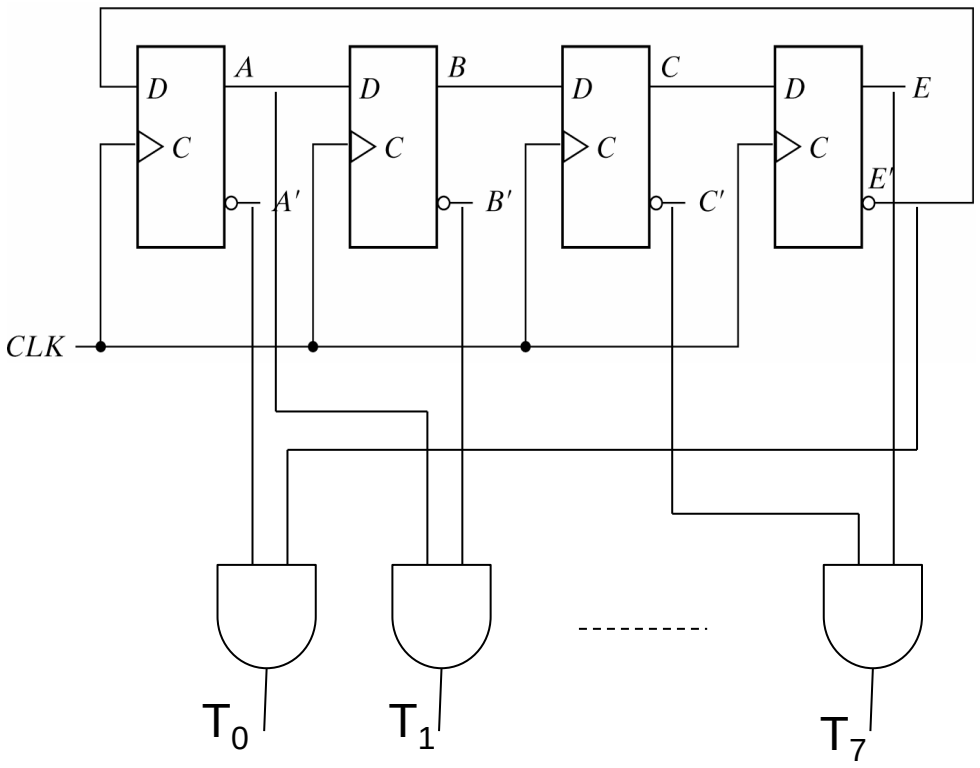
	A	B	C	E
t_0	0	0	0	0
t_1	1	0	0	0
t_2	1	1	0	0
t_3	1	1	1	0
t_4	1	1	1	1
t_5	0	1	1	1
t_6	0	0	1	1
t_7	0	0	0	1
t_8	0	0	0	0

Johnson Counter

- A k-bit switch-trail ring counter will go through a sequence of 2k states.
- These 2k states can be decoded by 2k decoding gates to provide 2k timing signals.
- The decoding gates can be two-input AND gates.

	A	B	C	E	decoding
t_0	0	0	0	0	A'E'
t_1	1	0	0	0	AB'
t_2	1	1	0	0	BC'
t_3	1	1	1	0	CE'
t_4	1	1	1	1	AE
t_5	0	1	1	1	A'B
t_6	0	0	1	1	B'C
t_7	0	0	0	1	C'E

Johnson Counter



	A	B	C	E	decoding
t_0	0	0	0	0	A'E'
t_1	1	0	0	0	AB'
t_2	1	1	0	0	BC'
t_3	1	1	1	0	CE'
t_4	1	1	1	1	AE
t_5	0	1	1	1	A'B
t_6	0	0	1	1	B'C
t_7	0	0	0	1	C'E