DIGITAL TECHNICS

Dr. Bálint Pődör

Óbuda University,
Microelectronics and Technology Institute

7. LECTURE: REGISTERS, COUNTERS AND SERIAL ARITHMETIC CIRCUITS

2016/2017

1

7. LECTURE: REGISTERS, COUNTERS AND SERIAL ARITHMETIC CIRCUITS

1. Counters, general properties
2. Ripple counters and synchronous counters
3. Registers, general properties
4. Shift registers
5. Circular registers and counters based on them
6. Register applications: serial arithmetic circuits

COUNTERS (AND REGISTERS): AN INTRODUCTION

Counters and registers belong to the category of MSI sequential logic circuits. They have similar architecture, as both counters and registers comprise a cascaded arrangement of more than one flip flop with or without combinational logic devices. Both constitute very important building blocks of sequential logic, and different types of counter and register available in integrated circuit (IC) form are used in a wide range of digital systems.

While counters are mainly used in counting applications, where they either measure the time interval between two unknown time instants or measure the frequency of a given signal, registers are primarily used for the temporary storage of data present at the output of a digital circuit before they are fed to another digital circuit.

COUNTERS: GENERAL CONCEPTS

- The counter is a special case of the sequential circuits.
- Its function is to count the input pulses (clock signal) and store the result till the arrival of the next signal.
- The counting process, therefore consists of a series of storage and addition operations.
- The counters are built from flip-flops and of gate circuits.

CLASSIFICATION AND PROPERTIES

According to the direction of counting:

- up counter
- down counter
- up-down counter

According to state encoding:

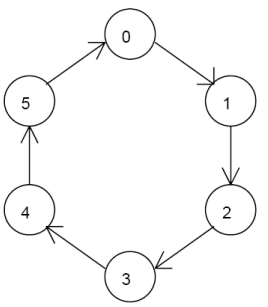
- binary
- decade (e.g. BCD)
- other

STATES OF A COUNTER

The state transition diagram of counters is of the form of a closed ring.
Example: mod 6 counter

The number of states, the counter goes through before recycling is the modulus of the counter.

The largest possible modulus of a n-bit counter is 2^n.



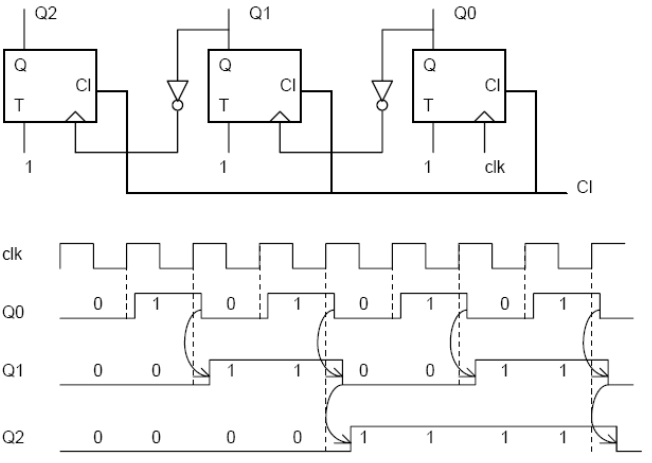
ASYNCHRONOUS AND SYNCHRONOUS COUNTERS

Many different types of electronic counters are available. They are all either asynchronous or synchronous type and are usually constructed using JK flip-flops.

• Asynchronous (ripple) counter: The input signal is applied to the clock input of the first FF, and the output of each FF is connected directly to the clock input of the next.

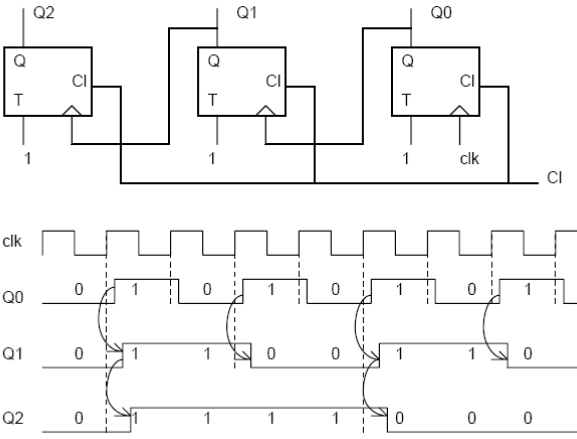
• Synchronous counter: all flip-flops are controlled by a common clock. Logic gates between each stage of the circuit control dataflow from stage to stage so that the desired count behaviour is realized.

RIPPLE COUNTER (UP)



Asynchronous binary up-counter with asynchronous clear. The delay of the outputs with respect to the clock input is increasing at each new stage.

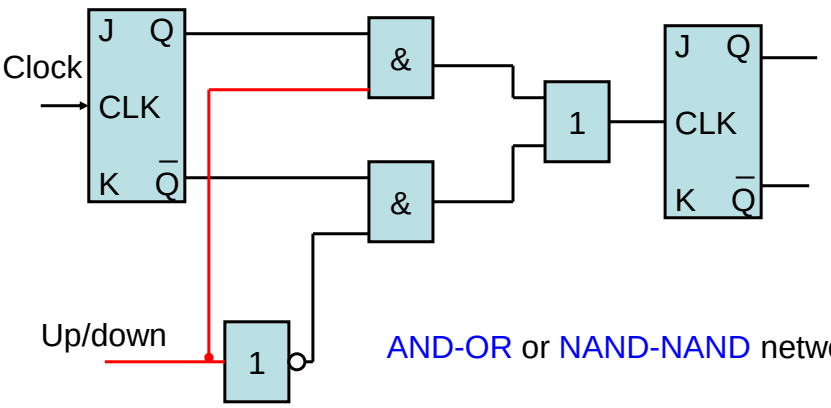
RIPPLE COUNTER (DOWN)



Asynchronous binary down-counter with asynchronous clear. The delay of the outputs with respect to the clock input is increasing at each new stage.

UP/DOWN BINARY RIPPLE COUNTER WITH JK FLIP-FLOPS

Scheme of the up/down control between each stages.



AND-OR or NAND-NAND network.

RIPPLE COUNTER: PROPAGATION DELAY

A major problem with ripple counters arises from the propagation delay of the flip-flops constituting the counter. The effective propagation delay in a ripple counter is equal to the sum of propagation delays due to different flip-flops. The situation becomes worse with increase in the number of flip-flops used to construct the counter, which is the case in larger bit counters.

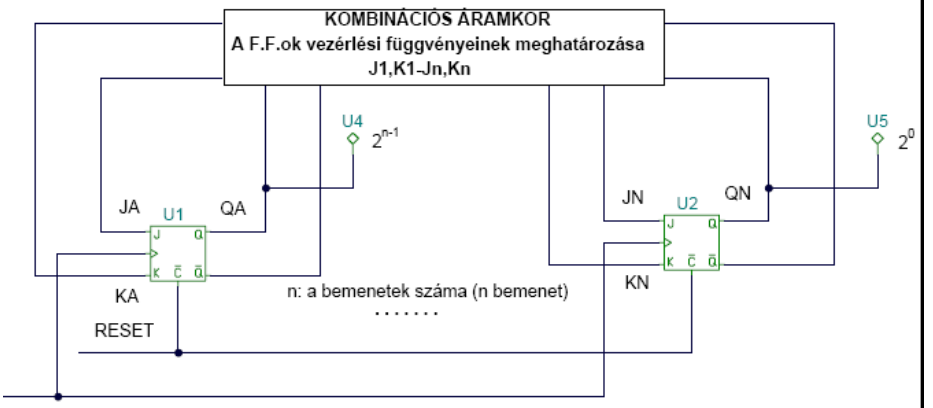
An increased propagation delay puts a limit on the maximum frequency used as clock input to the counter. We can appreciate that the clock signal time period must be equal to or greater than the total propagation delay. The maximum clock frequency therefore corresponds to a time period that equals the total propagation delay. If t_{pd} is the propagation delay in each flip-flop, then, in a counter with N flip-flops having a modulus of less than or equal to 2^N , the maximum usable clock frequency is given by $f_{max} = 1/(N \times t_{pd})$. Often the propagation delay times are specified in the case of flip-flops, one for LOW-to-HIGH transition (t_{pLH}) and the other for HIGH-to-LOW transition (t_{pHL}) at the output. In such a case, the larger of the two should be considered for computing the maximum clock frequency.

SYNCHRONOUS COUNTERS

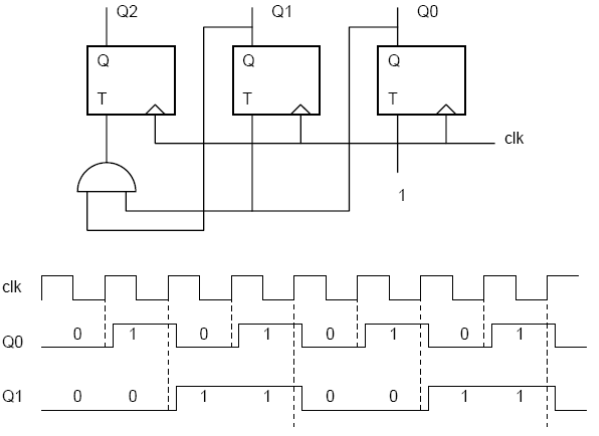
The propagation delay becomes prohibitively large in a ripple counter with a large count. On the other hand, in a synchronous counter, all flip-flops in the counter are clocked simultaneously in synchronism with the clock, and as a consequence all flip-flops change state at the same time. The propagation delay in this case is independent of the number of flip-flops used.

Since the different flip-flops in a synchronous counter are clocked at the same time, there needs to be additional logic circuitry to ensure that the various flip-flops toggle at the right time.

GENERAL SCHEMATICS OF SYNCHRONOUS COUNTER

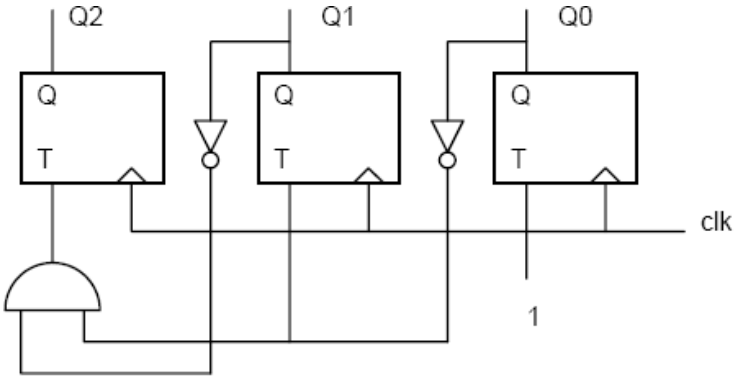


SYNCHRONOUS COUNTER (UP)



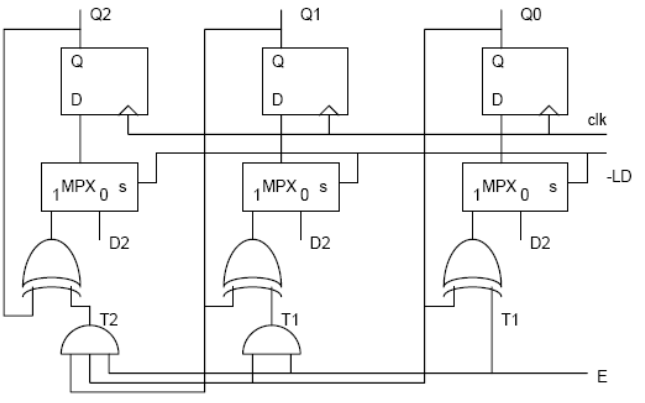
Functional diagram of the synchronous binary up-counter. The outputs change their states simultaneously, their delay with respect to the clock is small.

SYNCHRONOUS COUNTER (DOWN)



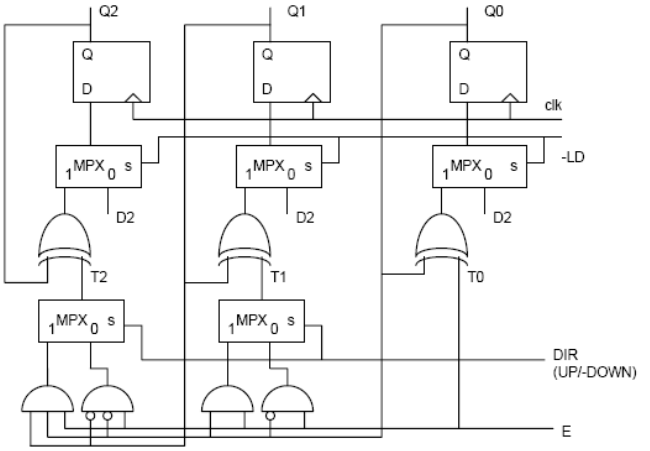
Functional diagram of the synchronous binary down-counter.

SYNCHRONOUS BINARY COUNTER



Functional diagram of the synchronous binary up-counter, with additional enable and (pre-)load functions.

SYNCHRONOUS BINARY UP/DOWN COUNTER



Functional diagram of the synchronous binary up/down counter, with additional enable and (pre-)load functions

SYNCR. VS ASYNCR. COUNTERS

It can be seen that a ripple counter requires less circuitry than a synchronous counter. No logic gates are used at all in the example above. Although the asynchronous counter is easier to construct, it has some major disadvantages over the synchronous counter.

First of all, the asynchronous counter is slow. In a synchronous counter, all the flip-flops will change states simultaneously while for an asynchronous counter, the propagation delays of the flip-flops add together to produce the overall delay. Hence, the more bits or number of flip-flops in an asynchronous counter, the slower it will be.

SYNCRONOUS VERSUS ASYNCRONOUS COUNTERS

Secondly, there are certain "risks" when using an asynchronous counter. In a complex system, many state changes occur on each clock edge and some ICs respond faster than others. If an external event is allowed to affect a system whenever it occurs (unsynchronized), there is a small chance that it will occur near a clock transition, after some IC's have responded, but before others have. This intermingling of transitions often causes erroneous operations. And the worse this is that these problems are difficult to foresee and test for because of the random time difference between the events.

REGISTERS

Registers are devices which are used to store and/or shift data entered from external sources. They are constructed by connecting a number of flip-flops in cascade.

A single flip-flop can store 1 bit of data, thus an n-bit register will require n flip-flops.

In a digital system such registers are generally used for temporary storage of data.

REGISTERS: PROPERTIES AND CLASSIFICATION

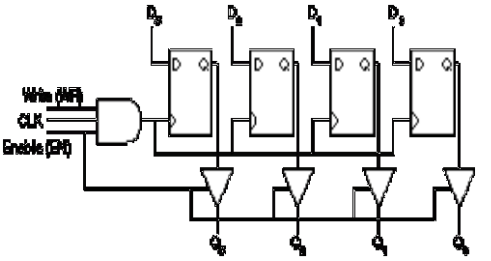
Classification according to internal structure and operation/function:

- storage register;
- shift register

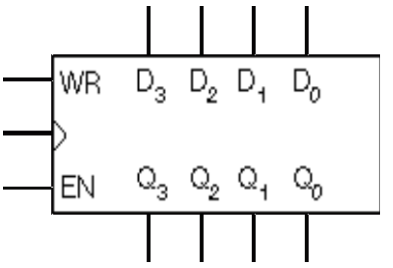
Storage register: it takes data from parallel inputs and copies it to the corresponding output when the registers are clocked. It can be used as a kind of “history”, retaining old information as the input in another part of the system, until ready for new information, whereupon, the registers are clocked, and the new data is “let through”. It is usually built using D flip-flops.

STORAGE REGISTERS

Gate-Level View



Chip-Level View

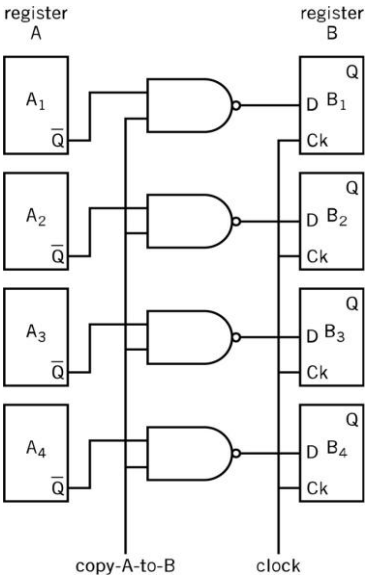


USES OF STORAGE REGISTERS

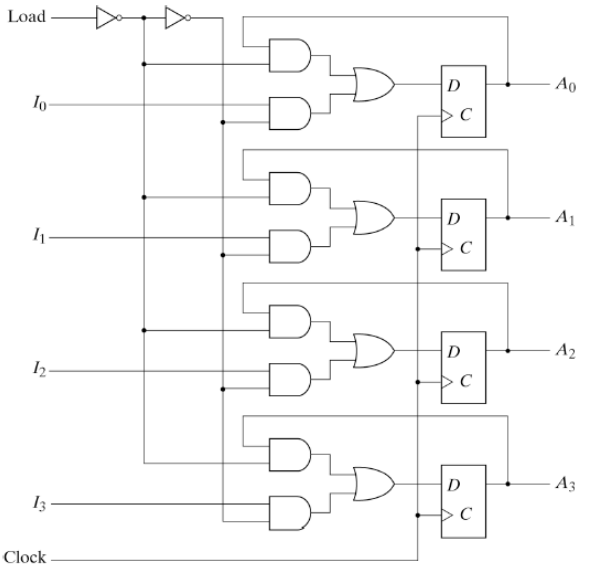
- arithmetical units;
- temporary storage between counter and display;
- code and signal conversion operations;
- input/output storage registers in μPs;
- intermediate storage functions in arithmetic/logic units (ALU);
- various types of other temporary storage functions.

EXAMPLE: REGISTER COPY OPERATION

- Uses both sequential and combinatorial logic

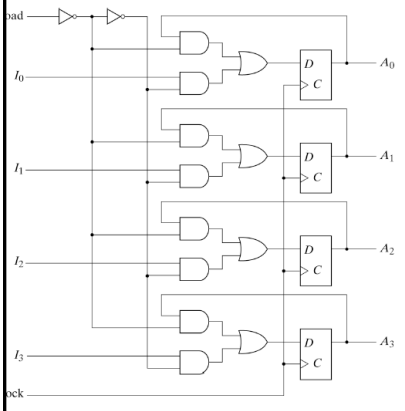


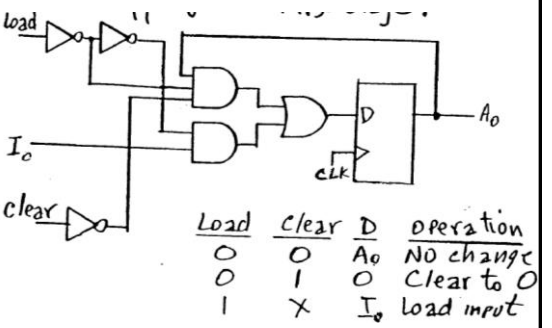
REGISTER: LOAD/HOLD



25

REGISTER: LOAD/HOLD/CLEAR





Load	Clear	D	operation
0	0	A_0	No change
0	1	0	Clear to 0
1	X	I_o	load input

26

SHIFT REGISTERS

- The shift function of a register allows the stored data to be moved serially from stage to stage or into or out if the register.
- Types of data movement:
  - serial shift right or left
  - parallel shift in to and out
- The shift is executed by the synchronizing or clock signal.
- Register are used for conversion of data form serial to parallel and vice versa. Also used as counters.
- In shift registers the flip-flop types used should not be transparent. Usually master-slave types are used.

REGISTERS: DESIGN ASPECTS

Collection of flip-flops with similar controls and logic

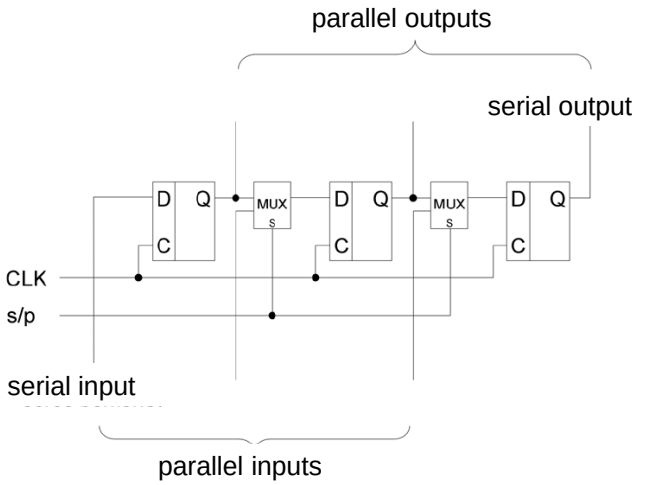
Stored values somehow related (e.g. form binary value)

Share clock, reset, and set lines

Similar logic at each stage

28

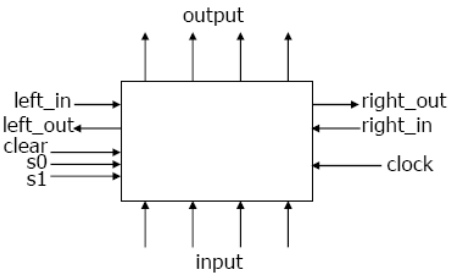
SERIES/PARALLEL INPUT REGISTER



29

UNIVERSAL SHIFT REGISTER

- Holds 4 values
  - serial or parallel inputs
  - serial or parallel outputs
  - permits shift left or right
  - shift in new values from left or right



clear sets the register contents and output to 0

s1 and s0 determine the shift function

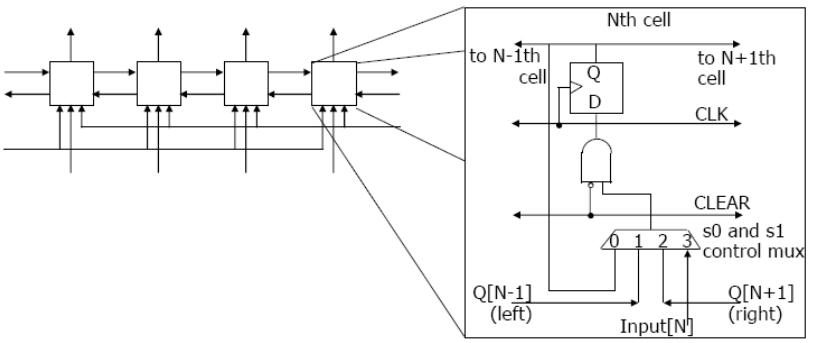
s0	s1	function
0	0	hold state
0	1	shift right
1	0	shift left
1	1	load new input

30

DESIGN OF UNIVERSAL SHIFT REGISTER

- Consider one of the four flip-flops
  - new value at next clock cycle:

clear	s0	s1	new value
1	–	–	0
0	0	0	output
0	0	1	output value of FF to left (shift right)
0	1	0	output value of FF to right (shift left)
0	1	1	input



31

SHIFT REGISTERS: CHARACTERISTIC EQUATIONS

Right shift register:

$$Q_i^n = Q_{i+1}^{n-1}$$

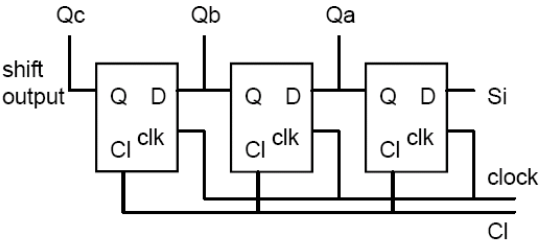
Left shift register:

$$Q_i^n = Q_{i-1}^{n-1}$$

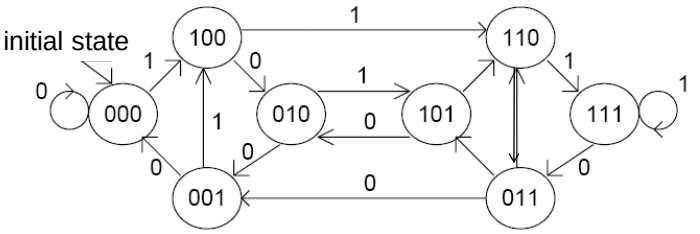
Bi-directional shift register
Right sift (M=1) - left shift (M = 0):

$$Q_i^n = \overline{M}\, Q_{i-1}^{n-1} + M\, Q_{i+1}^{n-1}$$

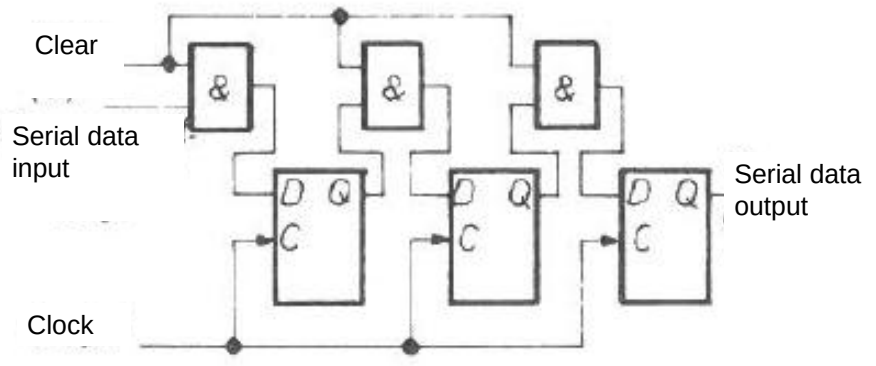
STATE TRANSITION DIAGRAM: LEFT SHIFT REGISTER



State transition diagram

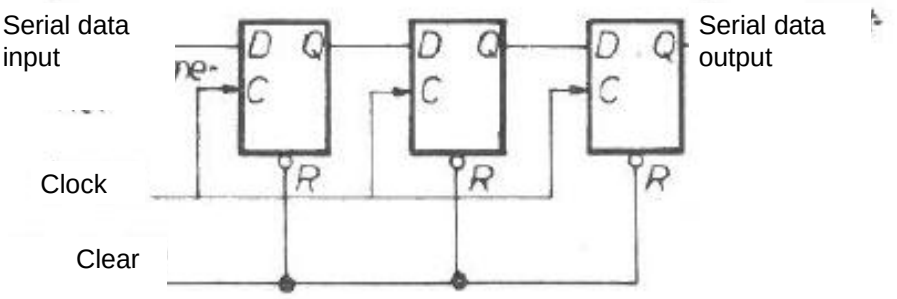


SHIFT REGISTER: SYNCHRONOUS CLEAR



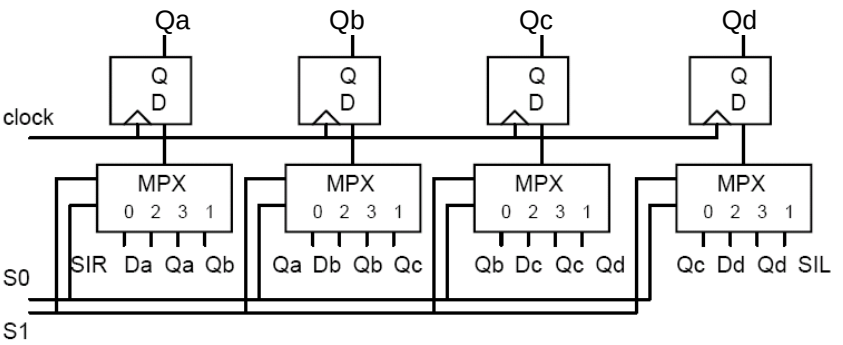
34

SHIFT REGSTER: ASYNCHRONOUS CLEAR



35

BI-DIRECTIONAL SHIFT REGISTER

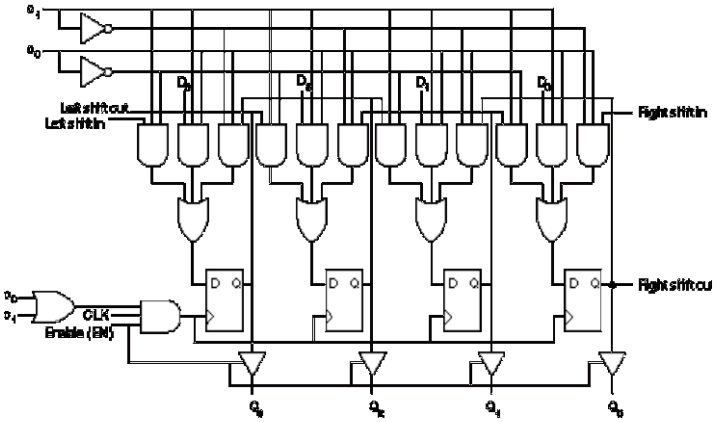


S1S0

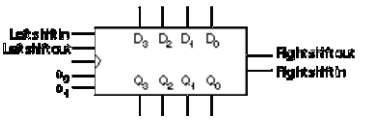
0 0 shift right (SHR)
0 1 shift left (SHL)
1 0 load (LOAD)
1 1 hold (HOLD)

Functional diagram of bi-directional shift register with load and hold facility

A LEFT-RIGHT SHIFT REGISTER WITH PARALLEL READ AND WRITE



Control s_1	s_0	Function
0	0	No change
0	1	Shift left
1	0	Shift right
1	1	Parallel load



SHIFT REGISTER COUNTERS

Both counters and shift registers are some kinds of cascade arrangement of flip-flops. A shift register, unlike a counter, has no specified sequence of states. However, if the serial output of the shift register is fed back to the serial input, we do get a circuit that exhibits a specified sequence of states. The resulting circuits are known as *shift register counters.*

Shift register counter – a circuit formed by a shift registers and combinational logic. The state diagram for this state machine is cyclic. This circuit does not necessarily count in ascending or descending order.

Ring counter – the simplest shift register counter. This circuit uses a *n*-bit shift register to obtain a counter with *n* states

SHIFT REGISTER COUNTERS

Shift register or ring counter is a counter composed of a circular shift register. The output of the last flop-flop is fed to the input of the first flip-flop.

Two types of ring counters:

Straight ring counter or Overbeck counter

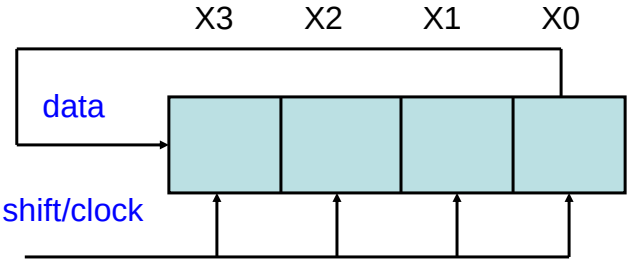
Twisted ring counter or Johnson or Möbius counter

RING COUNTER AND JOHNSON COUNTER

Straight ring counter or Overbeck counter connects the output of the last FF two the first FF input and circulates one (or zero) bit around the counter. One of the FFs must be pre-loaded with a 1 in order to operate properly.

Twisted ring counter or Johnson counter connects the complement of the output of the last FF to the input FF and circulates a stream of ones followed by zeros around the ring.

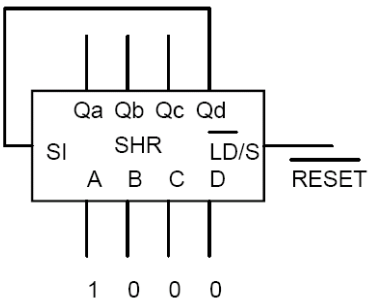
CIRCULAR REGISTER (1)



A circular register is formed by connecting (feeding back) the output of the last flip-flop to the (D-) input of the first flip-flop. In the circular register the clock keeps the binary information rotating. The bit pattern can be loaded in a parallel way.

RING COUNTER

The circular register can be operated as a ring counter. The counter is initially preset, e.g. by loading 1 into the first flip-flop. Then the *modulus* of the counter is equal to the number of flip-flops N. Appropriately changing the pre-loaded bit pattern, the *modulus* can be decreased.

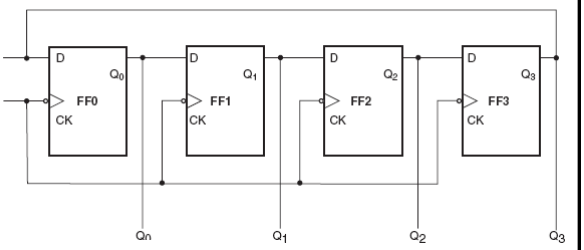


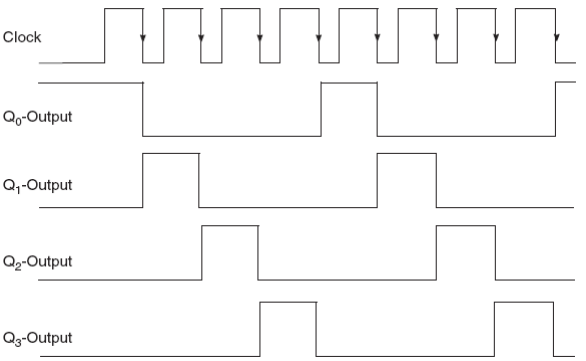
Encoding:

Qa	Qb	Qc	Qd
1	0	0	0
0	1	0	0
0	0	1	0
0	0	0	1

E.g. depending on the preloaded code word, an 4-bit ring counter can exhibit six different cycles.

4-BIT RING COUNTER OPERATION



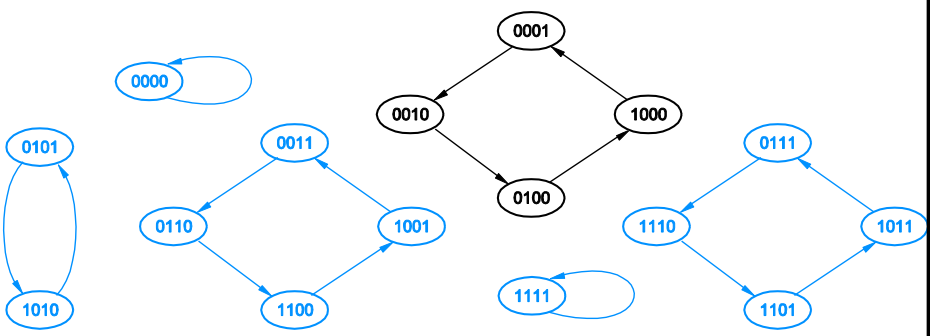


Timing waveforms

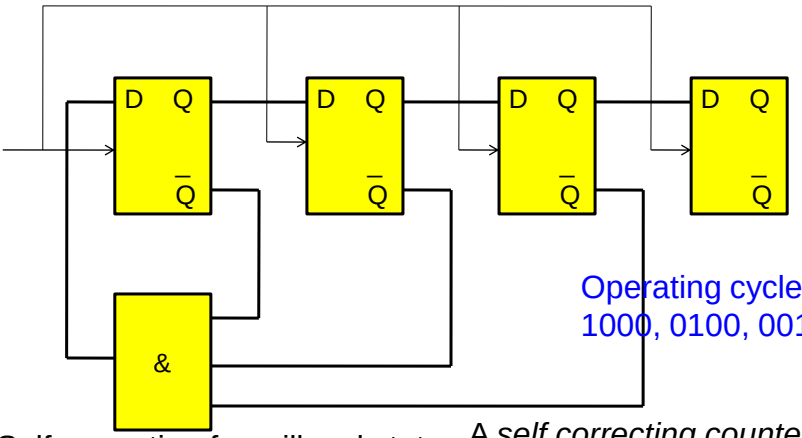
43

4-BIT RING COUNTER CYCLES

Depending on the preloaded code word, an 4-bit ring counter can exhibit six different cycles:



SELF CORRECTING RING COUNTER

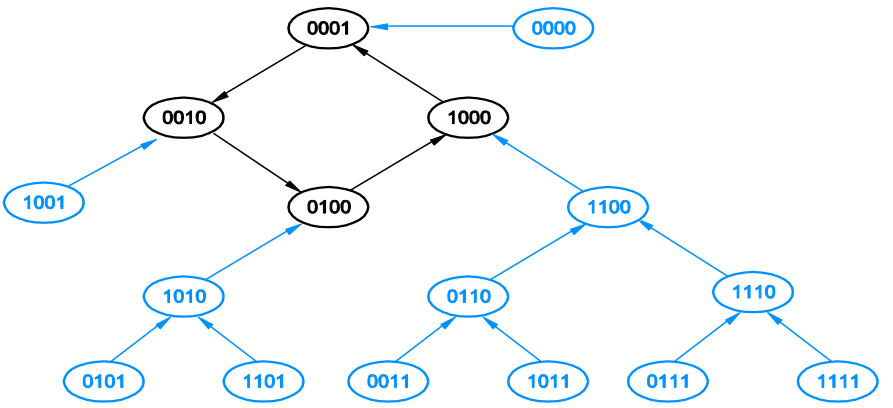


Operating cycle:
1000, 0100, 0010, 0001

Self correction from illegal state:
e.g. 1110⇒0111⇒0011⇒0001

A *self correcting counter* is designed so that all abnormal states have transitions leading to normal states.

SELF CORRECTING RING COUNTER

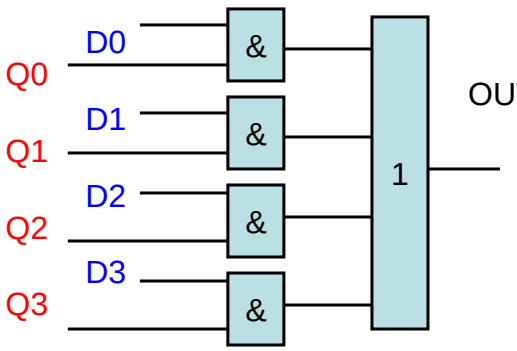


State transition diagram of the 4-bit self-correcting ring counter

RING COUNTER APPLICATION

A ring counter, instead of counting with binary numbers, counts with words that have a single high bit. These are ideal for timing a sequence of digital operations.

An application: time-division multiplexor



OUT: D0 D1 D2 D3 D0 D1 …

JOHNSON COUNTERS

A Johnson counter is a special case of a shift register, where the output of the last stage is inverted and fed back to the first stage.

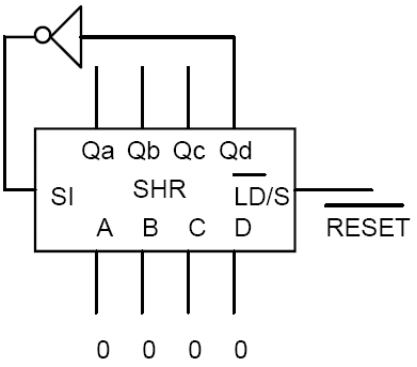
A pattern of bits equal to the length of the shift register thus circulates indefinitely.

These counters are sometimes called „walking ring" counters.

JOHNSON (MÖBIUS) COUNTER

Twisted-ring, Möbius or Johnson counter is a n-bit shift register whose serial input receives the complement of serial output. This counter has 2n states.

The feedback circuit is a single inverter. Beginning from a state of full 0s, the counter at first fills up itself with 1s, then with 0s. The modulus is 2N.

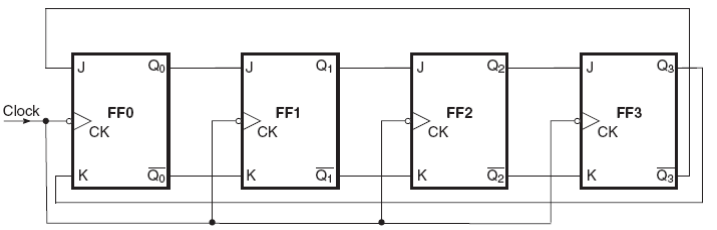


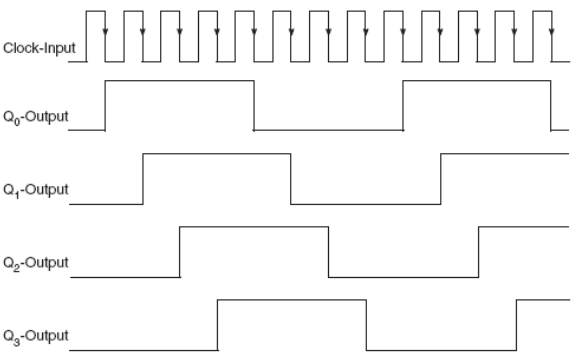
Encoding:

Qa, Qb, Qc, Qd:

0000, 1000, 1100, 1110, 1111, 0111, 0011, 0001

4-BIT JOHNSON COUNTER OPERATION

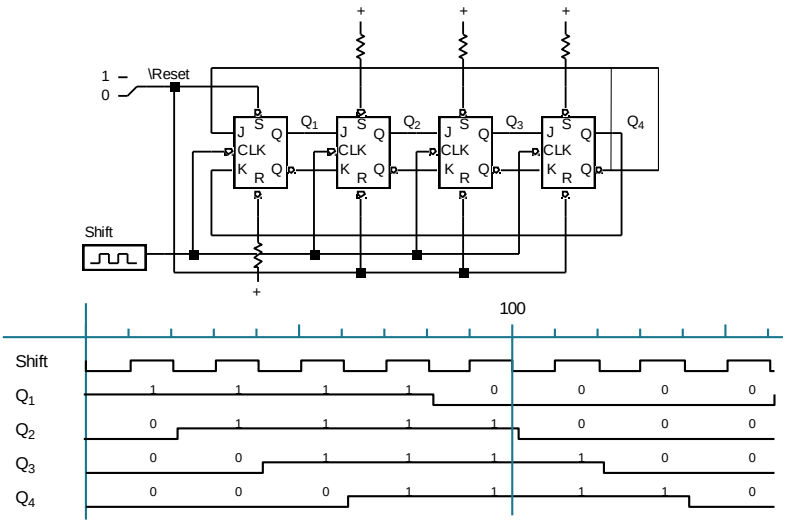




Timing waveforms

50

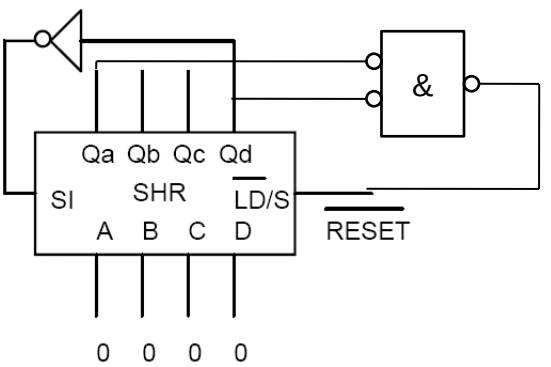
JOHNSON COUNTER



8 possible states, single bit change per state, useful for avoiding glitches (hazards)

SELF-CORRECTING JOHNSON COUNTER

Tetszőleges kezdőállapotból is belefut a normál ciklusba. Elv: A Johnson számláló előbb-utóbb előállít egy 0XX0 állapotot. Ez aktivizálja a LOAD (betöltés) funkciót, így beállítható a normál üzemmód.



52

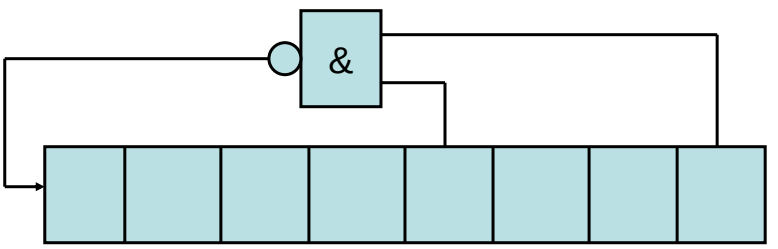
STRAIGHT AND TWISTED RING COUNTERS

Four-bit ring counter sequences

Straight ring/*Overbeck counter*					Twisted ring/*Johnson counter*				
State	Q0	Q1	Q2	Q3	State	Q0	Q1	Q2	Q3
0	1	0	0	0	0	0	0	0	0
1	0	1	0	0	1	1	0	0	0
2	0	0	1	0	2	1	1	0	0
3	0	0	0	1	3	1	1	1	0
0	1	0	0	0	4	1	1	1	1
1	0	1	0	0	5	0	1	1	1
2	0	0	1	0	6	0	0	1	1
3	0	0	0	1	7	0	0	0	1
0	1	0	0	0	*0*	0	0	0	0

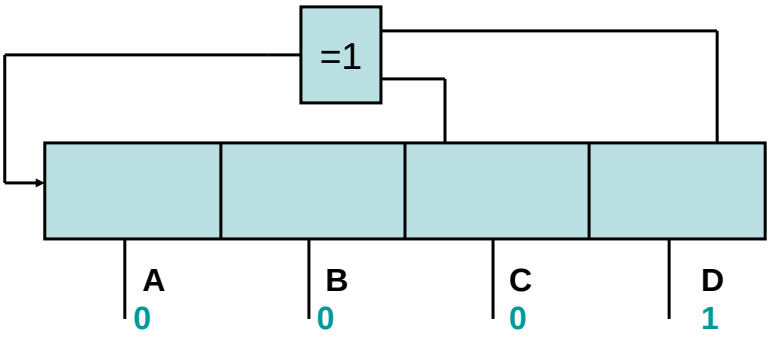
53

CIRCULAR REGISTER (2) (EXAMPLE)



The 8-bites circular register has 12 states with the feedback shown. Starting form the state 000...0, after the 4th then after the 16th clock cycle its state will be 11110000.

CIRCULAR REGISTER (2): RANDOM NUMBER GENERATOR



If the feedback network is an XOR gate, then using appropriate outputs of the register, the modulus of the counter will be 2^N -1.
For the circuit shown, with the given initial value the modulus will be the maximum possible i.e.15.

RANDOM NUMBER GENERATOR

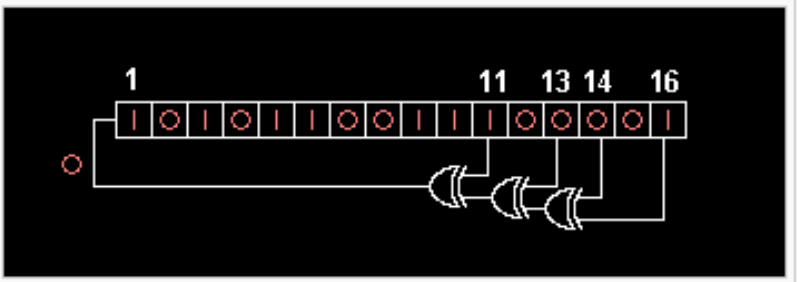
Important property: the sequence of codes is rather random.

(1) 0001, (8) 1000, (4) 0100, (2) 0010, (9) 1001, (12) 1100, (6) 0110, (11) 1011, (5) 0101, (10) 1010, (13) 1101, (14) 1110, (15) 1111, (7) 0111, (3) 0011.

No. of stages	No. of states	Serial No. of FFs to fed back
3	7	3, 2
4	15	4, 3 (shown above)
5	31	5, 3
6	63	6, 5
7	127	7, 6

(c.f. Benesóczky Z., Digitális tervezés funkcionális elemekkel ...)

FIBONACCI SERIES BASED FEEDBACK



A 16-bit Fibonacci LFSR. The feedback tap numbers in white correspond to a primitive polynomial in the table so the register cycles through the maximum number of 65535 states excluding the all-zeroes state. The state ACE1 hex shown will be followed by 5670 hex.

57

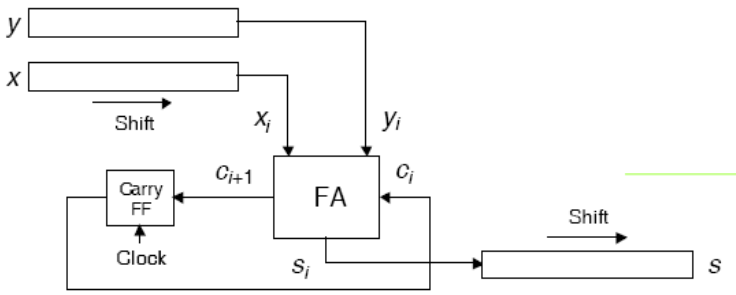
SERIAL ARITHMETICS

Register applications: serial arithmetic circuits

Bit-serial adder:
1-bit full adder plus registers

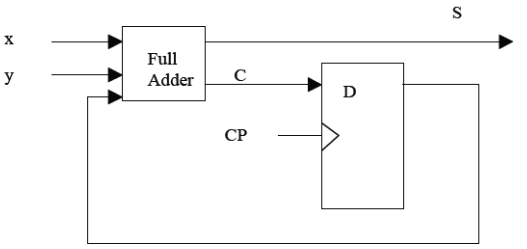
Serial/parallel multiplier (n-bit multiplicand, m-bit multiplier):
n-bit parallel adder (n 1-bit adders) plus registers

BIT-SERIAL ADDER

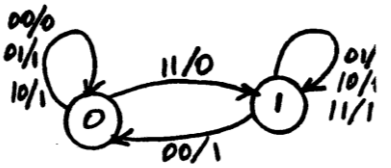


Functional diagram of the bit-serial adder.

OPERATION OF BIT-SERIAL ADDER



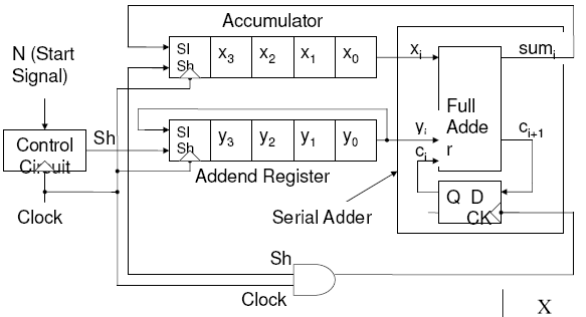
Present State Q	Inputs x	Inputs y	Next State Q	output S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1



$S = x \oplus y \oplus Q$

$Q(t+1) = xy + xQ + yQ$

SERIAL ADDER WITH ACCUMULATOR



0 1 0 1
0 1 1 1
1 1 0 0

	X	Y	c_i	sum_i	c_{i+1}
t_0	0101	0111	0	0	1
t_1	0010	1011	1	0	1
t_2	0001	1101	1	1	1
t_3	1000	1110	1	1	0
t_4	1100	0111	0	(1)	(0)

SERIAL/PARALLEL MULTIPLIER

Multiplication of binary numbers is usually implemented in microprocessors and microcomputers by using repeated addition and shift operations. Since the binary adders are designed to add only two binary numbers at a time, instead of adding all the partial products at the end, they are added two at a time and their sum is accumulated in a register called the accumulator register. Also, when the multiplier bit is '0', that very partial product is ignored, as an all '0' line does not affect the final result.

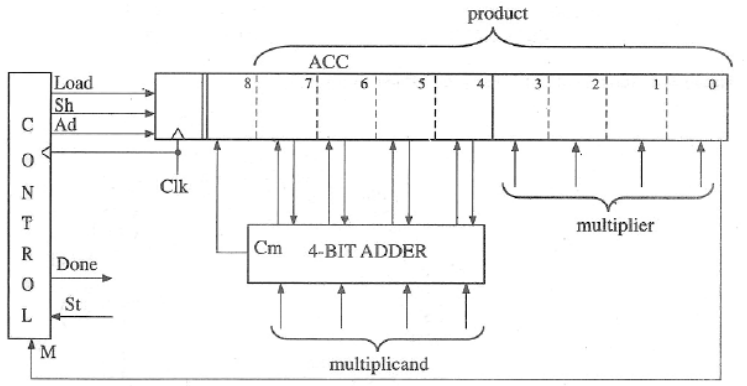
The basic hardware arrangement of such a binary multiplier would comprise shift registers for the multiplicand and multiplier bits, an accumulator register for storing partial products, a binary parallel adder and a clock pulse generator to time various operations.

4x4 BIT MULTIPLICATION (RECAPITULATION)

Partial Product Accumulation

				A3	A2	A1	A0
			Multiplicand	B3	B2	B1	B0
			Multiplier	A2 B0	A2 B0	A1 B0	A0 B0
			A3 B1	A2 B1	A1 B1	A0 B1	
		A3 B2	A2 B2	A1 B2	A0 B2		
	A3 B3	A2 B3	A1 B3	A0 B3			
S7	S6	S5	S4	S3	S2	S1	S0

4x4 BIT SERIAL/PARALLEL MULTIPLIER



Block diagram of a 4x4 bit serial/parallel multiplier

If multiplier bit 1 then add and shift

If multiplier bit 0 then shift

OPERATION OF THE MULTIPLIER (13x5)

Multiplicand 1 1 0 1 Mutiplier 0 1 0 1

8	7	6	5	4	3	2	1	0	
0	0	0	0	0	0	1	0	1	
0	1	1	0	1	0	1	0	1	ADD
0	0	1	1	0	1	0	1	0	SHIFT
0	0	0	1	1	0	1	0	1	SHIFT
1	0	0	0	0	0	1	0	1	ADD
0	1	0	0	0	0	0	1	0	SHIFT
0	0	1	0	0	0	0	0	1	SHIFT

13 x 5 = 65

MULTIPLICATION IN μP'S

Many microprocessors do not have in their ALU the hardware that can perform multiplication or other complex arithmetic operations such as division, determining the square root, trigonometric functions, etc. These operations in these microprocessors are executed through software. For example, a multiplication operation may be accomplished by using a software program that does multiplication through repeated execution of addition and shift instructions. Other complex operations mentioned above can also be executed with similar programs. Although the use of software reduces the hardware needed in the microprocessor, the computation time in general is higher in the case of software-executed operations when compared with the use of hardware to perform those operations.

PRACTICAL SSI/MSI EXAMPLE

Am25LS14A

8-Bit Serial/Parallel Two's Complement Multiplier

DISTINCTIVE CHARACTERISTICS

- Two's complement multiplication without correction
- Magnitude only multiplication
- Cascadable for any number of bits
- 8-bit parallel multiplicand data input
- 50MHz minimum clock frequency
- Second sourced by T.I. as the SN54LS/74LS384
- IMOX™ process with ECL internal

GENERAL DESCRIPTION

The Am25LS14A is an 8-bit by 1-bit sequential logic element that performs digital multiplication of two numbers represented in two's complement form to produce a two's complement product without correction. The device accepts an 8-bit multiplicand (X input) and stores this data in eight internal latches. The X latches are controlled via the clear input. When the clear input is LOW, all internal flip-flops are cleared and the X latches are opened to accept new multiplicand data. When the clear input is HIGH, the latches are closed and are insensitive to X input changes.

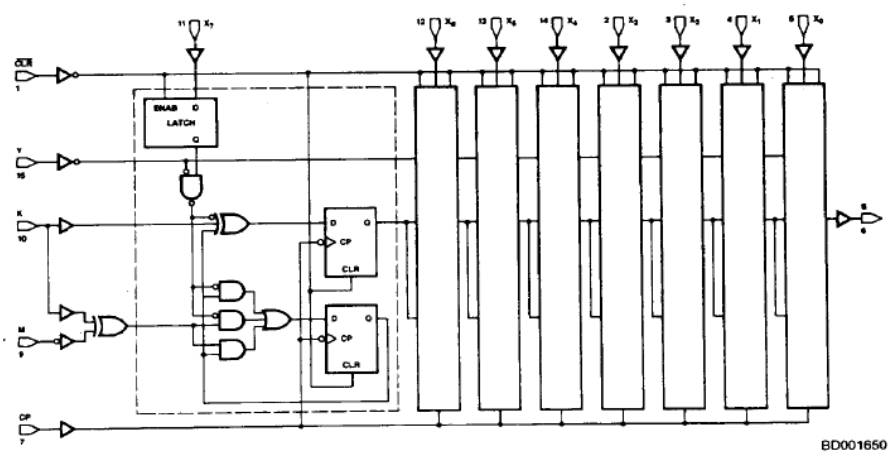
The multiplier word data is passed by the Y input in a serial bit stream – least significant bit first. The product is clocked out the S output least significant bit first.

GENERAL DESCRIPTION

The multiplication of an m-bit multiplicand by an n-bit multiplier results in an m + n bit product. The Am25LS14A must be clocked for m + n clock cycles to produce this two's complement product. Likewise, the n-bit multiplier (Y-input) sign bit data must be extended for the remaining m-bits to complete the multiplication cycle.

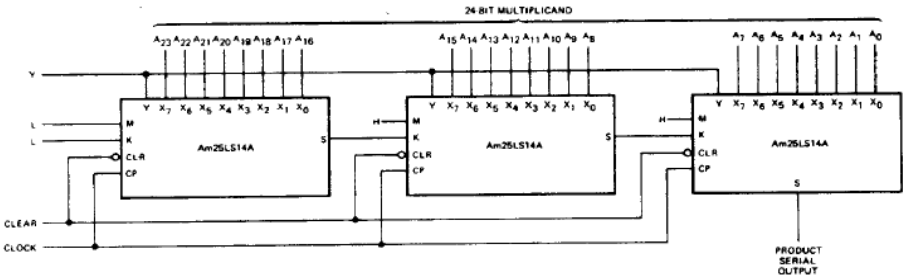
The device also contains a K input so that devices can be cascaded for longer length X words. The sum (S) output of one device is connected to the K input of the succeeding device when cascading. Likewise, a mode input (M) is used to indicate which device contains the most significant bit. The mode input is wired HIGH or LOW depending on the position of the 8-bit slice in the total X word length.

BLOCK DIAGRAM 8-BIT SERIAL/PARALLEL MUTIPLIER



Am25LS14A 8-bit serial/parallel two's complement multiplier

APPLICATIONS/EXTENSIONS



Basic 24-bit serial/parallel connection

END OF LECTURE