Digital Logic & Design (Lab)

Examination: Lab

Instructor: Muhammad Amin

Program: BS(CS)

Course Codes: CSC-201

EDP Codes: 102002078

Semester: Spring 2020

Date: July 8, 2020

Name = MUHAMMAD YASIR

ID = 15459

Q. Design and verify the logic circuit for the following:

1) Half adder using logic gates ?

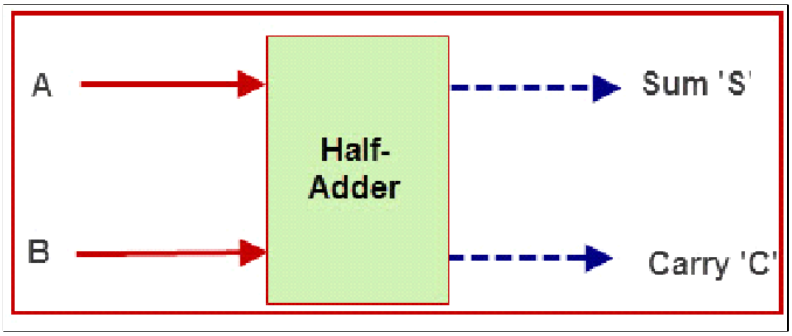
Half Adder :

The half-adder accepts two binary digits on its inputs and produces two binary digits on its outputs—a sum bit and a carry bit.

The half adder adds two binary digits called as augend and addend and produces two outputs as sum and carry; XOR is applied to both inputs to produce sum and AND gate is applied to both inputs to produce carry.

By using half adder, you can design simple addition with the help of logic gates.

Lets see an addition of single bits.



0+0 = 0

0+1 = 1

1+0 = 1

1+1 = 10

These are the least possible single-bit combinations. But the result for 1+1 is 10, the sum result must be re-written as a 2-bit output. Thus, the equations can be written as

0+0 = 00

0+1 = 01

1+0 = 01

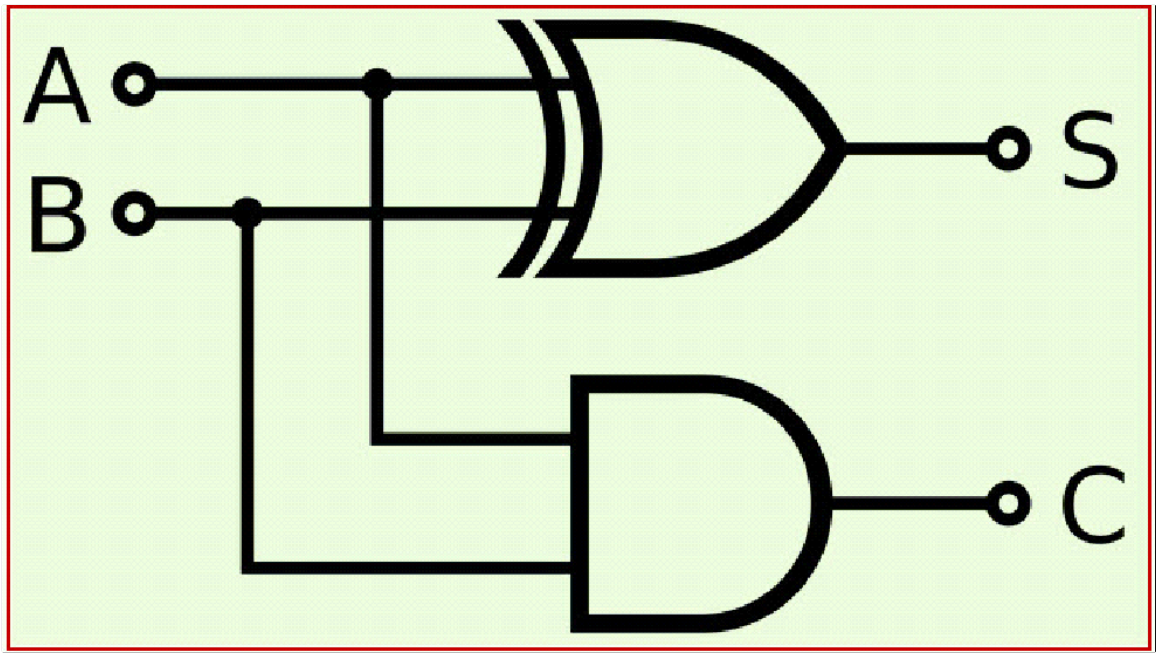
1+1 = 10

Half Adder Truth Table:

INPUTS		OUTPUTS	
A	B	SUM	CARRY
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Now it has been cleared that 1-bit adder can be easily implemented with the help of the XOR Gate for the output SUM and an AND Gate for the Carry. When we need to add, two 8-bit bytes together, we can be done with the help of a full-adder logic. The half-adder is useful when you want to add one binary digit quantities. A way to develop a two-binary digit adders would be to make a truth table and reduce it. When you want to make a three binary digit adder, do it again. When you decide to make a four digit adder, do it again. The circuits would be fast, but development time is slow.

Half Adder Logic Gate:



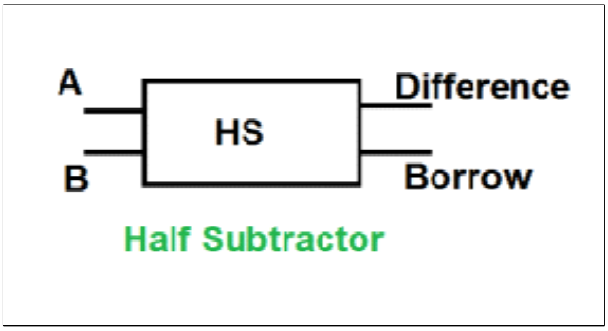
2) Half-subtractor using logic gate ?

Half subtractor :

Half subtractor is used to perform two binary digits subtraction. ... The circuit of the half subtractor can be built with two logic gates namely NAND and EX-OR gates. This circuit gives two elements such as the difference as well as the borrow. Half subtractor is used to perform two binary digits subtraction. Similarly, the subtractor circuit uses binary numbers (0,1) for the subtraction.

- Half Subtractor is a combinational logic circuit.
- It is used for the purpose of subtracting two single bit numbers.

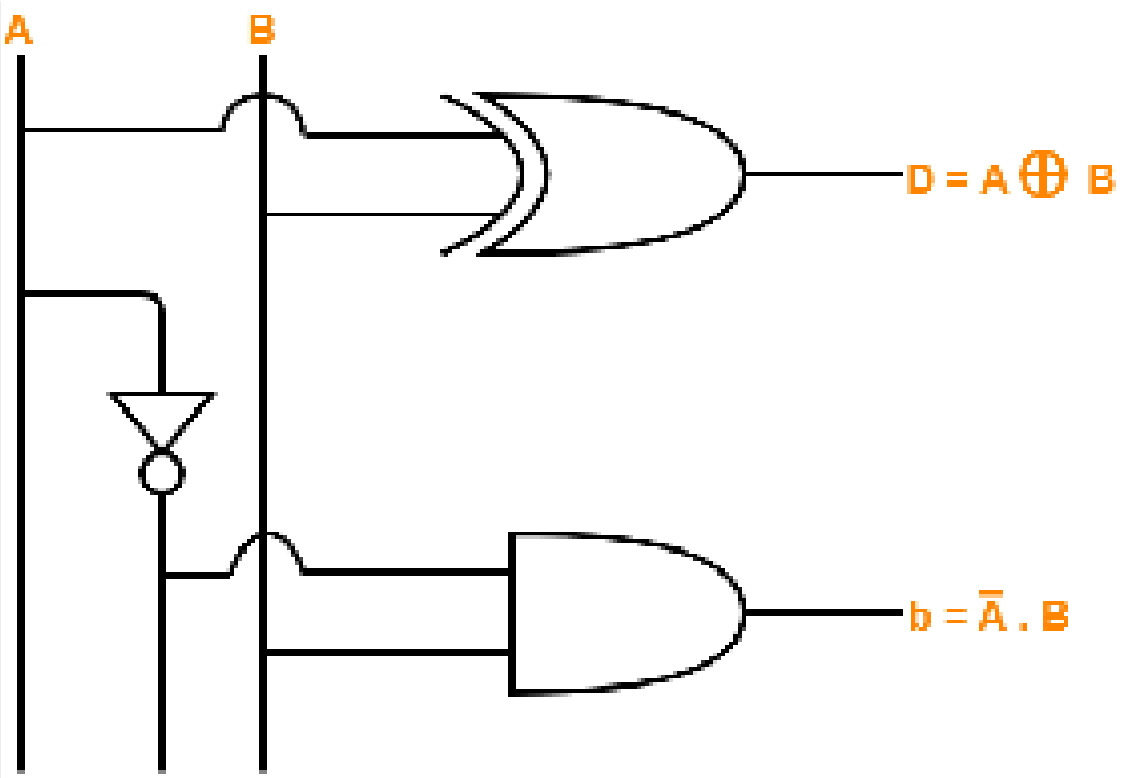
- It contains 2 inputs and 2 outputs (difference and borrow).



Truth Table:

A	B	Diff	Borrow
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

Logic Diagram of Half Subtractor :



Half Subtractor Logic Diagram

Application of Half Subtractor:

- Half subtractor is used to reduce the force of audio or radio signals
- It can be used in amplifiers to reduce the sound distortion
- Half subtractor is used in ALU of processor
- It can be used to increase and decrease operators and also calculates the addresses
- Half subtractor is used to subtract the least significant

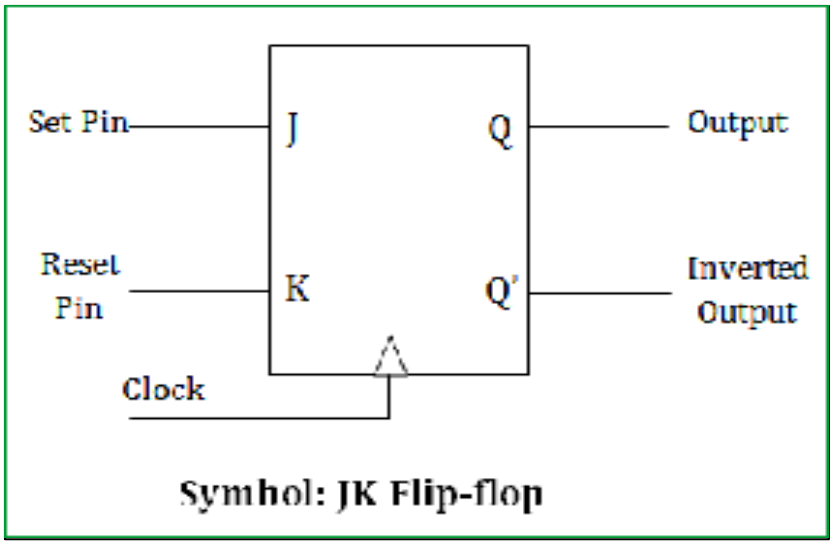
column numbers. For subtraction of multi-digit numbers, it can be used for the LSB.

3) J K Flip flop ?

JK Flip-flop:

The name JK flip-flop is termed from the inventor Jack Kilby from texas instruments. Due to its versatility they are available as IC packages. The major applications of JK flip-flop are Shift registers, storage registers, counters and control circuits. Inspite of the simple wiring of D type flip-flop, JK flip-flop has a toggling nature. This has been an added advantage. Hence they are mostly used in counters and PWM generation, etc. Here we are using NAND gates for demonstrating the JK flip flop

Whenever the clock signal is LOW, the input is never going to affect the output state. The clock has to be high for the inputs to get active. Thus, JK flip-flop is a controlled Bi-stable latch where the clock signal is the control signal.



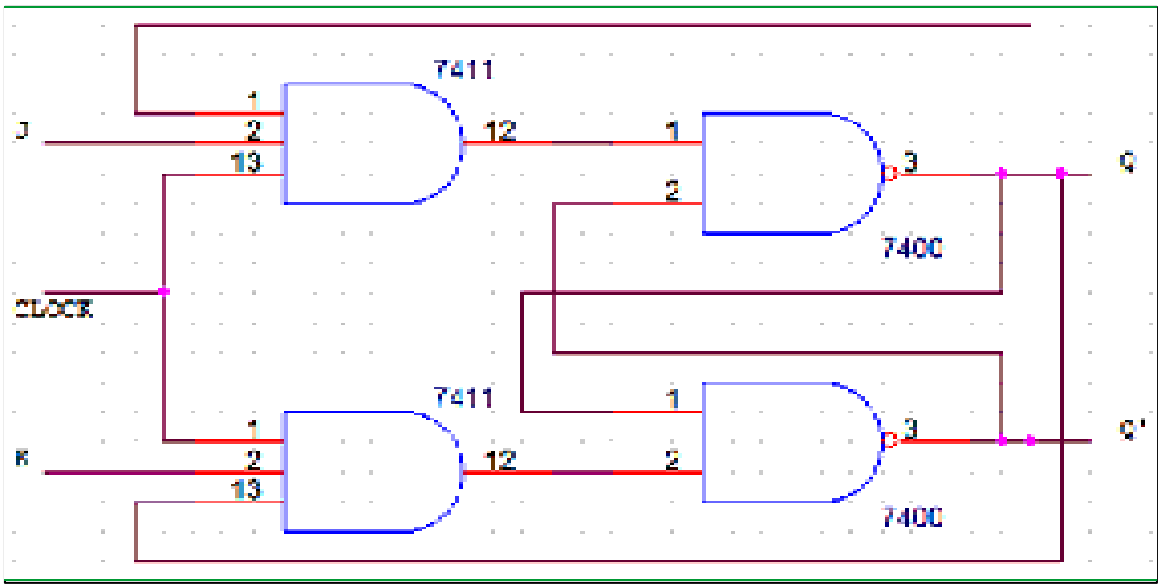
Truth table of JK Flip Flop:

Clock	INPUT	OUTPUT			
RESET	J	K	Q	Q	
X	LOW	X	X	0	1
HIGH	HIGH	0	0	No Change	
HIGH	HIGH	0	1	0	1
HIGH	HIGH	1	0	1	0
HIGH	HIGH	1	1	Toggle	
LOW	HIGH	X	X	No Change	
HIGH	HIGH	X	X	No Change	
HIGH	HIGH	X	X	No Change	

The J (Jack) and K (Kilby) are the input states for the JK flip-flop. The Q and Q represents the output states of the flip-flop. According to the table, based on the inputs, the

output changes its state. But, the important thing to consider is all these can occur only in the presence of the clock signal. This, works like SR flip-flop for the complimentary inputs and the advantage is that this has toggling function.

JK Flip-Flop using Logic Gates:



Representation of JK Flip-Flop using NAND Gates

4) Serial in-serial Out shift register ?

Serial In Serial Out (SISO) shift registers :

Serial In Serial Out (SISO) shift registers are a kind of shift registers where both data loading as well as data retrieval to/from the shift register occurs in serial-mode. Figure 1 shows a n-bit synchronous SISO shift register sensitive to positive edge of the clock pulse. Here the data word which is to be stored is fed bit-by-bit at the input of the first flip-flop. Further it is seen that the inputs of all other flip-flops (except the first flip-flop FF1) are driven by the outputs of the preceding ones say for example, the input of FF2 is adriven by the output of FF1. At last the data stored within the register is obtained at the output pin of the nth flip-flop in serial-fashion.

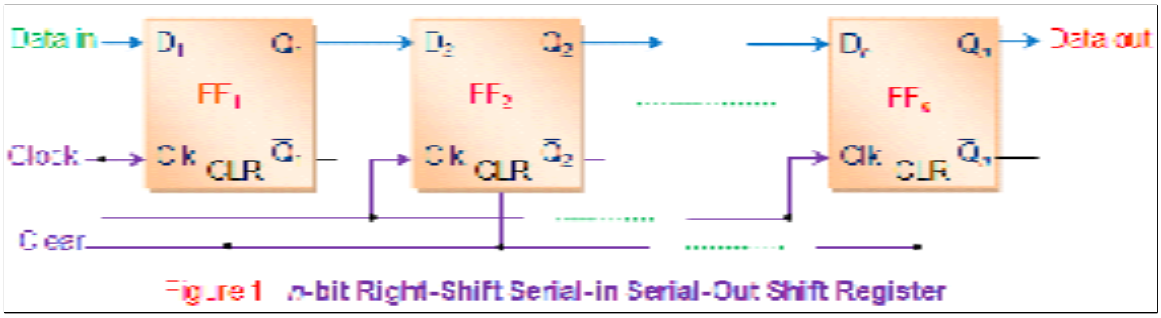


Figure 1 n-bit Right-Shift Serial-in Serial-Out Shift Register

Initially all the flip-flops in the register are cleared by applying high on their clear pins. Next the input data word is fed serially to FF1.

This causes the bit appearing at the D1 pin (B1) to be stored into FF1 as soon as the first leading edge of the clock appears. Further at the second clock tick, B1 gets stored into FF2 while a new bit enters into FF1 (B2).

This kind of shift in data bits continues for every rising edge of the clock pulse. This indicates that for every single clock pulse the data within the register moves towards right by a single bit. Thus the design shown in Figure 1 is regarded as a right-shift SISO shift register. Following the data transmission as explained, one can note that the first bit of an input word appears at the output of nth flip-flop for the nth clock tick. On applying further clock cycles, one gets the next successive bits of the input data word as the serial output (Table I). The waveforms pertaining to the same are shown by Figure 2.

Clock Cycle	Data in	Q_1	Q_2	...	Q_n = Data out	
1	D_0	D_0	0	...	0	
2	D_1	D_1	E_0	...	0	
3	D_2	D_2	E_1	...	0	
4	D_3	D_3	E_2	...	0	
5	D_4	D_4	E_3	...	0	
6	D_5	D_5	E_4	...	0	
.	.	.	.		.	
.	.	.	.	...	.	
n	B_n	B_n	E_{n-1}		B_1	Ser a Output Bits of SISO (Right-Shift) Shift Register
n+1	B_{n+1}	B_{n+1}	E_n	...	B_2	
.	.	.	.		.	
.	.	.	.	...	.	

Table I Data Movement in Right-Shift SISO Shift

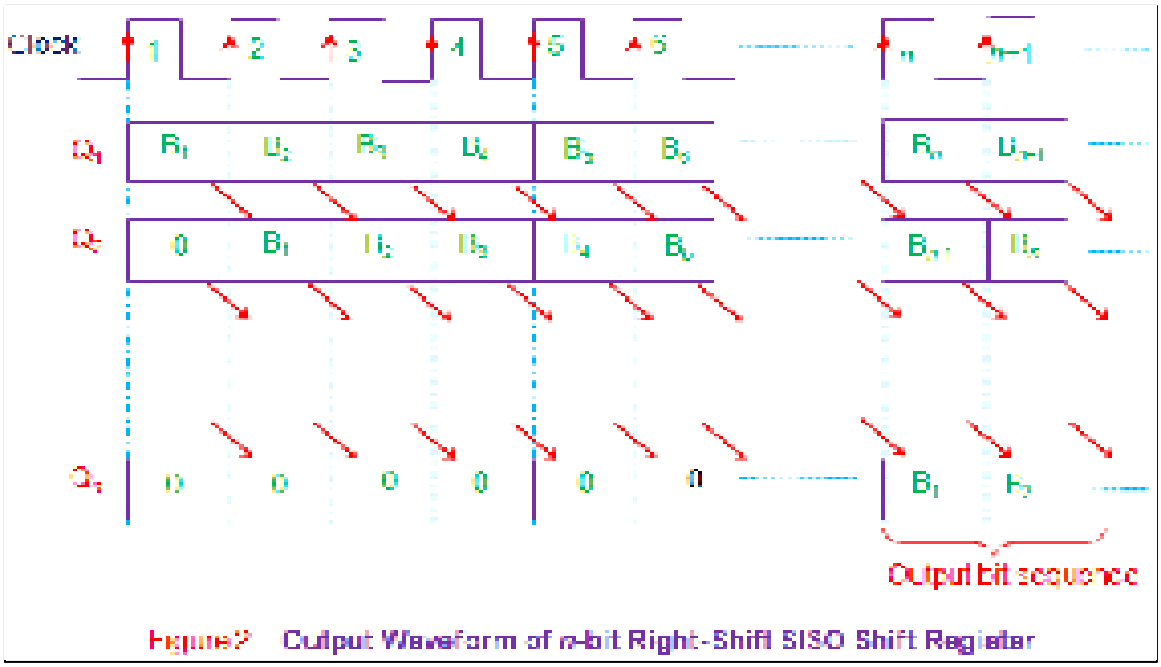


Figure2 Output Waveform of n-bit Right-Shift SISO Shift Register

Similar to the right-shift SISO shift-register shown, there can exist a left-shift SISO shift-register also (Figure 3). However the working principle remains the same except the fact that the data movement will be from right to left.

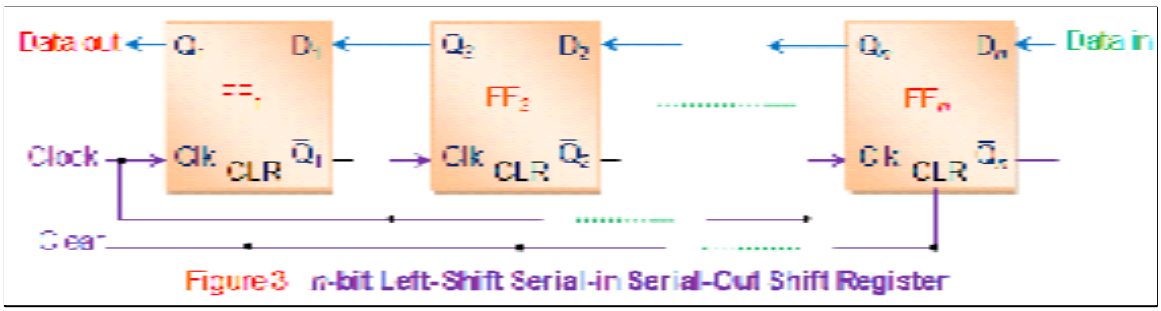
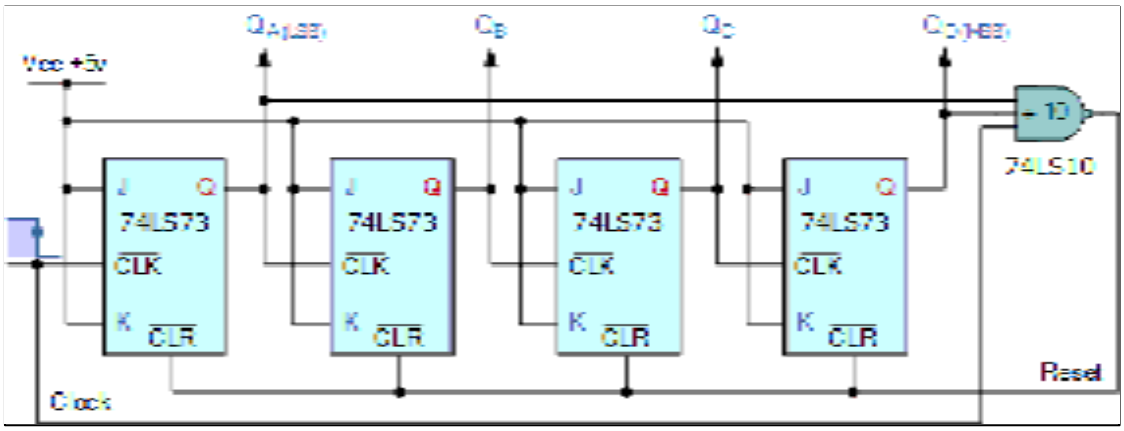


Figure3 n-bit Left-Shift Serial-in Serial-Out Shift Register

5) Asynchronous BCD Counter ?

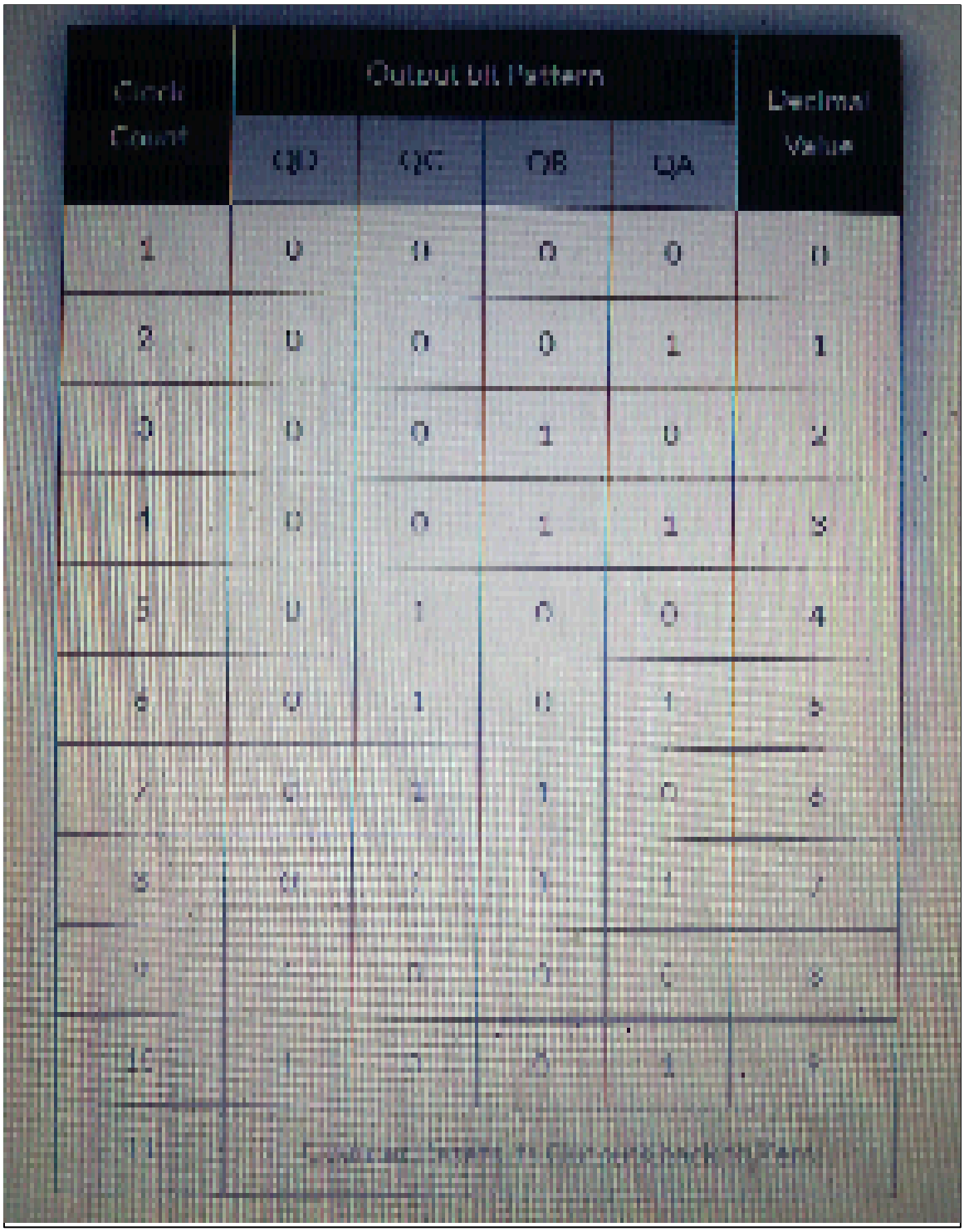
Asynchronous BCD Counter :



This type of asynchronous counter counts upwards on each trailing edge of the input clock signal starting from 0000 until it reaches an output 1001 (decimal 9). Both outputs QA and QD are now equal to logic 1. On the application of the next clock pulse, the output from the 74LS10 NAND gate changes state from logic 1 to a logic 0 level.

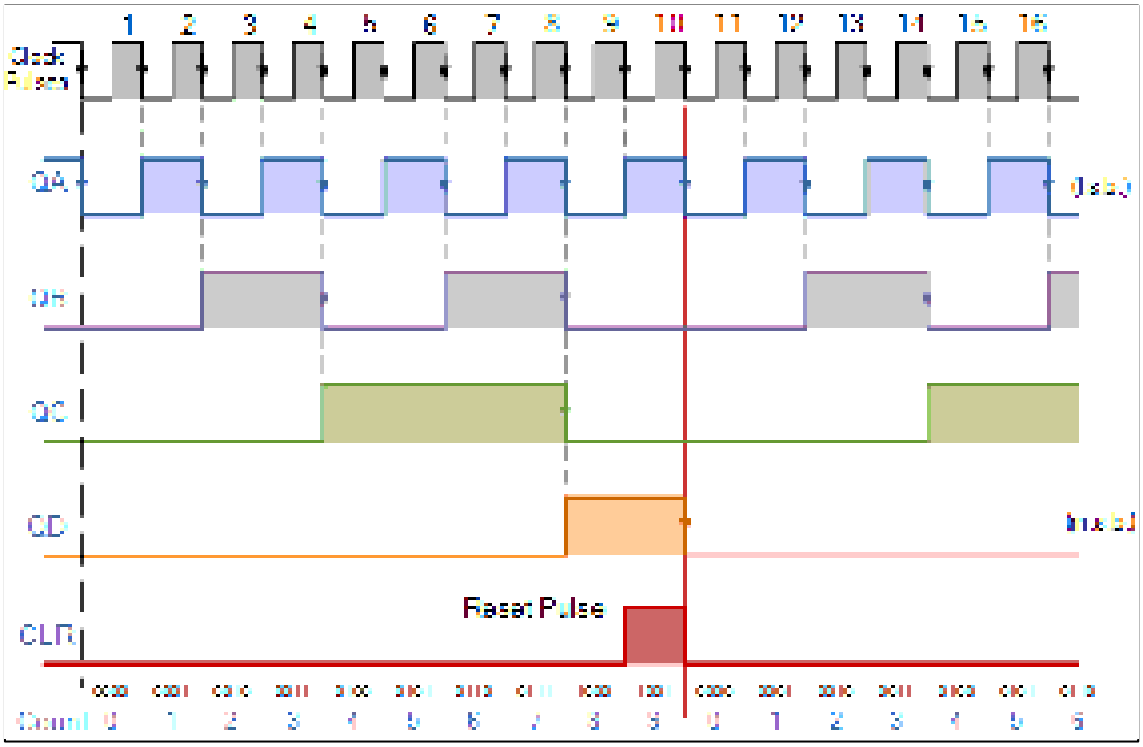
As the output of the NAND gate is connected to the CLEAR (CLR) inputs of all the 74LS73 J-K Flip-flops, this signal causes all of the Q outputs to be reset back to binary 0000 on the count of 10. As outputs QA and QD are now both equal to logic 0 as the flip-flops have just been reset, the output of the NAND gate returns back to a logic level 1 and the counter restarts again from 0000. We now have a decade or Modulo-10 up-counter.

BCD Counter Truth Table:



Clock Count	Output bit Pattern				Decimal Value
	QD	QC	QB	QA	
1	0	0	0	0	0
2	0	0	0	1	1
3	0	0	1	0	2
4	0	0	1	1	3
5	0	1	0	0	4
6	0	1	0	1	5
7	0	1	1	0	6
8	0	1	1	1	7
9	1	0	0	0	8
10	1	0	0	1	9
11	Invalid State				

BCD Counter Timing Diagram :



By using the same idea of truncating counter output sequences, the above circuit could easily be adapted to other counting cycles be simply changing the connections to the inputs of the NAND gate or by using other logic gate combinations.