1. Truth Table for the 2-bit Arithmetic and Logic Units

Inputs				Equal Greater	Less	Arithmetic	Output	
A1	A0	B1	B0	Magnitude(E)	A >B	A <B	Operation	
0	0	0	0	1	0	0	Multiply	**00**
0	0	0	1	0	0	1	Add	**01**
0	0	1	0	0	0	1	Add	**10**
0	0	1	1	0	0	1	Add	**11**
0	1	0	0	0	1	0	Subtract	**01**
0	1	0	1	1	0	0	Multiply	**00**
0	1	1	0	0	0	1	Add	**11**
0	1	1	1	0	0	1	Add	**10**
1	0	0	0	0	1	0	Subtract	**01**
1	0	0	1	0	1	0	Subtract	**10**
1	0	1	0	1	0	0	Multiply	**10**
1	0	1	1	0	0	1	Add	**01**
1	1	0	0	0	1	0	Subtract	**11**
1	1	0	1	0	1	0	Subtract	**10**
1	1	1	0	0	1	0	Subtract	**01**
1	1	1	1	1	0	0	Multiply	**11**

From the above truth table logical expressions for each output can be expressed as follows:

A = B : A1'A0'B1'B0' + A1'A0B1'B0 + A1A0B1B0 + A1A0'B1B0'

: A1'B1' (A0'B0' + A0B0) + A1B1 (A0B0 + A0'B0')

: (A0B0 + A0'B0') (A1B1 + A1'B1')

: (A0 Ex-Nor B0) (A1 Ex-Nor B1)

A < B : A1'B1 + A0'B1B0 + A1'A0'B0

A > B : A1B1' + A0B1'B0' + A1A0B0'

Selected output solution

Input 1: (A = 00, B = 00)

multiply:

00 * 00 = 00 output = 00

Input 2: (A = 00, B = 01)

Arithmetic operation: Add

```
  00
+ 01
----
  01
```

output = 01

Input 3: (A = 00, B = 10)

Arithmetic operation: Add

```
  00
+ 10
----
  10
```

Output = 10

Input 4: (A = 00, B = 11)

Arithmetic operation: Add

```
  00
+ 11
----
  11
```

output = 11

Input 5: (A = 01, B = 00)

Subtract.

```
  01
- 00
----
  01
```

output = 01

Input 6: (A = 01 B = 01)

multiply 01 * 01 = 01

Multiplexer selection:

The control signals dictate which output is chosen

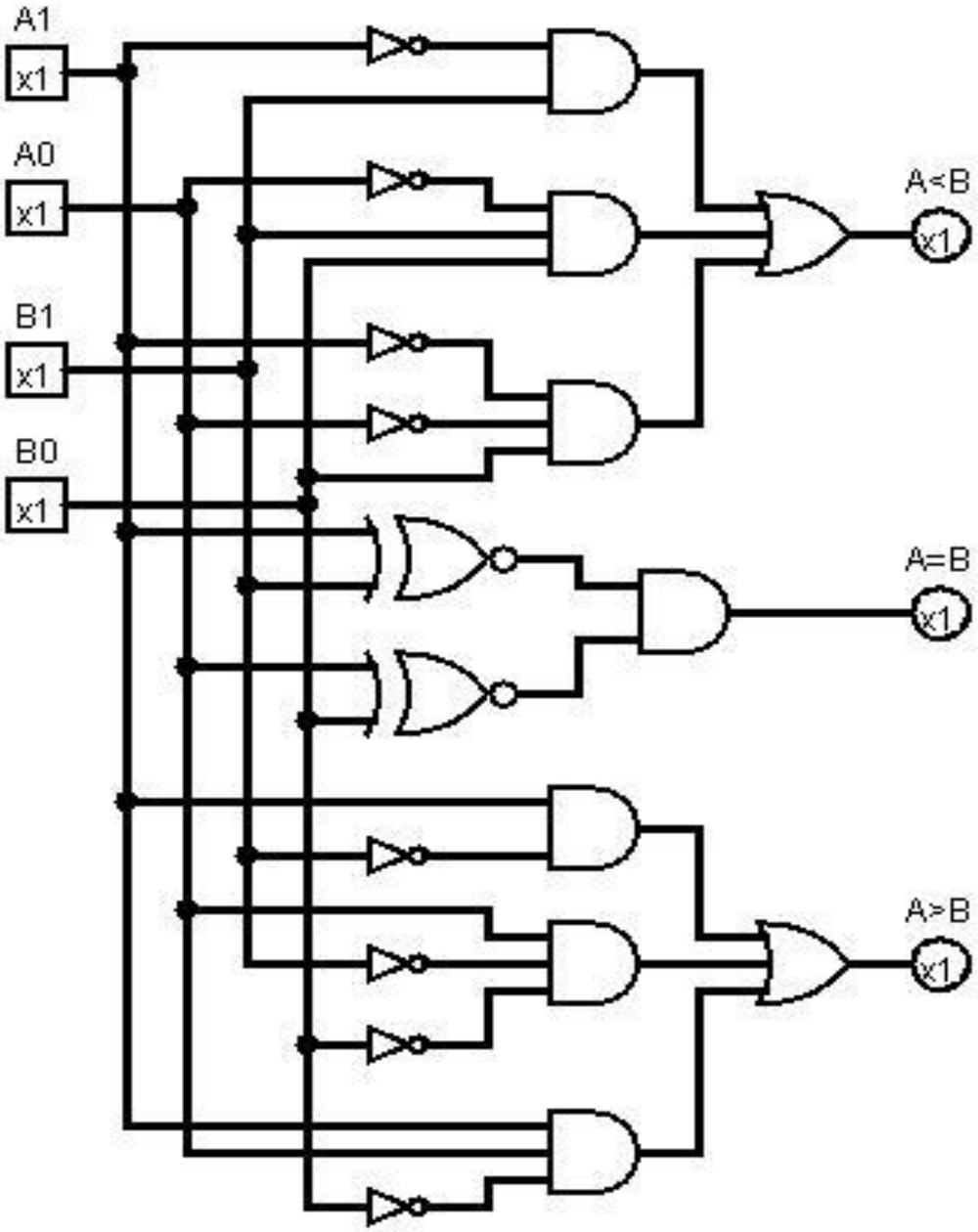
Since f = 1 (both A & B are equal) the multiplexers select the output of the multiplier where F1 = 0 and F0 = 0 output = 00

Input 7: 01 Add

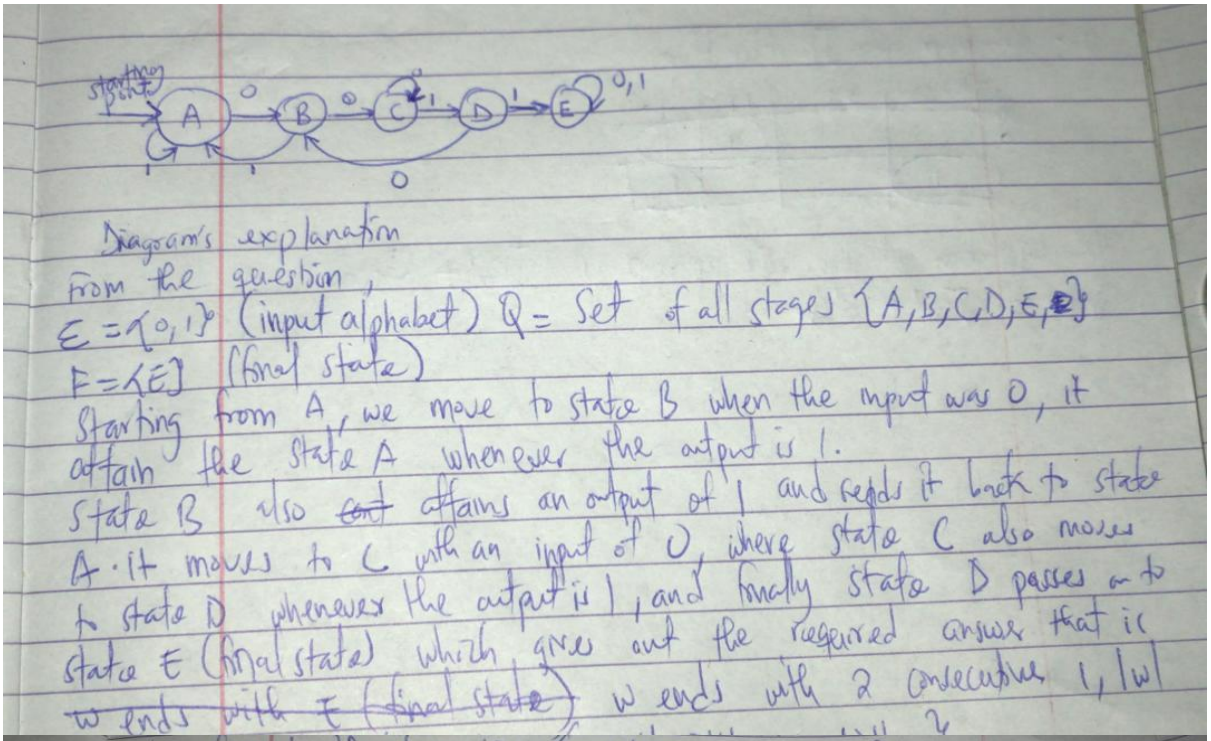
```
  10
----
  11
```

output = 11

Logic circuit diagram



2. A Finite State Machine has an ending input, it is made up of two types, that's finite state machine with output and finite state machine without output . In the given question finite state machine without output can be used. (Deterministic Finite State Automata).



Diagram's explanation
from the question,
$\Sigma = \{0,1\}$ (input alphabet) Q = Set of all stages $\{A,B,C,D,E\}$
$F = \{E\}$ (final state)
Starting from A, we move to state B when the input was 0, it attain the state A whenever the output is 1.
State B also ~~ent~~ attains an output of 1 and reads it back to state A. it moves to C with an input of 0, where state C also moves to state D whenever the output is 1, and finally state D passes on to state E (final state) which gives out the required answer that is ~~w ends with E (final state)~~ w ends with 2 consecutive 1, |w| has four length string thus {0011, 0111, or 1011...}

26. Transition table for the finite state machine.

States	Inputs 0	Inputs 1
A	B	A
B	C	A
C	C	D
D	B	E
E	E	E

Explanation: When either of the state attains an input $\Sigma = \{0,1\}$ where does it go, does it stay in the same state or it moves to another state.

b. **JK flip flop:** The JK flip flop is just like the **SR Flip flop** but there are some slide changes in the JK's. with Jk flip flop there is no change in the state when both J and K are **low.** To get better understanding of the JK flip flop there is a need to know everything about the NAND gate, especially their truth table.

NAND Gate Truth Table

A	B	C	AND Output	NAND
0	**0**	**0**	**0**	**1**
0	**0**	**1**	**0**	**1**
0	**1**	**0**	**0**	**1**
0	**1**	**1**	**0**	**1**
1	**0**	**0**	**0**	**1**
1	**0**	**1**	**0**	**1**
1	**1**	**0**	**0**	**1**
1	**1**	**1**	**1**	**0**

Explanation: NAND Gate is the complement of the AND Gate, where 0 changes to 1 and 1 changes to 0

with the help of the **NAND** gate, we can derive characteristics table, k-map and the excitation table for JK flip flop.

Characteristics Table for JK flip flop.

J	K	Q_n	Q_{n+1}	
x	x	0/1	0/1	Q_n
0	**0**	**0**	**0**	Q_n
0	**0**	**1**	**1**	
0	**1**	**0**	**0**	**Reset (0)**
0	**1**	**1**	**0**	
1	**0**	**0**	**1**	**Set(1)**
1	**0**	**1**	**1**	
1	**1**	**0**	**1**	$Q_{n'}$ **Toggle**
1	**1**	**1**	**0**	

Explanation:

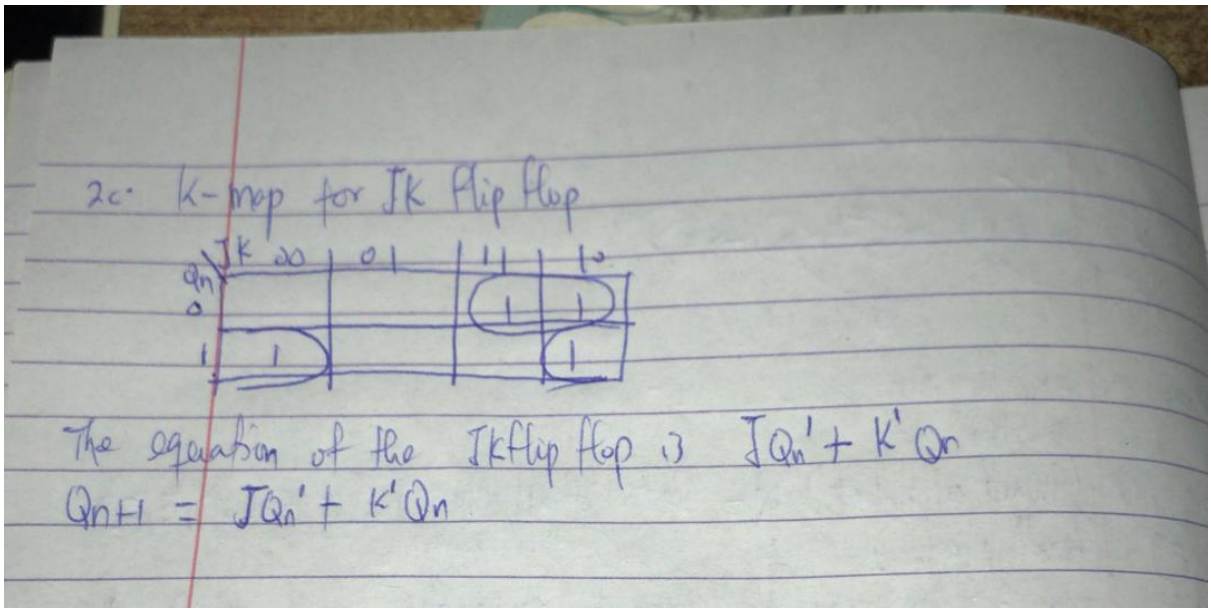
a. When J and K are zero(Qn) the flip flop will return its current state.

b. When J is 0 and K is 1, irrespective of the previous state the current state will be 0 for both.

c. Also, when j is 1 and k is 0 the current state will be 1, irrespective of the previous state.

d. When both J and K are 1 there would be a toggle where 1 will have a complement of 0 and too will have a complement of 1.

From the table we can draw a k-map to simplify the expression. From the table the minterms(m2, m3, m4 and m6) had an output of 1 each in the current state(Q_{n+1}), therefore we draw a k-map to simplify the given expression.



JK flip flop Excitation Table

Q_n	Q_{n+1}	J	K	Description from characteristic table
0	0	0	x	In Qn and reset column J is not changing but K is changing.
0	1	1	X	From set and toggle column j is not changing but k is changing, so we assign it to x
1	0	x	1	From reset and toggle column j is changing but k remains 1.
1	1	x	0	Toggle happens here, meaning we complement Qn for toggle state.

Circuit diagram for the JK flip flop