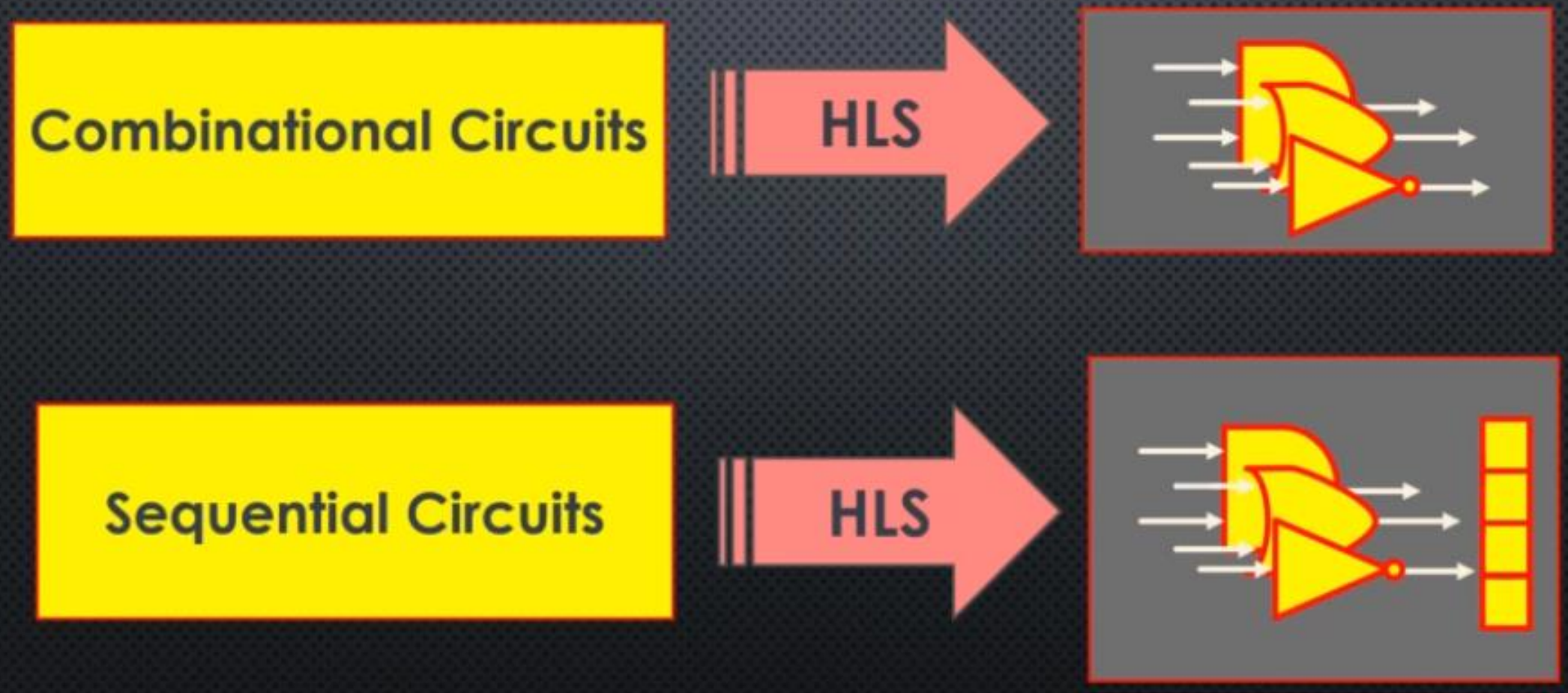
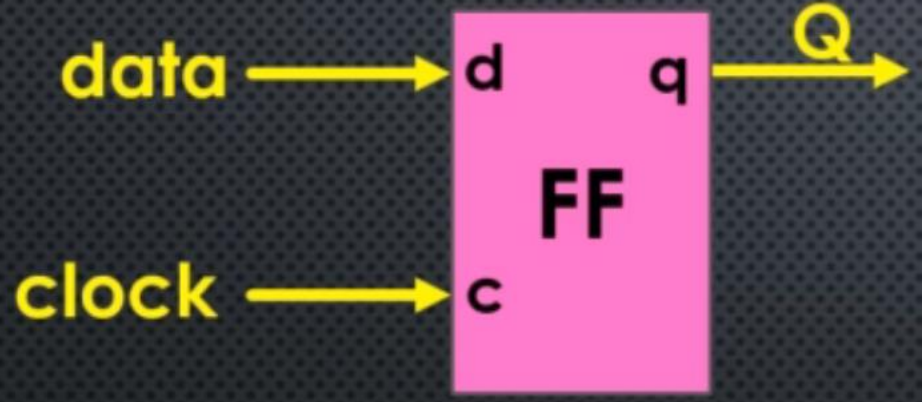
SEQUENTIAL CIRCUITS-DFF

Lecture-8

COMBINATIONAL CIRCUITS

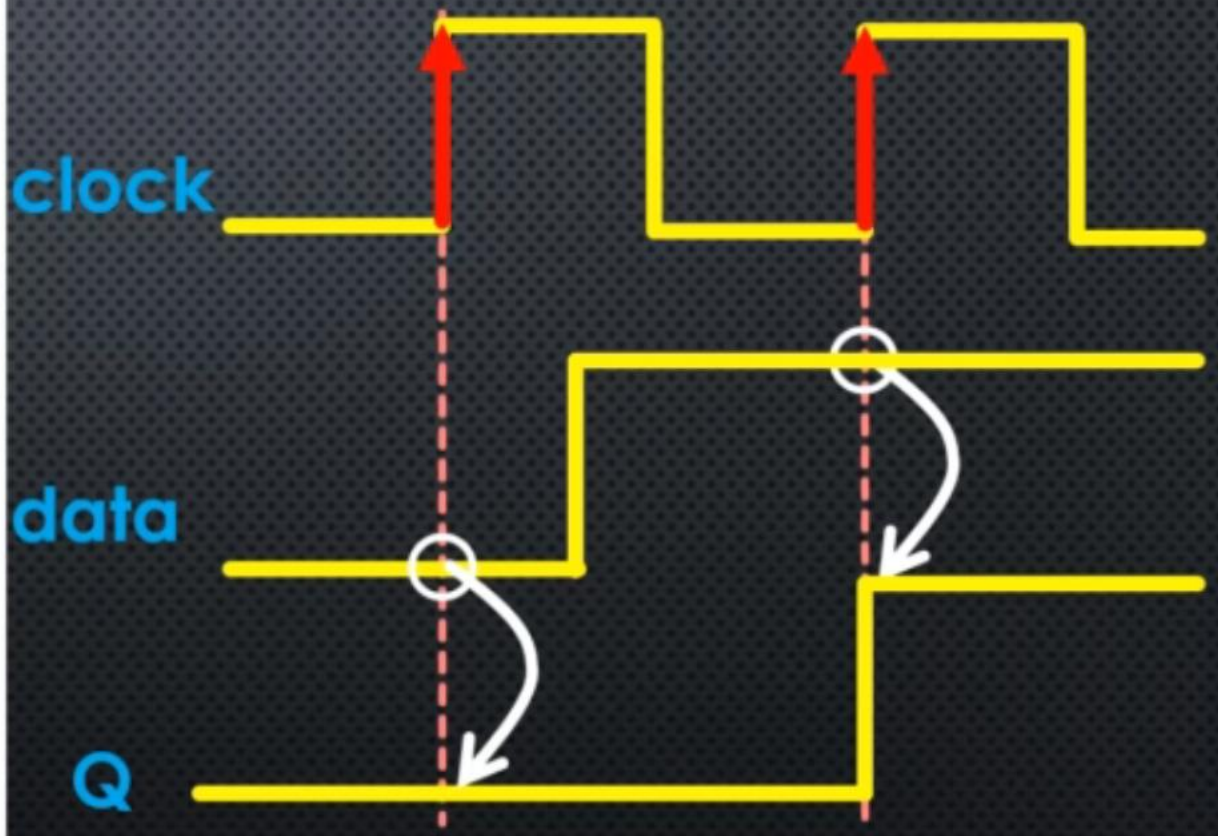


FLIP-FLOP

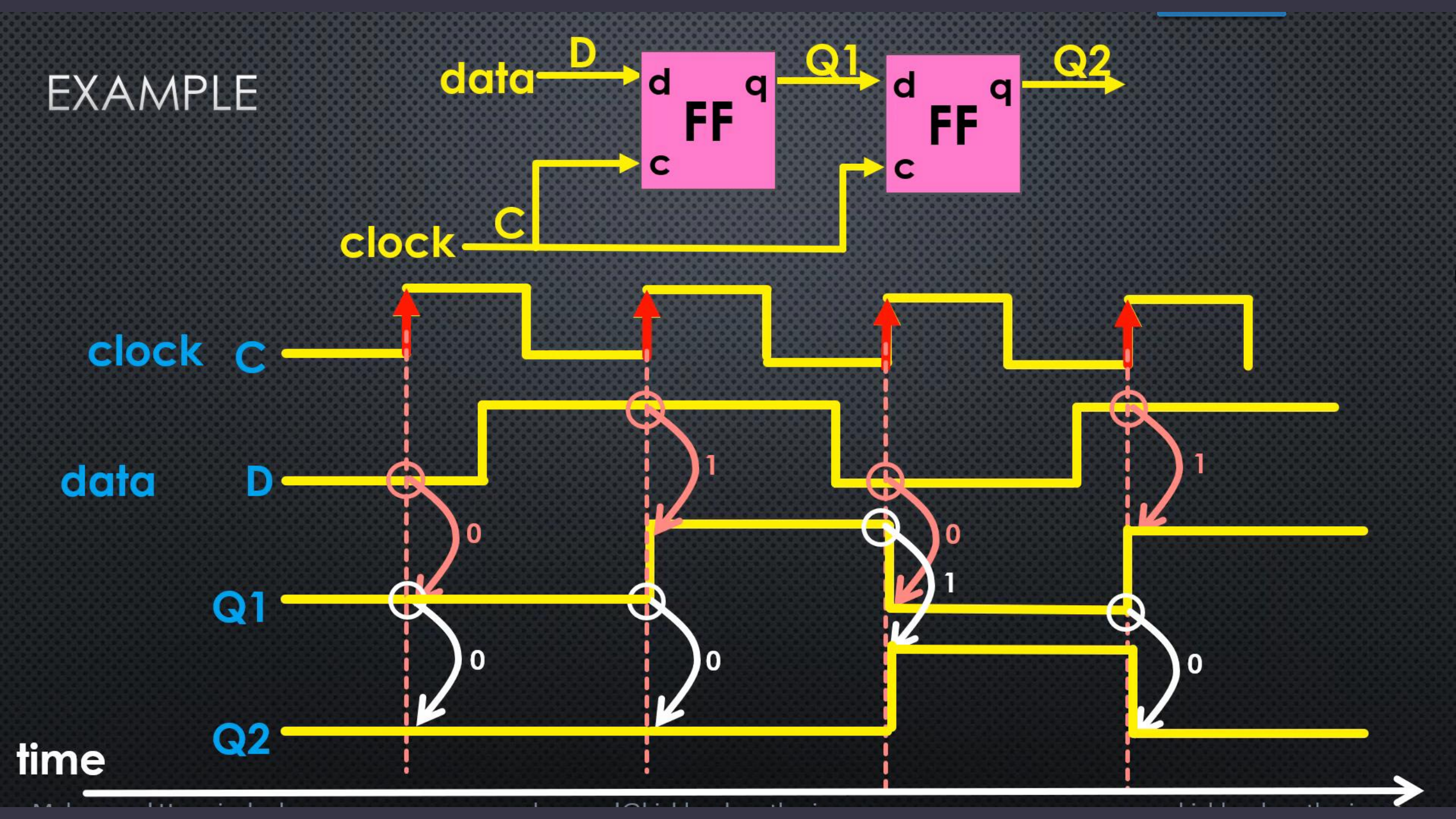


Characteristic Table

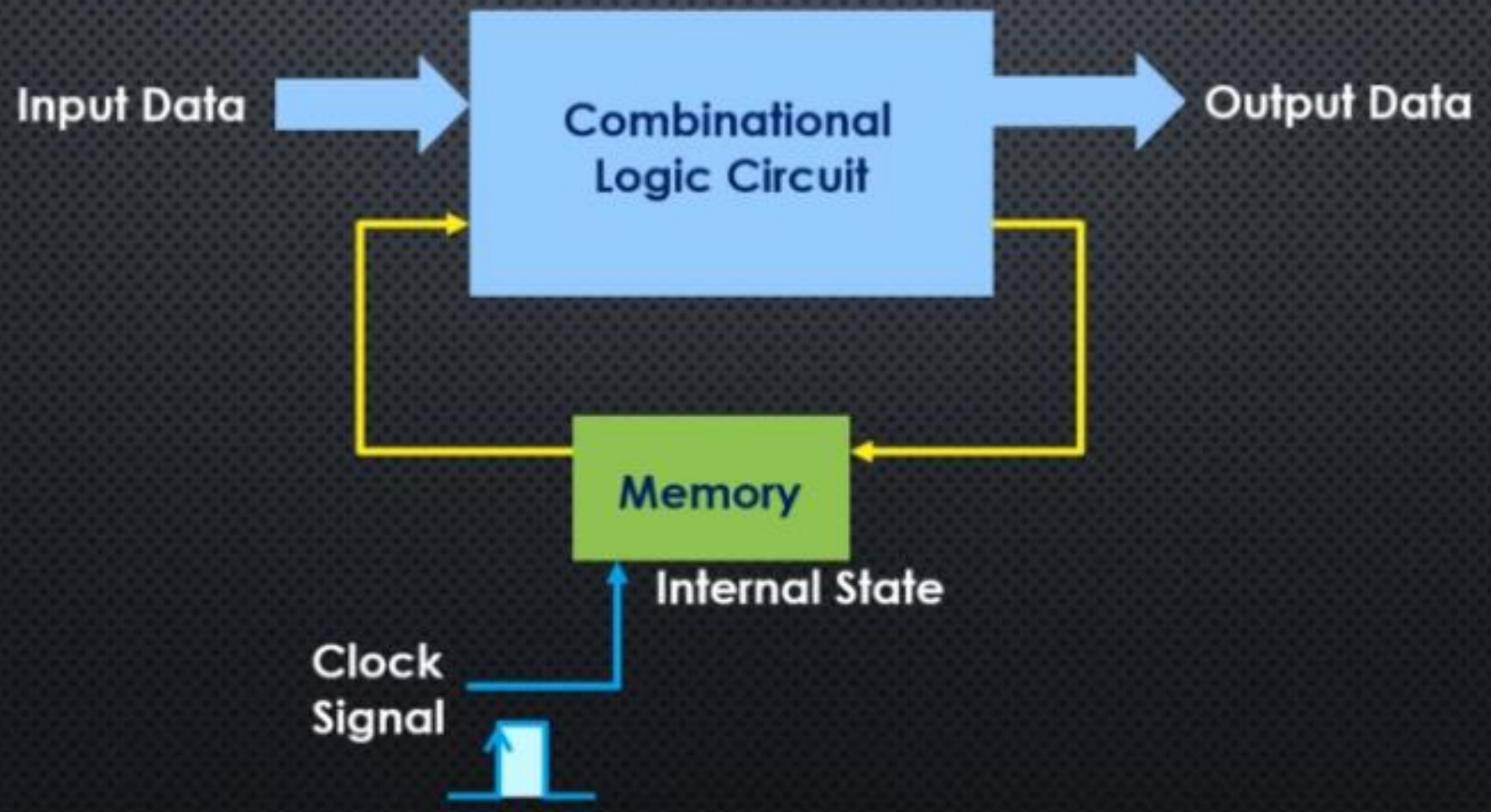
D	C	Q
1	↑	1
0	↑	0



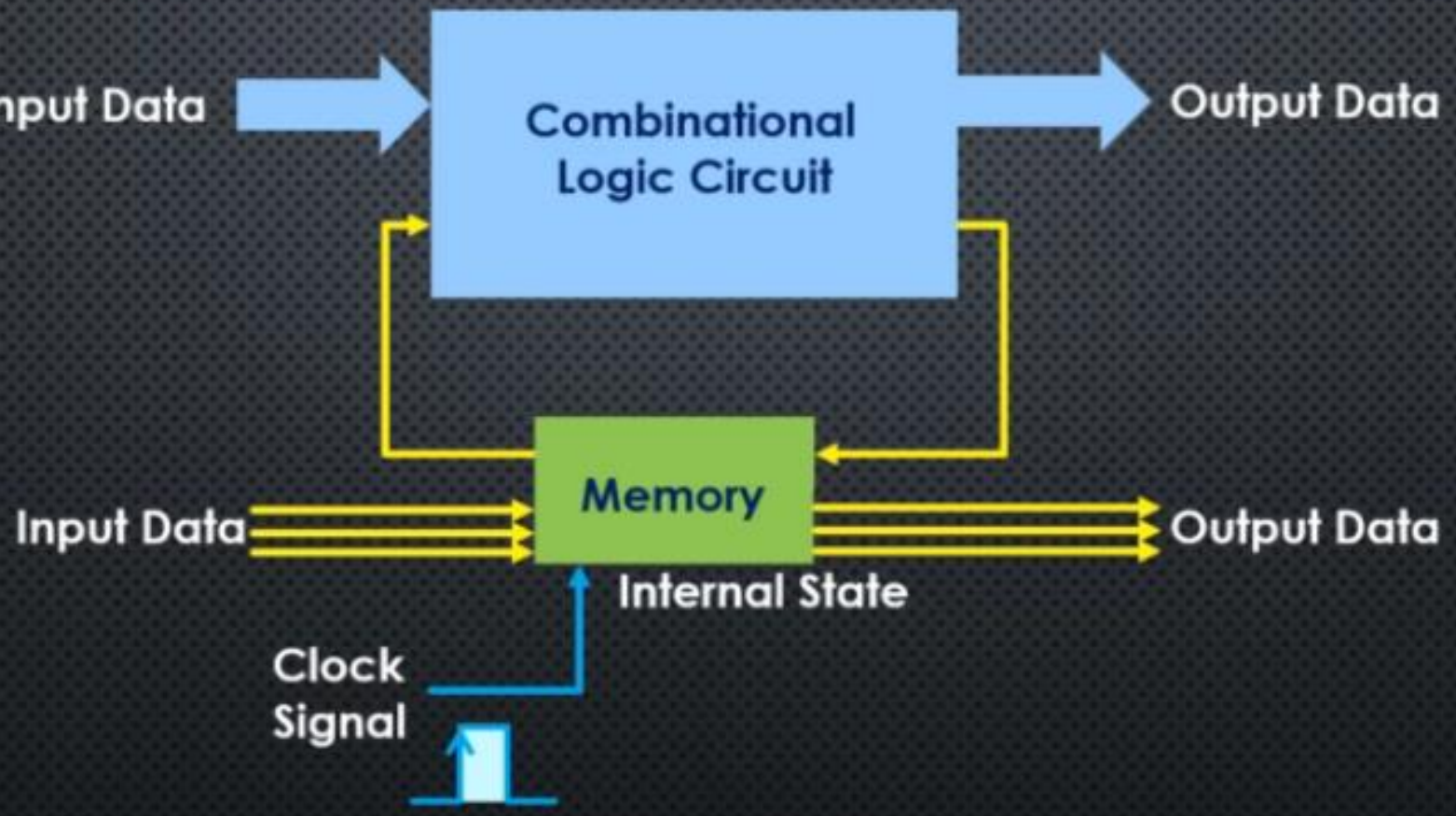
EXAMPLE


data
D
d
q
FF
c
Q1
d
q
FF
c
Q2
clock
C
clock C
data D
Q1
Q2
time
0
1
0
1
0
1
0
0

SEQUENTIAL CIRCUIT



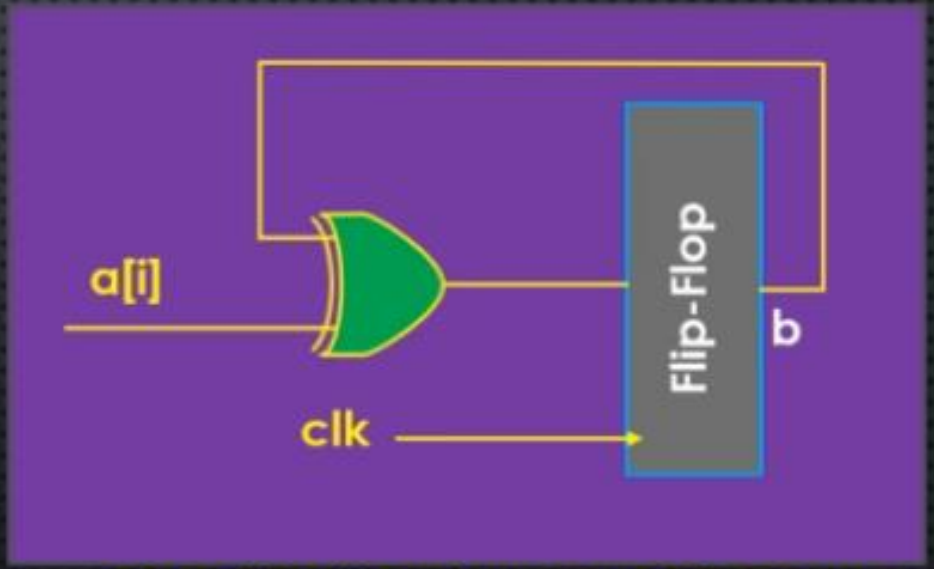
MORE DETAILS



EXAMPLE

```
bool sequential_parity(ap_uint<N> a)
{
  bool b=0;
  for (int i = 0; i < n; i++)
    b = b^a[i];
  return b;
}
```

Hardware design described in C

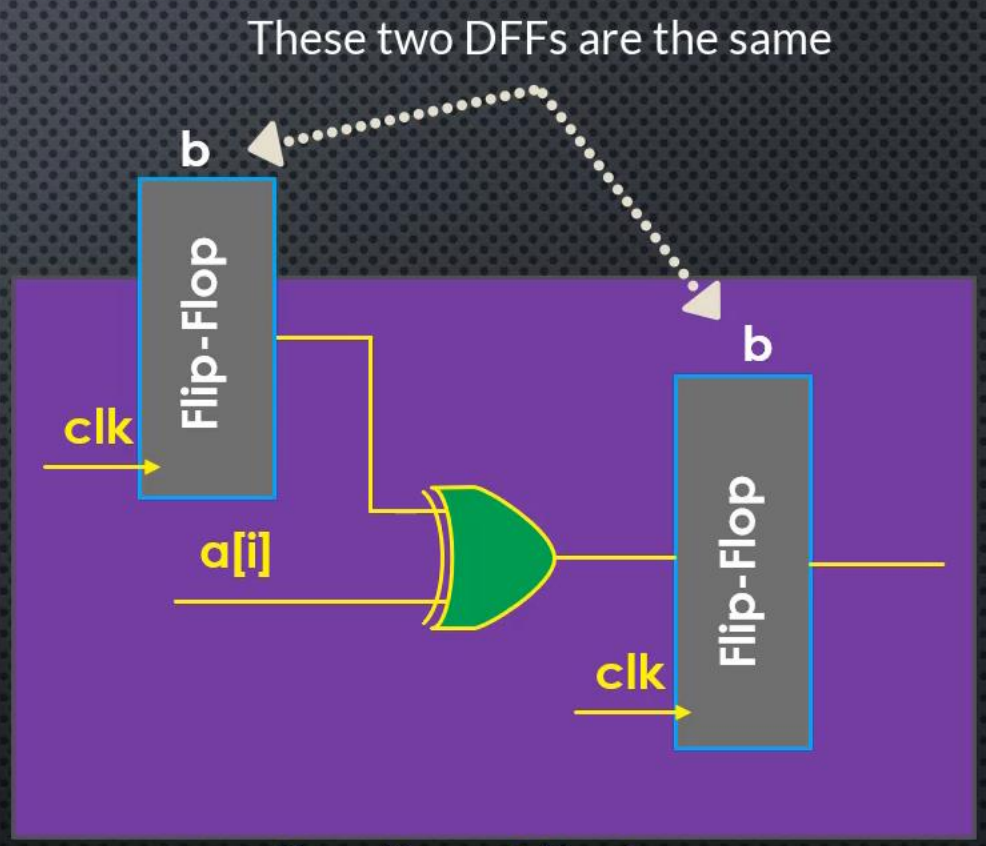


Simplified hardware architecture

EXAMPLE: ONE ITERATION

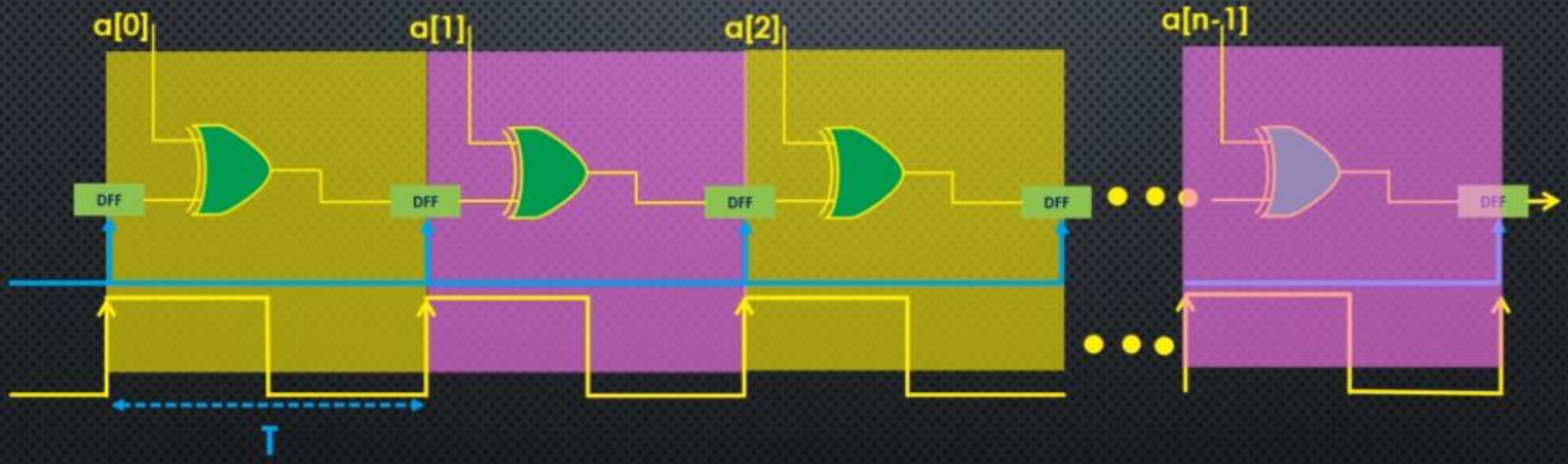
```
bool sequential_parity(ap_uint<N> a)
{
  bool b=0;
  for (int i = 0; i < n; i++)
    b = b^a[i];
  return b;
}
```

Hardware design in C

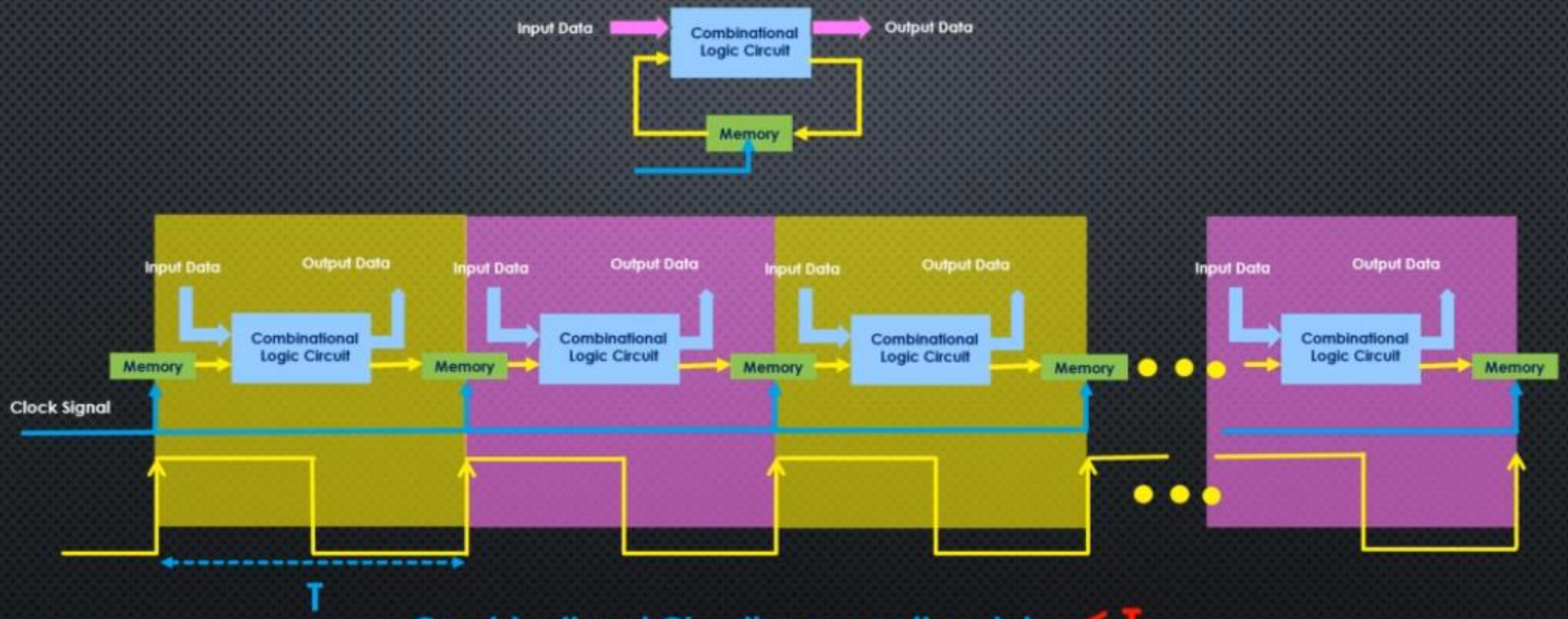


One iteration of the loop

EXAMPLE: SEQUENCE OF EXECUTION

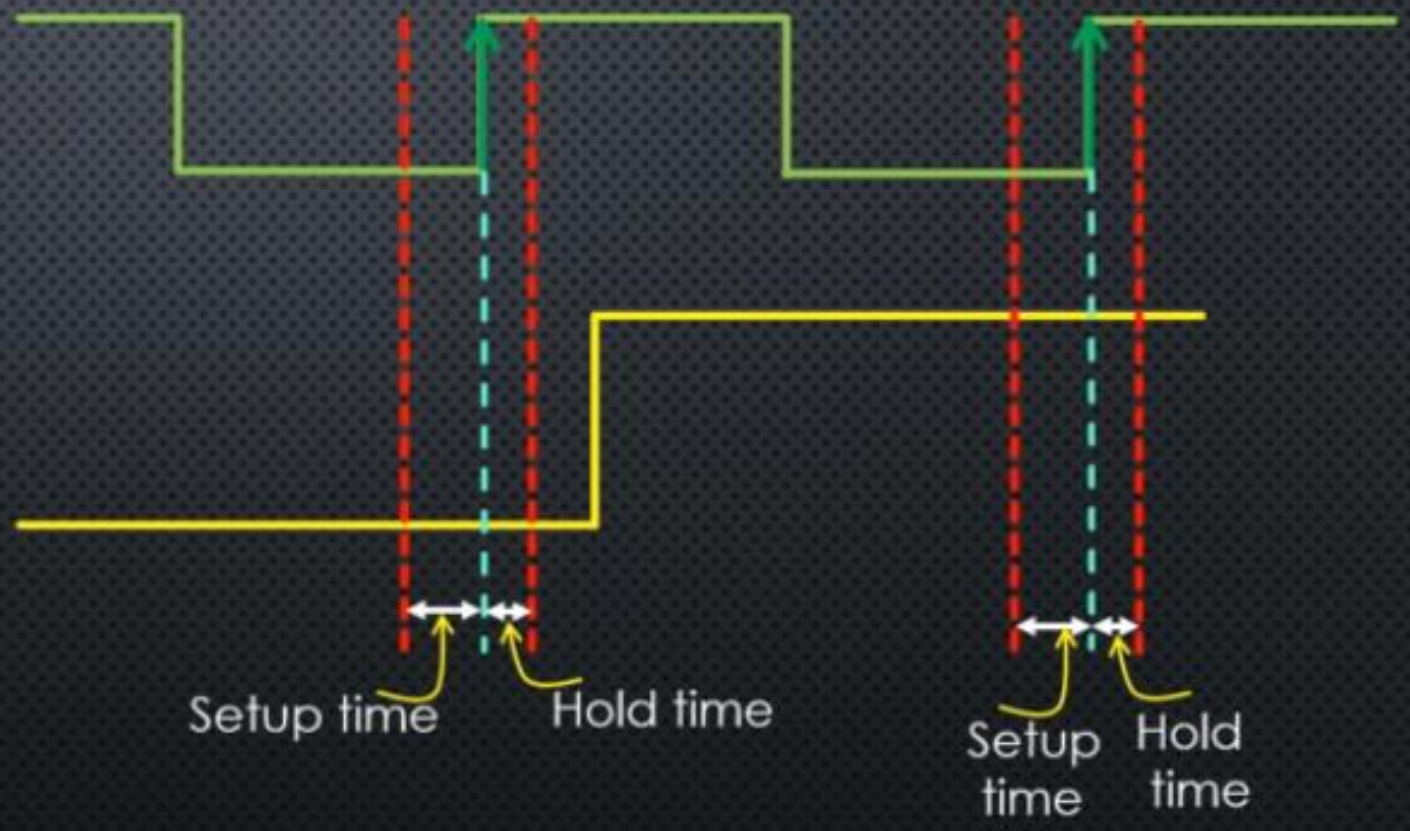


SEQUENTIAL CIRCUITS: SEQUENCE OF EXECUTION

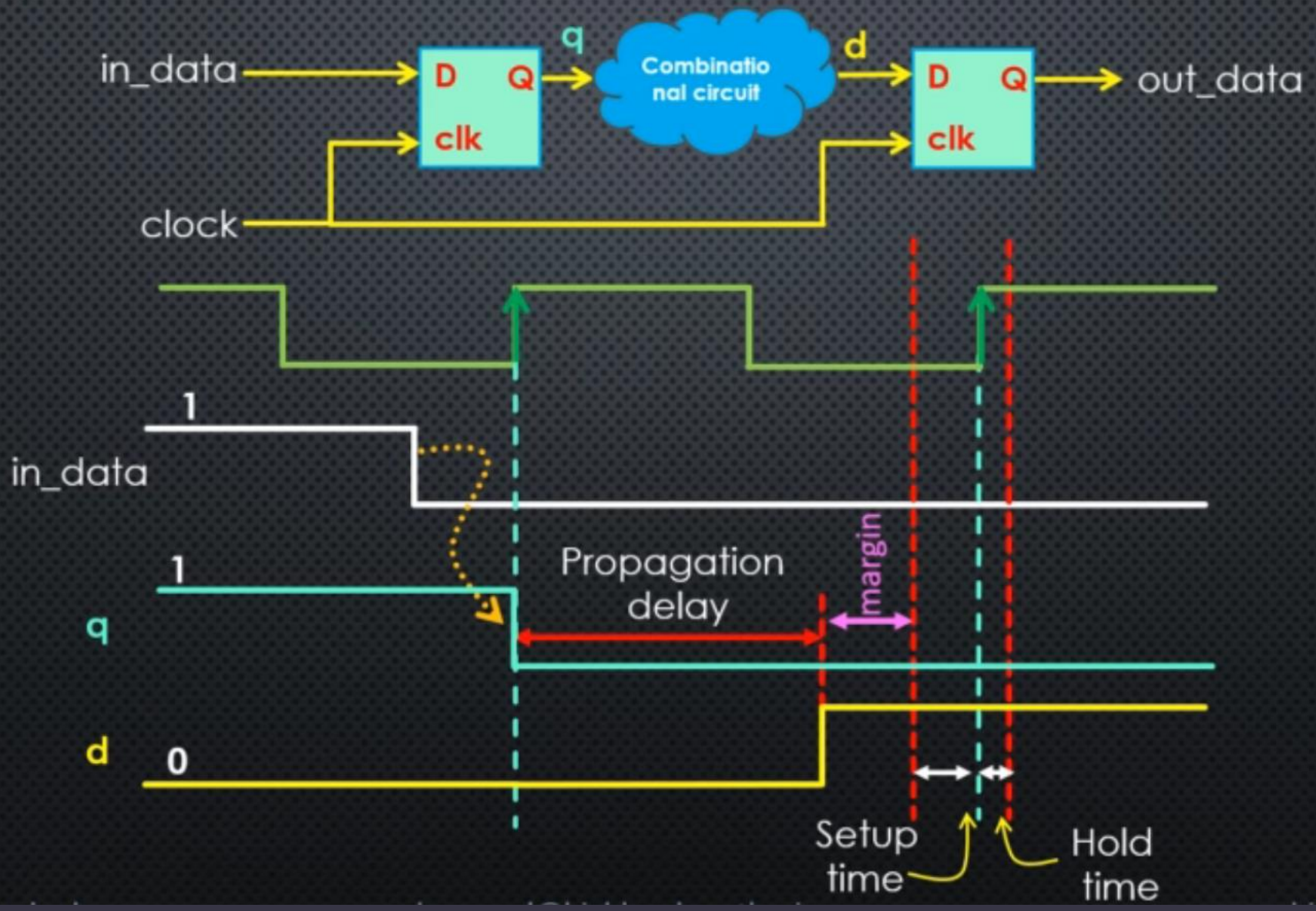


Combinational Circuit propagation delay < T

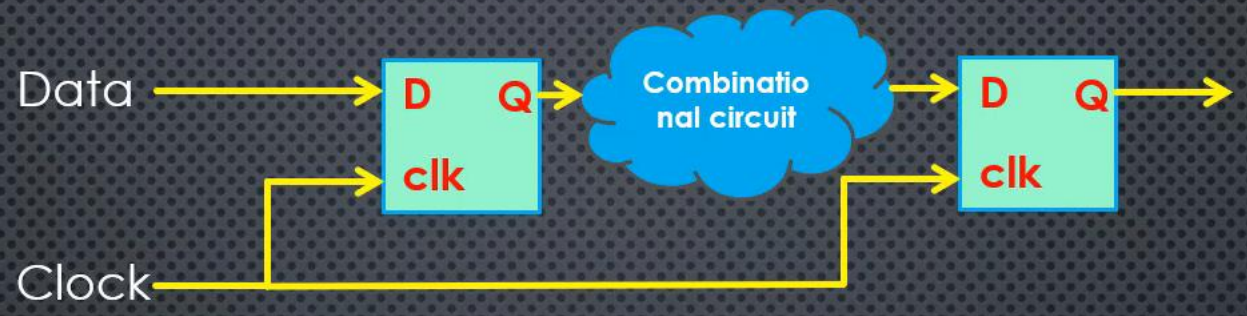
SETUP AND HOLD TIME

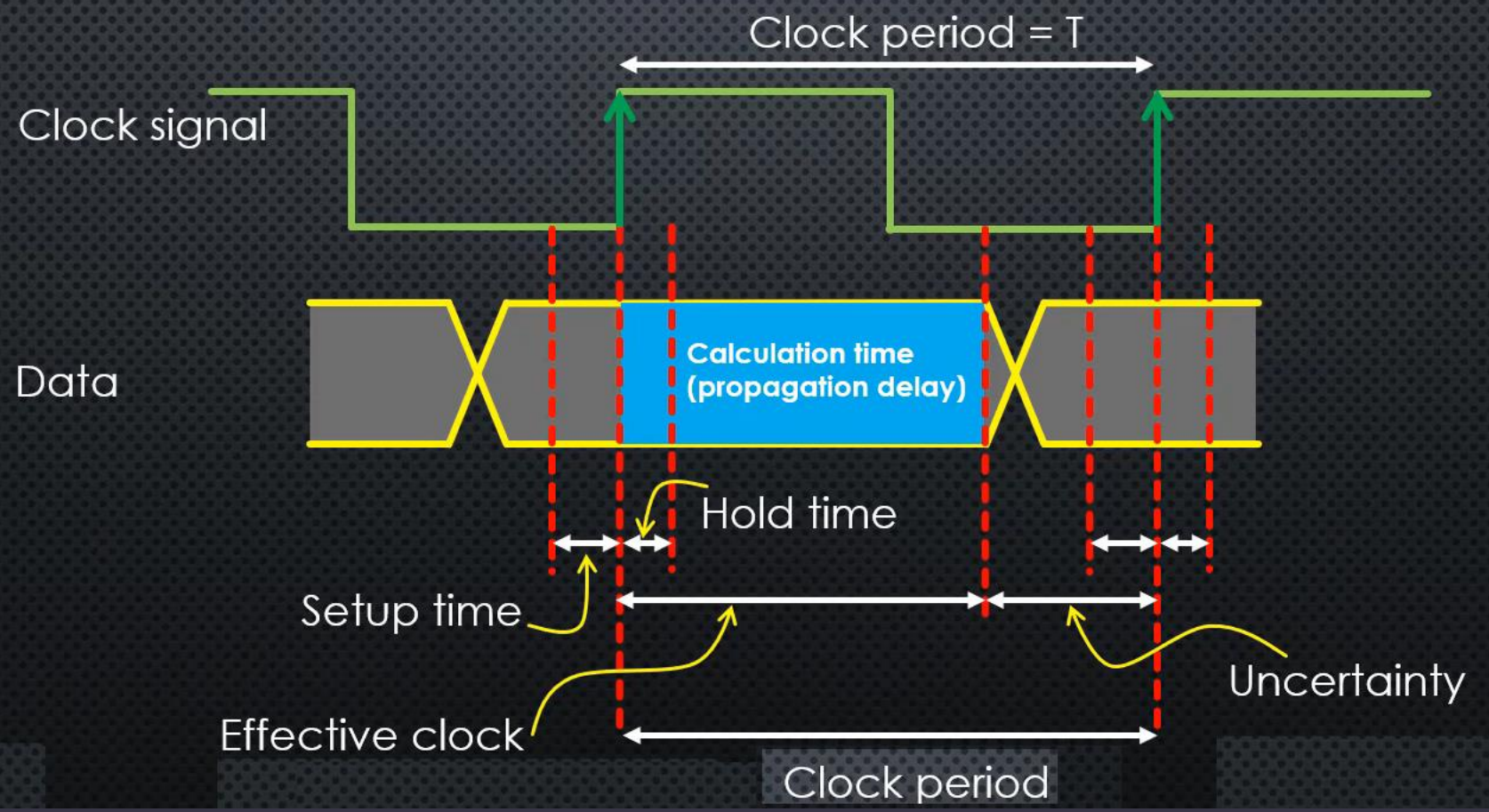


PROPAGATION DELAY & SETUP/HOLD TIME

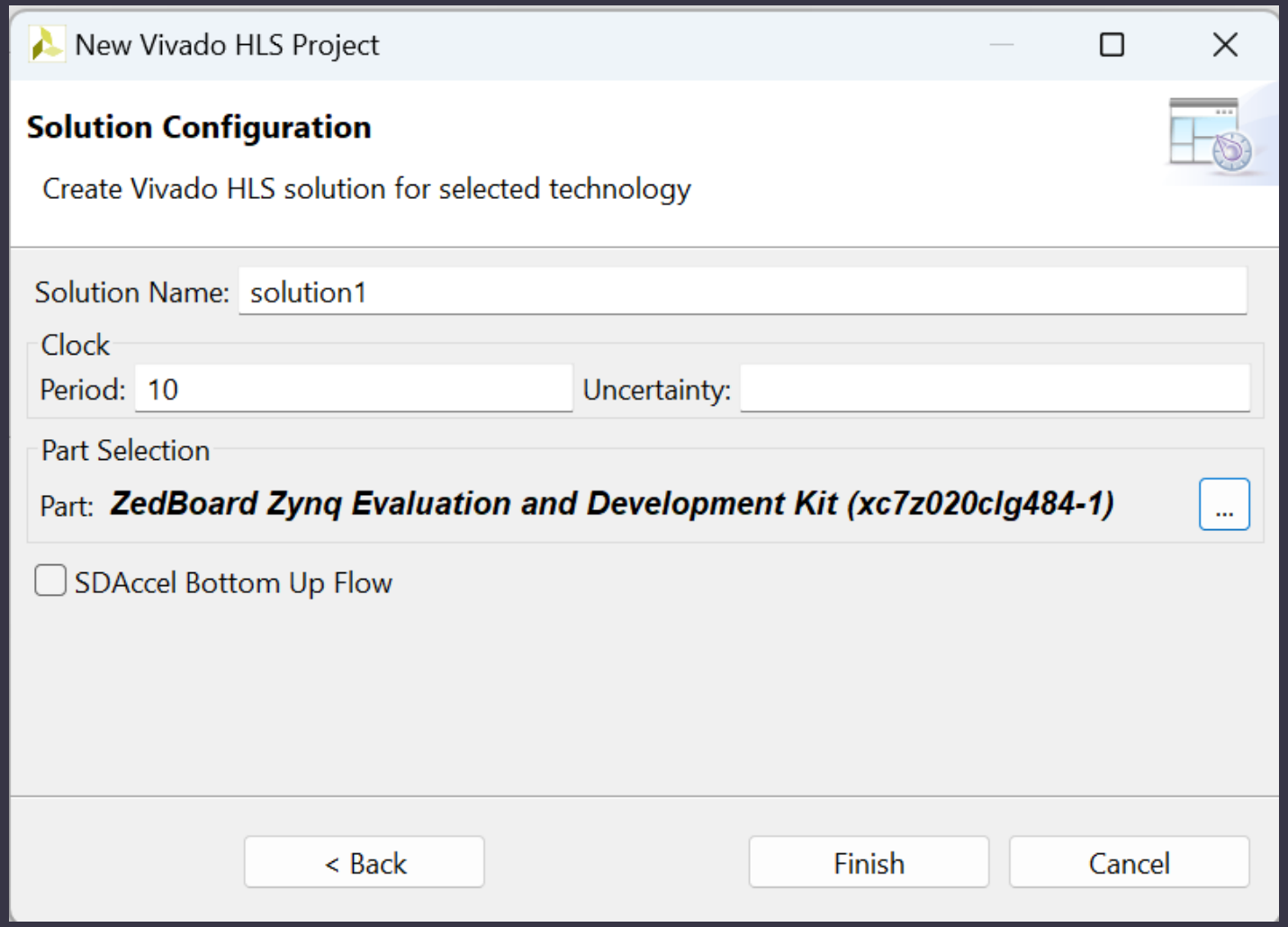


CLOCK ANATOMY

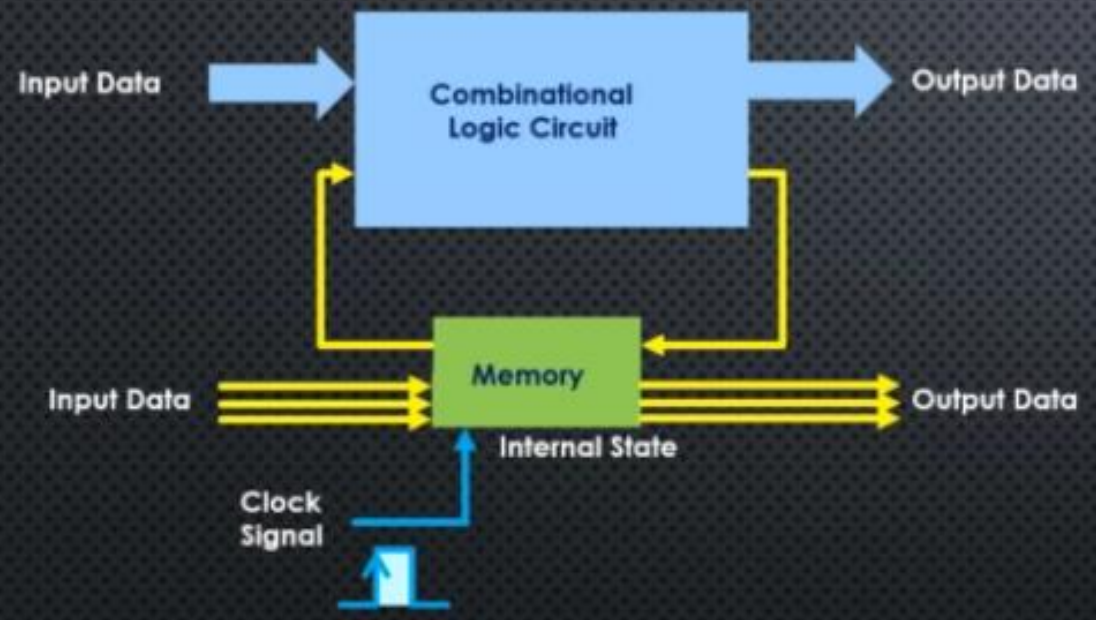


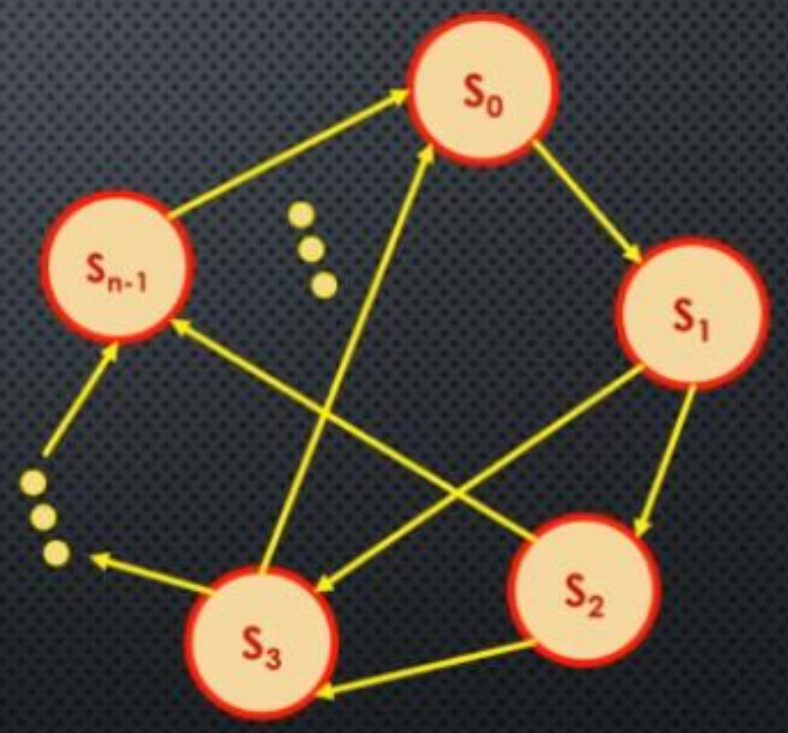


CLOCK CONSTRAINT IN HLS

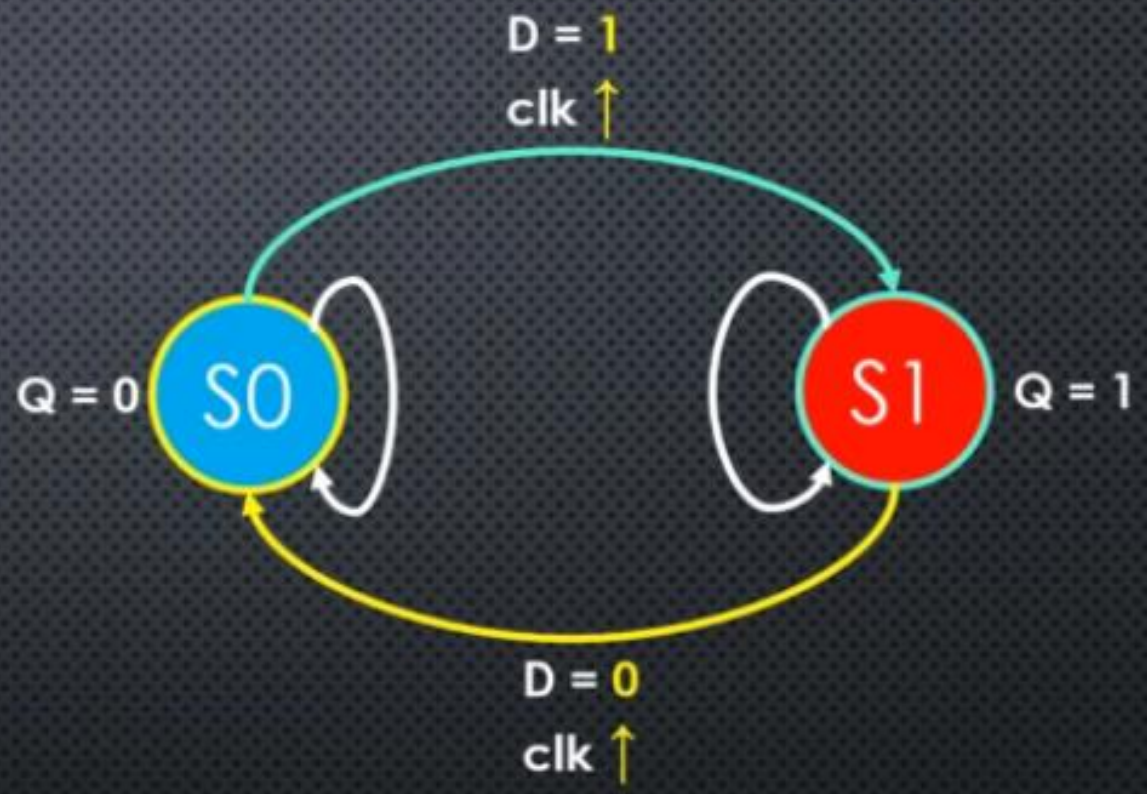


STATE

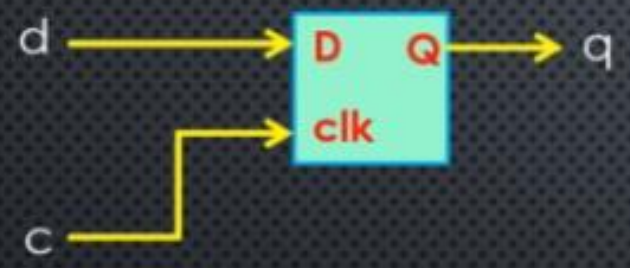

Input Data
Combinational Logic Circuit
Output Data
Memory
Input Data
Output Data
Internal State
Clock Signal

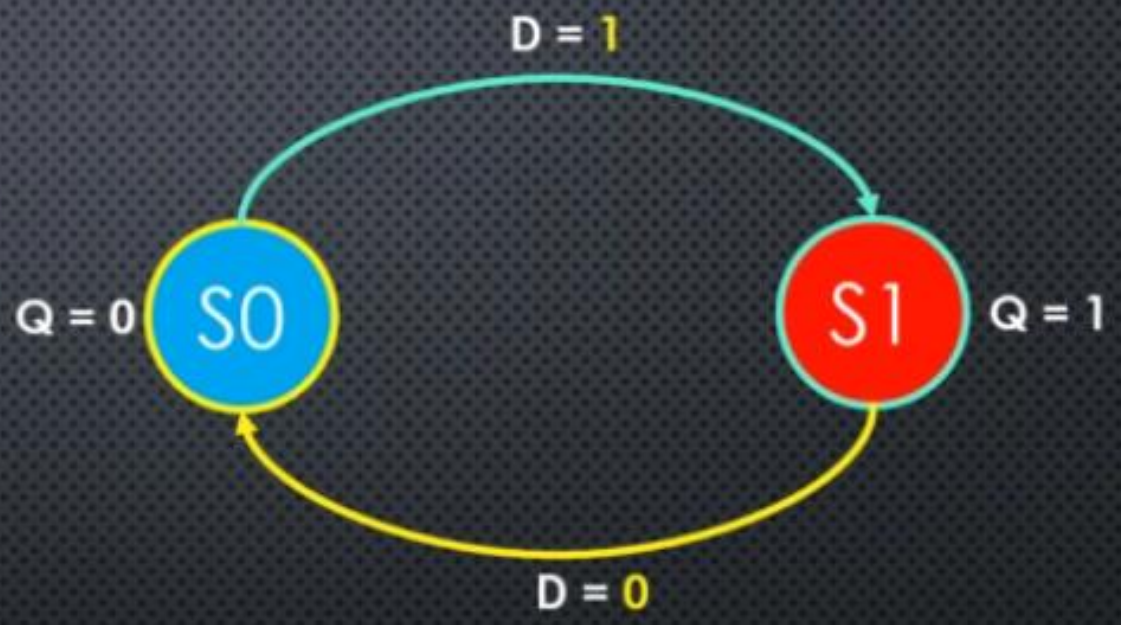

S_0
S_{n-1}
S_1
S_2
S_3

DFF STATE

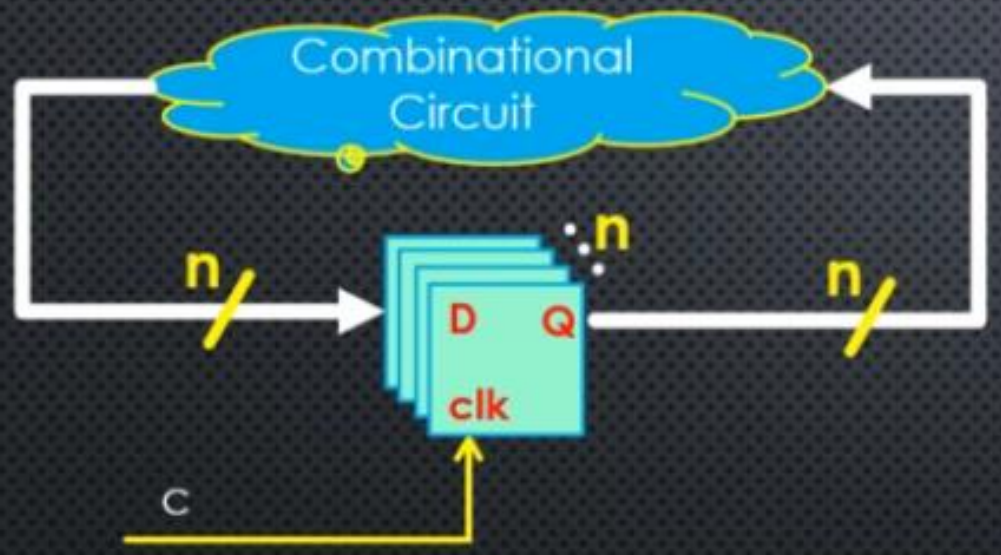


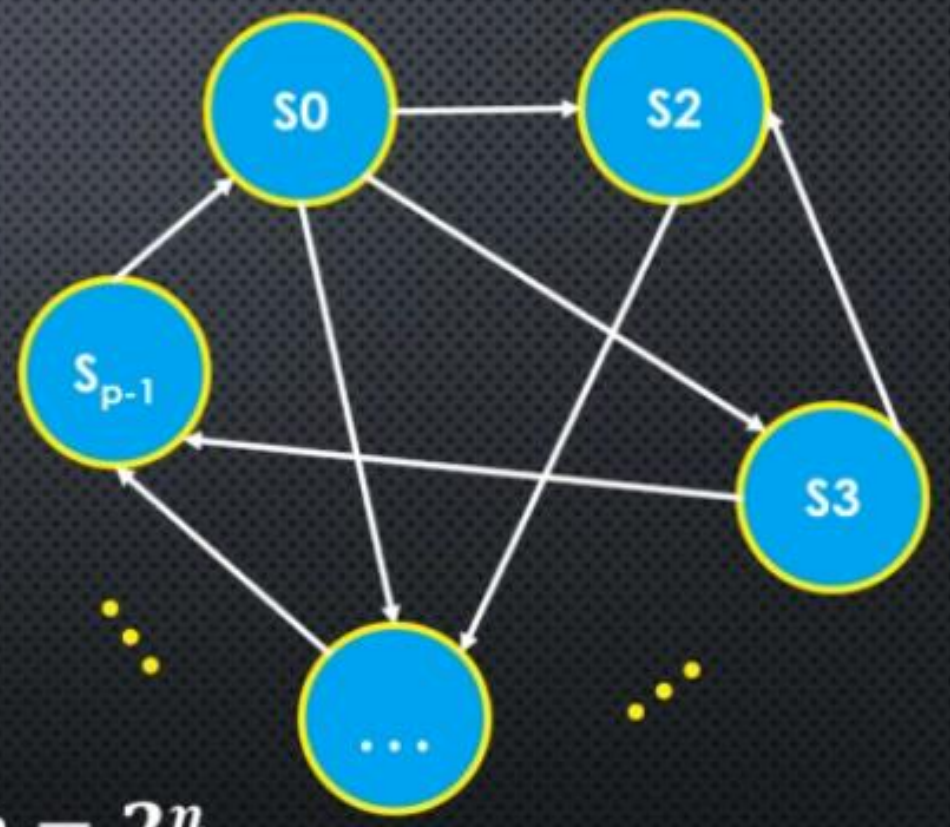
STATE DIAGRAM





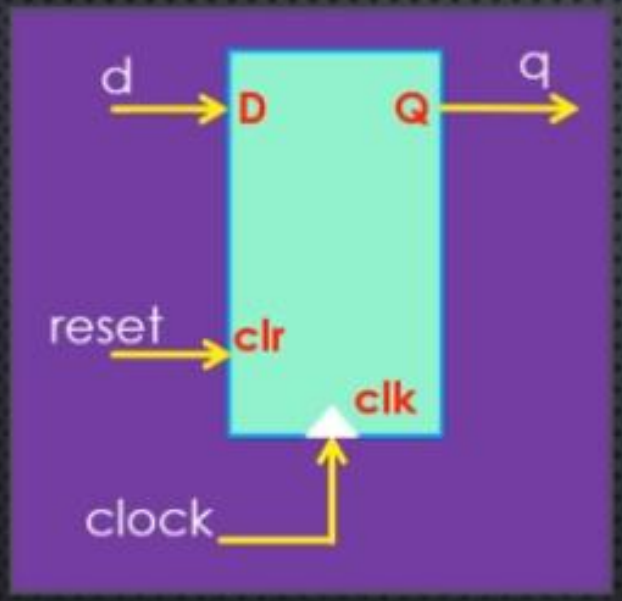
STATE DIAGRAM ATE



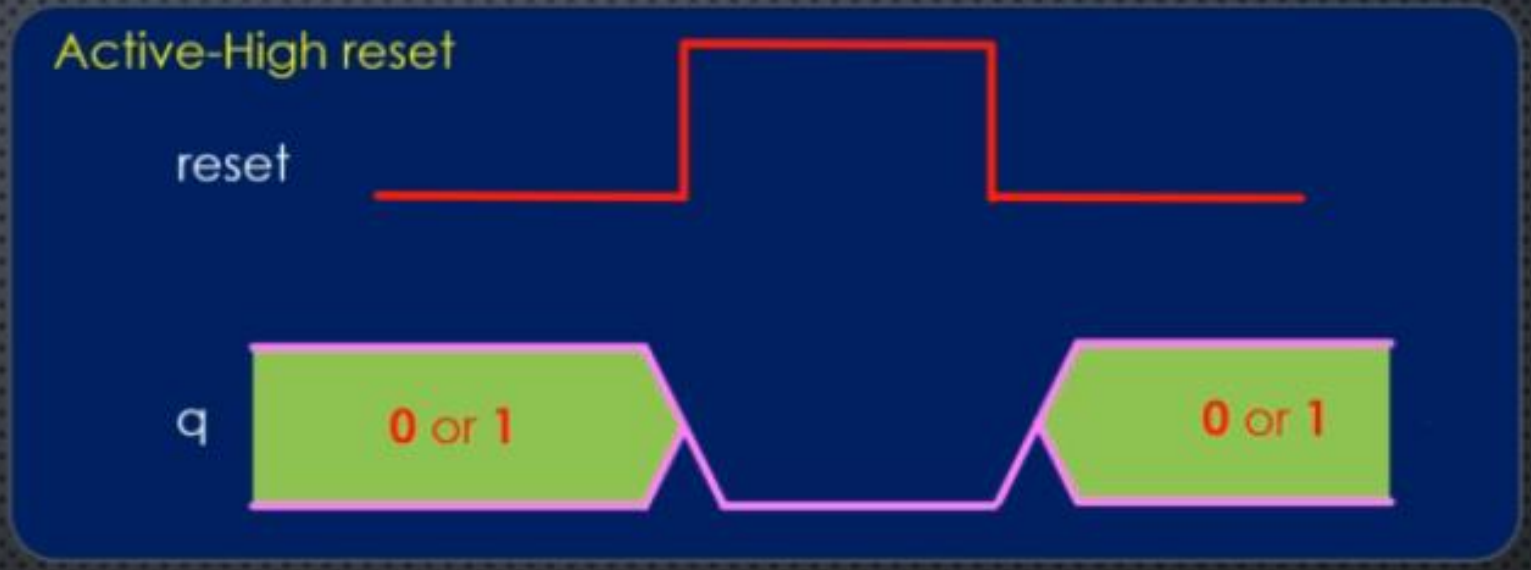


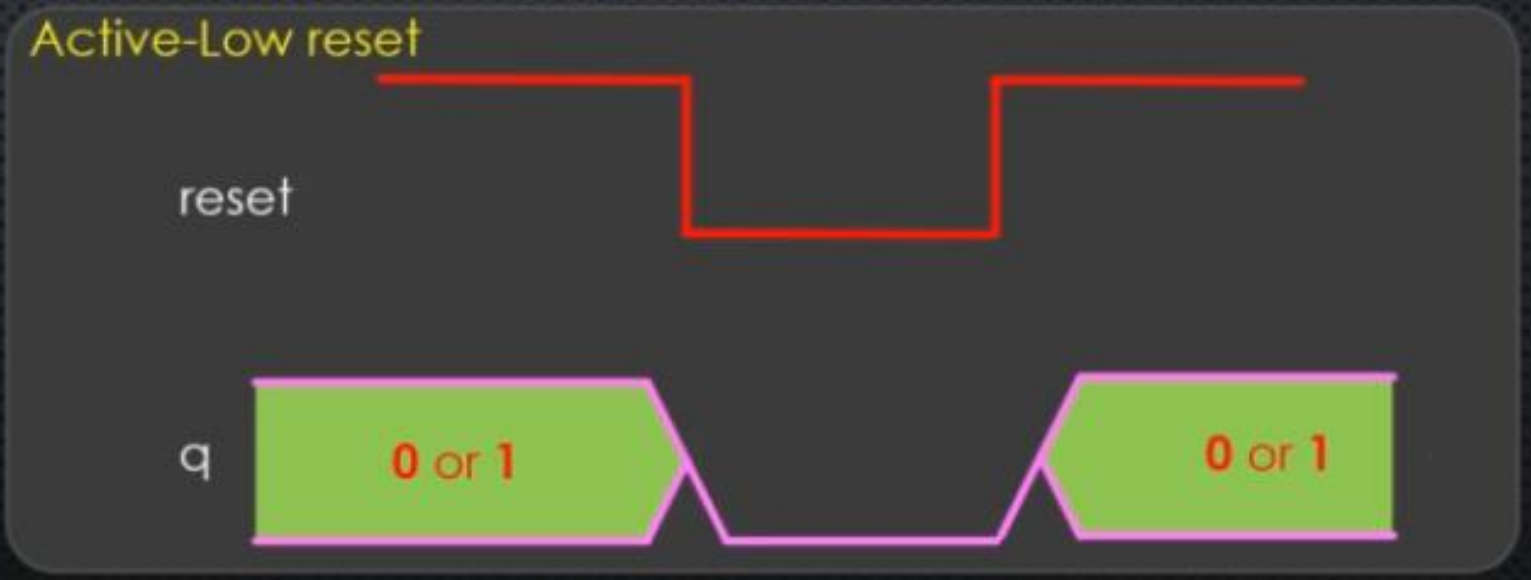
$$p = 2^n$$

RESET SIGNAL

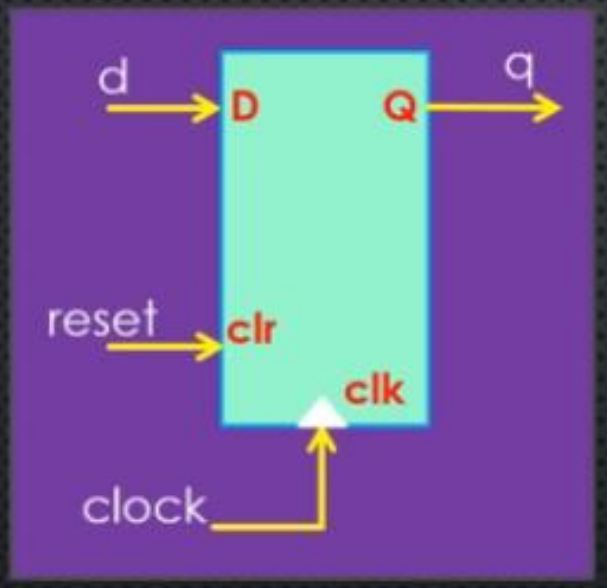


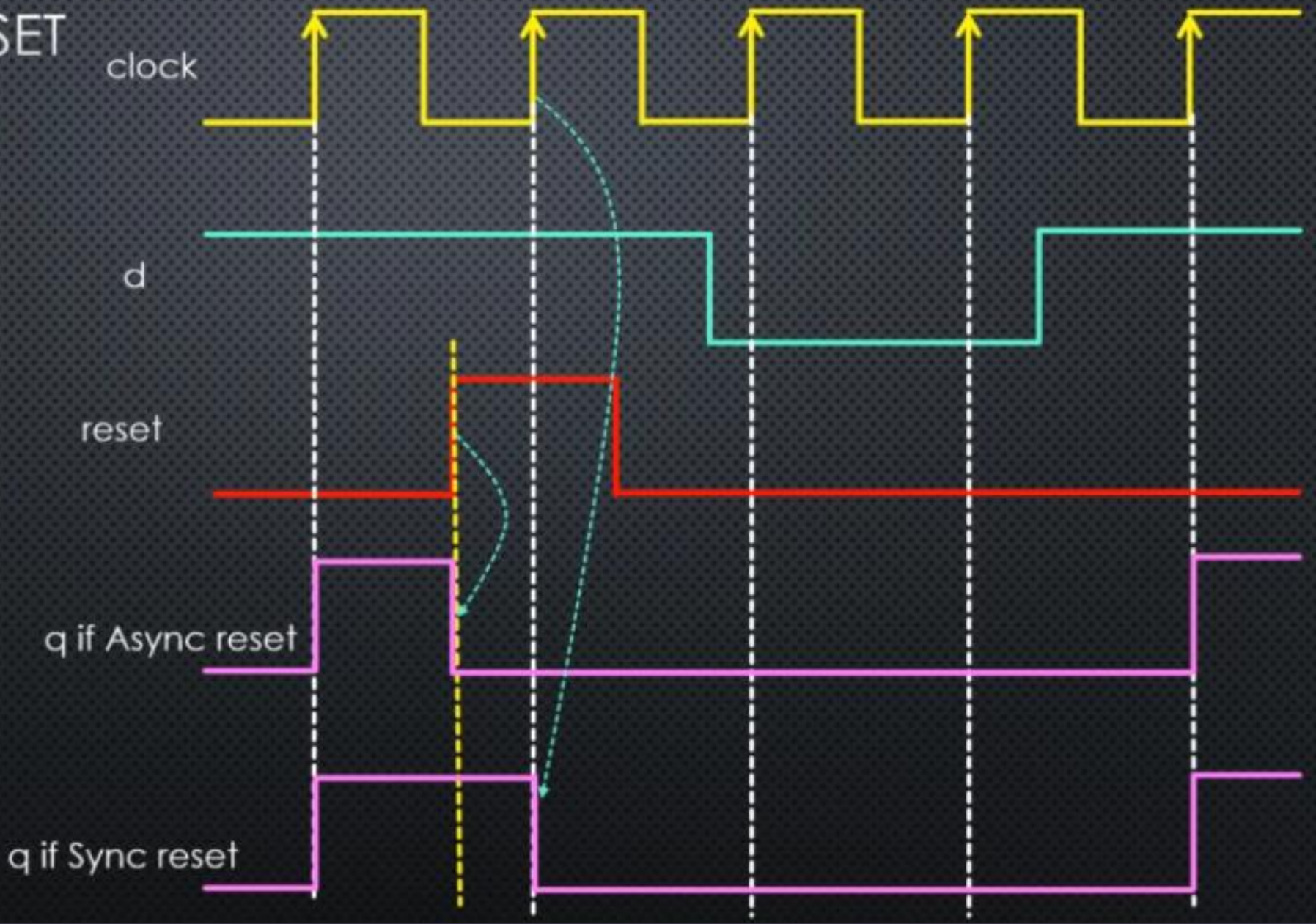
- Active High
- Active Low





ASYNC/SYNC RESET



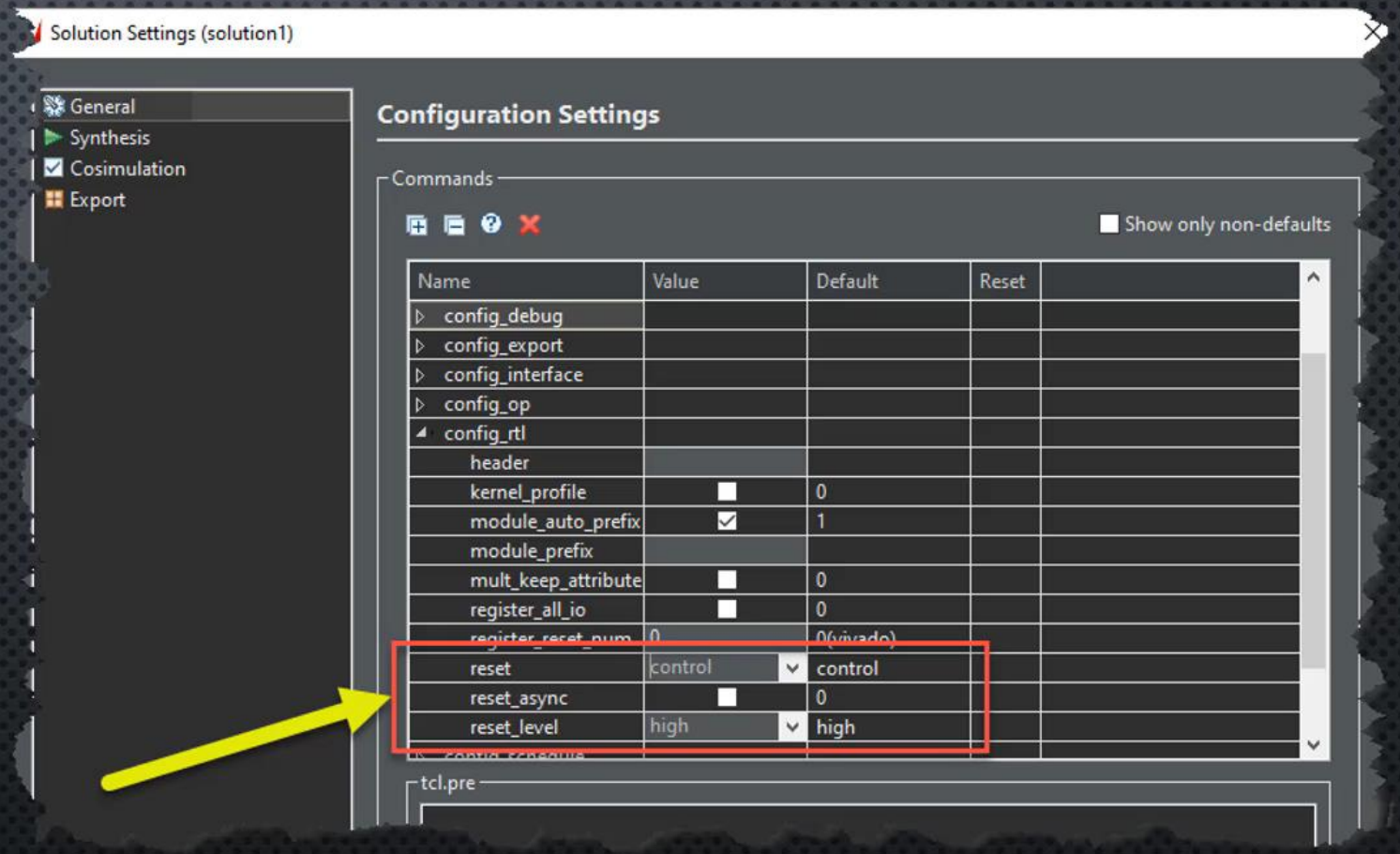


RESET SIGNAL CONFIGURATION IN HLS

reset: none
control
state
all

reset async:

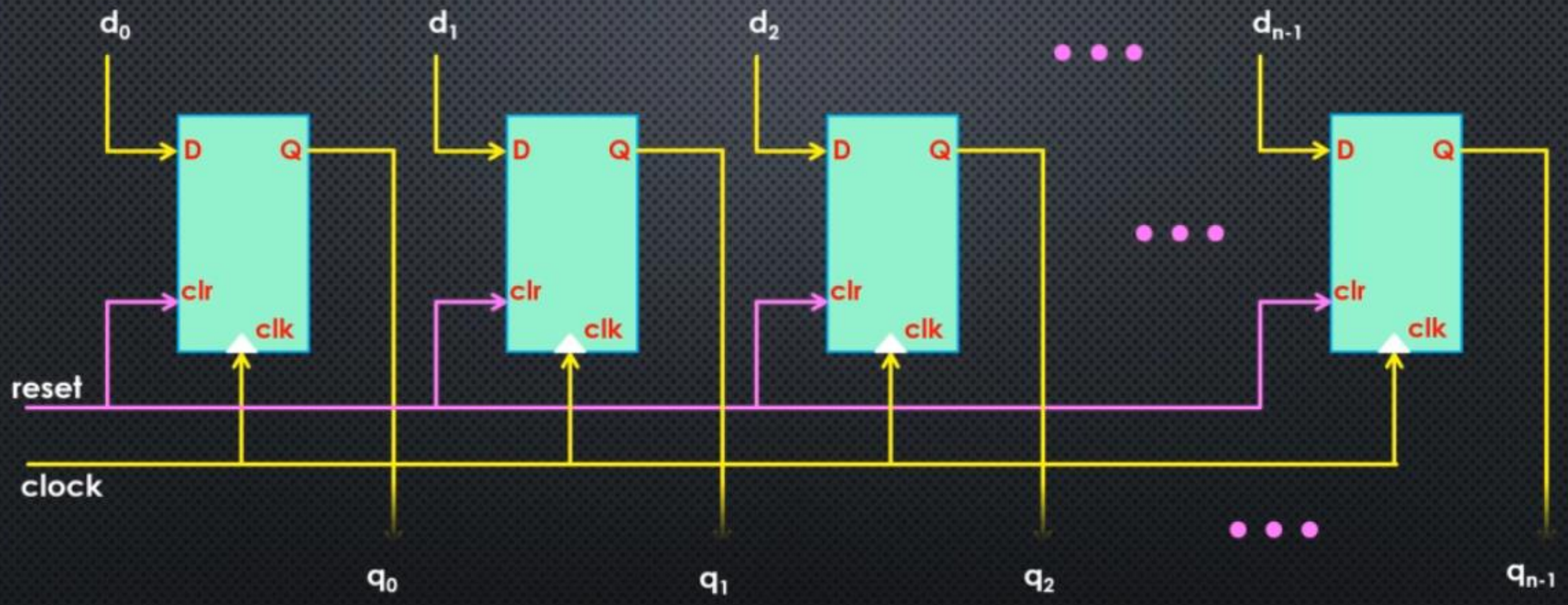
reset level: low
high



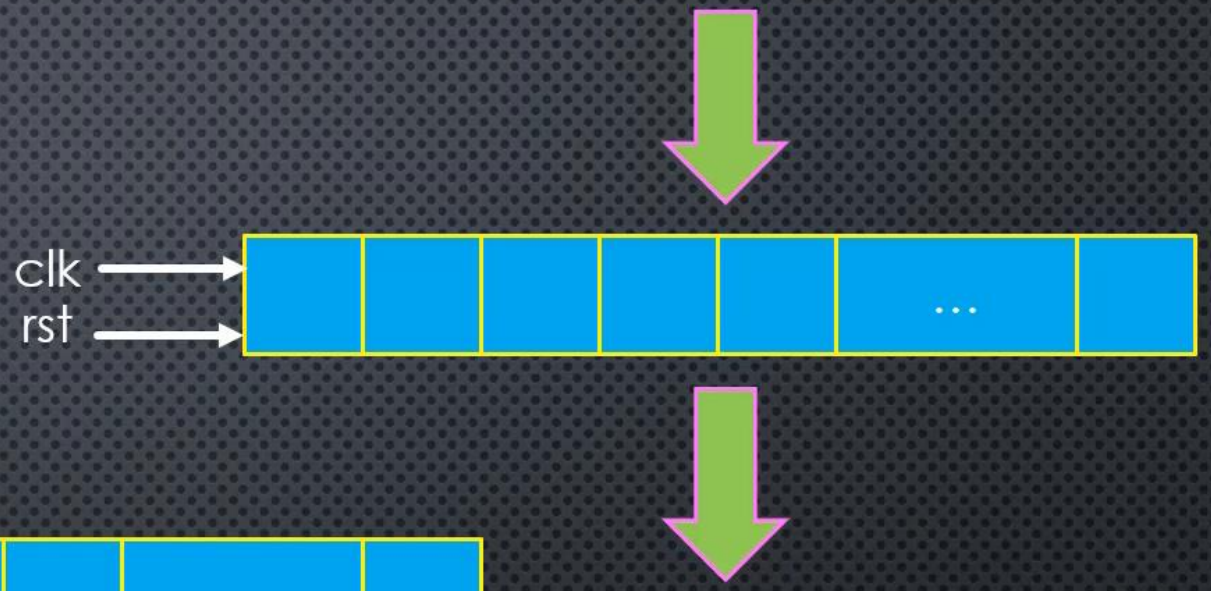
RESET SIGNAL IN HLS

```
void dffs_register(bool d, ap_uint<3> &q) {
#pragma HLS INTERFACE ap_none port=d
#pragma HLS INTERFACE ap_none port=q
#pragma HLS INTERFACE ap_ctrl_none port=return


  static ap_uint<3> reg = 0b000;

  reg = reg >> 1;
  reg[2] = d;

  q = reg;
}
```

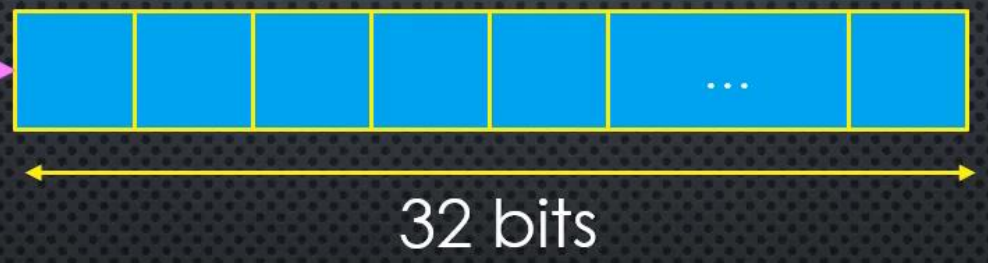
SIMPLE REGISTER STRUCTURE



REGISTERS IN HLS



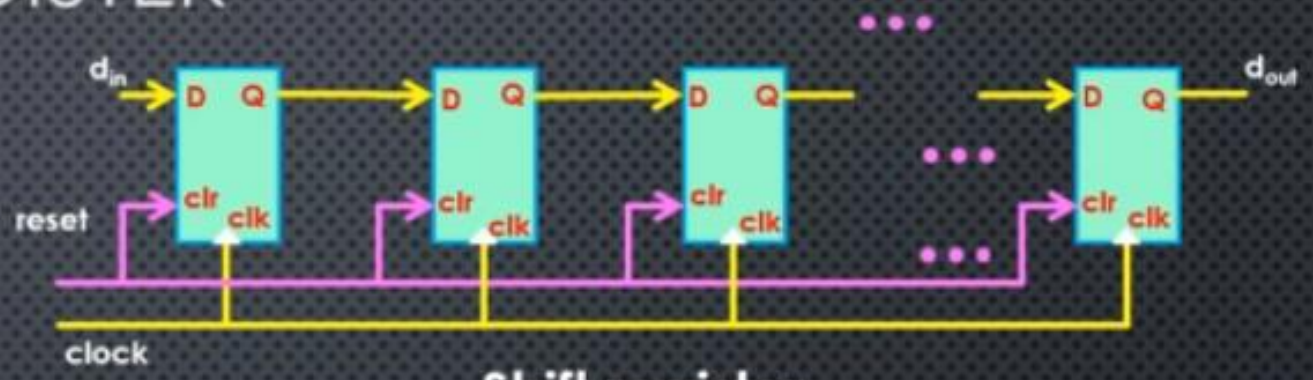

```
void func(...) {
    static int x;
    ...
    ...
}
```



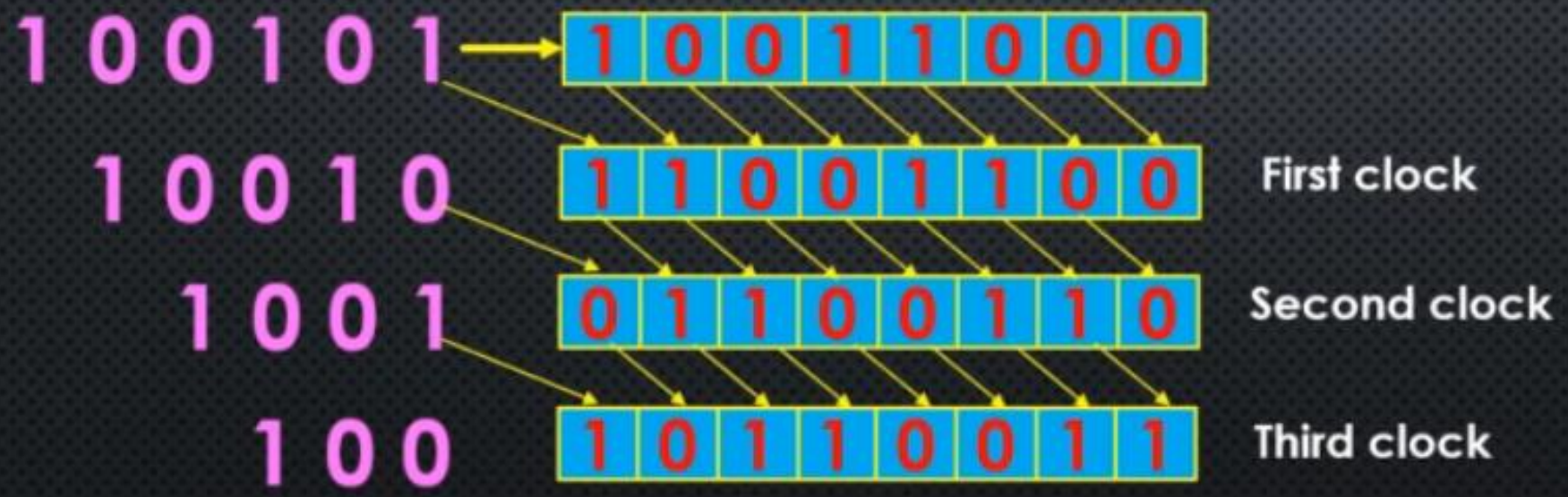

```
for (unsigned int i = 0; i < 1024; i++)
{
  ...
}
```

10-bit register

REGISTER TYPES - SHIFT REGISTER



Shift register

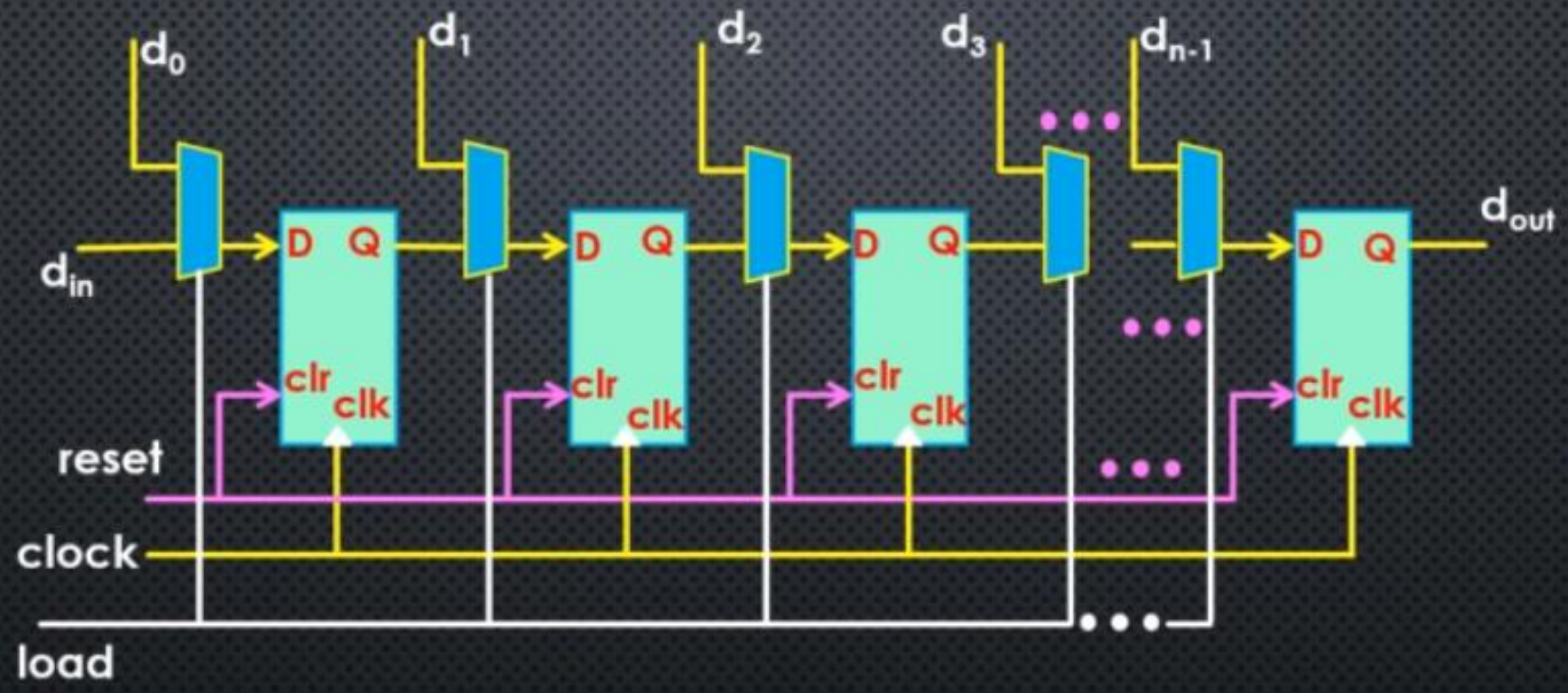


First clock

Second clock

Third clock

REGISTER TYPES – PARALLEL LOAD SHIFT REGISTER



Parallel load shift register

REGISTER TYPES IN HLS

```
void func(...) {
    static ap_int<10> x;
    ...
    x = 2;
    ...
    x = x >> 2;
    ...
    x = x << 4;
    ...
    x[5] = 10;
    ...
}
```

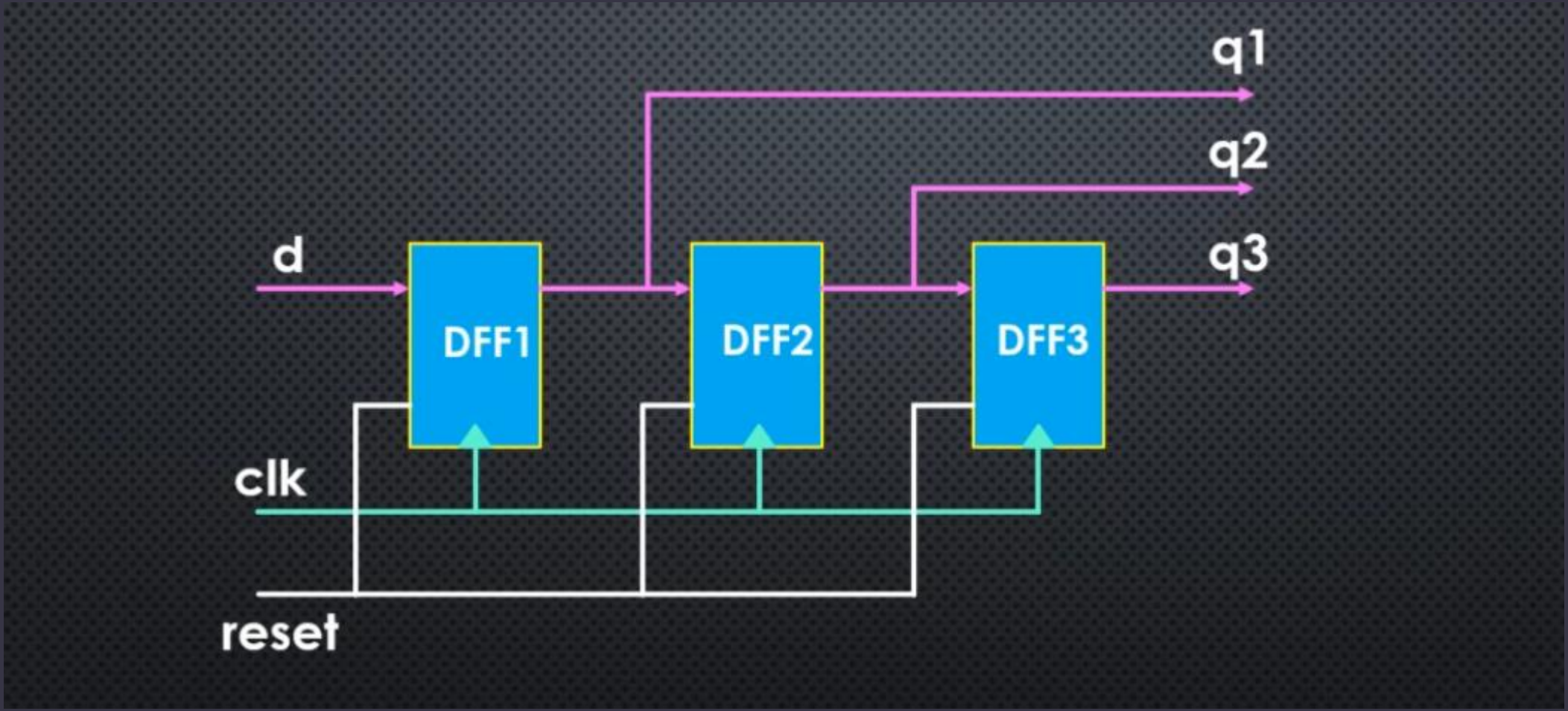
Parallel load

Shift right

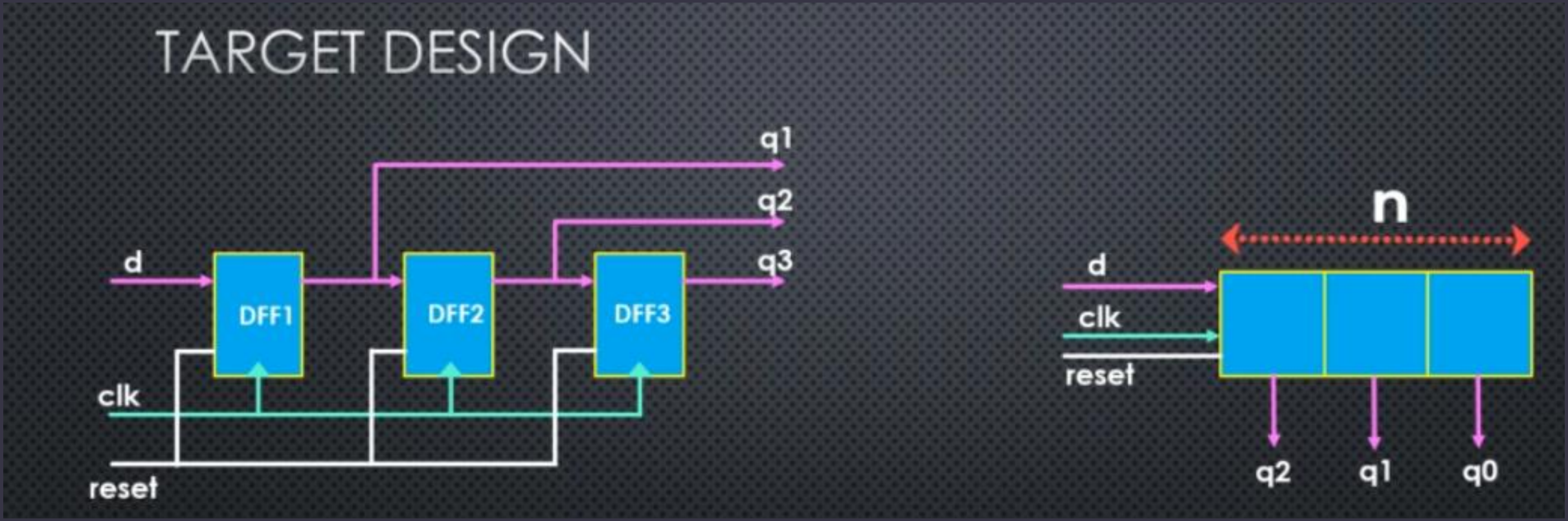
Shift left

Bit access

BASIC EXAMPLE WITH VITIS-1



BASIC EXAMPLE WITH VITIS-2



Any Question…

Thank you