

```

1. 1. /*
2   2     Name the gates, switches and clocks.
3   3     Number of inputs and outputs
4   4     does not need to be defined for DTYPE
5   5 */
6
7   6 DEVICES D1:DTYPE,
8   7         D2:DTYPE,
9   8         N1:NAND 2,
10  9         C1:CLOCK 8, # Period is a power of 2
11 10         S1:SWITCH 0,
12 11         S2:SWITCH 1,
13 12         S3:SWITCH 0 ;
14
15 13 /* connect inputs and outputs */
16 14 CONNECT S1 > D1.SET,
17 15         S1 > D2.SET,
18 16         S2 > D1.DATA,
19 17         S3 > D1.CLEAR,
20 18         S3 > D2.CLEAR,
21
22 19         C1 > D1.CLK,
23 20         C1 > D2.CLK,
24
25 21         D1.Q > D2.DATA,
26 22         D2.Q > N1.I1,
27 23         D2.QBAR > N1.I2 ;
28
29 24 /* monitor certain signals */
30 25 MONITOR D1.QBAR,
31 26         N1 ;
32
33 27 END

```

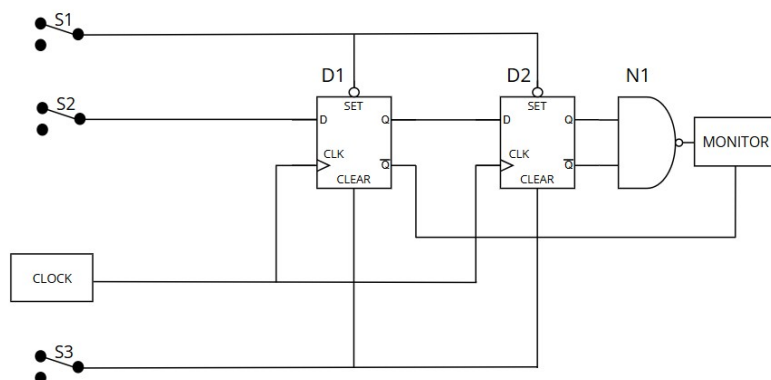


Figure 1: Visual logic circuit design for EBNF (example one) (1)

```

2.
1  /* This configuration is a full-adder */
2
3  /* name all devices */
4  DEVICES X1:XOR,
5          X2:XOR,
6          A1:AND 2,
7          A2:AND 2,
8          NO1:NOR 2,
9          O1:OR 2,
10         S1:SWITCH 1,
11         S2:SWITCH 1,
12         S3:SWITCH 0 ;
13
14 /* connect inputs and outputs */
15 CONNECT S1 > X1.I1,
16          S1 > A1.I1,
17          S2 > X1.I2,
18          S2 > A1.I2,
19          S3 > X2.I2,
20          S3 > A2.I2,
21          X1 > X2.I1,
22          X1 > A2.I1,
23          X2 > NO1.I1,
24          A1 > O1.I1,
25          A2 > O1.I2,
26          O1 > NO1.I2 ;
27
28 /* monitor particular signals */
29 MONITOR X2, # this is the summer bit
30         O1, # this is the carry bit
31         NO1 ;
32
33 END

```

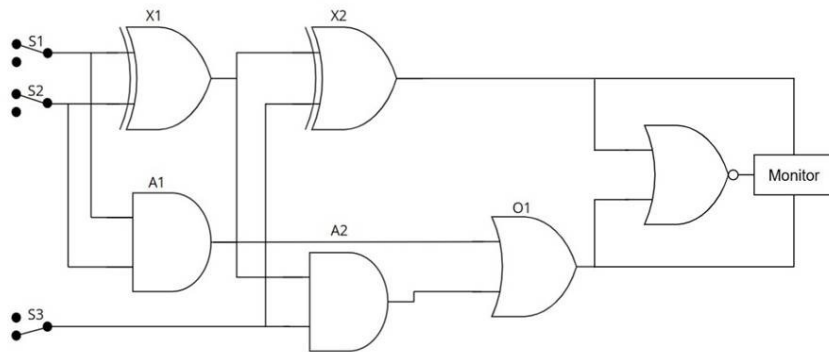
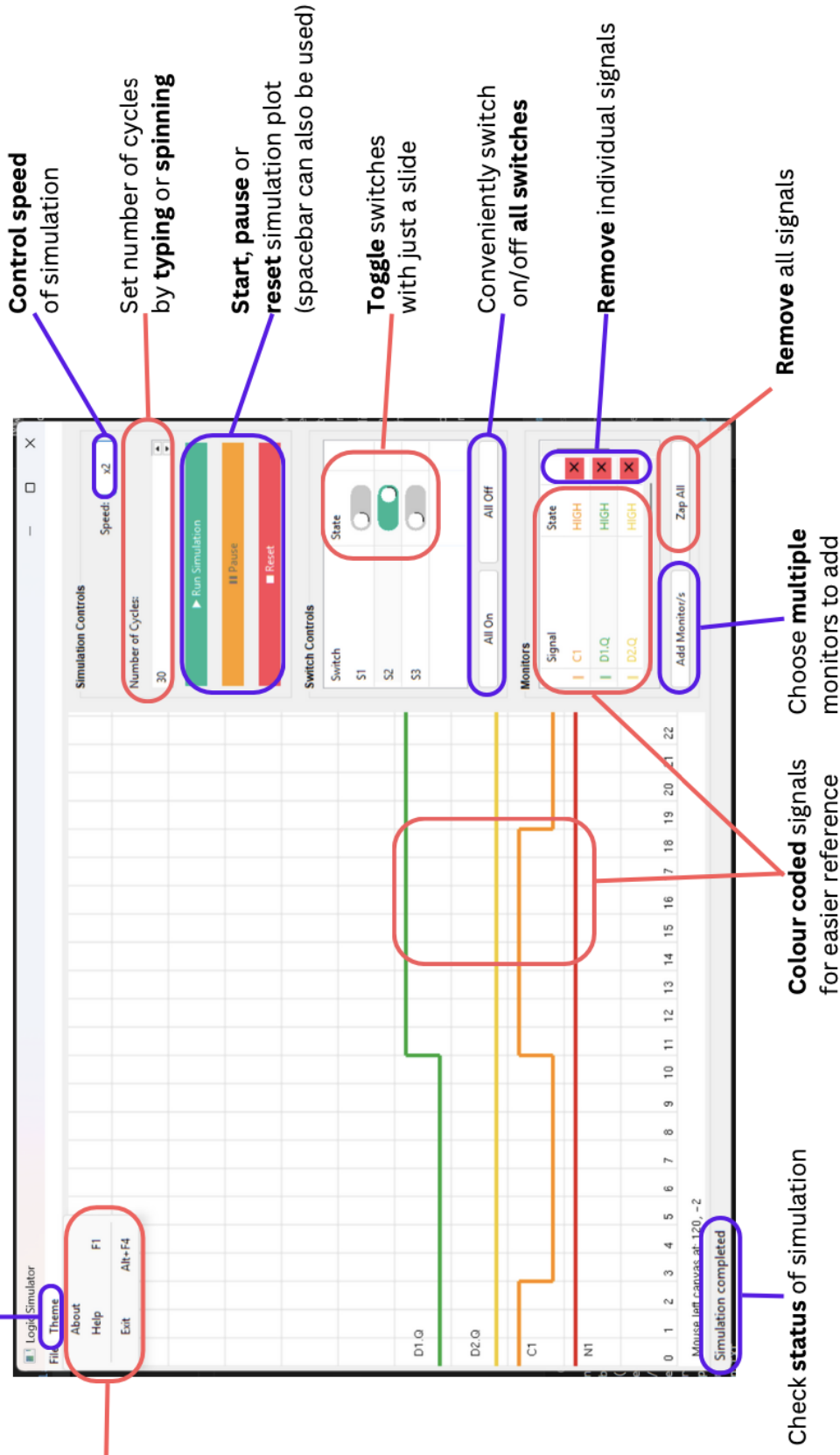


Figure 2: Visual logic circuit design for EBNF (example two) (1)

Switch between **Light** and **Dark mode** for power saving

Go-to **resource** for understanding how to use the simulator effectively



### Simulate your own logic circuit.

1. Clone our repository
2. Change directory to `GF2_logic_simulator/logsim`
3. Code the circuit using LogicHDL language
4. Run `python logsim.py <filename.txt>` in the terminal
5. GUI will pop up if successful.
6. Add signals to monitor
7. Start simulation

### Simulate a full-adder and a flip flop

1. Clone our repository
2. Change directory to `GF2_logic_simulator/logsim`
3. In the terminal, you can run these predefined files
  - a. `python logsim.py full_adder.txt`
  - b. `python logsim.py flip_flop.txt`
4. GUI will pop up if successful.
5. Add signals to monitor
6. Start simulation