SNS COLLEGE OF TECHNOLOGY
(An Autonomous Institution)
COIMBATORE-35
Accredited by NBA-AICTE and Accredited by NAAC – UGC with A++ Grade
Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai

23EET202 / DIGITAL ELECTRONICS AND LINEAR INTEGRATED CIRCUITS

II YEAR / III SEMESTER

UNIT-III: DESIGN OF SEQUENTIAL CIRCUITS, PLD, VHDL

Verilog HDL – PROGRAMMING - 1

TOPIC OUTLINE

VHDL – Types

Basic Gates – programming

Adders – programming

Subtractors – programming

Mux/Demux - programming

HDL

HDL - Hardware Descriptive Language: Describes the behaviour of Digital Circuits

Classified:

1. VHDL: Descriptive and deterministic language used to design digital electronic ICs for Specific applications and automation
Eg: PLD, CPLD, FPGA

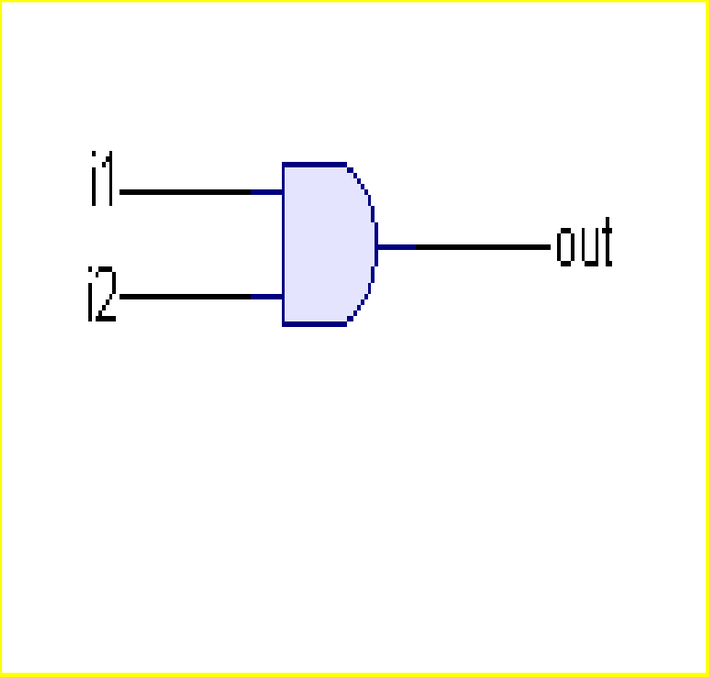
2. Verilog HDL: Hardware Modelling language used to model digital electronic IC – For learners, simple as C programming
Eg: Combinational and simple sequential circuits

VHDL - VERILOG®HDL

Types:

1. Structural Module – Gates, HA, FA, HS, FS
2. Data Flow Module – Encoder, decoder, mux, demux
3. Behavioral Module – Counter, Register, FF

AND Gate



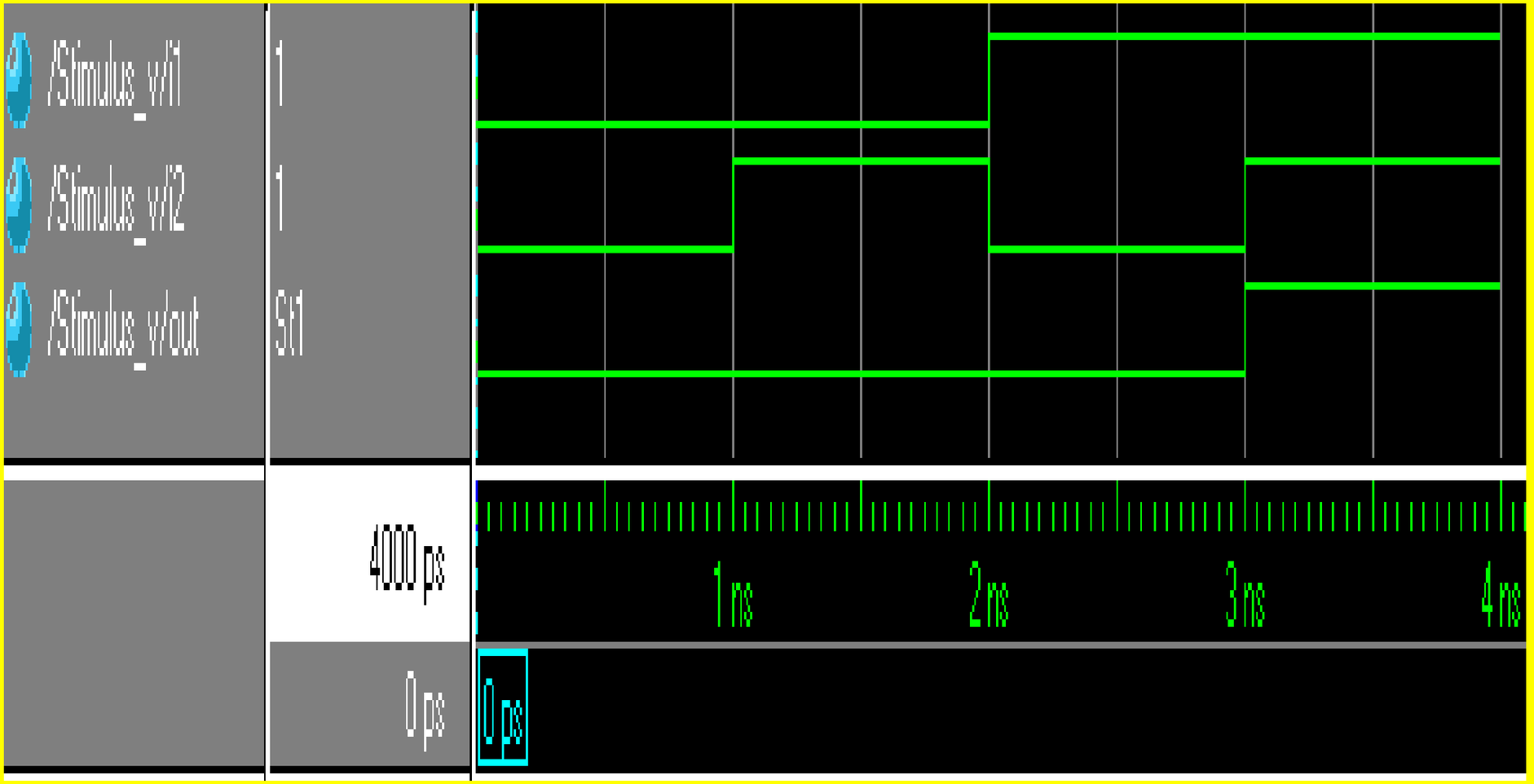

```
// Module Name:    Andgate
module Andgate(i1, i2, out);
    input i1;
    input i2;
    output out;
          and (out,i1,i2);
endmodule
```

Structural module….

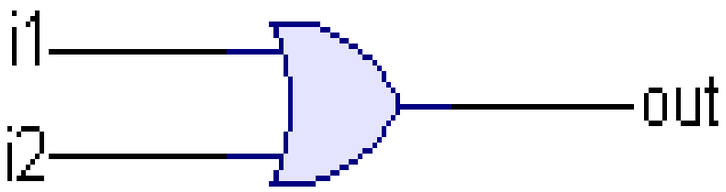
Output

```
#                    AND Gate
#          ---------------------------------------------
#          Input1          Input2          Output
#          ---------------------------------------------
#            0               0               0
#            0               1               0
#            1               0               0
#            1               1               1
#          ---------------------------------------------
```

Waveform


/Stimulus_v/i1
/Stimulus_v/i2
/Stimulus_v/out
1
1
St1
4000 ps
0 ps
0 ps
1 ns
2 ns
3 ns
4 ns

OR Gate

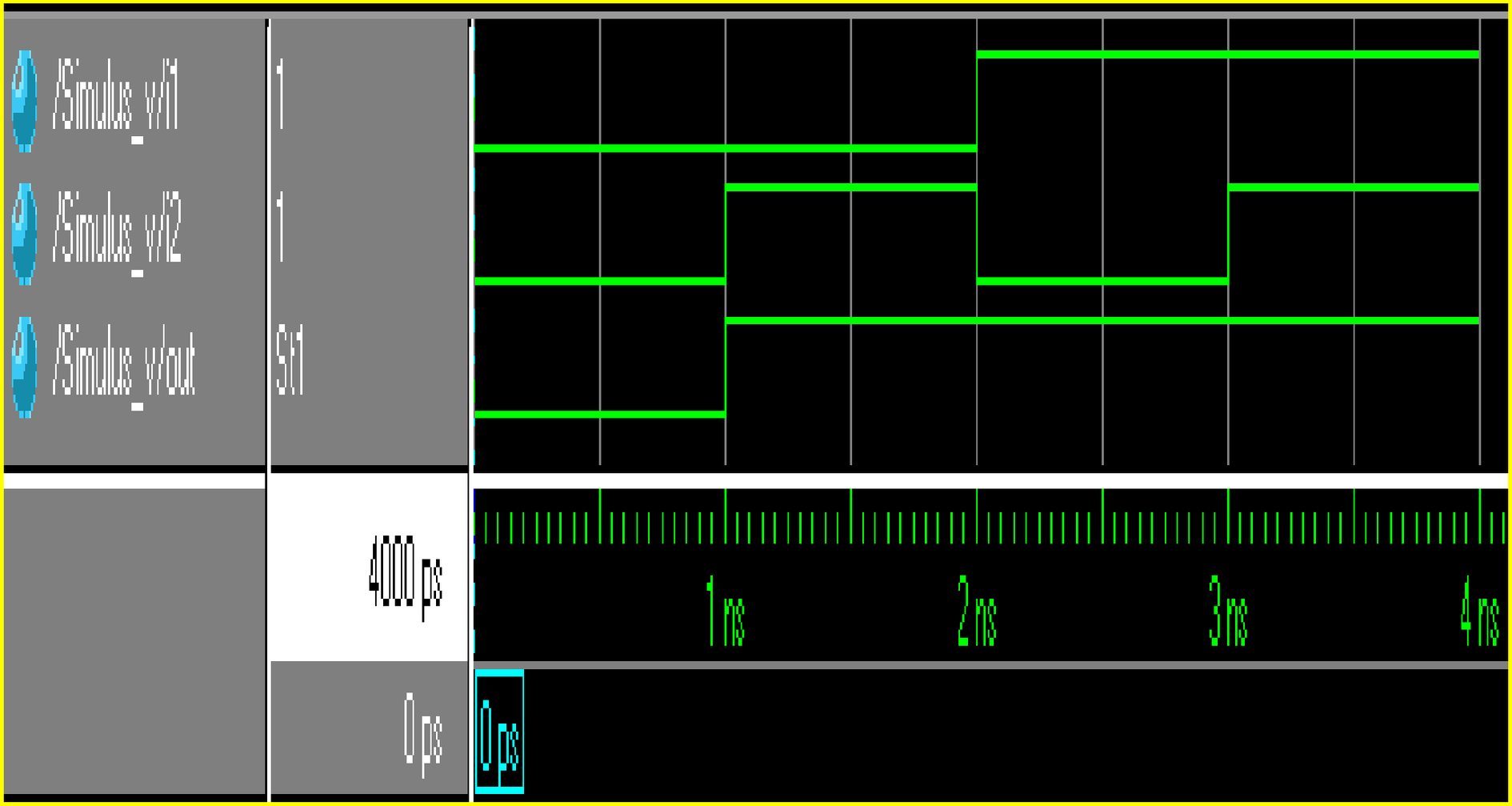



```
// Module Name:    Orgate
module Orgate(i1, i2, out);
    input i1;
    input i2;
    output out;
     or(out,i1,i2);
endmodule
```

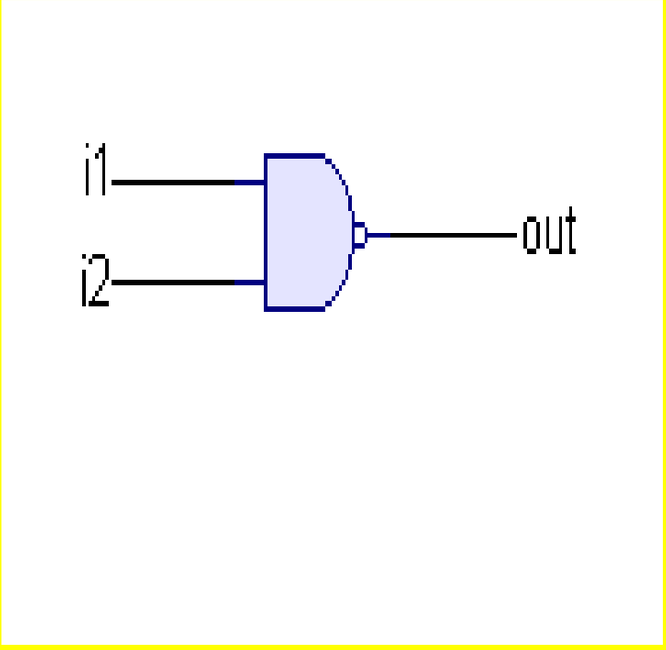
Output

```
#                  OR Gate
#          ---------------------------------------
#          Input1          Input2          Output
#          ---------------------------------------
#            0               0               0
#            0               1               1
#            1               0               1
#            1               1               1
#          ---------------------------------------
```

Waveform


/Stimulus_v/f1
/Stimulus_v/f2
/Stimulus_v/out
1
1
St1
4000 ps
1 ns
2 ns
3 ns
4 ns
0 ps
0 ps

NAND Gate

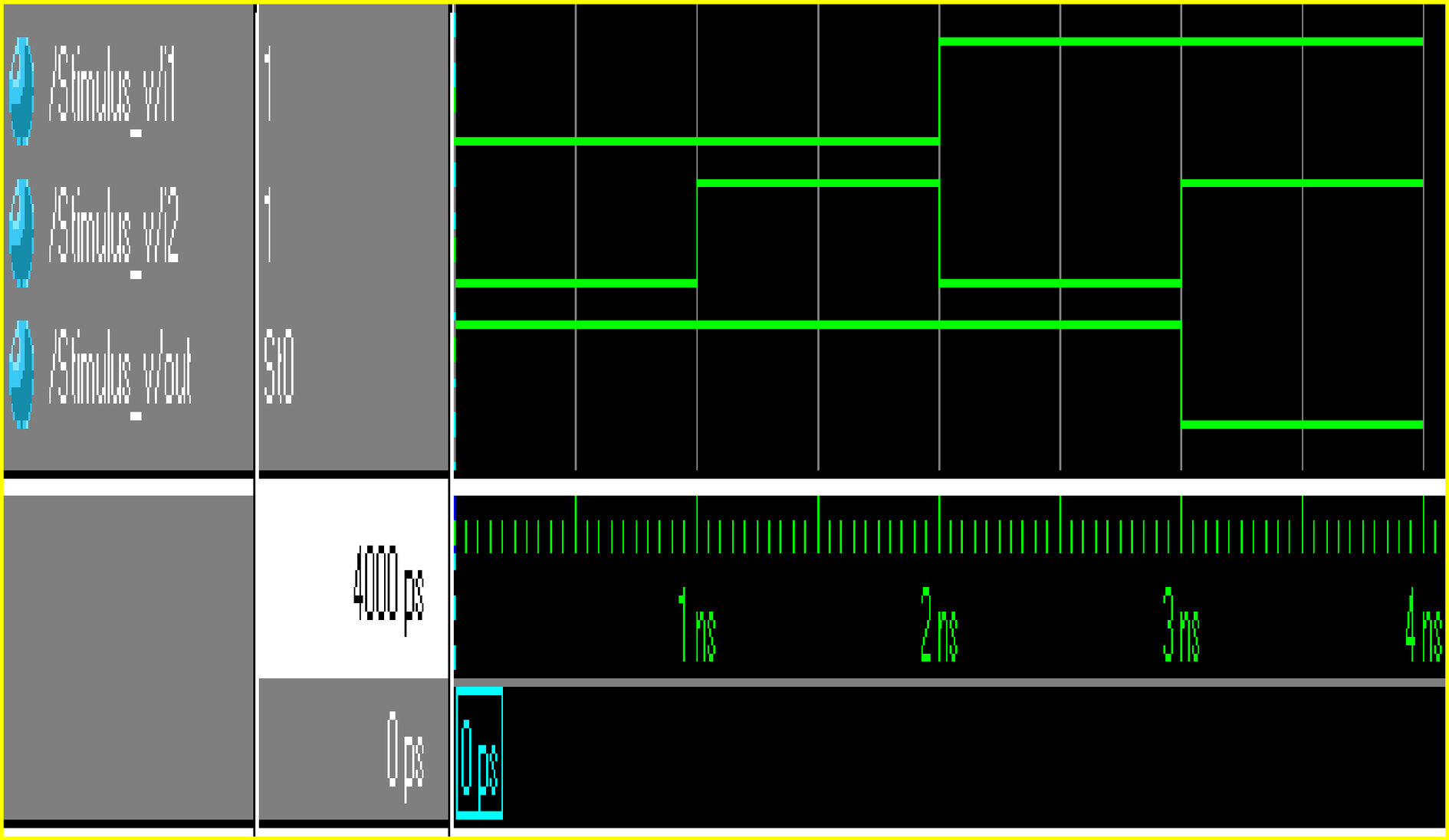



```
// Module Name:    Nandgate
module Nandgate(i1, i2, out);
    input i1;
    input i2;
    output out;
     nand(out,i1,i2);
endmodule
```

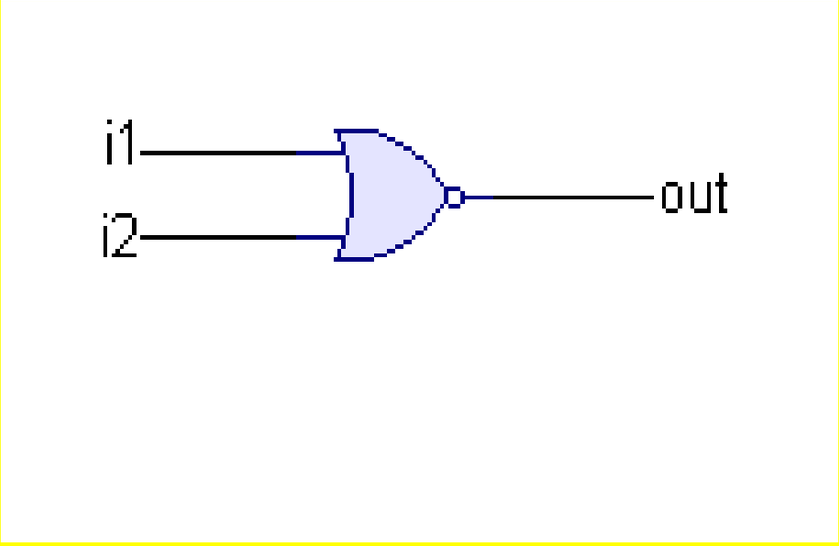
Output

```
#                    NAND Gate
#          ---------------------------------------------
#          Input1          Input2          Output
#          ---------------------------------------------
#            0               0               1
#            0               1               1
#            1               0               1
#            1               1               0
#          ---------------------------------------------
```

Waveform


/Stimulus_vin1
/Stimulus_vin2
/Stimulus_vout
1
1
St0
40000 ps
0 ps
1 ns
2 ns
3 ns
4 ns
0 ps

NOR Gate

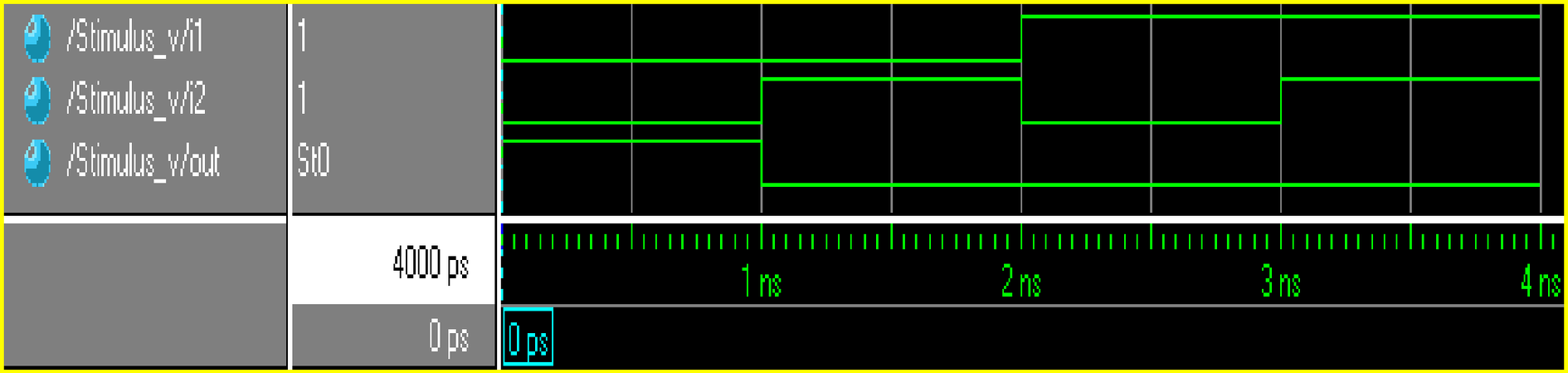



```
// Module Name:    Norgate
module Norgate(i1, i2, out);
    input i1;
    input i2;
    output out;
        nor(out,i1,i2);
endmodule
```

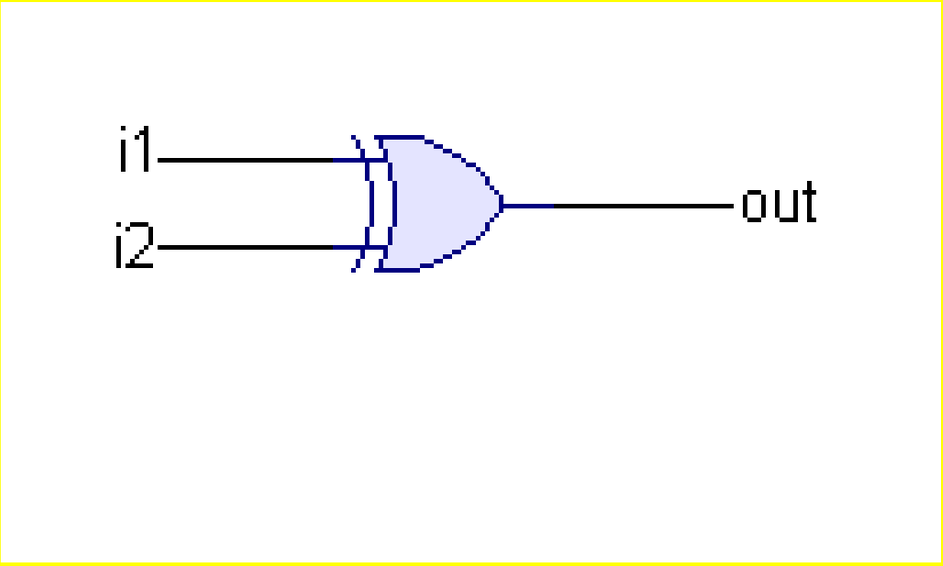
Output

```
#                    NOR Gate
#          ---------------------------------------------
#          Input1          Input2          Output
#          ---------------------------------------------
#            0               0               1
#            0               1               0
#            1               0               0
#            1               1               0
#          ---------------------------------------------
```

Waveform


/Stimulus_v/i1
/Stimulus_v/i2
/Stimulus_v/out
1
1
St0
4000 ps
0 ps
1 ns
2 ns
3 ns
4 ns
0 ps

XOR Gate

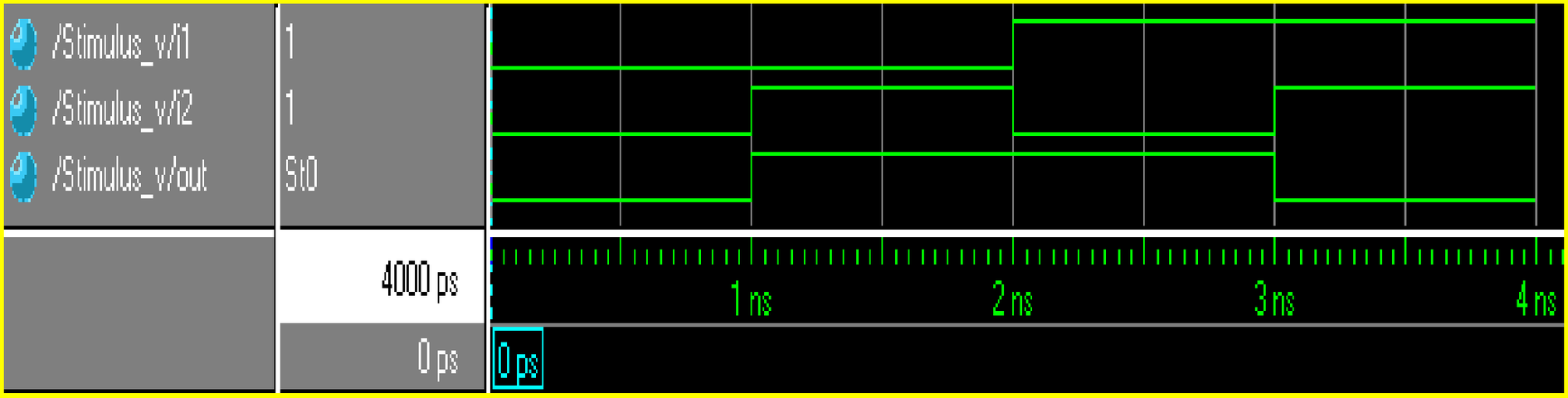



```
// Module Name:    Xorgate
module Xorgate(i1, i2, out);
    input i1;
    input i2;
    output out;
    xor(out,i1,i2);
endmodule
```

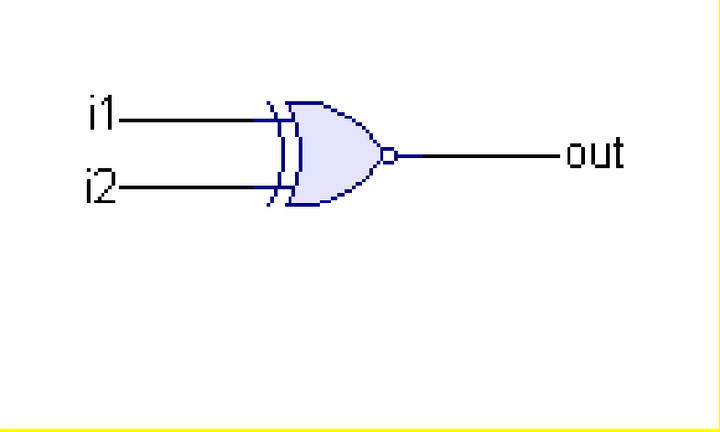
Output

```
#                    XOR Gate
#          ---------------------------------------------
#          Input1          Input2              Output
#          ---------------------------------------------
#             0               0                  0
#             0               1                  1
#             1               0                  1
#             1               1                  0
#          ---------------------------------------------
```

Waveform


/Stimulus_v/i1
/Stimulus_v/i2
/Stimulus_v/out
1
1
St0
4000 ps
0 ps
0 ps
1 ns
2 ns
3 ns
4 ns

XNOR Gate

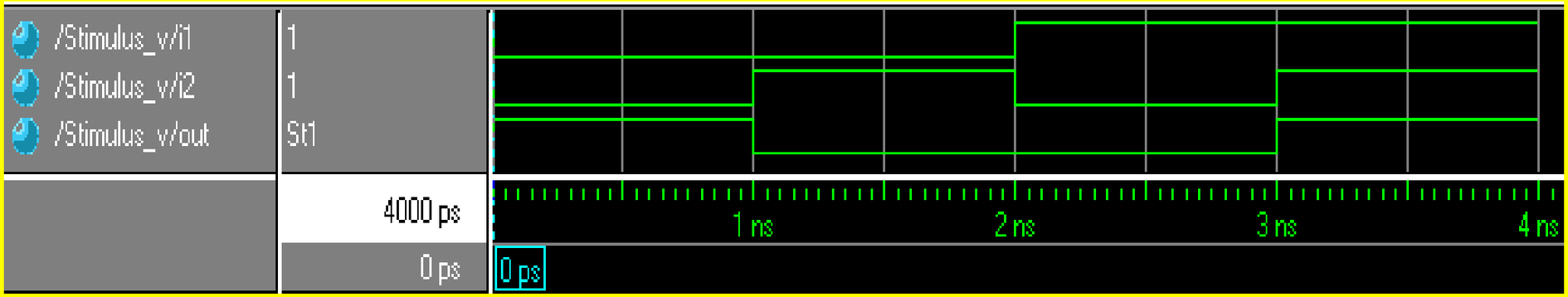



```
// Module Name:    Xnorgate
module Xnorgate(i1, i2, out);
    input i1;
    input i2;
    output out;
    xnor(out,i1,i2);
endmodule
```

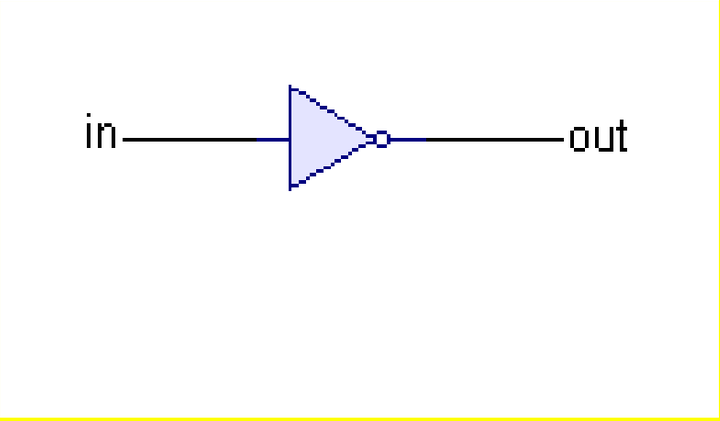
Output

```
#                    XNOR Gate
#          ---------------------------------------------
#          Input1          Input2          Output
#          ---------------------------------------------
#            0               0               1
#            0               1               0
#            1               0               0
#            1               1               1
#          ---------------------------------------------
```

Waveform


/Stimulus_v/i1
/Stimulus_v/i2
/Stimulus_v/out
1
1
St1
4000 ps
0 ps
1 ns
2 ns
3 ns
4 ns
0 ps

NOT Gate

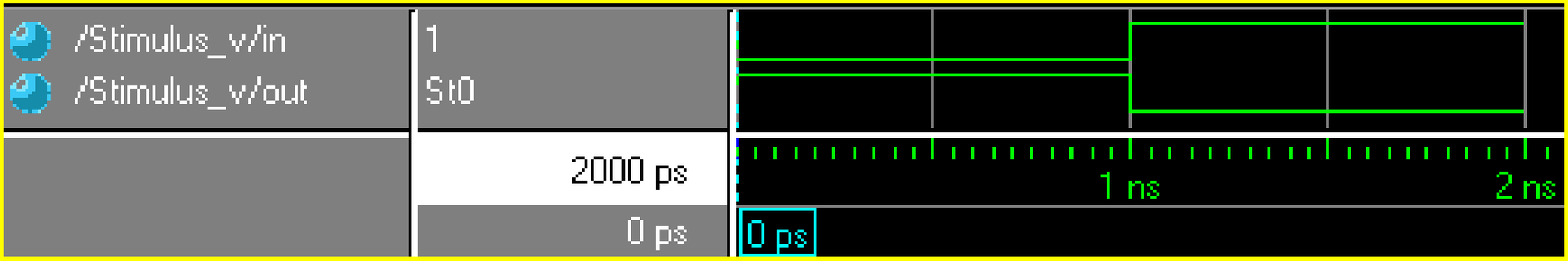



```
// Module Name:    Notgate
module Notgate(in, out);
    input in;
    output out;
    not(out,in);
endmodule
```

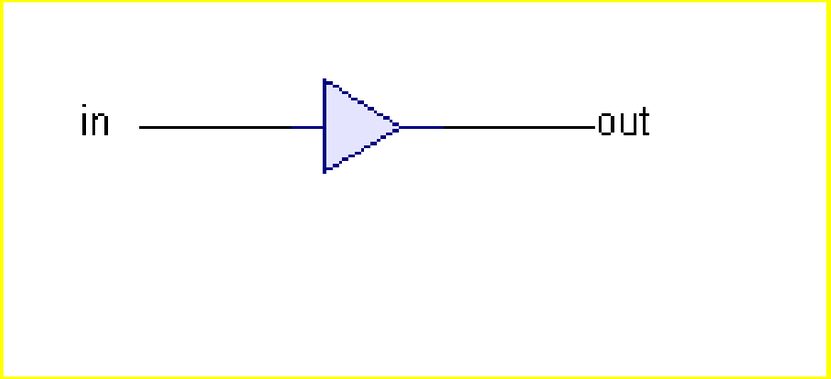
Output

```
#          NOT Gate
#      ----------------------------
#      Input          Output
#      ----------------------------
#        0              1
#        1              0
#      ----------------------------
```

Waveform



Buffer

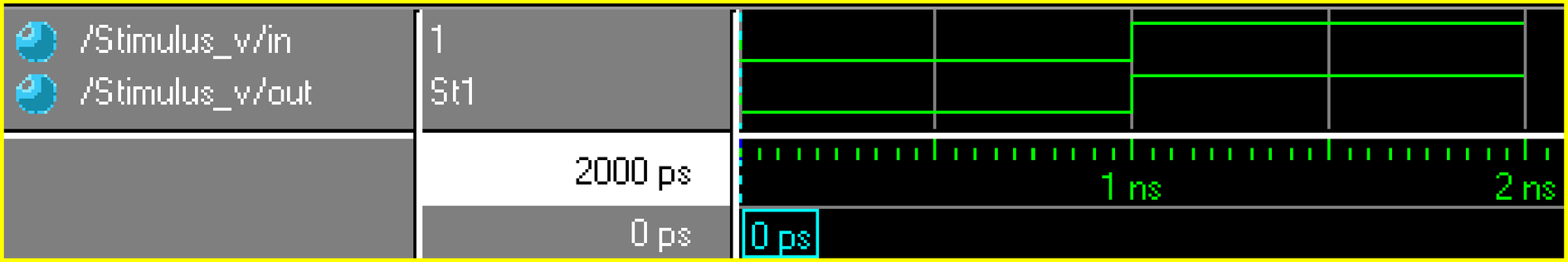



```
// Module Name:    Buffer
module Buffer(in, out);
    input in;
    output out;
    buf(out,in);
endmodule
```

Output

```
#            BUFFER
#      ------------------------
#      Input          Output
#      ------------------------
#        0              0
#        1              1
#      ------------------------
```

Waveform



RECAP..

Thank You