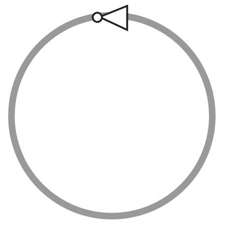
NCL Sandbox 1
Basics

https://github.com/karlfant/NCL_sandbox/basics

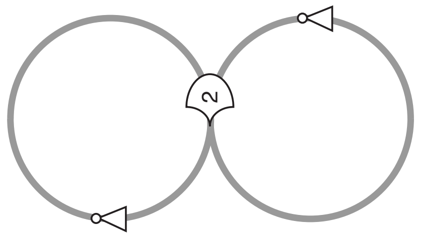
www.karlfant.net/sandbox

Karl Fant
Aug, 2015

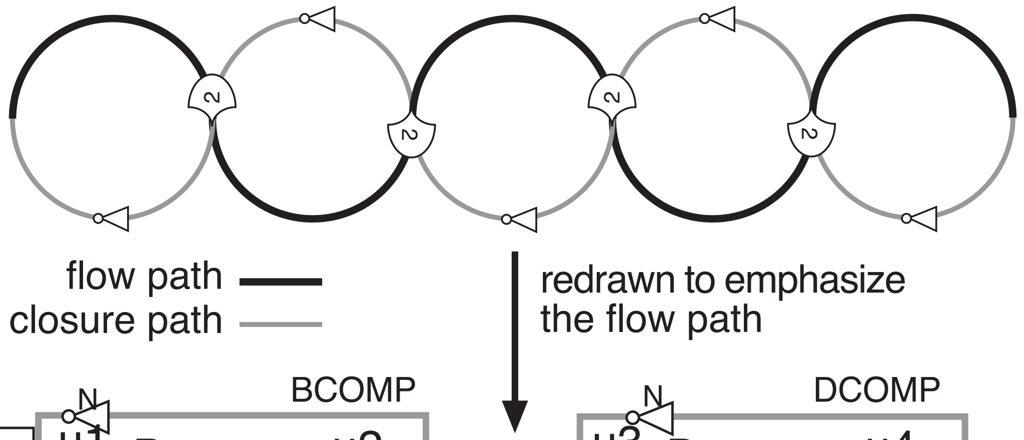
Oscillation

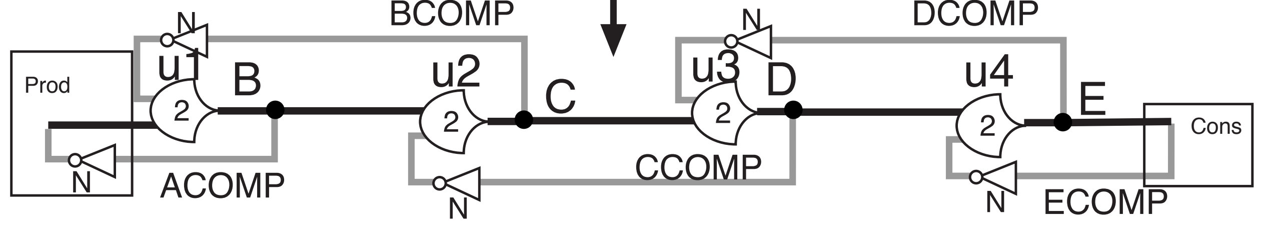


Linked Oscillations

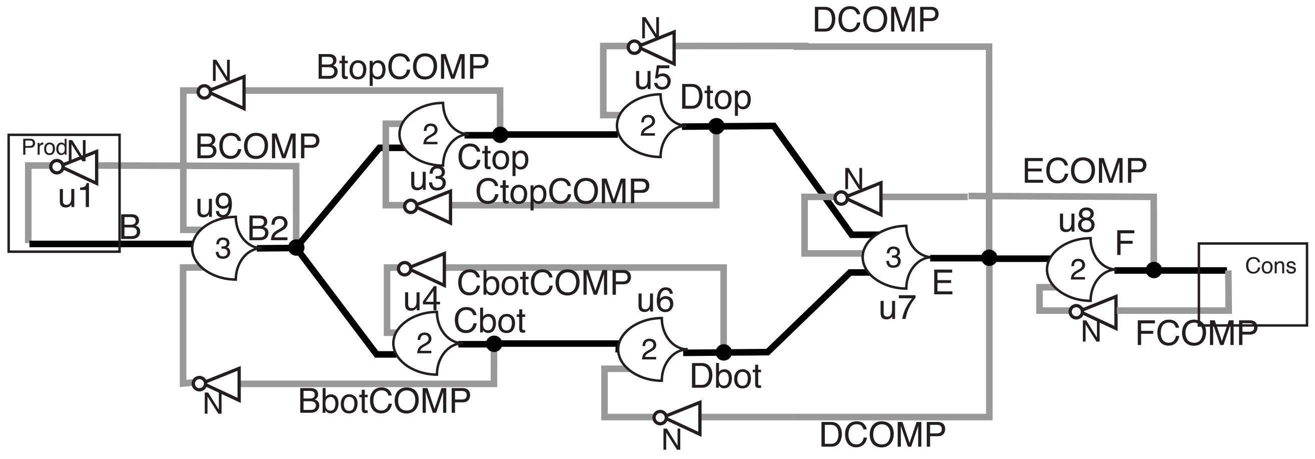


Pipeline of Linked Oscillations: The Flow Path

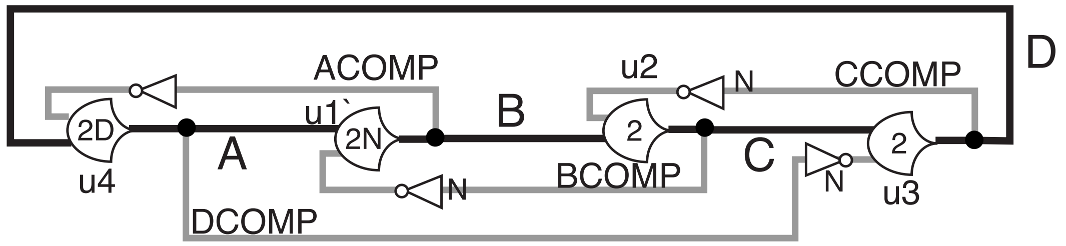




pipeline1.v

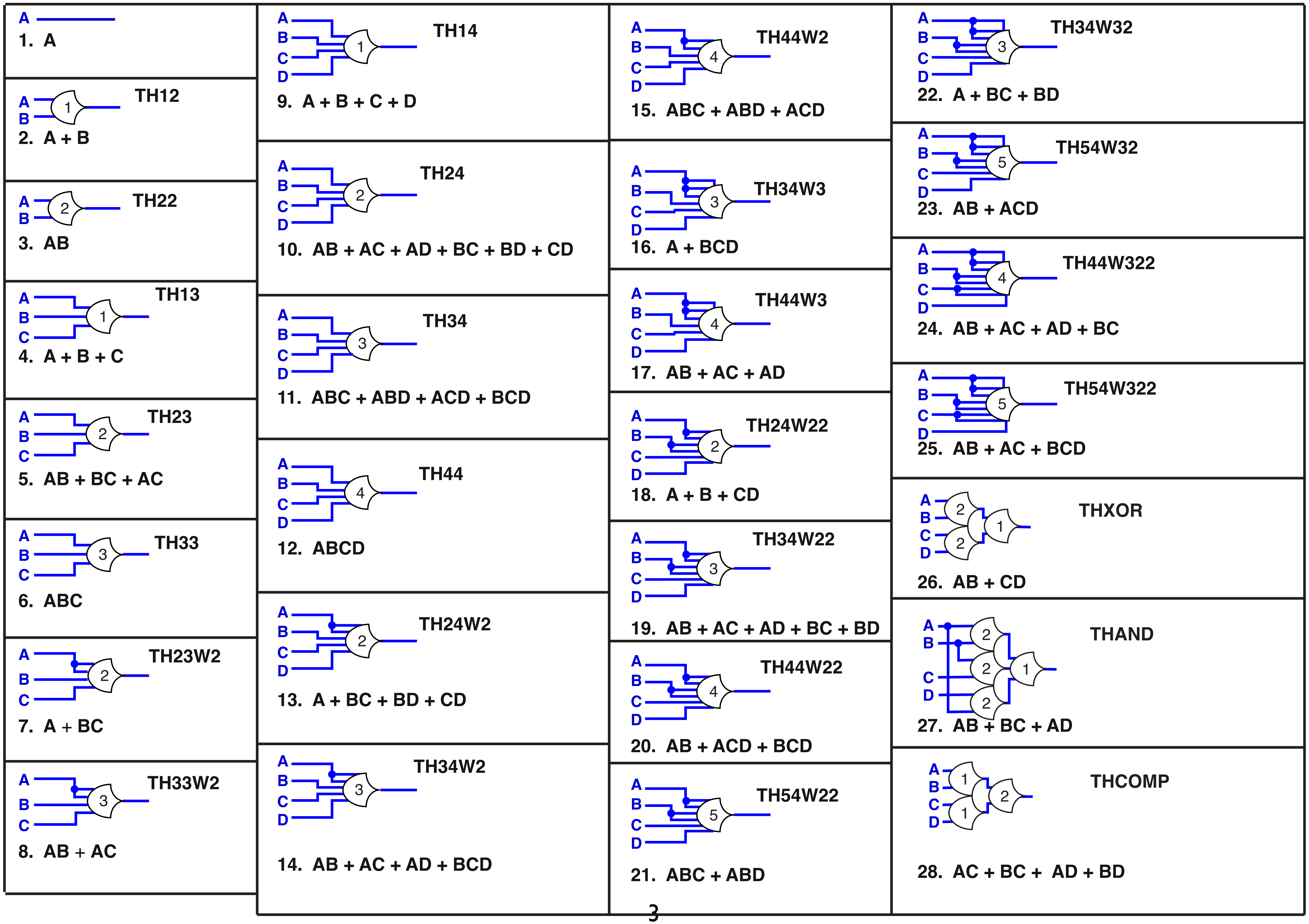


fanoutfanin1.v



ring1.v

NCL Dual Threshold Logic Functions



1. A

TH12
2. A + B

TH22
3. AB

TH13
4. A + B + C

TH23
5. AB + BC + AC

TH33
6. ABC

TH23W2
7. A + BC

TH33W2
8. AB + AC

TH14
9. A + B + C + D

TH24
10. AB + AC + AD + BC + BD + CD

TH34
11. ABC + ABD + ACD + BCD

TH44
12. ABCD

TH24W2
13. A + BC + BD + CD

TH34W2
14. AB + AC + AD + BCD

TH44W2
15. ABC + ABD + ACD

TH34W3
16. A + BCD

TH44W3
17. AB + AC + AD

TH24W22
18. A + B + CD

TH34W22
19. AB + AC + AD + BC + BD

TH44W22
20. AB + ACD + BCD

TH54W22
21. ABC + ABD

TH34W32
22. A + BC + BD

TH54W32
23. AB + ACD

TH44W322
24. AB + AC + AD + BC

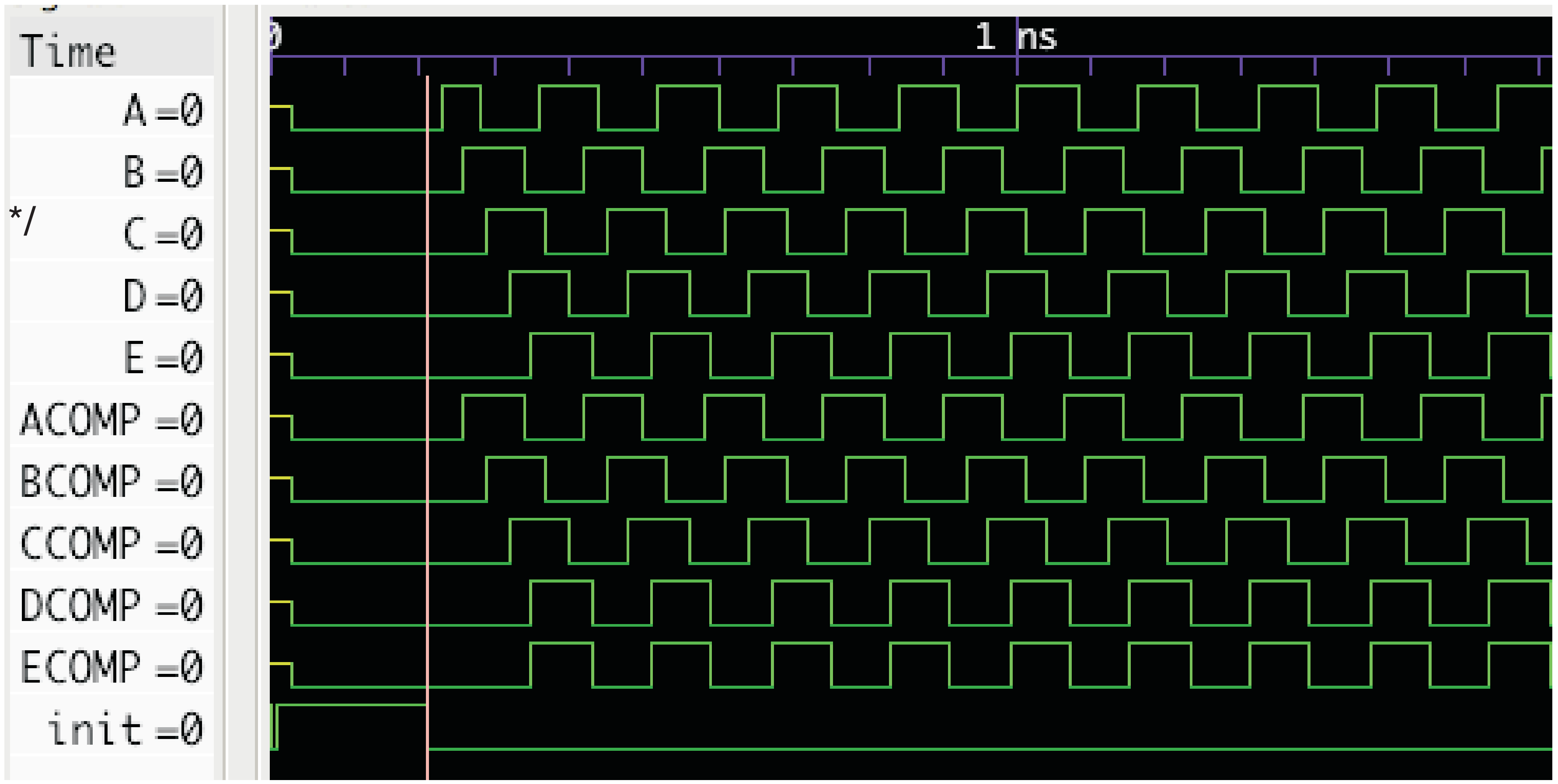
TH54W322
25. AB + AC + BCD

THXOR
26. AB + CD

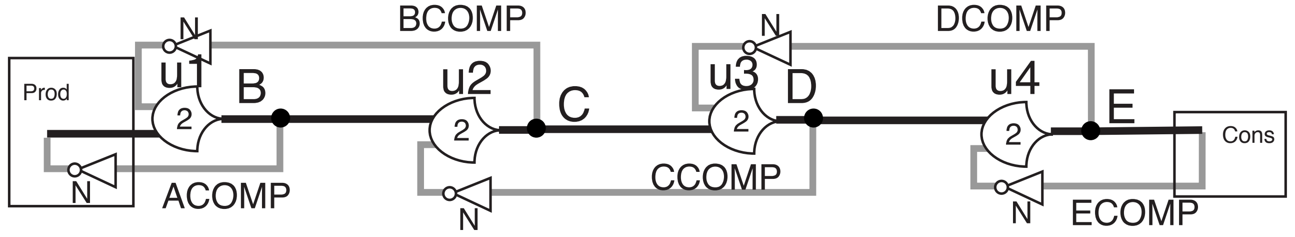
THAND
27. AB + BC + AD

THCOMP
28. AC + BC + AD + BD

pipeline1.v

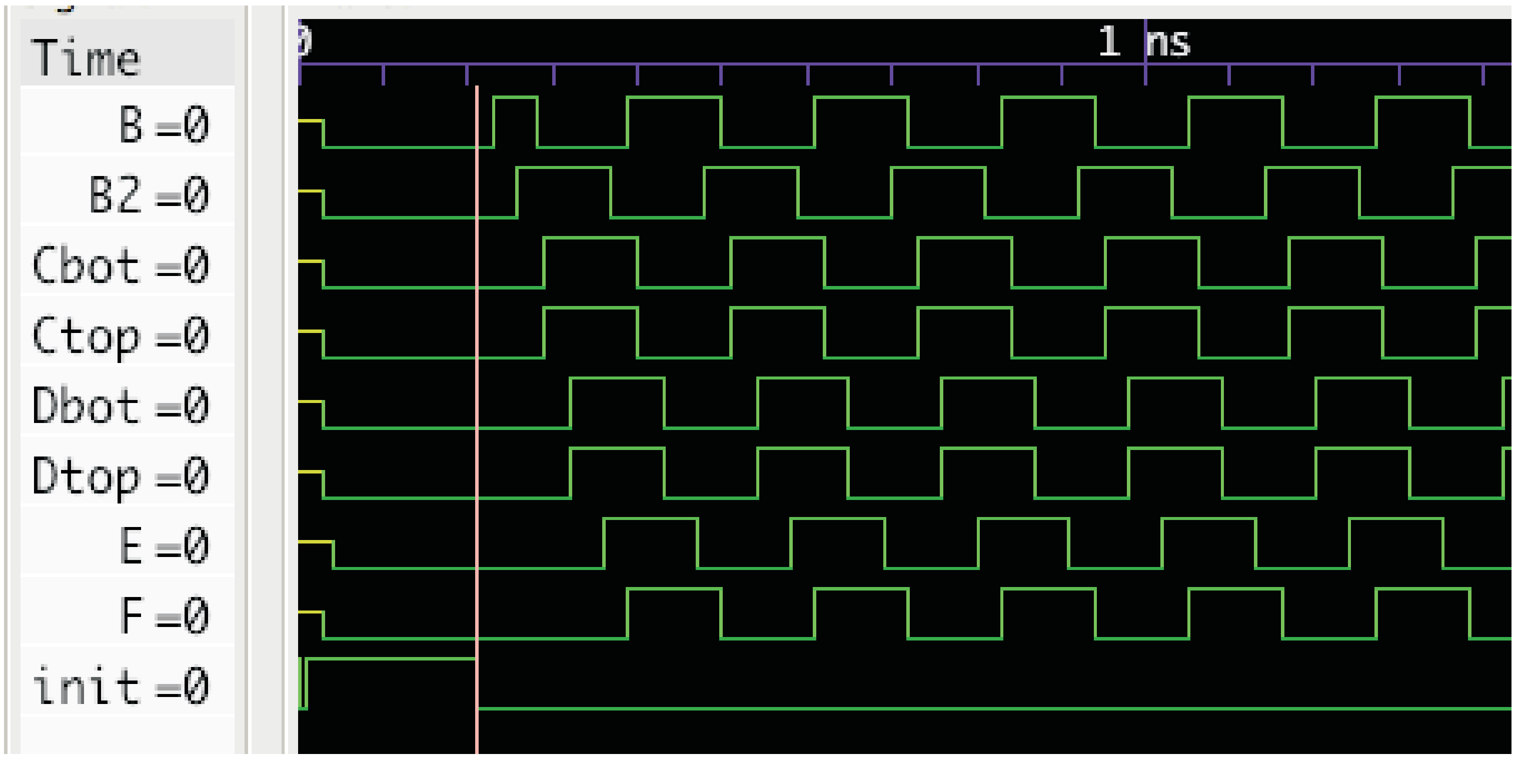


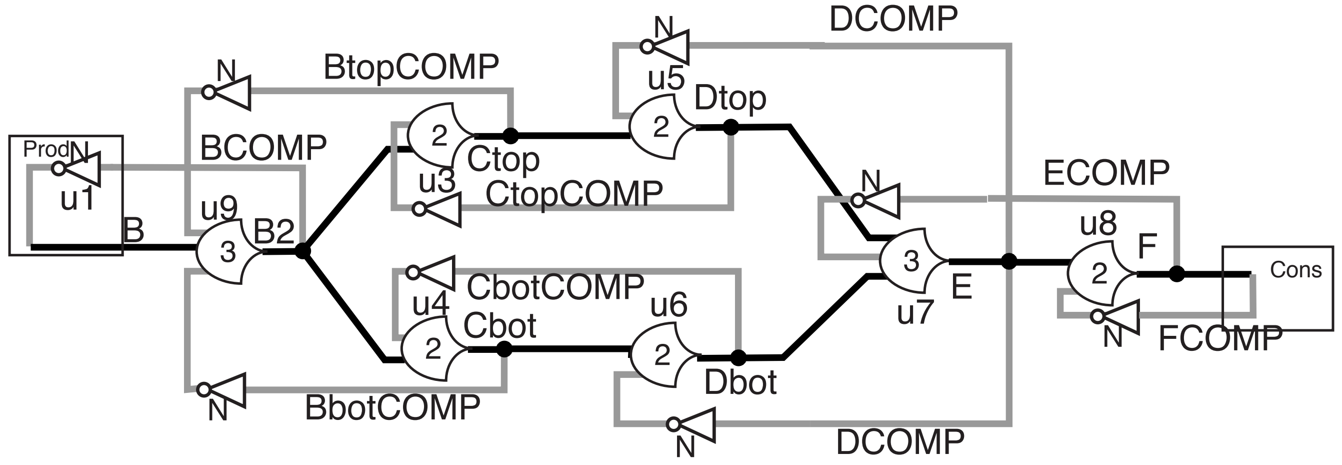
```
module pipeline1;
reg init = 0;
  /* Make an init that pulses once. */
 initial begin
     # 0 init = 1;
     # 20 init = 0;
     # 1000 $stop;
  end
initial
 begin
    $dumpfile("pipeline1.vcd");
    $dumpvars(0,pipeline1);
 end

///// Testbench
/////////////////////////////
///// Circuit Under Test

// 4 stage pipeline
wire  A, B, C, D, E;
wire ACOMP, BCOMP, CCOMP, DCOMP, ECOMP;
THnotN  A0(A, ACOMP, init); // auto produce A input
Pipecomponent u1(B, BCOMP, A, ACOMP, init);
Pipecomponent u2(C, CCOMP, B, BCOMP, init);
Pipecomponent u3(D, DCOMP, C, CCOMP, init);
Pipecomponent u4(E, ECOMP, D, DCOMP, init);
assign ECOMP = E;  // auto consume E output
endmodule

module Pipecomponent(output Z, input ZCOMP, input A, output ACOMP, input init);
wire enable;
THnotN  u0(enable, ZCOMP, init);
TH22  u1(Z, A, enable);
assign ACOMP = Z;
endmodule
```



fanoutfanin1.v

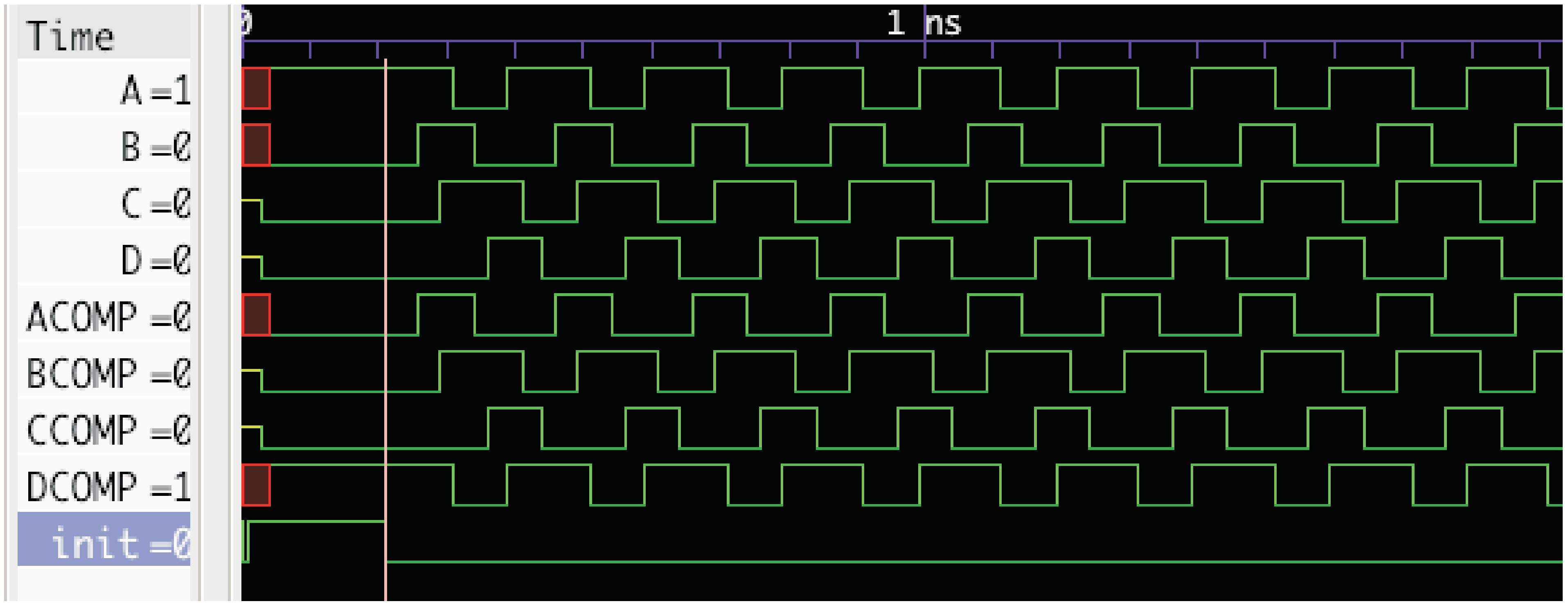
```
module Pipefanin(output Z, input ZCOMP, input
A, input B, output ACOMP, input init);
wire enable;
THnotN  u0(enable, ZCOMP, init);
TH33  u1(Z, A, B, enable);
assign ACOMP = Z;
endmodule

module Pipecomponent(output Z, input ZCOMP,
input A, output ACOMP, input init);
wire enable;
THnotN  u0(enable, ZCOMP, init);
TH22  u1(Z, A, enable);
assign ACOMP = Z;
endmodule
```

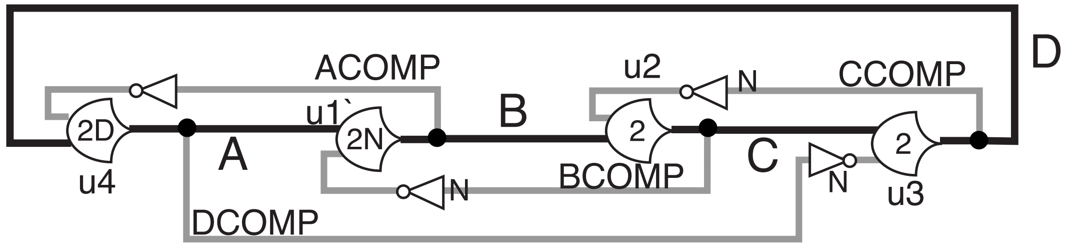




```
/////////////////////////////
///// Circuit Under Test

// fanout fanin pipeline
THnotN  A0(B, BCOMP, init); // auto produce B input
TH22 u10 (B2COMP, BtopCOMP, BbotCOMP);
Pipecomponent u9 (B2, B2COMP, B, BCOMP, init); // fanout
Pipecomponent u3 (Ctop, CtopCOMP, B2, BtopCOMP, init);
Pipecomponent u4 (Cbot, CbotCOMP, B2, BbotCOMP, init);
Pipecomponent u5 (Dtop, DCOMP, Ctop, CtopCOMP, init);
Pipecomponent u6 (Dbot, DCOMP, Cbot, CbotCOMP, init);
Pipefanin u7 (E, ECOMP, Dtop, Dbot, DCOMP, init);  // fanin
Pipecomponent u8 (F, FCOMP, E, ECOMP, init);
assign FCOMP = F;  // auto consume F output
endmodule
```

fanoutfanin1.v

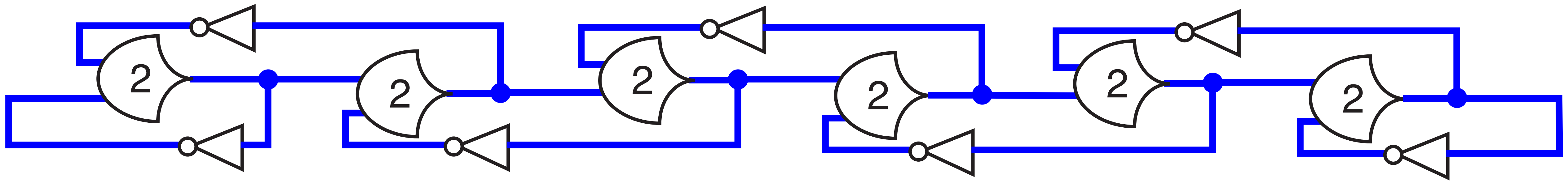




```
/////////////////////////////
///// Circuit Under Test

// one rail 4 stage ring
PipecomponentN u1(B, BCOMP, A, ACOMP, init);
Pipecomponent u2(C, CCOMP, B, BCOMP, init);
Pipecomponent u3(D, DCOMP, C, CCOMP, init);
PipecomponentD u4(A, ACOMP, D, DCOMP, init);
endmodule

module Pipecomponent(output Z, input ZCOMP,
input A, output ACOMP, input init);
wire enable;
THnotN  u0(enable, ZCOMP, init);
TH22  u1(Z, A, enable);
assign ACOMP = Z;
endmodule
```

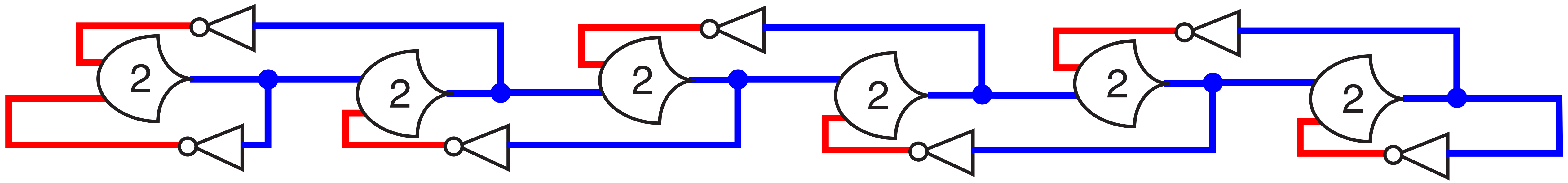
```
module PipecomponentN(output  Z, input ZCOMP,
input A, output ACOMP, input init);
wire enable;
THnotN  u0(enable, ZCOMP, init);
TH22N  u1(Z, A, enable, init);
assign ACOMP = Z;
endmodule

module PipecomponentD(output  Z, input ZCOMP,
input A, output ACOMP, input init);
wire enable;
THnot  u0(enable, ZCOMP);
TH22D  u1(Z, A, enable, init);
assign ACOMP = Z;
endmodule
```

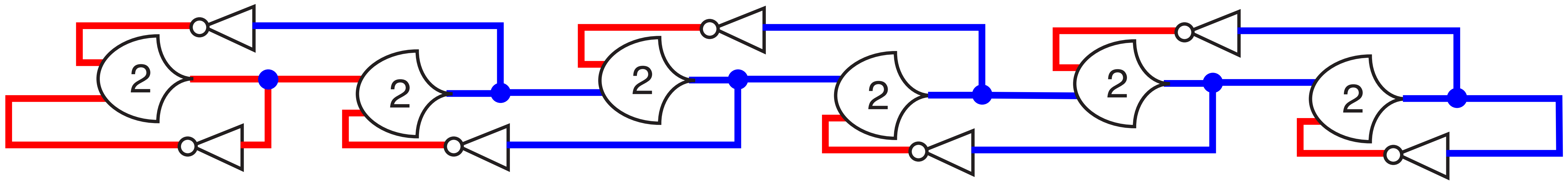
Init forces output of inverters to low

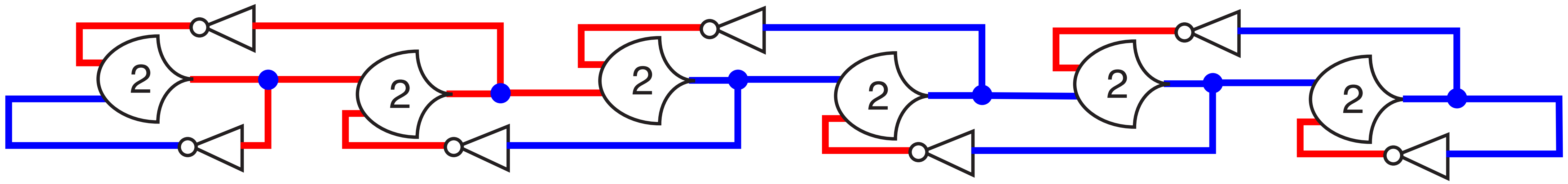


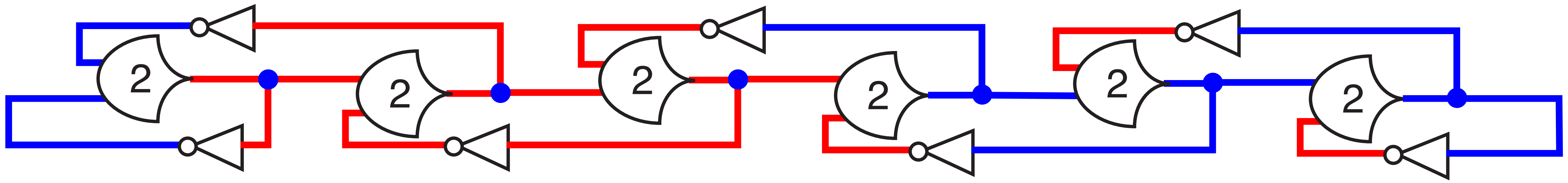
The low value propagates through the structure to the input of the inverters setting the entire structure to low

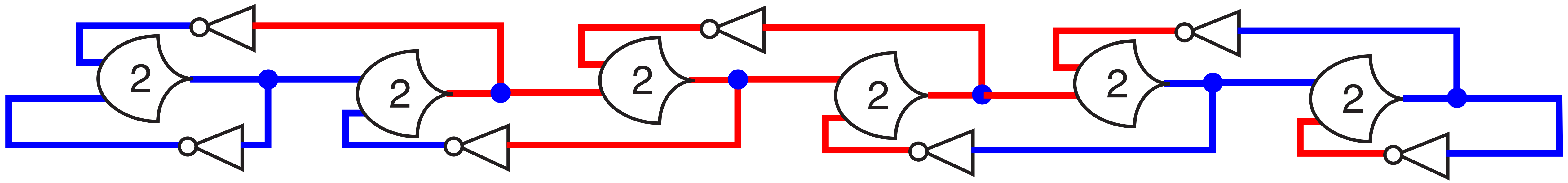
Init is released
the inverters transition to high
and a transition to high wavefront begins flowing




2
2
2
2
2
2

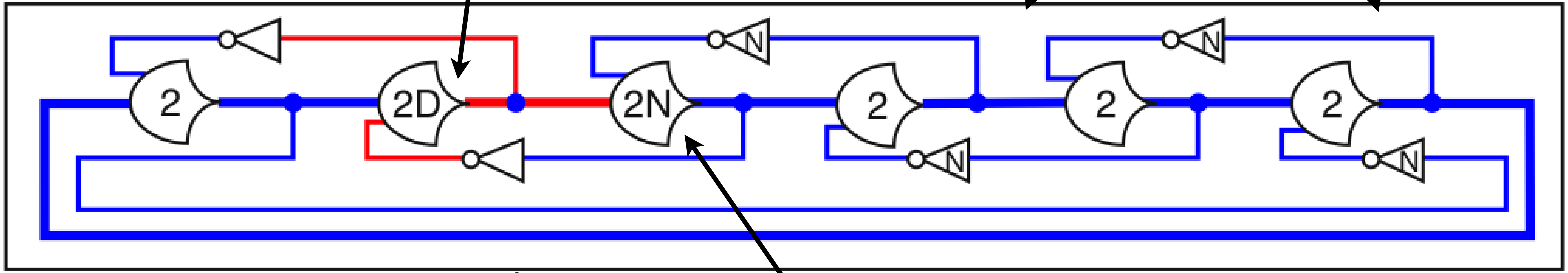

2
2
2
2
2
2


2
2
2
2
2
2


2
2
2
2
2
2

All inverters except the two associated with the flow path initializations are initialized to low to allow the flow of low through the entire flow path

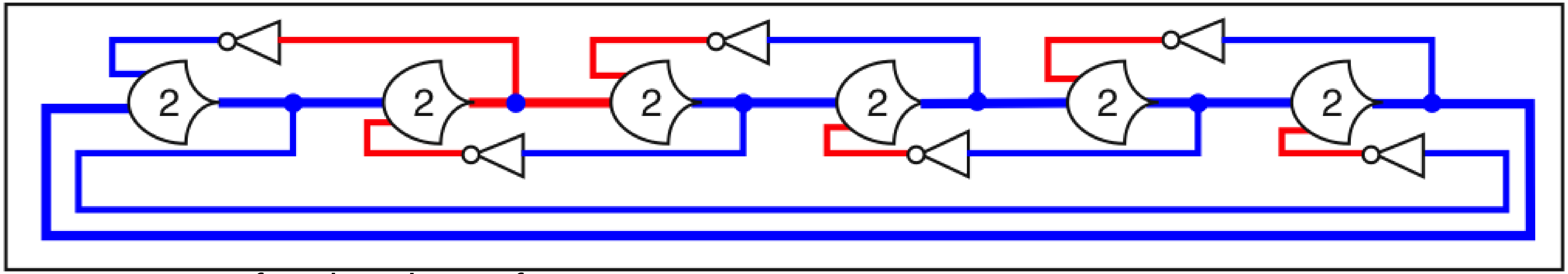
A ring does not have any auto produce end caps so a data wavefront must be explicitly initialized in the flow path of the ring.



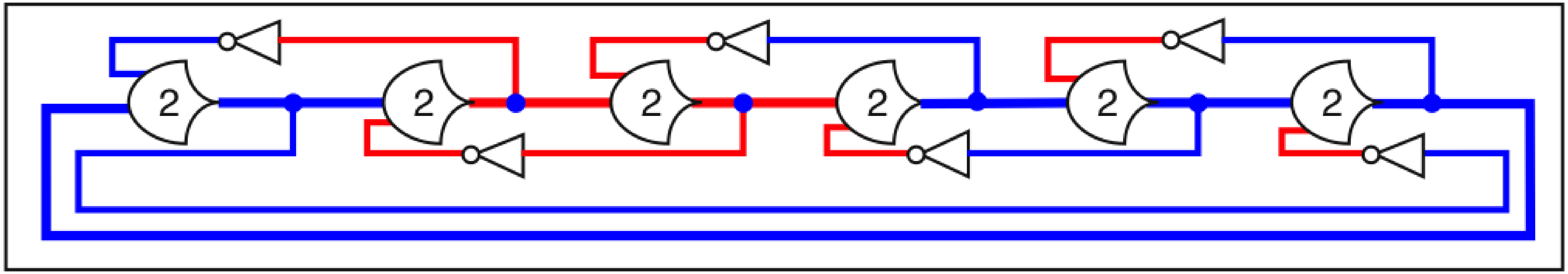
ring state just prior to release of init

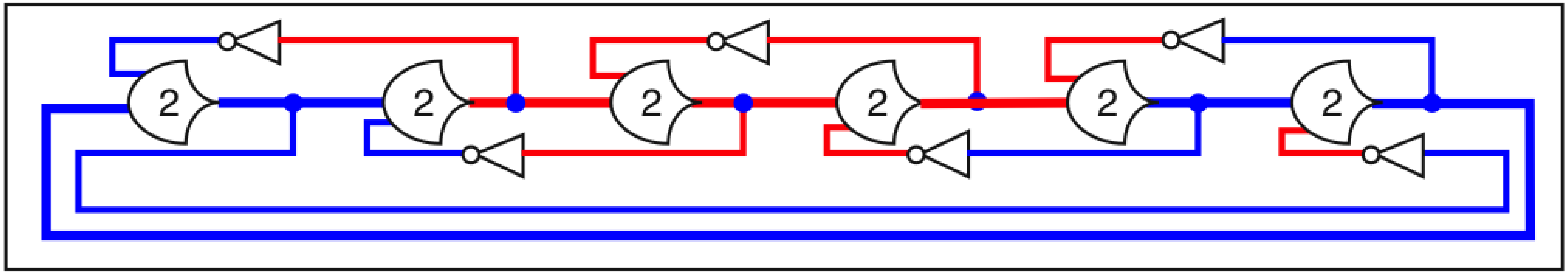
The flow path must be initialized to low immediately following the high initialization to isolate the high value during initialization and to initiate the flow of low through the rest of the flow path

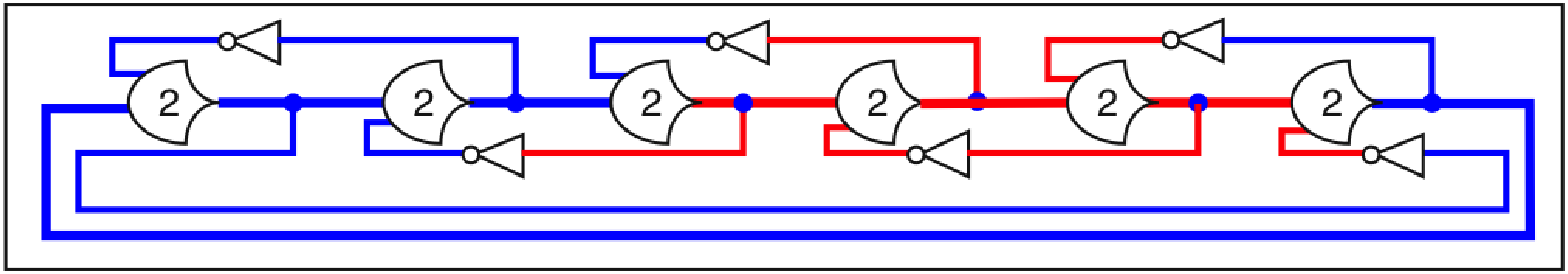
Init is released
the initialized inverters transition to high
and the transition to high wavefront
begins flowing

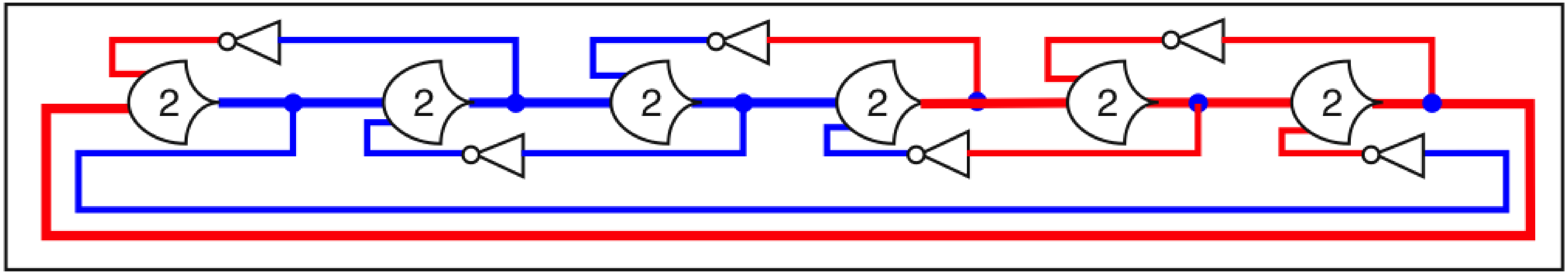


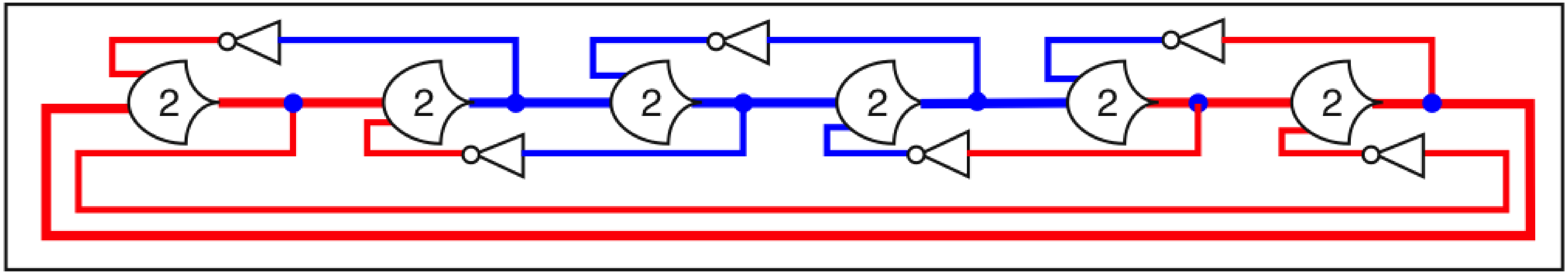
ring state just after the release of init

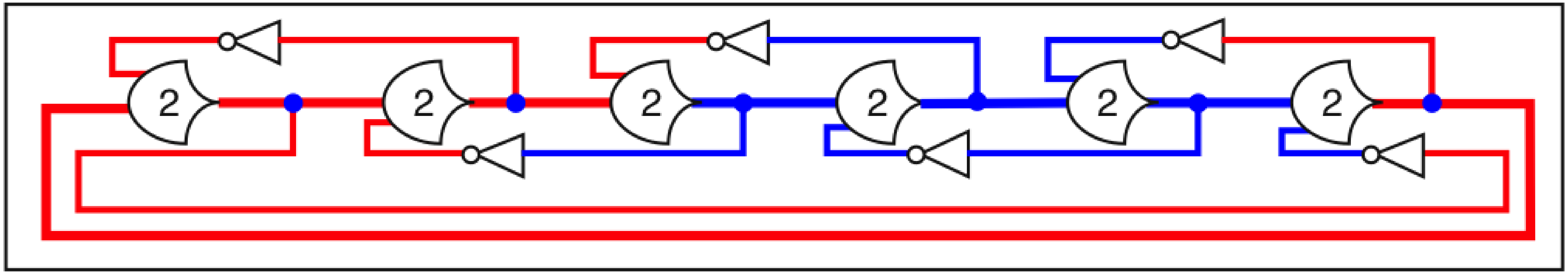

2
2
2
2
2
2


2
2
2
2
2
2


2
2
2
2
2
2


2
2
2
2
2
2


2
2
2
2
2
2


2
2
2
2
2
2