Symbolic Specification

The Multi-rail Token

token {3:0} A;

Specifies 4 OR related { } rails named 0, 1, 2, and 3

Only one rail will transition to data per oscillation

The rails are referenced by name as A/0, A/1, A/2, A/3

The Flow component

(source -> destination){**flow**; body; **close**}

Specifies a shared completeness behavior linking flow paths A and B

```
(A -> B){
      flow A-> B;
      A -> B;
      close A <- B/#;
}
```

default specification →

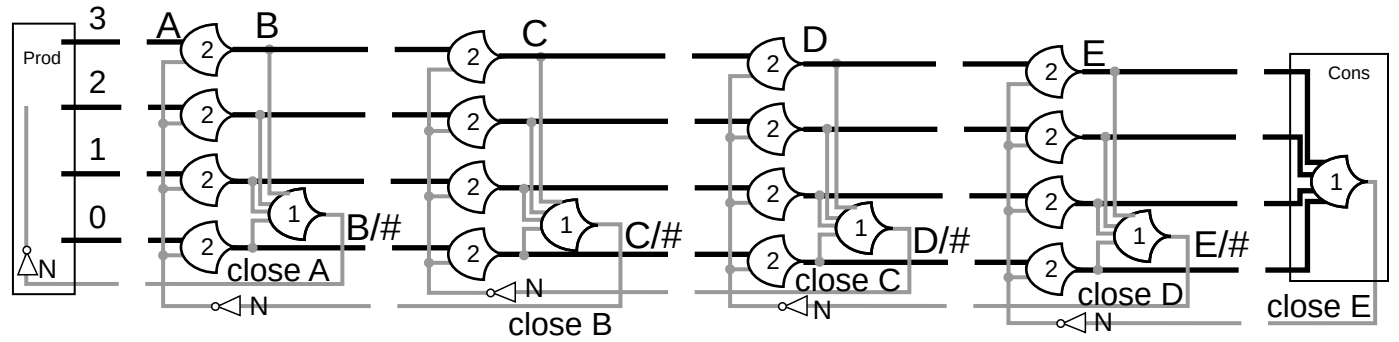
(A -> B);

(-> B); source

(A ->); sink

Four-Rail Four-Component Pipeline

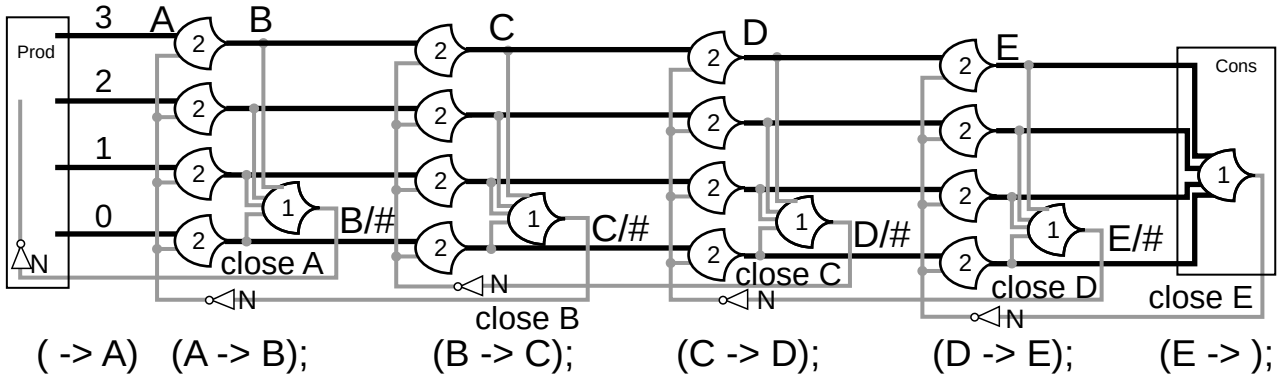
Individual flow components



symbolic specification

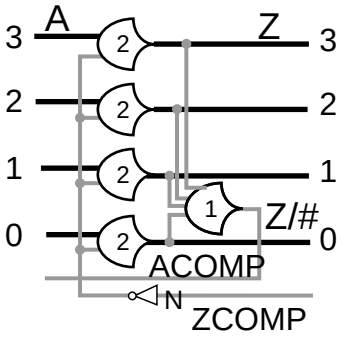
token {3:0} A, B, C, D, E;
(-> A);
(A -> B);
(B -> C);
(C -> D);
(D -> E);
(E ->);

Pipeline of linked components



pipeline4.v

```
token {3:0} A, B, C, D, E;
( -> A);  // A0
(A -> B);  // u1
(B -> C);  // u2
(C -> D);  // u3
(D -> E);  // u4
(E -> );     // u5
```



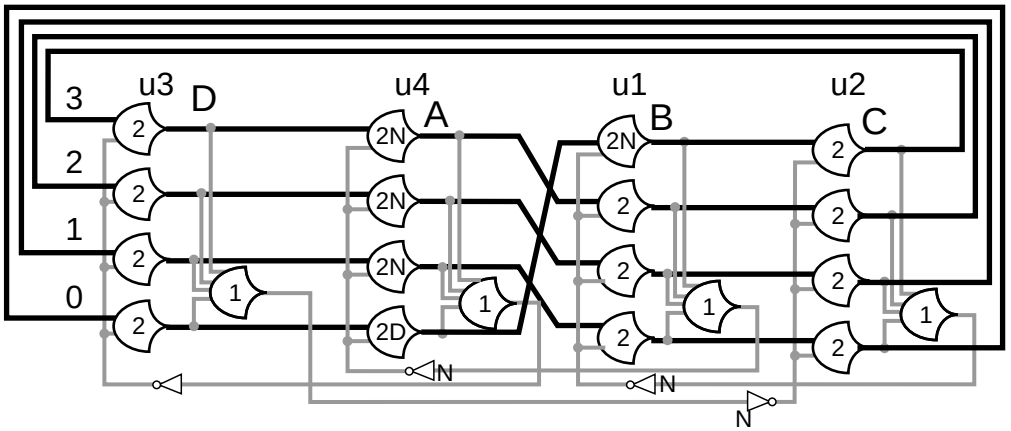

```
////////////////////////////
///// Circuit Under Test
wire [3:0] A, B, C, D, E;
wire ACOMP, BCOMP, CCOMP, DCOMP, ECOMP;

// 4 rail 4 oscillation pipeline
THnotN  A0(A[0], ACOMP, init); // auto produce A input
assign A[1] = 0;  // auto produce constant rails
assign A[2] = 0;  // auto produce constant rails
assign A[3] = 0;  // auto produce constant rails
Pipecomponent u1(B[3:0], BCOMP, A[3:0], ACOMP, init);
Pipecomponent u2(C[3:0], CCOMP, B[3:0], BCOMP, init);
Pipecomponent u3(D[3:0], DCOMP, C[3:0], CCOMP, init);
Pipecomponent u4(E[3:0], ECOMP, D[3:0], DCOMP, init);
TH14 u5 (ECOMP, E[0], E[1], E[2], E[3]);  // auto consume E output
endmodule

module Pipecomponent(output [3:0] Z, input ZCOMP, input [3:0] A,
output ACOMP, input init);
wire enable;
THnotN  u0(enable, ZCOMP, init);
TH22  c1(Z[0], A[0], enable);
TH22  c2(Z[1], A[1], enable);
TH22  c3(Z[2], A[2], enable);
TH22  c4(Z[3], A[3], enable);
TH14 c5 (ACOMP, Z[0], Z[1], Z[2], Z[3]);
endmodule
```

ring4.v

```
ring4{
token {3:0} A, B, C, D: A/0(D);
(D -> A);  //u4
(A -> B){  //u1
   flow A -> B;
      A/3 -> B/2;
      A/2 -> B/1;
      A/1 -> B/0;
      A/0 -> B/3;
   close A <- B/# };
(C -> D);  //u3
(B -> C);  //u2
}
```




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

wire [3:0] A, B, C, D;
wire ACOMP, BCOMP, CCOMP, DCOMP;

// 4 rail 4 oscillation 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 [3:0] Z, input ZCOMP,
input [3:0] A, output ACOMP, input init);
wire enable;
THnotN  u0(enable, ZCOMP, init);
TH22  u1(Z[0], A[0], enable);
TH22  u2(Z[1], A[1], enable);
TH22  u3(Z[2], A[2], enable);
TH22  u4(Z[3], A[3], enable);
TH14 u5 (ACOMP, Z[0], Z[1], Z[2], Z[3]);
endmodule
```

```
module PipecomponentN(output [3:0] Z, input ZCOMP,
input [3:0] A, output ACOMP, input init);
wire enable;
THnotN  u0(enable, ZCOMP, init);
TH22  u1(Z[0], A[1], enable);  // rotate rails
TH22  u2(Z[1], A[2], enable);
TH22  u3(Z[2], A[3], enable);
TH22N  u4(Z[3], A[0], enable, init); // block inited data
TH14 u5 (ACOMP, Z[0], Z[1], Z[2], Z[3]);
endmodule

module PipecomponentD(output [3:0] Z, input ZCOMP,
input [3:0] A, output ACOMP, input init);
wire enable;
THnot  u0(enable, ZCOMP);
TH22D  u1(Z[0], A[0], enable, init);
TH22N  u2(Z[1], A[1], enable, init);
TH22N  u3(Z[2], A[2], enable, init);
TH22N  u4(Z[3], A[3], enable, init);
TH14 u5 (ACOMP, Z[0], Z[1], Z[2], Z[3]);
endmodule
```

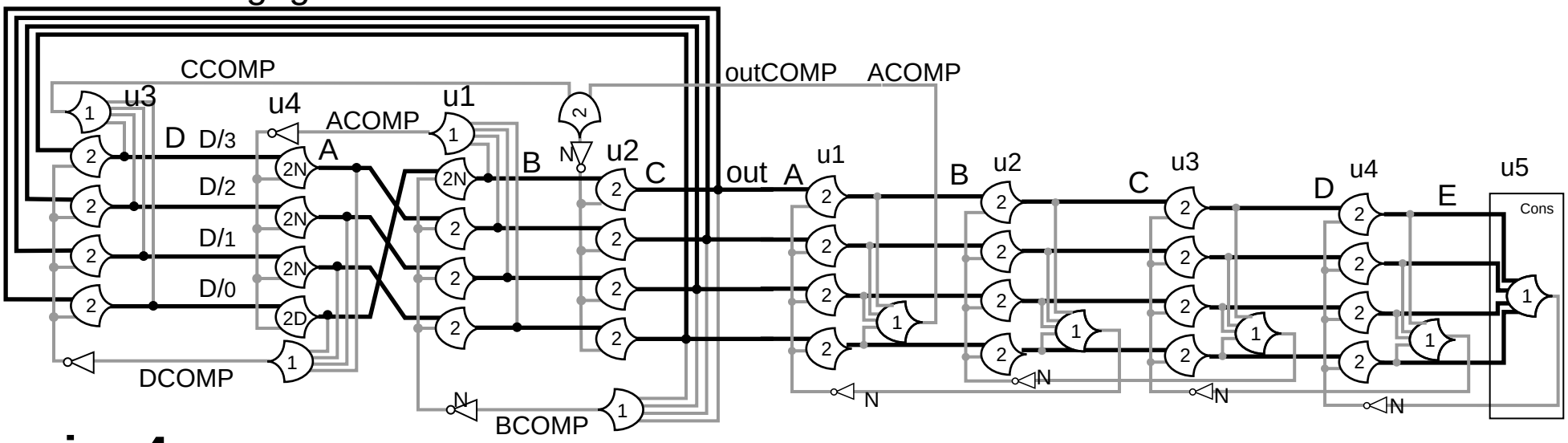
ring4pipe4.v

```
ring4gen( -> out){
token {3:0} A, B, C, D: A/0(D);
(D -> A);  //u4
(A -> B){  //u1
   flow A -> B;
      A/3 -> B/2; A/2 -> B/1; A/1 -> B/0; A/0 -> B/3;
   close A <- B/# };
(B -> C);  //u2
(C -> D, out)  // u3
   close C <- [D/#, out/#] ;  //u5
}
```

```
token {3:0} A, B, C, D;

ring4gen( -> A);  // TB1
(A -> B);  // u1
(B -> C);  // u2
(C -> D);  // u3
(D -> E);  // u4
(E -> );
```

ring4gen.v TB1



ring4gen.v

```
module ring4gen (output wire [3:0] out, input outCOMP, input init);
wire [3:0] A, B, C, D;
wire ACOMP, BCOMP, CCOMP, DCOMP, CCOMP2;
// 4 stage ring
PipecomponentN u1(B, BCOMP, A, ACOMP, init);
Pipecomponent u2(C, CCOMP2, B, BCOMP, init);
Pipecomponentshift u3(D, DCOMP, C, CCOMP, init);
PipecomponentD u4(A, ACOMP, D, DCOMP, init);
TH22 u5(CCOMP2, CCOMP, outCOMP);
assign out = C;
endmodule
```

ring4pipe4.v

```
// 4 rail ring producing 4 rail output
ring4gen TB1 (A[3:0], ACOMP, init);

// 4 rail rail pipeline receiving A flow from ring
PipecomponentP u1(B[3:0], BCOMP, A[3:0], ACOMP, init);
PipecomponentP u2(C[3:0], CCOMP, B[3:0], BCOMP, init);
PipecomponentP u3(D[3:0], DCOMP, C[3:0], CCOMP, init);
PipecomponentP u4(E[3:0], ECOMP, D[3:0], DCOMP, init);

TH14 u5 (ECOMP, E[0], E[1], E[2], E[3]);  // auto consume
endmodule
```