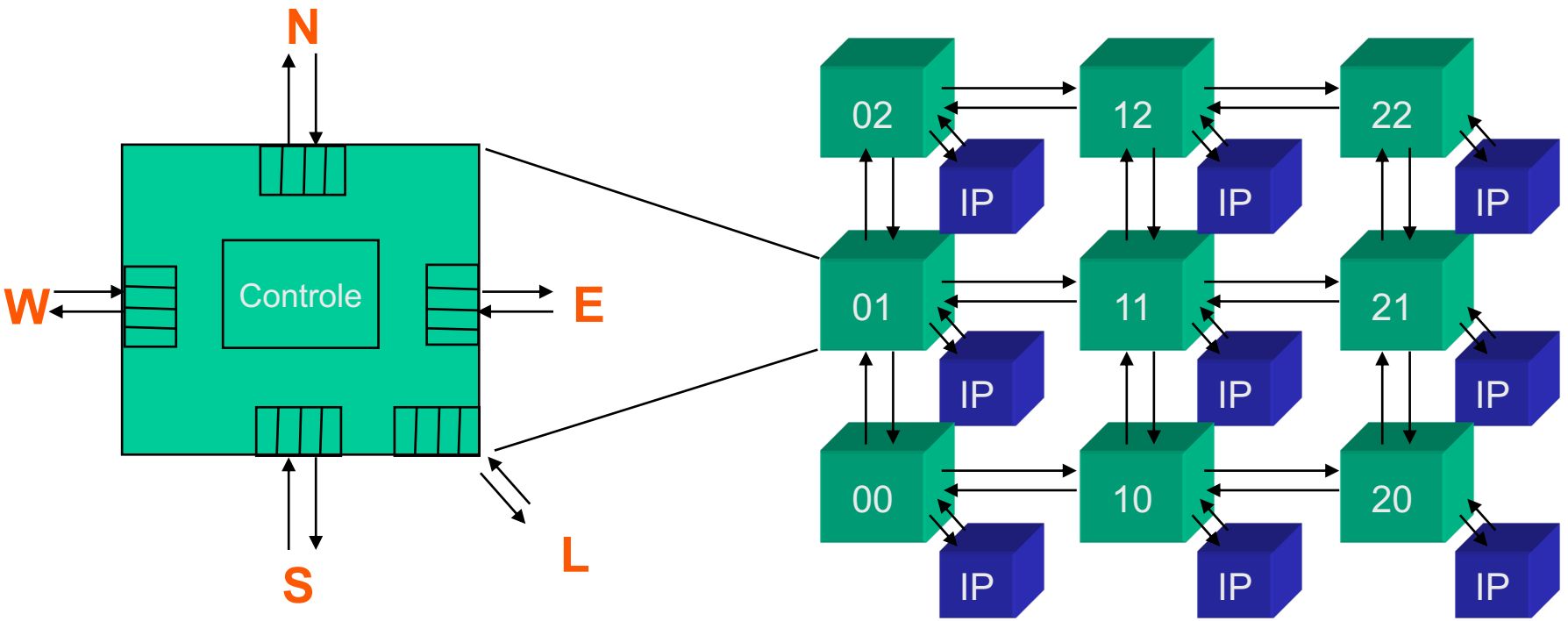
HERMES

NoC Hermes

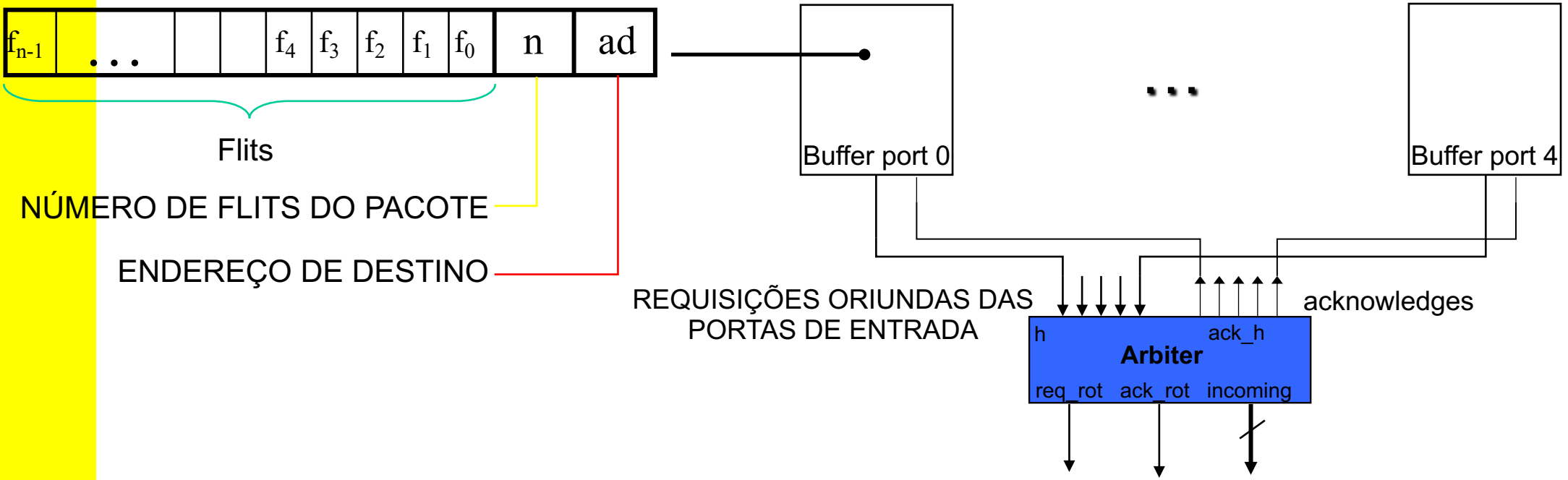
Características

- Todo IP é conectado a um roteador (rede direta)
- Endereçamento XY
- Até 5 portas unidirecionais por roteador (N / S / W / E + Local)
- Bufferização de entrada



Pacotes

- Características
  - Apenas o primeiro *flit é* tratado (endereço de destino)
  - Arbitragem rotativa das prioridades de acesso às portas de saída (*round robin*)



Árbitro e roteamento

- Características
  - Apenas o primeiro *flit é* tratado (endereço de destino)
  - Arbitragem rotativa das prioridades de acesso às portas de saída (*round robin*)

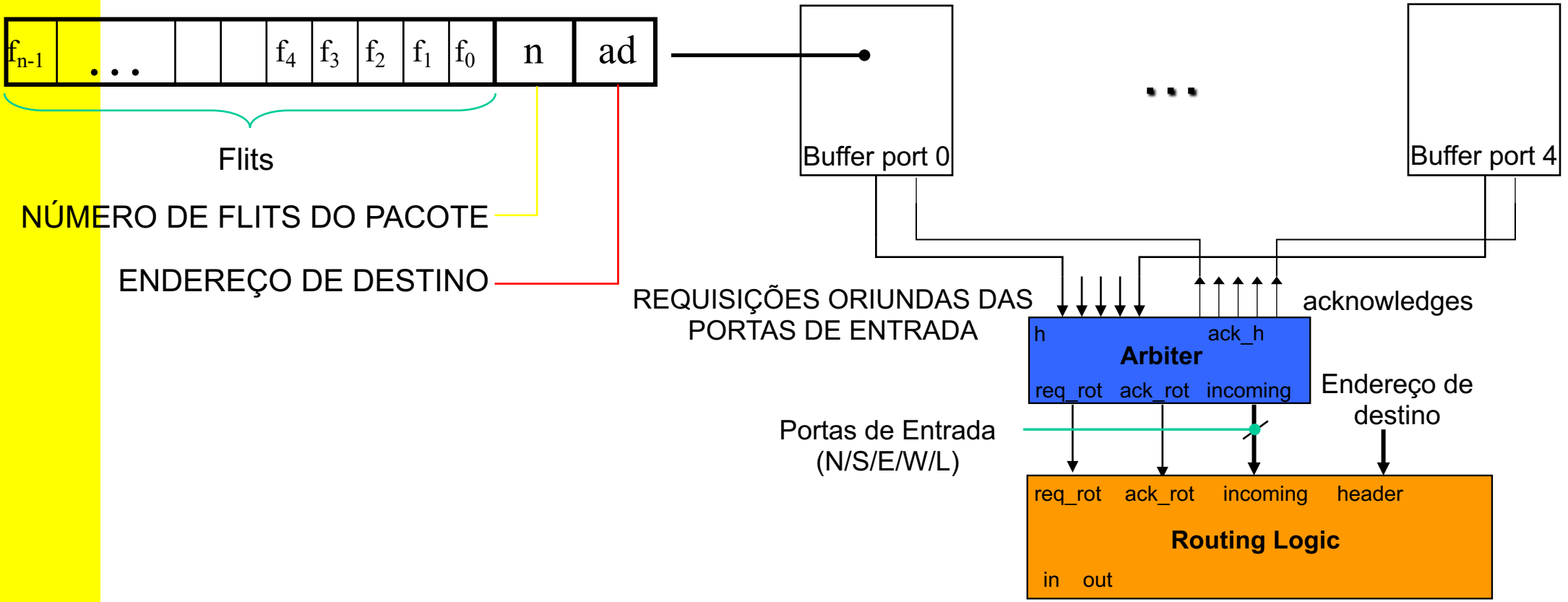


Tabela de roteamento

- Características
  - Apenas o primeiro *flit é* tratado (endereço de destino)
  - Arbitragem rotativa das prioridades de acesso às portas de saída (*round robin*)

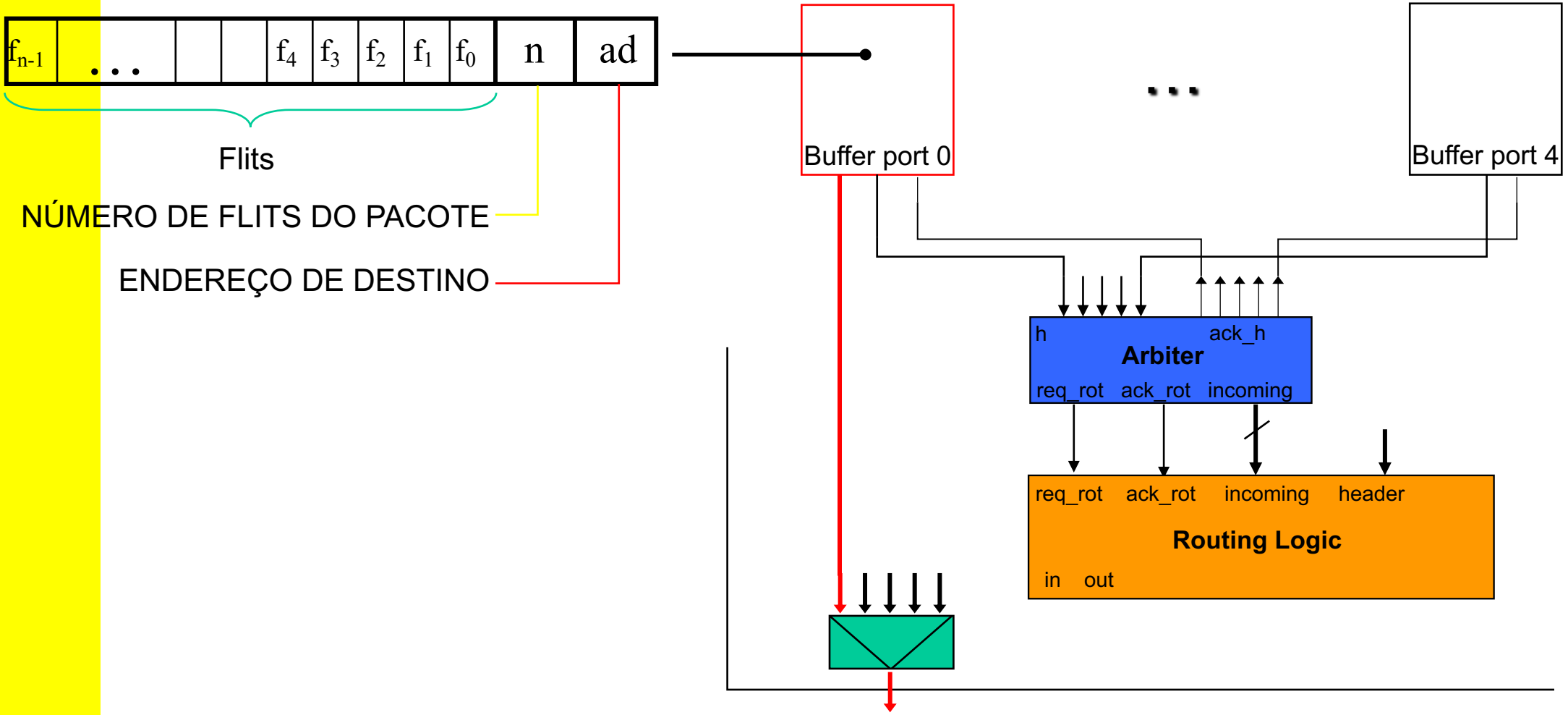
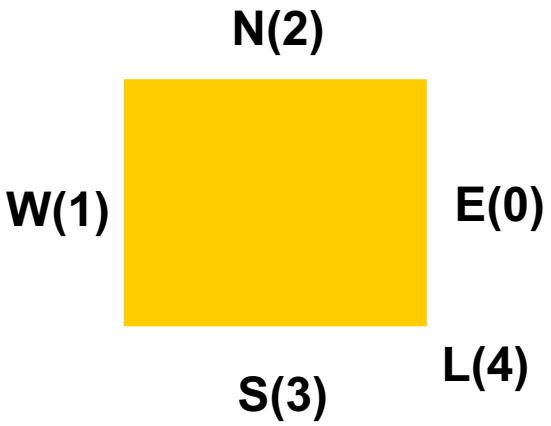


Tabela de roteamento

- Tabela de conexão
  - Quando uma conexão é estabelecida o caminho entre a porta de entrada e saída é armazenado
  - Tres vetores são utilizados: IN, OUT, FREE

	0 – E	1 – W	2 – N	3 – S	4 – L
Free	1	1	1	1	1
In					
Out					



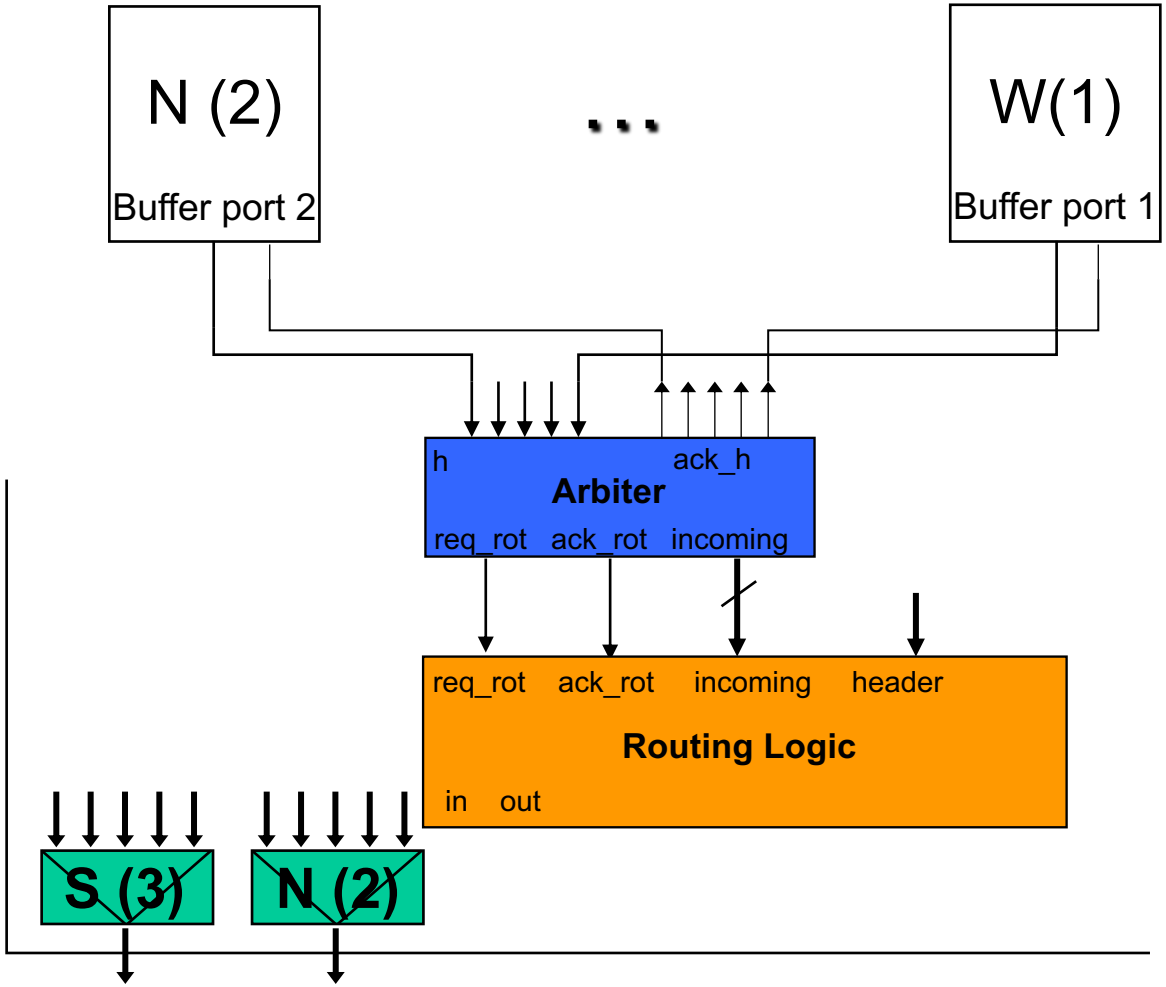
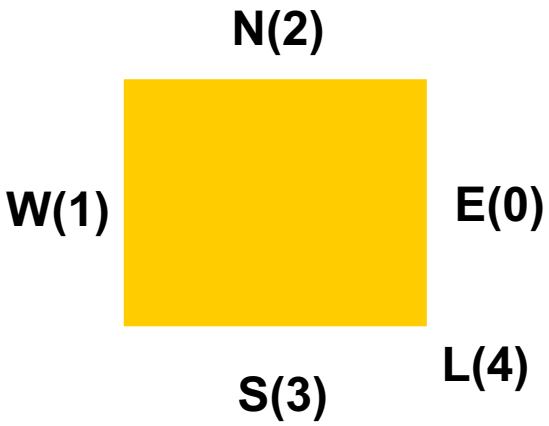


Tabela de roteamento

- Tabela de conexão
  - Quando uma conexão é estabelecida o caminho entre a porta de entrada e saída é armazenado
  - Tres vetores são utilizados: IN, OUT, FREE

	0 – E	1 – W	2 – N	3 – S	4 – L
Free	1	1	1	1	1
In					
Out					



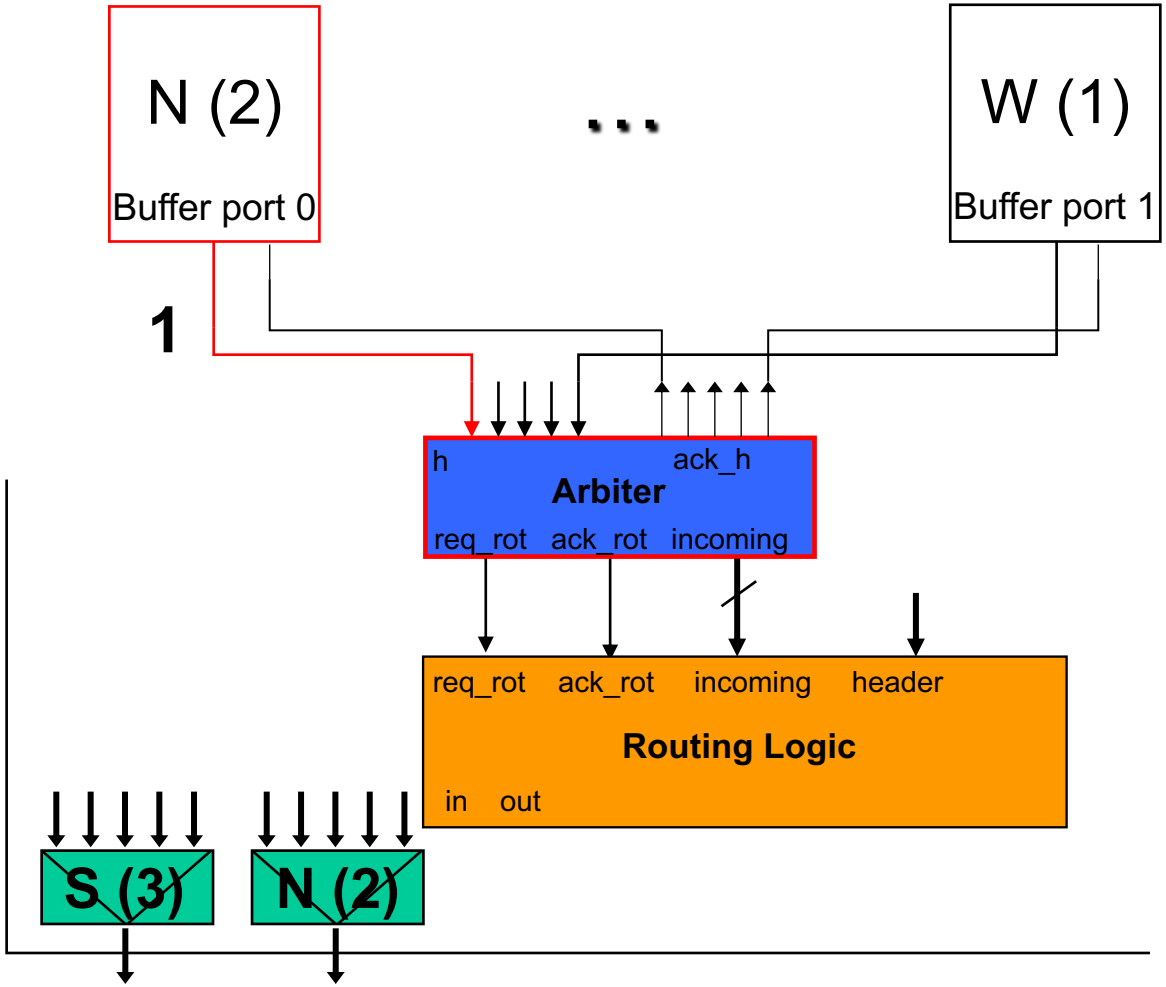
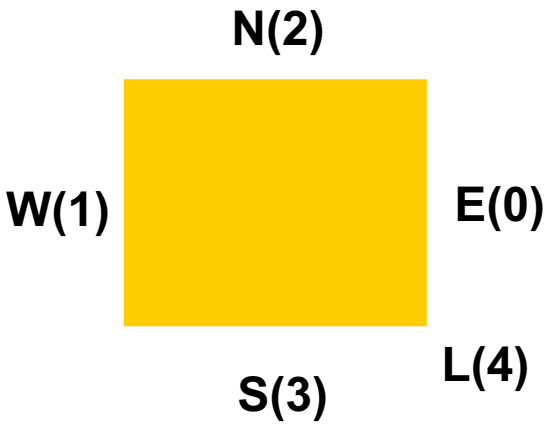


Tabela de roteamento

- Tabela de conexão
  - Quando uma conexão é estabelecida o caminho entre a porta de entrada e saída é armazenado
  - Tres vetores são utilizados: IN, OUT, FREE

	0 – E	1 – W	2 – N	3 – S	4 – L
Free	1	1	1	1	1
In					
Out					



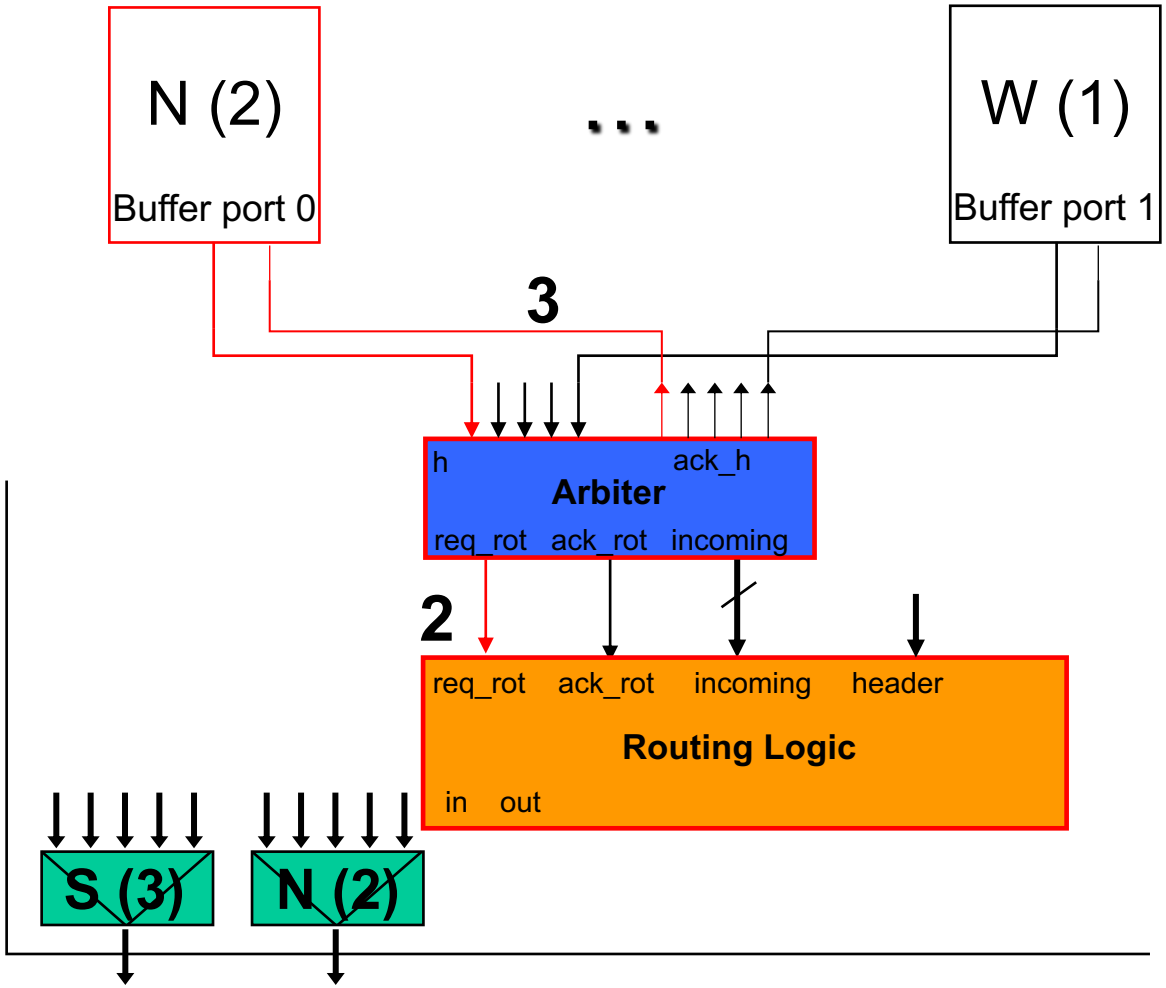
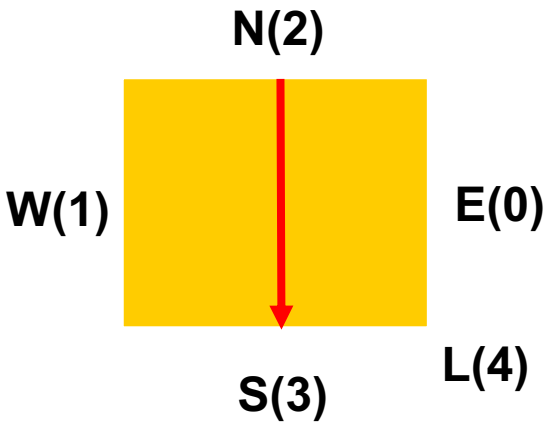


Tabela de roteamento

- Tabela de conexão
  - Quando uma conexão é estabelecida o caminho entre a porta de entrada e saída é armazenado
  - Tres vetores são utilizados: IN, OUT, FREE

	0 – E	1 – O	2 – N	3 – S	4 – L
Free	1	1	1	0	1
In			3		
Out				2	



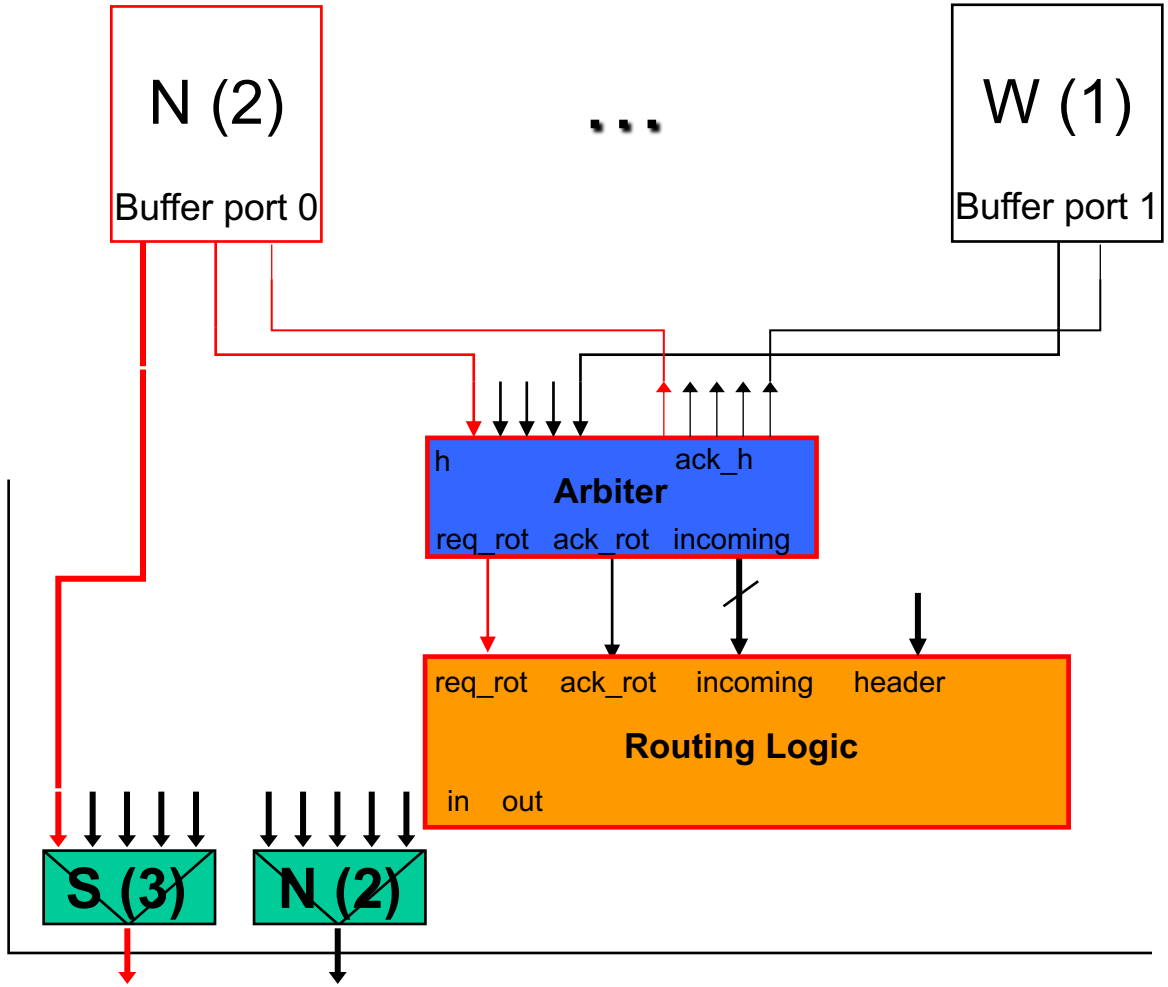
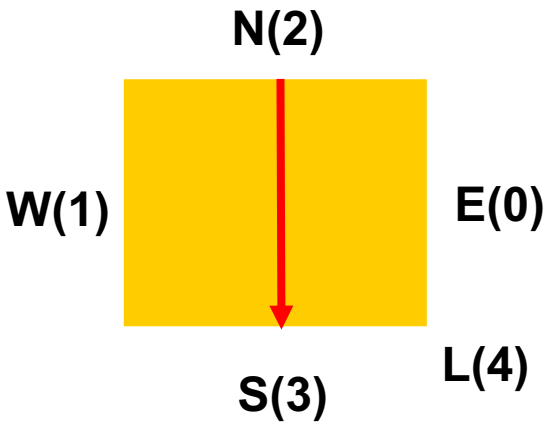


Tabela de roteamento

- Tabela de conexão
  - Quando uma conexão é estabelecida o caminho entre a porta de entrada e saída é armazenado
  - Tres vetores são utilizados: IN, OUT, FREE

	0 – E	1 – O	2 – N	3 – S	4 – L
Free	1	1	1	0	1
In			3		
Out				2	



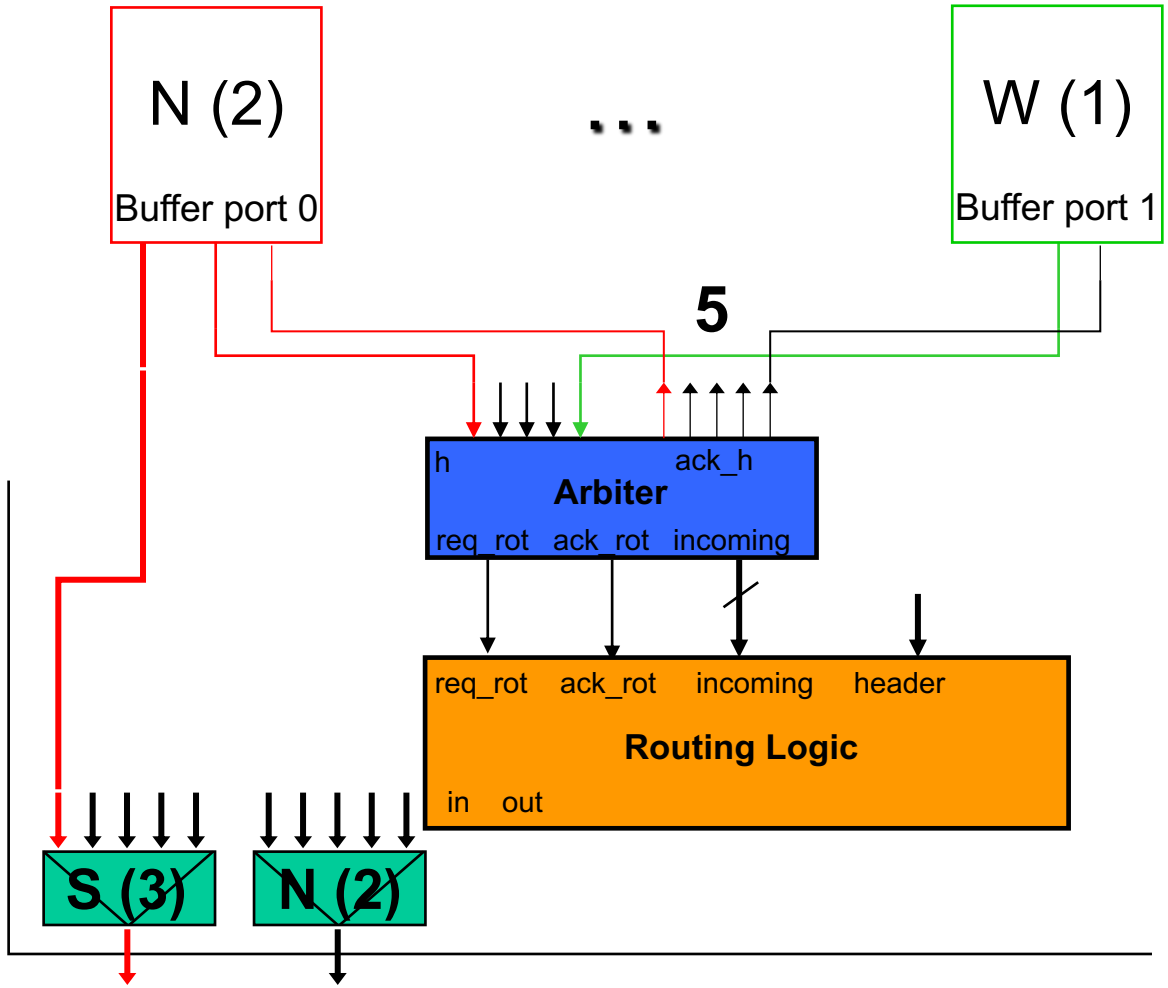
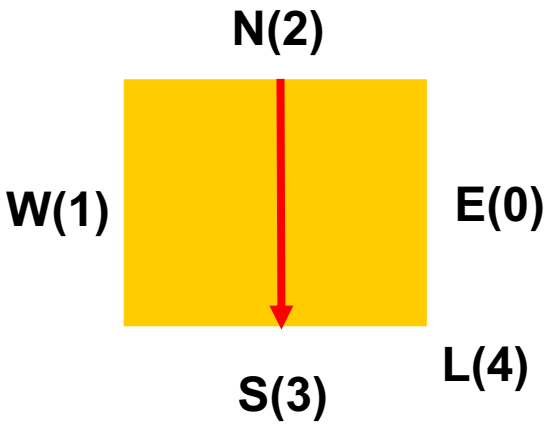


Tabela de roteamento

- Tabela de conexão
  - Quando uma conexão é estabelecida o caminho entre a porta de entrada e saída é armazenado
  - Três vetores são utilizados: IN, OUT, FREE

	0 – E	1 – O	2 – N	3 – S	4 – L
Free	1	1	1	0	1
In			3		
Out				2	



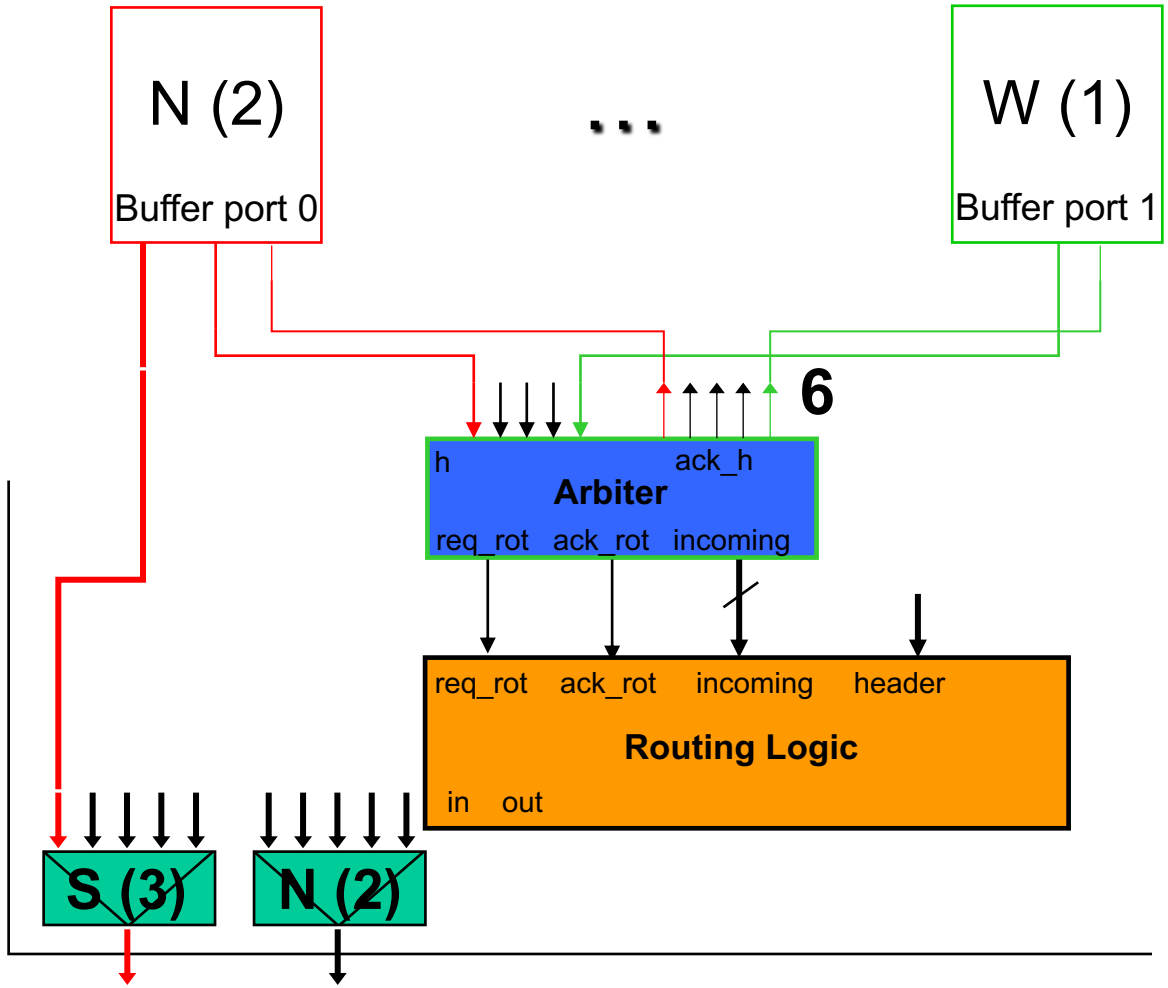
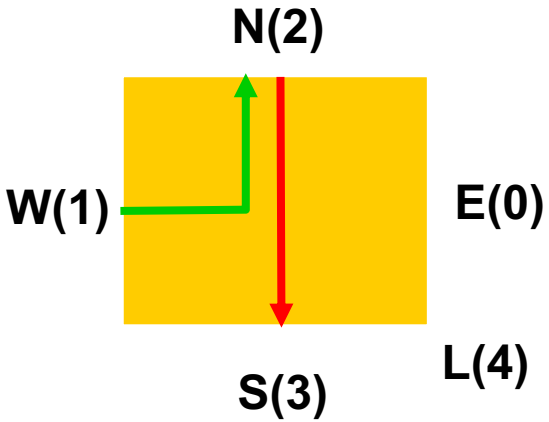
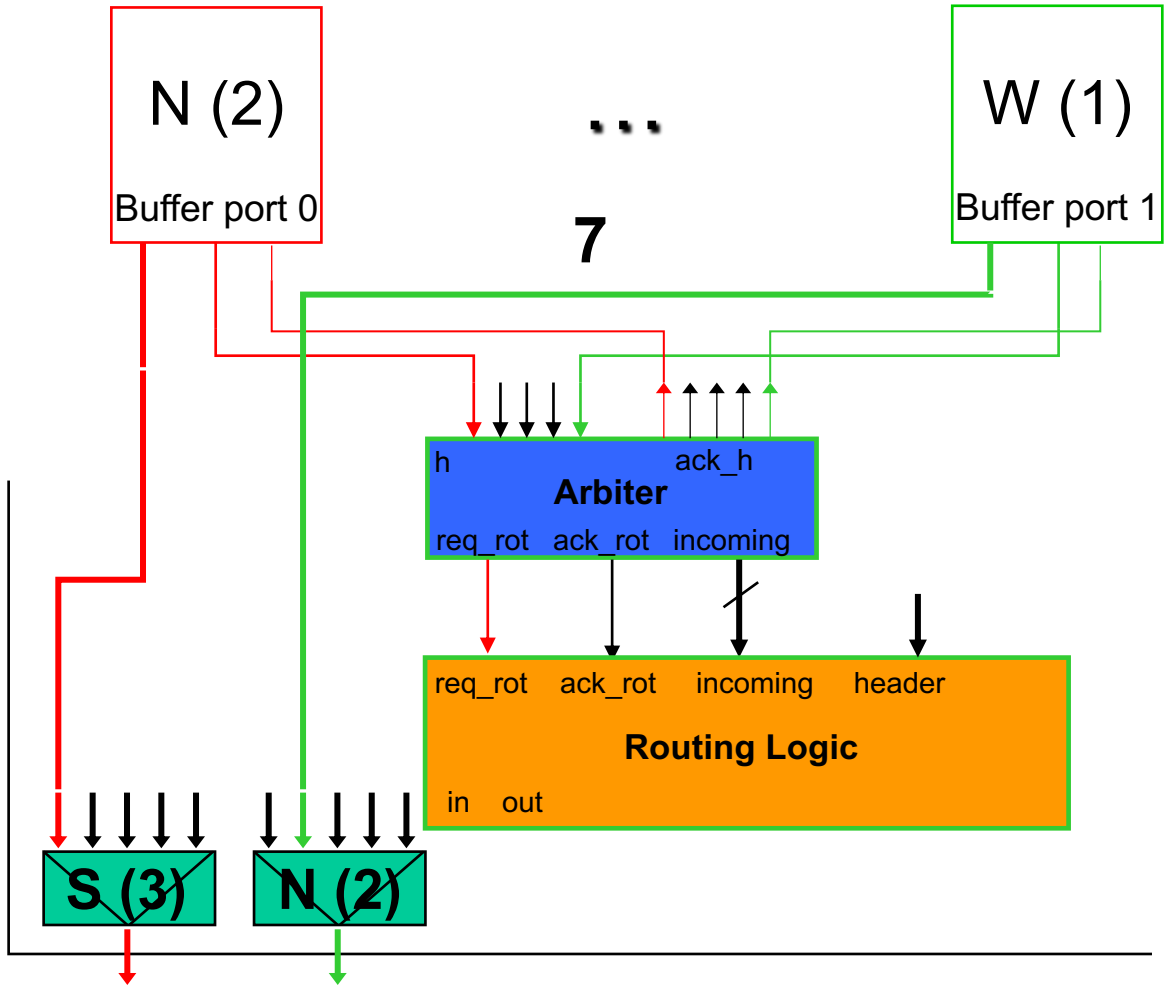


Tabela de roteamento

- Tabela de conexão
  - Quando uma conexão é estabelecida o caminho entre a porta de entrada e saída é armazenado
  - Tres vetores são utilizados: IN, OUT, FREE

	0 – E	1 – W	2 – N	3 – S	4 – L
Free	1	1	0	0	1
In		2	3		
Out			1	2	





Estrutura do código

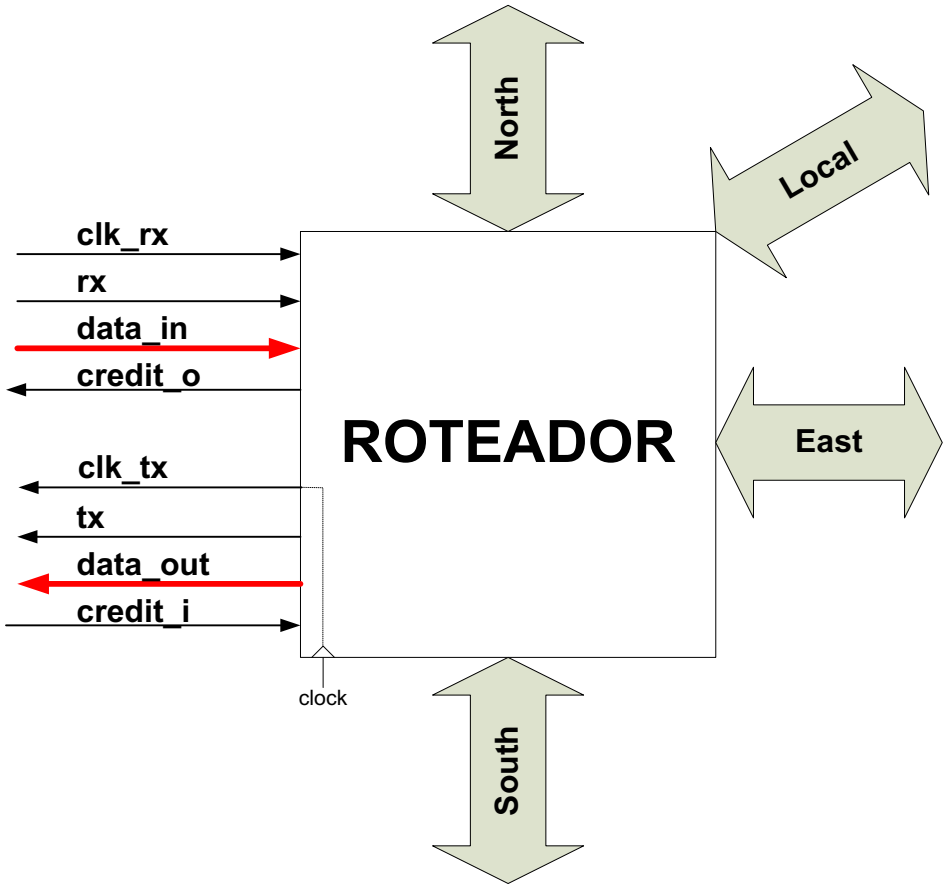
topNoC

- **Instanciação de roteadores**
- **Conexão entre os sinais, com os devidos aterramentos (portas não utilizadas)**

Estrutura do código

Roteador

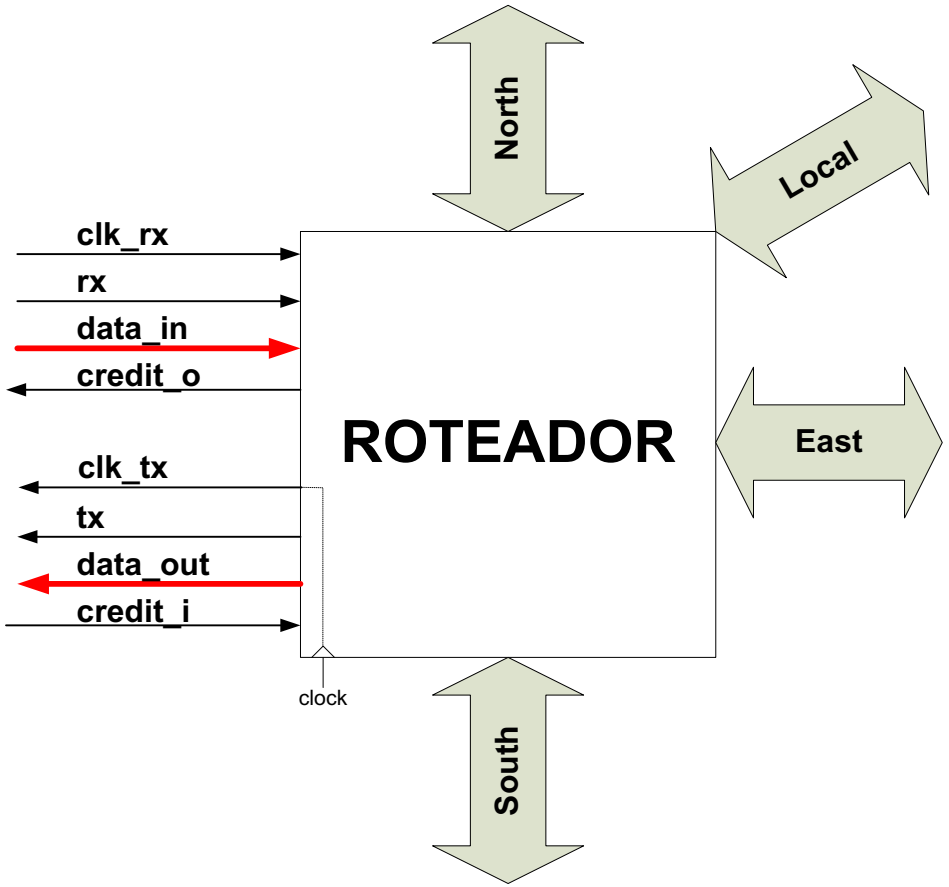
```
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.std_logic_unsigned.all;
use work.HermesPackage.all;

entity RouterCC is
generic( address: regmetadeflit);
port(
    clock:      in  std_logic;
    reset:      in  std_logic;
    clock_rx:   in  regNport;
    rx:         in  regNport;
    data_in:    in  arrayNport_regflit;
    credit_o:   out regNport;
    clock_tx:   out regNport;
    tx:         out regNport;
    data_out:   out arrayNport_regflit;
    credit_i:   in  regNport);
end RouterCC;

architecture RouterCC of RouterCC is

signal h, ack_h, data_av, sender, data_ack: regNport := (others=>'0');
signal data: arrayNport_regflit := (others=>(others=>'0'));
signal mux_in, mux_out: arrayNport_reg3 := (others=>(others=>'0'));
signal free: regNport := (others=>'0');
```



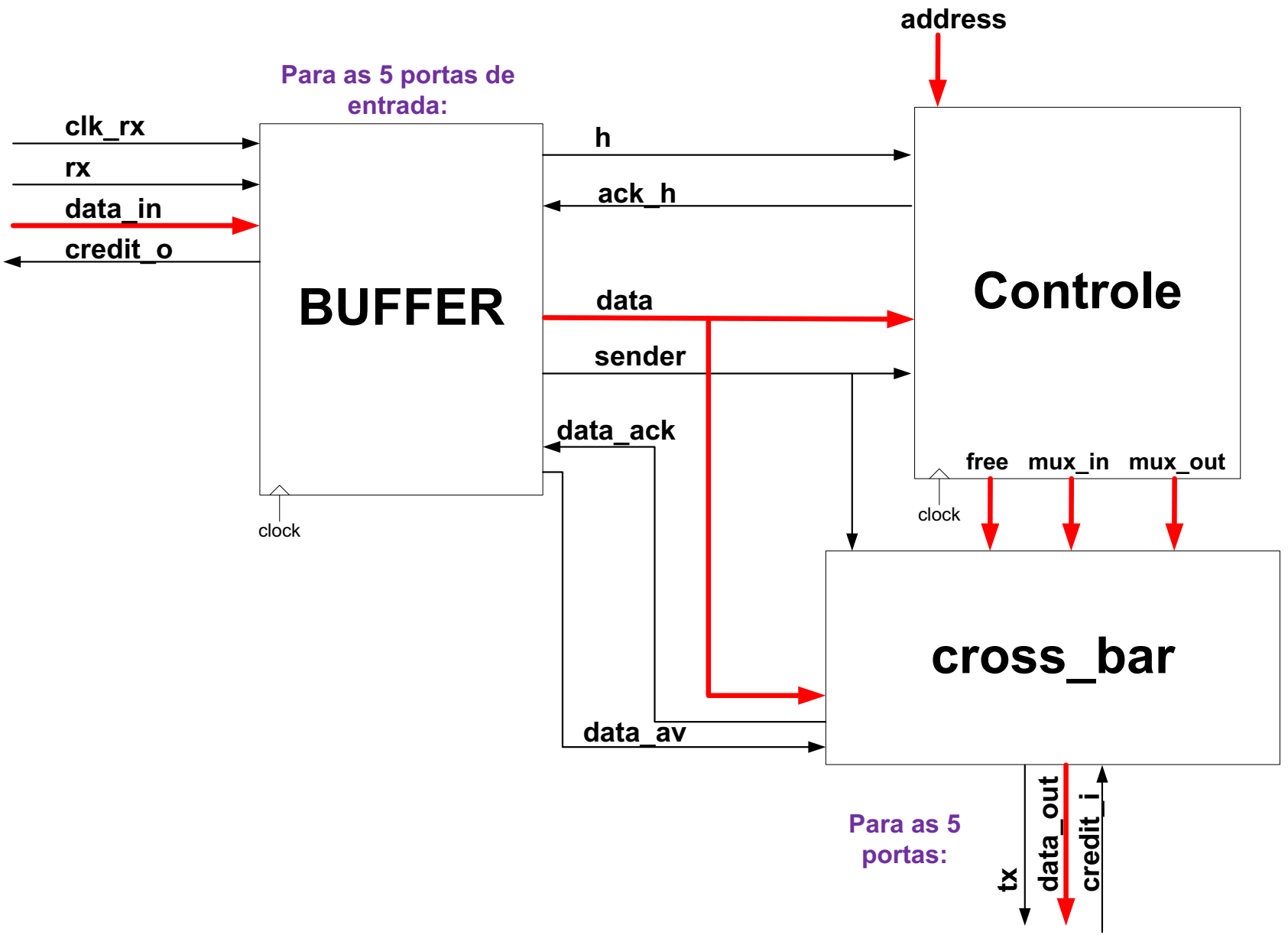
Estrutura do código

routerCC

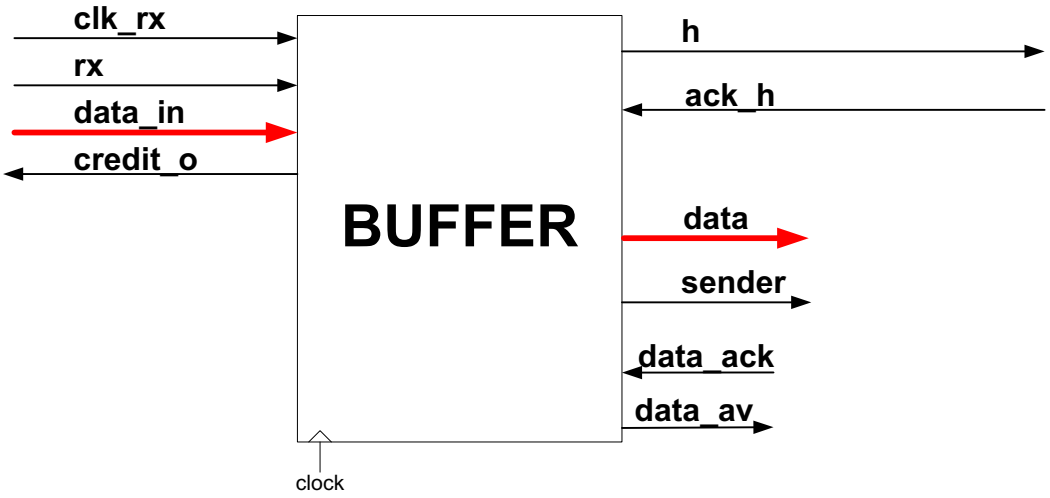
```
begin

    FEast : Entity work.Hermes_buffer
    port map(

    FWest : Entity work.Hermes_buffer
    port map(

    FNorth : Entity work.Hermes_buffer
    port map(

    FSouth : Entity work.Hermes_buffer
    port map(

    FLocal : Entity work.Hermes_buffer
    port map(

    SwitchControl : Entity work.SwitchControl(AlgorithmXY)
    port map(

    CrossBar : Entity work.Hermes_crossbar
    port map(

    CLK_TX : for i in 0 to(NPORT-1) generate
     clock_tx(i) <= clock;
    end generate CLK_TX;

end RouterCC;
```



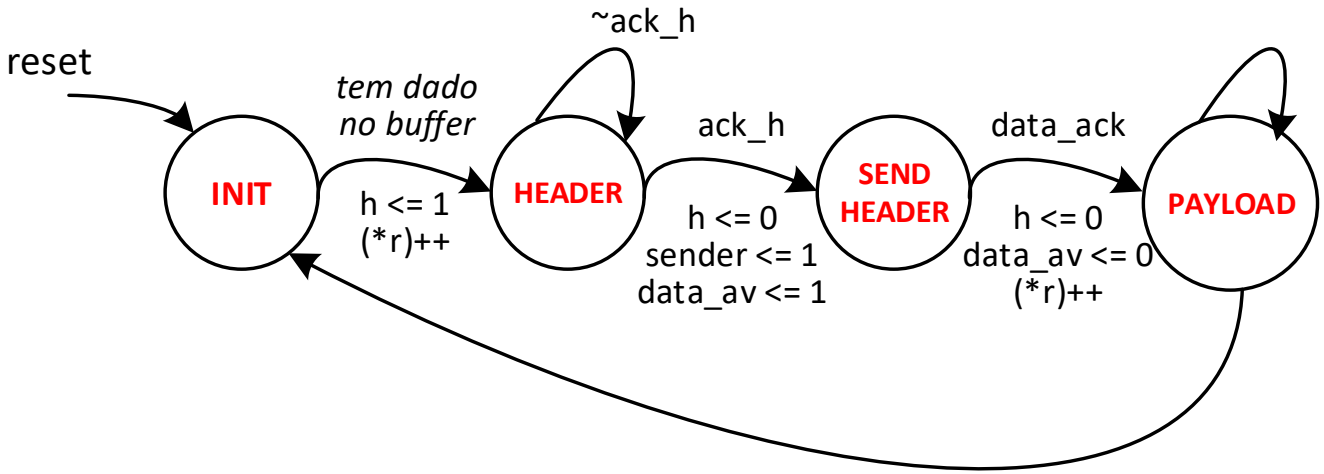
Roteador


address
Para as 5 portas de entrada:
clk_rx
rx
data_in
credit_o
BUFFER
clock
h
ack_h
data
sender
data_ack
data_av
Controle
free
mux_in
mux_out
clock
cross_bar
Para as 5 portas:
tx
data_out
credit_i

Buffer



FSM para ler do buffer



se há dado no buffer:
data_av <= 1

se o dado foi consumido:
{ trata tam_pacote

se final do pacote:
{ sender <= 0
data_av <=0
volta para INIT
}
senão:
{ (*r)++
continua em PAYLOAD
}
}

Buffer

FSM de controle