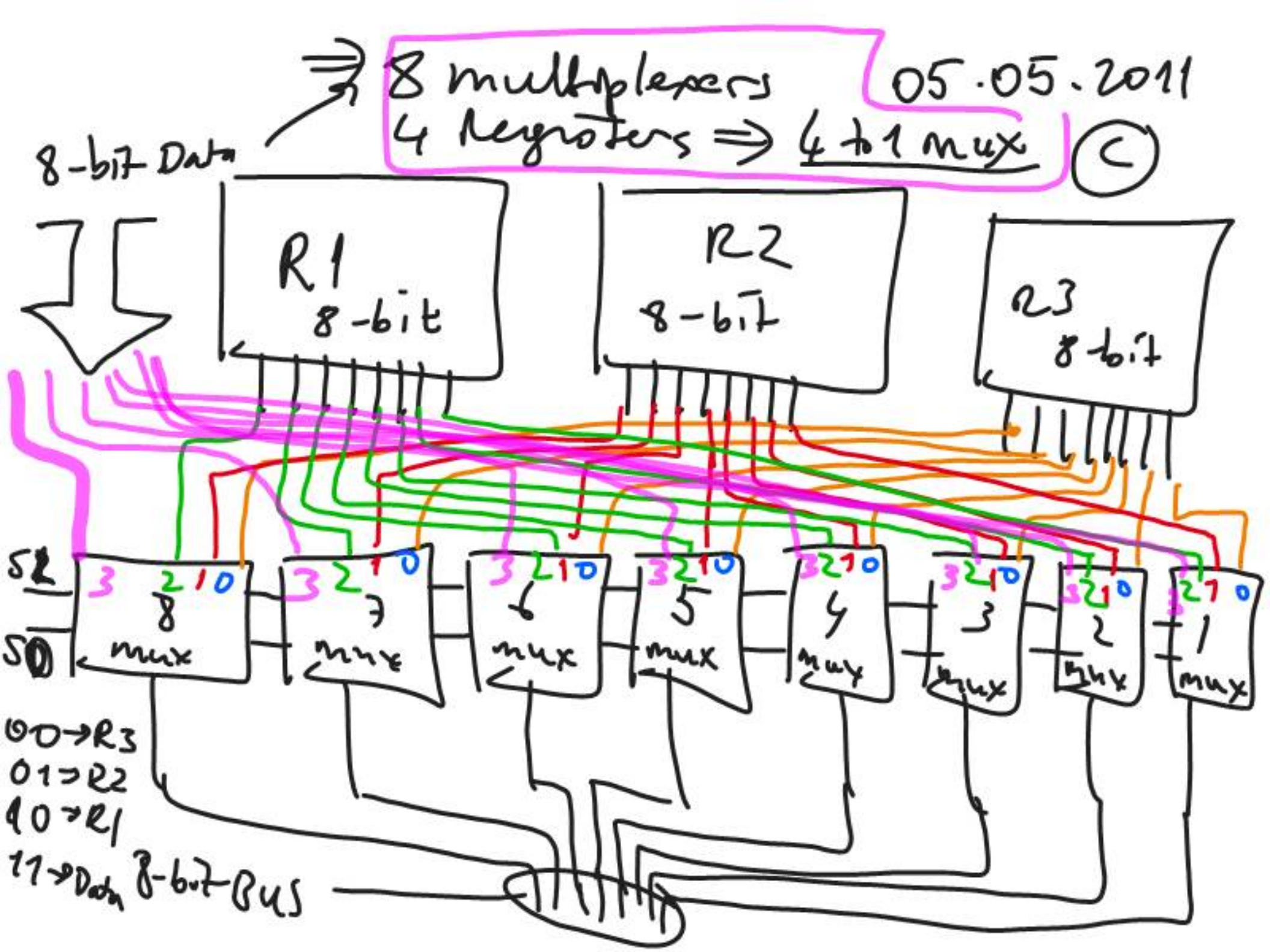

8 multiplexers
4 Registers ⇒ 4 to 1 mux
05.05.2011
C
8-bit Data
R1
8-bit
R2
8-bit
R3
8-bit
S1
S0
8
mux
7
mux
6
mux
5
mux
4
mux
3
mux
2
mux
1
mux
00→R3
01→R2
10→R1
11→Data 8-bit BUS

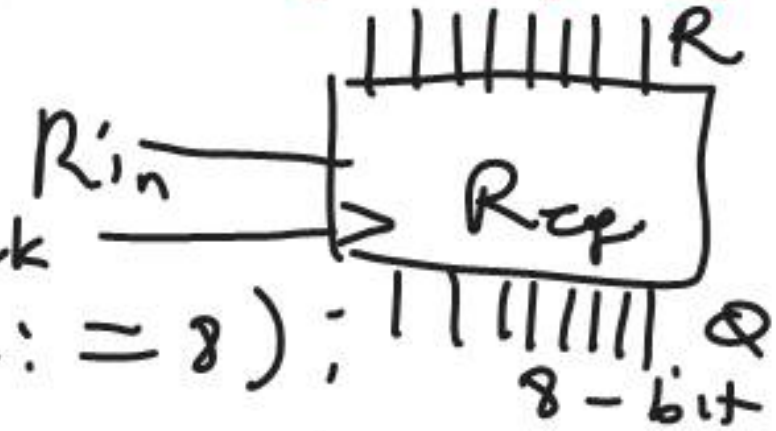
Code for 3-bit Buffers:


```
;
ENTITY Trin IS
GENERIC (N : INTEGER := 8);
PORT( X : IN STD_LOGIC_VECTOR( N-1 DOWN TO 0);
      E : IN STD_LOGIC;
      F : OUT STD_LOGIC_VECTOR(N-1 DOWN TO 0));
END Trin;
ARCHITECTURE Behaviour of Trin IS
BEGIN
F <= (OTHERS => 'Z') WHEN E = '0' ELSE X;
END Behaviour;
```

High impedance

Conditional concurrent statement

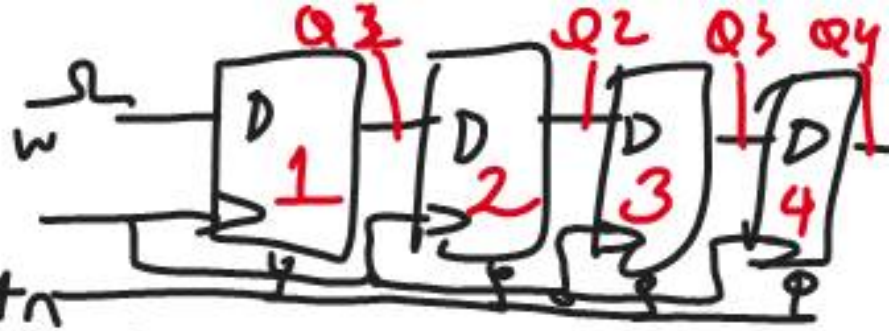
Code for n-bit registers (R_1, R_2, R_3)




```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
ENTITY Regn IS
GENERIC ( N : INTEGER := 8 );
PORT ( R : IN STD_LOGIC_VECTOR (N-1 DOWNTO 0);
       Rin, Clock : IN STD_LOGIC;
       Q : OUT STD_LOGIC_VECTOR(N-1 DOWNTO 0));
END Regn;
ARCHITECTURE Behavior OF Regn IS
BEGIN
PROCESS
BEGIN
```

```
WAIT UNTIL Clock'EVENT AND Clock='1';
  IF Rin='1' THEN
     Q <= R;
  END IF;
 END PROCESS;
END Behavior;
```

Control Circuit Code:




```
:
ENTITY SHIFTR IS  Resetn
GENERIC ( K: INTEGER := 4); -- 4 bit Register
PORT ( Resetn, Clock, W : IN STD_LOGIC;
       Q : BUFFER STD_LOGIC_VECTOR(1 TO K));
END Shifter;
ARCHITECTURE Behavior OF Shifter IS
 BEGIN
PROCESS (Resetn, Clock)
 BEGIN
```

-- Four Q bits are not forming a code. They are taken separately

```
IF Reset = '0' THEN                    -- asynchronous reset
    Q <= (OTHERS => '0');
ELSIF Clock'EVENT AND Clock = '1' THEN
    Genbits: FOR i IN K DOWNTO 2 LOOP;
        Q(i) <= Q(i-1);
    END LOOP;
    Q(1) <= W;
END IF;
END PROCESS;
END Behavior;
```

shift register

Q(4) <= Q(3)
Q(3) <= Q(2)
Q(2) <= Q(1)
Q(1) <= W

Declare all of these as PACKAGE :

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
PACKAGE SWAP_COMPONENTS IS
  COMPONENT Reg n
  GENERIC (N: INTEGER := 8);
  PORT (R: IN STD_LOGIC_VECTOR (N-1 DOWNTO 0);
        Rin, Clock: IN STD_LOGIC;
        Q : OUT Std_Logic_Vector (N-1 DOWNTO 0));
  END COMPONENT;
  COMPONENT SHIFTR -- The control register
  GENERIC (K: INTEGER := 4);
```

① Reg n

```
PORT (Reset, Clock, W: IN STD_LOGIC;
      Q : BUFFER STD_LOGIC_VECTOR (1 TO K));
END COMPONENT;
COMPONENT TRIN
 GENERIC(N: INTEGER := 8);
 PORT( X: IN STD_LOGIC_VECTOR (N-1 DOWNTO 0);
       E: IN STD_LOGIC;
       F: OUT STD_LOGIC_VECTOR(N-1 DOWNTO 0));
 END COMPONENT;
END SWAP_COMPONENTS;
```

②

③

Summary:

PACKAGE SWAP_COMPONENTS

① COMPONENT Reg n
Rin
Clock
R
Q
END COMPONENT;

② COMPONENT shifter
W
Clock
Reset
Q(1) Q(2) Q(3) Q(4)
END COMPONENT

③ COMPONENT TRIN
X
E
F
END COMPONENT

END SWAP_COMPONENTS;

SWAP CIRCUIT

```
USE work.SWAPCOMPONENTS.all;
ENTITY SWAP IS
PORT (Data: IN STD_LOGIC_VECTOR(7 DOWNTO 0);
      Reset, W: IN STD_LOGIC;
      Clock, Extern: IN STD_LOGIC;
      RIN Ext: IN STD_LOGIC_VECTOR (1 TO 3);
      Buswires: INOUT STD_LOGIC_VECTOR (7 DOWNTO 0));
END SWAP;
```

```
ARCHITECTURE Behavior OF SWAP IS
SIGNAL Rin, Rout, Q : STD_LOGIC_VECTOR (1 TO 3);
SIGNAL R1, R2, R3 : STD_LOGIC_VECTOR (7 DOWN TO 0);
BEGIN
```

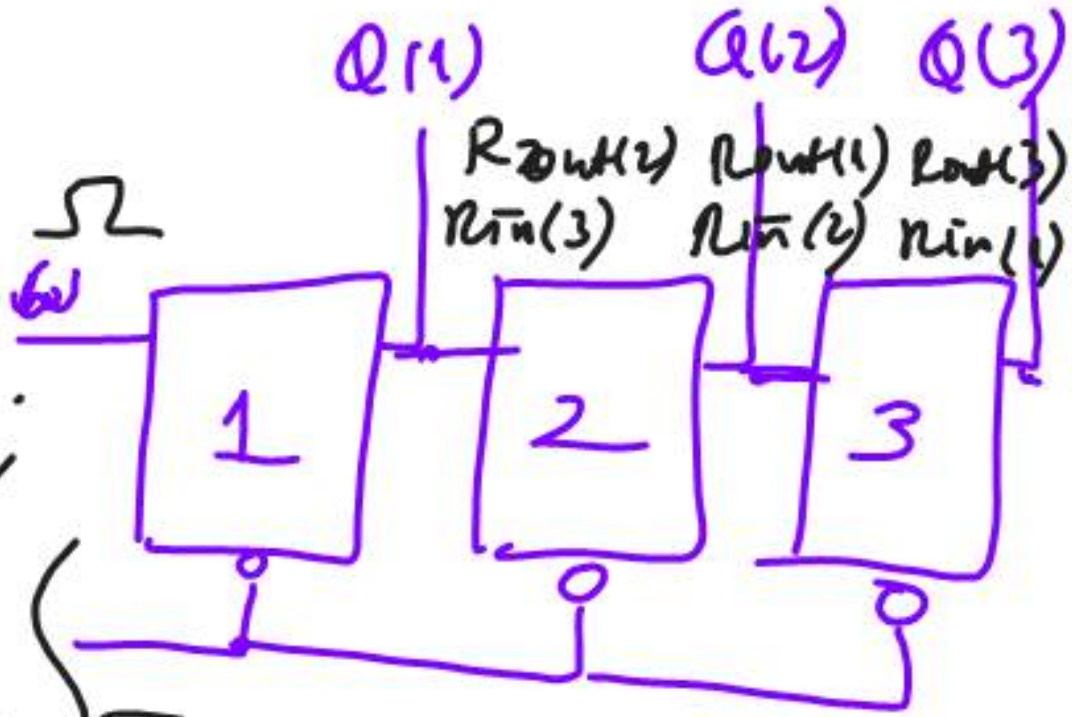
-- signals are either IN or OUT

```
Control : shifter GENERIC MAP (K => 3)
PORTMAP (Reset, Clock, w, Q);
Rin(1) <= Rin0ut(1) OR Q(3);
Rin(2) <= Rin0ut(2) OR Q(2);
Rin(3) <= Rin0ut(3) OR Q(1);
```

Rout(1) ⇐ Q(2)

Rout(2) ⇐ Q(1)

Rout(3) ⇐ Q(3);



tri_ext : TRIN PORT MAP

(Data, Extern, Buswires);

reg1 : regn PORTMAP

(Buswires, Rin(1), Clock, R1);

reg2 : regn PORTMAP

(Buswires, Rin(2), Clock, R2);

reg3 : regn PORTMAP (Buswires, Rin(3) Clock, R3);

R2 → R3

R1 → R2

R3 → R1

```
tri1 : tri PORTMAP (R1, Rout(1), BusWires);
tri2 : tri PORTMAP (R2, Rout(2), BusWires);
tri3 : tri PORTMAP (R3, Rout(3), BusWires);

END Behavior;
```

SWAP CIRCUIT: