Conditional Concurrent Signal Assignment

The conditional concurrent signal assignment statement is modeled after the “if statement” in software programming languages.

The general format for this statement is:

```
target_signal <= value1 WHEN condition1 ELSE
                 value2 WHEN condition2 ELSE
                 value3 WHEN condition3 ELSE
                          ...........
                 valueN;
```

When one or more of the signals on the right-hand side change value, the statement executes, evaluating the condition clauses in textual order from top to bottom. If a condition is found to be true, the corresponding expression is executed and the values are assigned to the target signal.

The conditions must evaluate to a boolean value. i.e, True or False

Example:

```
z_out <=   a_input WHEN (select = “00”) ELSE
           b_input WHEN (select = “01”) ELSE
           c_input WHEN (select = “10”) ELSE
           d_input WHEN (select = “11”) ELSE
           “X”; -- what am I?
```

Conditional Concurrent Signal Assignment

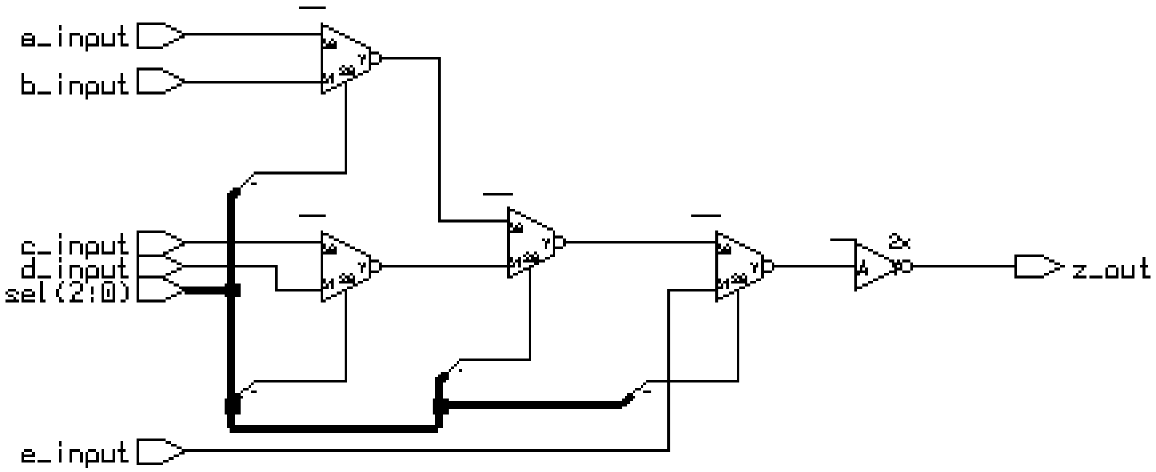
What happens when we don’t completely specify all the choices?

First, lets do it right.

```
--5:1 mux, 1 bit wide
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY mux5_1_1wide IS
  PORT(
     a_input   : IN STD_LOGIC;  --input a
     b_input   : IN STD_LOGIC;  --input b
     c_input   : IN STD_LOGIC;  --input c
     d_input   : IN STD_LOGIC;  --input d
     e_input   : IN STD_LOGIC;  --input e
     sel       : IN STD_LOGIC_VECTOR(2 DOWNTO 0);  --sel input
     z_out     : OUT STD_LOGIC  --data out
     );
END mux5_1_1wide;
ARCHITECTURE beh OF mux5_1_1wide IS
  BEGIN
     z_out <= a_input WHEN (sel = “000”) ELSE
              b_input WHEN (sel = “001”) ELSE
              c_input WHEN (sel = “010”) ELSE
              d_input WHEN (sel = “011”) ELSE
              e_input WHEN (sel = “100”) ELSE
             ‘X’;
 END beh;
```

When synthesized, we get:

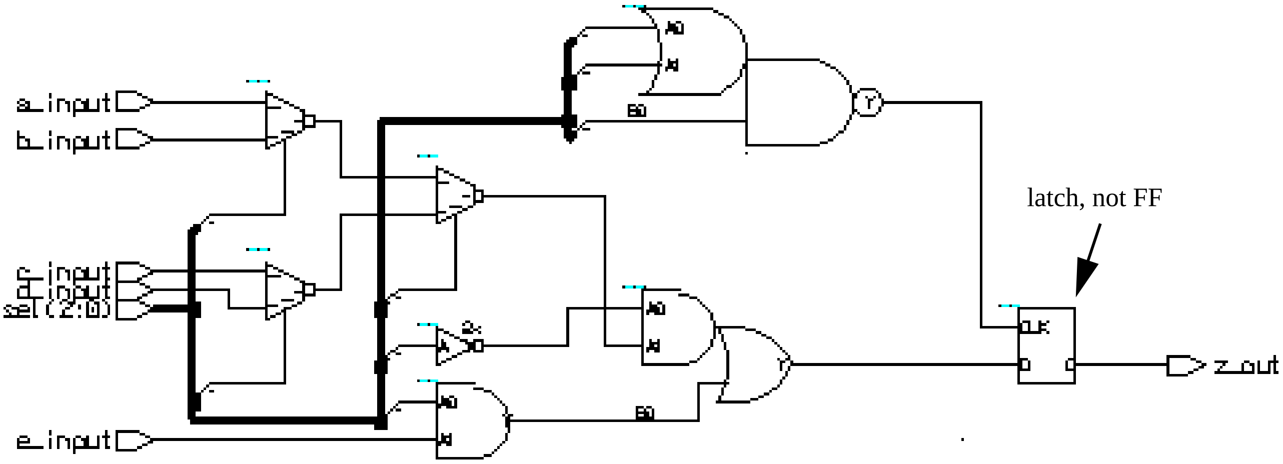


Conditional Concurrent Signal Assignment

Now let’s incompletely specify the choices.

```
ARCHITECTURE noelse OF mux5_1_1wide IS
  BEGIN
     z_out <= a_input WHEN (sel = “000”) ELSE
              b_input WHEN (sel = “001”) ELSE
              c_input WHEN (sel = “010”) ELSE
              d_input WHEN (sel = “011”) ELSE
              e_input WHEN (sel = “100”); -- no ending else
 END beh;
```

When synthesized:



latch, not FF

What happened?

- How does a transparent latch operate?
- What is the truth table for the decoder to the latch “clk” pin?

sel(2:0)	latch enable pin	behavior
000	1	latch is transparent
001	1	ditto
010	1	ditto
011	1	ditto
100	1	ditto
101	0	latch is in “hold” state
110	0	hold state
111	0	hold state