Comp 331: Intro to Computer Hardware

Decoders, Encoders, and Priority Encoders

Decoder maps $n$ inputs into exactly one of $2^n$ outputs

```
2-to-4 Decoder

\[ a_0 \rightarrow y_0 \]
\[ a_1 \rightarrow y_1 \]
\[ y_0 \rightarrow 0 \]
\[ y_1 \rightarrow 0 \]
\[ 0 \]
\[ 1 \]
\[ 0 \]
\[ 1 \]
\[ 0 \]
\[ 0 \]
```

Architecture decode38a of decode38a is

begin
\[ y(0) \triangleq \text{not } a(2) \text{ and not } a(1) \text{ and not } a(0); \]
\[ y(1) \triangleq \text{not } a(2) \text{ and not } a(1) \text{ and } a(0); \]
\[ y(2) \triangleq \text{not } a(2) \text{ and } a(1) \text{ and not } a(0); \]
\[ y(3) \triangleq \text{not } a(2) \text{ and } a(1) \text{ and } a(0); \]
\[ y(4) \triangleq a(2) \text{ and not } a(1) \text{ and not } a(0); \]
\[ y(5) \triangleq a(2) \text{ and not } a(1) \text{ and } a(0); \]
\[ y(6) \triangleq a(2) \text{ and } a(1) \text{ and not } a(0); \]
\[ y(7) \triangleq a(2) \text{ and } a(1) \text{ and } a(0); \]
end decode38a;

Encoder maps $2^n$ input into an $n$-bit binary number output

```
4-to-2 Encoder

\[ a_0 \rightarrow y_0 \]
\[ a_1 \rightarrow y_1 \]
\[ a_2 \rightarrow y_2 \]
\[ a_3 \rightarrow y_3 \]
```

```
architecture encode83a of encode83a is

begin
process(x)
variable valid_var: STD_LOGIC;
begin
\[ y(2) \triangleq x(7) \text{ or } (x(6) \text{ or } x(5) \text{ or } x(4)); \]
\[ y(1) \triangleq x(7) \text{ or } x(6) \text{ or } x(3) \text{ or } x(2); \]
\[ y(0) \triangleq x(7) \text{ or } x(5) \text{ or } x(3) \text{ or } x(1); \]
valid_var := '0';
for \( i \) in 7 downto 0 loop
valid_var := valid_var or x(i);
end loop;
valid <= valid_var;
end process;
end encode83a;
```

Priority Encoder

```
architecture pencode83 of pencode83 is

begin
process(x)
variable j :integer;
begin
\[ y \triangleq "000" ; \]
\[ \text{valid} \triangleq '0'; \]
for \( j \) in 0 to 7 loop
if \( x(j) = '1' \) then
\[ y \triangleq \text{conv_std_logic_vector}(j,3); \]
\[ \text{valid} \triangleq '1'; \]
end if;
end loop;
end process;
end pencode83;
```

Converts \( j \) to 3-bit STD_LOGIC_VECTOR signal

Requires: use IEEE.STD_LOGIC_arith.all

Exercise #1 – Program and Simulate (do not implement on BASYS-2 board) Example 25: 8-to-3 Encoder: Logic Equations

Note that the Fig 5.53 Simulation uses circular one counter as stimulator for input \( x \).

Instructor Check __

Exercise #2 – Program and Implement on BASYS-2 board
Example 26: 8-to-3 Priority Encoder. Note the use of \( dp \) for the valid bit

Instructor Check ____