Specifying Programmable Logic

- Marking 'X's on a PAL diagram is a nice way to visualize which fuses will be blown or left intact but we obviously need an electronic form for specifying fuse programming.
- A JEDEC file (.jed) is a standard file format that specifies the fuse programming.
- PLD programmers read a JEDEC file and program the PLD device based on the JEDEC file contents.
  - High-density PLD devices such as FPGAs may use vendor-dependent file formats instead of JEDEC format.

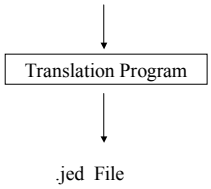
PLD Programming Options

- Some PLDs are One Time Programmable (OTP). Once they are programmed, they cannot be erased.
- Eraseable PLDs are either Electrically Erasable or UV-light erasable
  - UV erasable PLDs have a quartz window. The PLD must be removed from its socket, and placed under a UV light source for 10-20 minutes
- Some PLDs have volatile programming - that is, the programming contents is lost on power down.
  - Must be programmed from some non-volatile memory device on every power up.
- The 22V10 PLD we will use in Lab is UV-erasable and is non-volatile.

Creating a JEDEC File

- A JEDEC file was never meant to be produced manually - too cumbersome, error prone to do so.

P = A'BGH' + CD' + HIJ + BG'H



PLD Programming Language

Need a language that we can use to specify the logic that we want programmed into the PLD.

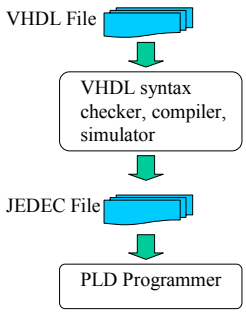
Need a translation program that will translate this language into a JEDEC file for the PLD programmer to use.

An early language was called PALASM (PAL ASseMbly language). PALASM is now obsolete.

We will use a language called VHDL, which is commonly used in industry and is an IEEE standard. A software program called WARP from Cypress Semiconductor will be used to translate the VHDL file into a JEDEC file.

Another common language is called Verilog. We will not use Verilog in this class.

PLD Programming



Boolean equations in VHDL

The boolean operators in VHDL are: not, and, or, nand, nor, xor, xnor

VHDL for F = AB' + C is

```
F <= (A and not(B)) or C ;
```

Signal name

Signal assignment operator

boolean operators

A semi-colon marks the end of the line

VHDL is not case sensitive. Capitalization used for emphasis only.

Operator Precedence

The *and,or,nor,nand,xor,xnor* operators all have the SAME precedence.

The *not* operator has higher precedence than any of these.

You MUST BE CAREFUL!!! For F= C + A'B

F <= C or not A and B;

is ACTUALLY the equation F = (C + A') B !!!!

Use lots of parentheses to make sure of precedence!!!

F <= C or ((not A) and B) ;

This has more parentheses than needed but improves readability.

A Complete VHDL Model

```
library ieee;
use ieee.std_logic_1164.all;
-- this is the entity declaration
entity majority is
   port ( A, B, C : in std_logic;     -- two dashes is a COMMENT in VHDL
          Y: out std_logic
        );
end majority;
-- this is the architecture declaration

architecture boolean of majority is

begin

Y <= (A and B) or (A and C) or (B and C);

end boolean;
```


Comments on example

- Italics used to indicate what are not keywords
- Most VHDL compilers/simulators require the File name to be the entity name with a .VHD extension (majority.vhd).
- Each VHDL model has two major sections.
  - The first section is the "entity" declaration. Declares the input/output ports.
  - The second part is the "architecture" declaration. Describes the logic that implements the function.

How much VHDL do YOU have to Know?

- VHDL is not the major thrust of this class; only a means of specifying the contents of a PLD
- You will always be provided with a 'template' file that has the entity already written and the architecture left blank.
- You will only have to provide the boolean equations.
- Other examples of VHDL models will be given here but these only demonstrate a tiny subset of VHDL. You will only be expected to write boolean equations in VHDL.

Using Internal Signals

If I have complex boolean equations, it might be easier to declare some internal signals. I have shown the architecture section of the previous example modified to use internal signals.

```
architecture boolean of majority is
  signal t1, t2, t3 : std_logic ;

begin
  t1 <=  A and B;
  t2 <=  A and C;
  t3 <=  B and C;
  Y <=  t1 or t2 or t3;

end boolean;
```


VHDL Template for SSN Decode (Entity)

```
library ieee;
use ieee.std_logic_1164.all;
use work.cypress.all;

entity ssncomb is
port (
-- inputs
  signal  A:  in std_logic;  -- MSB
  signal  B:  in std_logic;
  signal  C:  in std_logic;
  signal  D:  in std_logic;  -- LSB
-- outputs
  signal    F1,F2, F3, F4:  out std_logic

);
-- this pin assignment is specifically for 22V10!
attribute pin_numbers of ssncomb:entity is
 "A:2 B:3  C:4  D:5 " &
  " F1:22   F2:21  F3:20  F4:19";

end ssncomb;
```

Used to specify the mapping of inputs/outputs to pins on the 22V10 PLD

VHDL Template for SSN Decode (Architecture)

```
-- Architecture body

architecture a of ssncomb is
-- Solution for  SSN 458 70 2198

begin

--- STUDENTS: Put your statements starting on the next line

 F1 <=   (B and (not C)) or (A and (not D));
 F2 <=  ( (not A) and (not B) and (not C) and (not D))  or
       (B and C and D);
 F3 <=  A or ((not B) and (not C) and D) or ((not B) and C and (not D));

 F4 <= A or (not C) or (B and D) or ((not B) and (not D));

end a;
```


Compiling, Simulating, JEDEC File creation

Maxplus II can be used to compile/simulate your VHDL file. However, not all students have access to Maxplus II. Also, Maxplus II cannot produce a JEDEC file for a 22V10 PLD.

To make things easier, a WWW page has been created that will perform compilation, simulation, and JEDEC file creation.

Look at:

http://www.ece.msstate.edu/~reese/EE3714/webcad/ssncomblab.htm

This form requires that your VHDL file be accessible from the PC/Workstation running your web browser.

Compilation/Simulation Results

When you submit your VHDL file via the previous form, your file is shipped to a server in Electrical Engineering, compiled, and simulated.

If your file had no *syntax* errors, you will get back a HTML page that has the simulation results of your file, and a link to your JEDEC file.

PRINT OUT THIS PAGE!! (it has a URL that you will need in Lab in order to retrieve your JEDEC file for programming).

Verify that the simulation results matches your expected results. If they do not match, then your boolean equations are incorrect. You will not be allowed to use the PLD programmer if your simulation results are incorrect.

Compilation/Simulation Failure

- If your file has a syntax error, then the page will not have a link to a JEDEC file
  - Look at the compiler messages that come back, correct the syntax errors, and re-submit
- If the simulation results are not correct for inputs “0000” to “1001” , then check your boolean equations, and try again.
  - The inputs “1010” to “1111” are don’t cares so it does not matter what the outputs are for these input values (they will depend on how your treated the ‘X’s in your K-maps).

Before Lab (2nd Week)

- The PLD solution is to be done the 2nd week of the SSN Decode lab.
- You must have a printout of a successful VHDL compilation/simulation run for your prelab.
  - This WWW page is not deleted after the compilation/simulation run - you can access this page in the Lab and download the JEDEC file to the PC in the lab for PLD programming
  - Do NOT print the JEDEC file.
- You will not be allowed to program your PLD if the simulation results are incorrect