PALs, GALs & CUPL

A lesson on Programmable Logic Devices and programming them using CUPL

What is Programmable Logic?

- Digital integrated circuits where the Boolean function can be determined by the user.
- PLDs can replace several specific purpose ICs in a digital design.
- A single PLD is functionally equivalent to a specific device containing from 5 to 10,000 gates.
- Typically PLDs implement Boolean functions using Sum Of Minterms (SOM) or Sum of Products (SOP) form.
 SOM and SOP use a AND-OR gate structure.

Basic Architecture of PLDs

- PLDs contain a pre-defined general architecture for a user to program a design into the device.
 Generally consists of one or more arrays of AND and OR terms
- Also contain flip-flops and latches to be used as storage elements for inputs and outputs
- Consist of programmable AND terms feeding fixed OR terms.

All inputs can be AND-ed together but specific AND terms are dedicated to specific OR terms.

Pins can be used as Input, Output, I/O with tri-state enables, also may have output registers.

PLD Programming

PLDs are manufactured in a "blank" or "erased" form. Programming is performed in concept blowing out fuses between inputs, AND gates, and OR gates in the generic AND-OR structure. An erased PLD has all fuses intact. Actual "fuses" may be implemented as:

<u>Type</u>	<u>Function</u>
fuses	one-time programmable
EPROM	uv light erasable
CMOS	electrically erasable
EEPROM	

<u>Advantages</u>

low cost Reprogrammable fast, easy reprogramming Fuses and Logic Implementation



A blown fuse acts like the input does not exist (or a logic 1 at the input).

PLD Advantages:

- •reduce IC package count
- •board space
- •power
- •shorten design time
- •allow for future changes
- (maintainability)
- •improve reliability (fewer packages)
- •generally faster
- •smaller inventory

Programmable Logic Devices I

EECE143 Lecture 4



© J. Chris Perez 2001

PAL Example: Given functions w, x, y, and z. Implement with <u>one</u> PAL4H4.

Given: Sum of Minterms w(A,B,C,D) = Σ (2,12,13) x(A,B,C,D) = Σ (7,8,9,10,11,12,13,14,15) y(A,B,C,D) = Σ (0,2,3,4,5,6,7,8,10,11,15) z(A,B,C,D) = Σ (1,2,8,12,13) ... After Simplification w = ABC' + A'B'CD' x = A + BCD y = A'B + CD + B'D'z = w + AC'D' + A'B'C'D

Programmable Logic Devices I



© J. Chris Perez 2001

Programmable Array Logic

- PALXXYZ
- XX : Number of possible inputs
- Y : L = Combinational R = Registers
- Z : Number of possible outputs
- Examples:
- PAL16R4 : 16 inputs with 4 registered outputs
- PAL16L8 : 16 inputs with 8 outputs

Generic Array Logic

Designed to emulate many common PALs Reduces number of different devices Electronically erasable Such as GAL16V8, GAL22V10

GAL16V8

- 8 output Macrocells
- Pin 1: Clock
- Pin 11: Active-low Output Enable
 - Must be tied low to use Registered outputs
 - Must be declared in .PLD file
- Can emulate various PAL devices
- 8 Product Terms per Output

GAL22V10

- 10 output Macrocells
- 24 Pin Package
- Pin 1: Clock
- Pin 13: Active-low Output Enable
- Can emulate various PAL devices
- Pin 14,23 = 8 Product Terms
- Pins 15,22 = 10 Product Terms
- Pins 16,21 = 12 Product Terms
- Pins 17,20 = 14 Product Terms
- Pins 18,19 = 16 Product Terms

WinCUPL Programming Language

WinCUPL is a software package that runs on an PC. It performs most of the work in translating a PLD design into a programming file. The programming file can be used to program an IC to implement the desired logic functions. Marquette University (College of Engineering) has a site license for CUPL.

•	Translates PLD Design into	! Invert
	programming file.	& AND
•	CUPL expressions written in Sum of Products form. <i>Limit</i>	# OR
	of 7 product terms	\$ XOR

CUPL Programming

- Combinational
 - Use equations or Truth table
 - Examples1.PLD, Example2.PLD
- Sequential
 - Use Equations or State Machine
 - Example3.PLD, Example4.PLD

Combinational Logic Implementation

- Problem: CUPL expressions must be written for the function. Sum of Products (SOP) is the form CUPL uses. There is a limit of 7 product terms.
- Given: A sum of products equation.
- Find: A complement form of the equation also in sum of products.

Example: A = x'y + xy'z

- In CUPL this becomes:
- Example: B = x'y'z' + xy' + yz

In CUPL this becomes:

A = !x&y # x&!y&z;

B = !x&!y&!z # x&!y # y&z;

Example1.pld

Name	Example1;	
Partno	none;	
Date	06/01/98;	
Revision	01;	
Designer	Bruce Hoeppner;	
Company	EECE143;	
Assembly	none;	
Location	EN365;	
Device	G16V8;	
/ * * * * * * * * * *	* * * * * * * * * * * * * * * * * * * *	/
/* Example	e program to implement the following equations st	/
/* in CUPI	J Programming language *	/
/ *	*	/
/* A =	x'y + xy'z *	/
/* B =	x'y'z' + xy' + yz *	/
/ *	*	/
/ * * * * * * * * * *	* * * * * * * * * * * * * * * * * * * *	1

Example1.pld

```
/** Inputs **/
Pin 2 = x;
Pin 3 = y;
Pin 4 = z;
/** Outputs **/
Pin 19 = A;
Pin 18 = B;
/** Declarations and Intermediate Variable Definitions **/
/** Logic Equations **/
A = !x&y # x&!y&z;
B = !x&!y&!z # x&!y # y&z;
/** End of example1.pld source code **/
```

EECE143 Lecture 4

Example2.pld

Name	Example2;
Partno	example2.pld;
Date	06/01/98;
Revision	01;
Designer	Bruce Hoeppner;
Company	EECE143;
Assembly	PLD Source Code;
Location	EN365;
Device	G16V8;

```
/*
  Example program to implement the following equations*/
/*
  in CUPL programming language using Truth Tables
                                * /
/*
                                * /
/*
 A = x'y + xy'z
                                * /
/*
  B = x'y'z' + xy' + yz
                                * /
/*
                                * /
/*
                                * /
  Allowable Target Device Types:
```

Programmable Logic Devices I

EECE143 Lecture 4

```
/** Inputs **/ Example2.pld
Pin 2 = xi
Pin 3 = y;
Pin 4 = z;
/** Outputs **/
Pin 19 = A;
Pin 18 = B;
/** Declarations and Intermediate Variable Definitions **/
FIELD INPUT = [x, y, z];
FIELD OUTPUT = [A,B];
/** Truth Table **/
TABLE INPUT => OUTPUT {
0=>'b'01;
1=>'b'00;
2=>'b'10;
3=>'b'11;
4=>'b'01;
5=>'b'11;
6=>'b'00;
7=>'b'01;
}
                                                © J. Chris Perez 2001
/** End of example2.pld source code **/
```

Sequential Circuit Implementation of Programmable Logic

Given: The state diagram for a sequence recognizer (0110) Problem: Implement using one PAL16R4 or GAL16V8

State	Input	Next	Output
ΒA	S	ΒA	found
0 0	0	01	0
0 0	1	0 0	0
01	0	01	0
01	1	10	0
10	0	01	0
10	1	11	0
11	0	0 0	1
11	1	0 0	0



© J. Chris Perez 2001

Insert into K-Maps



DB = s(B'A) + s(BA')

DA = s'B' + BA'

Let: pin 1 = clock, pin 2 = s, pin 19 = found, pin 17 = B, pin 16 = A, and pin 11 = !enable Written in CUPL form: B.d = s&!B&A # s&B&!A; A.d = !s&!B # B&!A; The output equation is:



found = !s&B&A;

EECE143 Lecture 4

Example3.pld

Name	Example3;
Partno	example3.pld;
Date	06/01/98;
Revision	01;
Designer	Bruce Hoeppner;
Company	EECE143;
Assembly	PLD Source Code;
Location	EN365;
Device	G16V8;

```
/*
  Example program to implement the following equations*/
/*
  in CUPL programming language using Equations
                                   * /
/*
                                   * /
/*
  DA = s'B' + BA'
                                   * /
   DB = sB'A + sBA'
/*
                                   * /
/*
   found = s'BA
* /
Allowable Target Device Types:
/*
                                  * /
© J. Chris Perez 2001
```

```
/** Inputs **/
Pin 1 = clock;
Pin 2 = si
                            /* combinational data input */
Pin 11 = !enable;
/** Outputs **/
Pin 19 = found;
Pin 17 = B;
Pin 16 = A;
/** Declarations and Intermediate Variable Definitions **/
/** Equations **/
B.d = s\&!B\&A \# s\&B\&!A;
A.d = !s\&!B # B\&!A;
found = !s\&B\&A;
/** End of example3.pld source code **/
```

EECE143 Lecture 4

Example4.pld

Name	Example4;
Partno	example4.pld;
Date	06/01/98;
Revision	01;
Designer	Bruce Hoeppner;
Company	EECE143;
Assembly	PLD Source Code;
Location	EN365;
Device	G16V8;

```
/** Inputs **/
Pin 1 = clock;
Pin 2 = si
                           /* combinational data input */
Pin 11 = !enable;
/** Outputs **/
Pin [14..15] =[01..0];
Pin 19 = found;
                                  /* combinational output
*/
/** Declarations and Intermediate Variable Definitions **/
field state = [Q1..0];
$define S0 'b'00
$define S1 'b'01
$define S2 'b'10
$define S3 'b'11
```

Experiment #5: Programmable Logic Devices 1

Programmable Logic Devices as replacements/improvements to standard logic ICs.

Goals:

Learn to use programmable logic devices as MSI TTL & CMOS replacements.

Prelab:

- 1. Create PAL source code that implements equations D, E &F from Experiment #1.
- 2. Create PAL source code that implements the sequential circuit for the circuit of Figure 4.1 (state diagram) of Experiment #4.
- 3. Compile all PAL source codes. Bring source code listings (on paper) and floppy disk containing the files to lab. The files should be error free at the beginning of the lab period.

Compiling PLD File

- Use WinCUPL or use another text editor to create your file
- Save your file with the extension .PLD
- Compile using WinCUPL make sure you have no errors.
- Save your .JED file on diskette

EECE143 Lecture 4

Using WinCUPL

Located in Engineering Applications\ EECE\ Wincupl folder

After creating and editing your PLD file:

Open Options|Compiler menu item or type Ctrl+W

WinCupl:	Total Designer _ 🗆 🗙
<u>File E</u> dit <u>V</u> i	ew <mark>Options</mark> <u>R</u> un <u>U</u> tilities <u>W</u> indow <u>H</u> elp
	Compiler Ctrl+W Simulator Ctrl+L ⊻HDL Ctrl+E
🛗 C:\MY M	USIC\VARCOUNT.PLD 📃 🗆 🔀 PROJ 🛌
Name Partno Date Revision Designer Company Assembly Location Device	varcount; varcount; 10/15/98; 01; Guam; BECE143; X000X; X000X; G22V10;
/*************************************	**************************************
•	Line:1 Col:1 INS CAPS NUM SCRL

Programmable Logic Devices I

General	Library	Device
Output Files	Minimization	Optimization
Download ▼ JEDEC/POF/PR ► HL ► ASCII/Hex	General G I Absolut ☐ List ☐ Expand I PDIF ☐ PLA	e Jed Macro
)oc File Options Fuse Plot Equations		М

Output Files	Minimization	Optimization
General	Library	Device
Secure Device		
Deactivate Unused	OR Terms	
Simulate		
One-hot-bit State Ma	achine	
JEDEC name = PLD	name	
View Simulation Res	ults	

Select Download JEDEC/PDF/PRG

Select JEDEC name = PLD name

Output Files	Minimia	ation	Optimization
General	Library		Device
Device Type:		Device	Selection:
iAL		g22v10	
ATTICE MACH MAPL Micron PAL 6 - 14		g22v10 gds22 g22v10cp g22v10cp g24v10 g24v10ma	
Package Type: © DIP © LCC	Dev	ice In File	

Select Device: Either g16v8 for GAL16V8 Or g22v10 for GAL22V10 DIP Package

To compile your program, Select Run|Device Dependent Compile menu item

If you get errors, check the Messages window for error locations and make corrections to your code.



Programming PLD File

- Use Allpro programmer in EN365 or the ChipMaster 88 Programmers in the digital lab or the EECE Open Lab
- Be sure to follow instructions for programming for each programmer.

ChipMaster 6000 Intelligent Universal Programmer



Using the CM6000 Universal Programmer

- 1. Load the device
- 2. Select a device
- 3. Perform Blank Check
- 4. Erase device
- 5. Read jedec file
- 6. Program device
- 7. Verify

EECE143 Lecture 4

Insert Device into the socket

- 1. Make sure the busy light is off.
- 2. Lift the lever of the ZIF socket up.
- 3. Insert the device. Align the device with the bottom of the socket with pin 1 facing the top of the socket.
- 4. Press the lever of the ZIF socket down.



© J. Chris Perez 2001

Startup



Programmable Logic Devices I

EECE143 Lecture 4

- 1. Type ALT-C or Select the Device|Change Device menu item.
- 2. Select the device required: Use the mouse: Click on 'All', EPROM', 'PROM', 'PLD', or 'MPU'

Use the keyboard: Press TAB until the cursor is flashing in the 'Type' box. Use the up and down arrows to go to the appropriate type. Press the space bar to select the type.

3. Enter the part number in the 'Search' box.

Use mouse: Click on 'Search'

Use keyboard: Press TAB until the cursor flashes in the 'Search' box. Type in the part number.

- 4. Use the mouse to select a vendor.
- 5. Select a device and click 'OK. Use the TAB key to skip between various screens, use the arrow keys to move around each screen. Press <ENTER> to select the vendor/device.

Select Device

la			🗾 ок
<u>/</u> endor	<u>D</u> evice		
Atmel Catalyst Cypress Dallas	GAL16LV8/C/Z GAL16LV8D GAL16V8 GAL16V8 GAL16V8	GAL22LV10 GAL22V10/B/C/D GAL26CLV12 GAL26CV12/B/C	Cance HELP
Exel Fujitsu HOLTEK Hyundai(Hynix) ICT Intel ISSI Lattice Lucent(AT&T)	GAL16V8D GAL18V10/B GAL20LV8/C/Z GAL20LV8D GAL20RA10/B GAL20V8 GAL20V8A/B/C/D/Z GAL20VV10	GAL6001B GAL6002B ispLSI1016 *44 ispLSI1016 @44 ispLSI1016(OLD) *44 ispLSI1016E *44 ispLSI1024 @68 ispLSI1024(OLD) @68	Iype All EPRON PROM PLD MPU
Type PLD A	dapter NONE	Man.Code	Dev.Code

Load File into Memory

- 1. Press ALT-L for Load File or select the File|Load File menu item.
- 2. Press <TAB> to move the cursor to the list of files in the default directory
- 3. Select a file with the up and down arrow keys.
- 4. Press <ENTER>.
- 5. Select a file type with the up and down arrow keys. You want to select a jedec (.JED) file
- 6. Press <ENTER> to load the file.

Programmable Logic Devices I

EECE143 Lecture 4

Open ? X	Read Jedec (.JED)
Look jn: Cece143 Ecce143 EXP61.jed	File
File name: EXP61 Files of type: JEDEC Files (*.jed) Cancel	elligent Universal Programmer btions Djagnostics Help AutolD Edit Vector Blank Read Prog. Verify Erase Corrig Option
C Open as read-only	5.0a Serial# /004//85
Name	g22v10 Library DLIB-h-40-1
Partno	Thu May 31 09:04:06 2001
Revision	exp61
Date	exp61
Designer	01
Company	10/15/98
Security OFF	Guam
>>Read file co Device : Latt Adapter : NON Fuse : 589 Check Sum : 000 File : Z:\p Note : For Help, press F1	mplete ! ice GAL22V10/B/C/D NE 2 Vector: 0 DA7D5h erezjc\EECE143\EXP61.jed Current Count: 0 Max Failure: 5 Count: 0000003

Programmable Logic Devices I

EECE143 Lecture 4

Program Device

👹 ChipMaster-6000 Intelligent Universal Programmer		
<u>File</u> Project <u>D</u> evice <u>O</u> ptions Djagnostics <u>H</u> elp		
Save Load Select AutolD Edit Vector Blank Read Prog. Verify Era:	se Comp. Proct.	
Revision U1		
Date 10/15/98		
Designer Guam		
Company		
Bead file complete L		
Read IIIe complete !		
Brank checking		
Verifuing		
No test vectors in the buffer		
Programming complete.		
>>00:00:09.62	-	
۲ () () () () () () () () () (
Device : Lattice GAL22V10/B/C/D Adapter : NONE Pin : 24 Fuse : 5892 Vector : 0 Check Sum : 0000A7D5h	Current Count : 1 Target Count : 100 Current Failure : 0 Max Failure : 5	
File : Z:\perezjc\EECE143\EXP61.jed	OnOff Reset	
Note :	Alarm Config	
I For Help, press F1	Count 0000004	

Blank check the device

📅 ChipMaster-6000 Intelligent Universal Programmer - 🗆 × File Project Device Options Diagnostics Help Read Prog. Verify Erase Select AutoID Edit Vector Coption Save Load Blank Comp Lattice GAL22V10/B/C/D Generated by ChipMaster-6000 on Wed May 30 08:51:52 2001 ChipMaster-6000 at LPT 1 Firmware Version 2.20 Blank checking ... Blank check complete. >>00:00:02.84 Device : Lattice GAL22V10/B/C/D Current Count 1 Target Count : 100 Adapter : NONE Pin: 24 Current Failure : 0 Fuse: 5892 Vector: Max Failure Check Sum : 00000064h **OnOff** Reset File : No File Note : Config Alarm Count 0000003 For Help, press F1

Insert the device in the socket.

- 1. Select the device algorithm.
- 2. Press the ALT-B.

Erase the device

This option is only available for devices that support chip erase.

- 1. Insert the device in the socket.
- 2. Select the device algorithm.
- 3. Press CTRL-F1
- 4. Click OK at the Are you sure you want to erase the device?

Program the device

Before you program the device you have to load the file into memory.

- 1. Insert device into the socket.
- 2. Select the device algorithm.
- Press ALT-P. or Click on the Prog. Button.

ChipMaster-6000 Intelligent Universal Programmer	
<u>File</u> Project <u>D</u> evice <u>D</u> ptions Djagnostics <u>H</u> elp	
Save Load Select AutolD Edit Vector Blank Read Prog. Verify Erase	Comp. Proct. Corific Option
Dete 10/15/98	<u> </u>
Designer Guam	
Company	
Security OFF	
Read file complete !	
Blank checking	
Programming	
Verifying	
No test vectors in the buffer !	
Programming complete.	
>>00:00:09.62	
	F
Device : Lattice GAL22V10/B/C/D	Current Count
Adapter: NUNE Pin: 24	Current Failure
Check Sum : 000047DEb	May Failure
Cileck Sull . 0000Ar DSil	Maxianure
File : Z:\perezjc\EECE143\EXP61.jed	OnOff Reset
Note :	Alarm Config
I For Help, press F1	Count 0000004

Verify the device

- 1. If the device is not inserted, insert it into the socket.
- 2. Select the device algorithm.
 - The CM6000 verifies the device with the contents of the memory buffer. If the correct file is not loaded, load it now.
- 3. Press ALT-V.

After programming, the CM6000 automatically verifies.

You are done programming your chip.

AllPro 88 Programmer



© J. Chris Perez 2001

Using the AllPro 88 Programmer

- Load Device into programmer
- Choose Library and Device from library
- Blank Check Device (optional)
- Read Jed file
- Program Device

Load Device in Programmer

Place chip so Pin 1 faces front of programmer.



Programmable Logic Devices I

Choose Library and Device from library



Programmable Logic Devices I

EECE143 Lecture 4

Choose Library and Device from library



© J. Chris Perez 2001

EECE143 Lecture 4

Choose Library ...

	Library List	
Name	Description	Version
PLDOTH	Atmel FLD Library	- UZ.28C
FLBCYP	Cypress Semiconductor PLD Library	- UZ 38C
PLDEXL	Exel Microelectronics, Inc. PLD Library	- UZ 280
PLDGED	AMI/GOULD Semiconductor PLD Library	- 112 200
PLBICT	International CMOS Technology FLB Library	- 117 780
PLDINT	Intel FLD Library	- 117 390
PLDLAT	Lattice Semiconductor PLD Library	- 112 300
PLDLA2	Lattice High Density and Digital Switches	- U2 680
PLDNS1	Mational Semiconductor PLD Library #1	- 87 780
PLDWSZ	Mational Semiconductor PLD Library #2	- U2.380
PLOPLX	PLX Technology, Inc. PLD Library	- VZ.28C
	essage Center	0.000
Use Cursor to pick type in t ESC abort	Keys and the Enter Key a selection or he code of selection. s without selection.	

Programmable Logic Devices I

EECE143 Lecture 4

Choose Device ...

	Device List
Device Mame	Description
BB RALZEXS	RAL28X8 - Subset of CAL28VUIAR
89 RALZBX4	RAL28X4 - Subset of GAL28VIIA
54 GAL6881	Generic Array Logic
15 GALGOBIE	Generic Array Logic
24 GAL68828	Generic firray Logic
58 GAL18V18/8	Generic Array Logic
17 GALZBRA18/B	Generic Array Logic
241 GALZORA18/B	Generic Array Logic (With - UFS)
18 GALZZU18/B/BQ/C CC	Generic Array Logic
148 GALZZU18/8/80/C	Generic Array Logic (With - UES)
745 GALZZLV10	Supports GALZZLV18C/Z/ZD/D
(Harrison and and and and and and and and and an	Library = PLDLA1
Jessage Cente	Operator Input
ise Cursor Keus and the	Enter Ken
to pick a selection	. or
type in the code of s	election.
TWO allowed a state of	
Loc anorts without se	lection.

Blank Check Device (optional)



Read formatted File ...

Select your .jed file from your floppy diskette

PA	IN MENU
levice select Hank check device Load device Program device Verify device Secure device Functional test device	 Edit data map Read formatted file Write formatted file Utilities menu Nodify operational flags Checksum device image Quit
This function writes the contents of Allpro's data buffer out to the device and burns it's fuses accordingly. A verify operation is performed automatically.	Derator Input

EECE143 Lecture 4

Program Device ...



... and you are done.