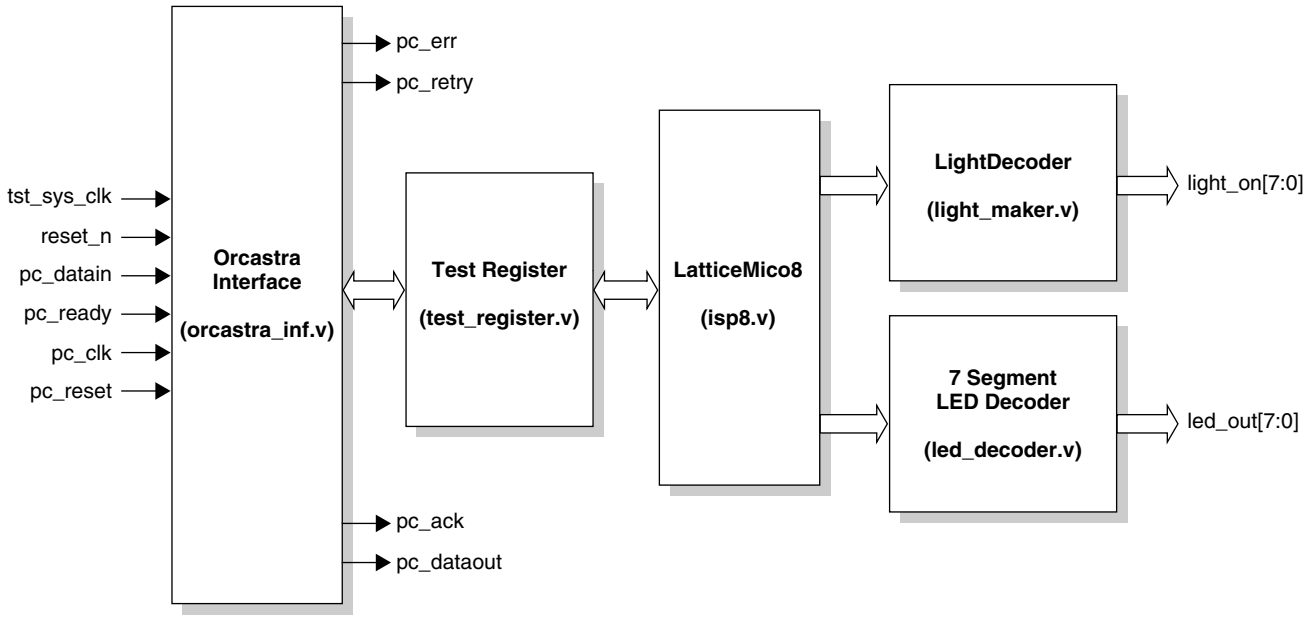
Using the LatticeMico8 Microcontroller with the LatticeXP Evaluation Board

July 2007 Technical Note TN1095

Introduction

The LatticeMico8™ is a flexible 8-bit microcontroller optimized for Lattice's leading edge families. This document describes the operation and use of a demonstration program for the LatticeMico8 on the LatticeXP™ Standard and Advanced Evaluation Boards. The program can demonstrate Fibonacci number or Up/Down counters. Along with the ability to send interrupts, the user can also control and monitor the board via an ORCAstra-style interface.

Figure 1. Block Diagram of LatticeMico8 Test Program



The following are the steps required to run the LatticeMico8 microcontroller core on the LatticeXP Evaluation Board:

1. Convert the assembly file (isp8_demo.s) to a ROM initialization file
2. Synthesize the Verilog source codes
3. Functional and timing simulation
4. Use ispLEVER® for generating JEDEC
5. Set up the board
6. Program the board using ispVM®
7. Control and monitor the board from ORCAstra

Note: This demo requires the user to download and unzip the following files from Lattice web site:

- LatticeMico8.zip
- LatticeMico8_Tools.zip
- MatticeMico8_Demo.zip

Step 1: Converting the Assembly File to a ROM Initialization file

Follow the instructions below to convert the assembly file.

- Copy all the files in the directory `LatticeMico8_Tools` to directory `C:\<your_directory>\LatticeMico8_Demo\test`
- Open a DOS command window
- Change the directory to `C:\<your_directory>\LatticeMico8_Demo\test`
- In the command line, type: `isp8asm_win -o prom_init.v -l -ve ..\asm\isp8_demo.s`
- A new ROM initialization file (prom_init.v) is generated

Step 2: Synthesizing the Verilog Source Codes

After obtaining the `prom_init.v` file, the next step is to synthesize the Verilog source codes. The file name for the memory initialization file remains unchanged as `prom_init.v` since the file `isp8.v` will call the `prom_init.v` directly.

- Open Synplify Synthesis
- Add these files in the following order:
  ```
  ~/LatticeMico8/ver2.4/models/xp/sim/dpram32x8.v
  ~/LatticeMico8/ver2.4/models/xp/sim/prom.v
  ~/LatticeMico8/ver2.4/models/xp/sim/spram32x8.v
  ~/LatticeMico8/ver2.4/models/xp/sim/spram16x8.v
  ~/LatticeMico8/ver2.4/models/xp/sim/spram16x9.v
  ~/LatticeMico8/ver2.4/models/xp/syn/xp.v
  ~/LatticeMico8/source/isp8.v
  ~/LatticeMico8_Demo/source/test_register/test_register.v
  ~/LatticeMico8_Demo/source/orcastra_inf/orcastra_inf.v
  ~/LatticeMico8_Demo/source/led_decoder/led_decoder.v
  ~/LatticeMico8_Demo/source/light_maker/light_maker.v
  ~/LatticeMico8_Demo/source/top/isp8_top_system.v
  ```
- Add the following source directory in the **Include** path:
  ```
  ~/LatticeMico8_Demo/source/config3
  ~/LatticeMico8_Demo/test
  ```
- In the **Implementation Options** dialog box set the following options under **Device** tab:
  - Lattice Technology = **LATTICE XP**
  - Part = **LFXP10C**
  - Speed = **-5**
  - Package = **F256C (Standard Board) / F388C (Advanced Board)**
- Click **Run**

Note: The above procedure is for synthesizing LatticeMico8 and the demo source codes. Users can synthesize LatticeMico8 (isp8.v) by itself from the LatticeMico8 directory by using the Tcl script located in the Synthesis directory. Before using this Tcl script, please modify the PROJPATH variable in the Tcl script to resemble the current project directory.

Step 3: Functional and Timing Simulation

- Open ModelSim Simulator
- From the menu bar, select **File -> Change Directory**, and set the folder to `~/LatticeMico8_Demo/simulation`

- For functional simulation, from the menu bar, select **Tools -> TCL -> Execute Macro**, click on the **Script** directory and select `demo_func_sim.do`
- For timing simulation, the .vo and .sdf files must already be generated by ispLEVER Project Navigator and located at `~/LatticeMico8_Demo/par/top`. From the menu bar, select **Tools -> TCL -> Execute Macro**, click on the **Script** directory and select `demo_time_sim.do`.

Note: The `demo_time_sim.do` may need to be modified to match the .vo and .sdf file names.

Step 4: Using ispLEVER for Generating JEDEC

- Open ispLEVER Project Navigator
- To set the target device, in the **Device Selector** dialog box, select the following:
  - Family = **LatticeXP**
  - Device = **LFXP10C or LFXP10E**
  - Speed Grace = **-5**
  - Package Type = **FPBGA256 (Standard Board) / FPBGA388 (Advanced Board)**
  - Operating Conditions = **Commercial**
- From the Project Navigator, double click on **Pre-Map Preference Editor** to set the pin types and pin locations

Table 1. Pin Assignment for LatticeXP Advanced & Standard Boards

Signal Name	Pin Location		I/O Type
	Standard Board	Advanced Board	
tst_sys_clk	A7	A10	LVCMOS33
reset_n	L12	H3	LVCMOS33
light_on_0	D5	H1	LVCMOS33
light_on_1	A3	B16	LVCMOS33
light_on_2	B3	B18	LVCMOS33
light_on_3	B2	C18	LVCMOS33
light_on_4	A2	C19	LVCMOS33
light_on_5	B1	C20	LVCMOS33
light_on_6	F5	W16	LVCMOS33
light_on_7	C5	A16	LVCMOS33
led_out_0	J14	D2	LVCMOS33
led_out_1	K16	F3	LVCMOS33
led_out_2	K15	F1	LVCMOS33
led_out_3	L14	E2	LVCMOS33
led_out_4	L13	E1	LVCMOS33
led_out_5	K13	D1	LVCMOS33
led_out_6	N16	F2	LVCMOS33
pc_err	H16	B4	LVCMOS33
pc_retry	H12	C4	LVCMOS33
pc_ack	F13	A18	LVCMOS33
pc_dataout	G14	C3	LVCMOS33
pc_clk	C15	A3	LVCMOS33
pc_ready	F16	B3	LVCMOS33
pc_datain	E14	A2	LVCMOS33
pc_reset	C16	A4	LVCMOS33

- Double click on **Generate Timing Simulation Files**. This process will generate .vo and .sdf files that can be used for timing simulation
- Double click on **Generate Data File** (JEDEC) to get the .jed file that will be used later with ispVM System to program the evaluation board

Step 5: Setting up the Board

For the LatticeXP Standard Evaluation Board:

- Set the VCCIO pins of all banks of eight jumpers (JP1, JP2, JP12, JP11, JP13, JP14, JP10, and JP3) to 3.3V
- The VCC core is set and defaulted to 1.2 V. Don't change the setting of the VCC core, but keep it as its original setting at 1.2V
- Use a 1.2 power supply and a 3.3V power supply that can provide adequate current to the board
- To interface with ORCAstra software, connect the parallel port pins as mentioned in Table 2

Note: Unlike the LatticeXP Advanced Board, the Standard board does not have seven-segment display on the board. However, user can easily get a regular seven-segment display such as HDSP7501 and connect it with the pins mentioned in Table 1.

For the LatticeXP Advanced Evaluation Board:

- Set the V_{CCIO} pins of all banks to 3.3 V by putting all eight jumpers J2-J4 to J1
- Use a 5V power supply that can provide adequate current to the board
- To interface with ORCAstra software, connect the following parallel port pins to the board pins

Table 2. ORCAstra Interface Connections

Parallel Port Pins	Signal Name	Direction	Board/Pin Locations	
			Standard Board	Advanced Board
2	(tied to 15)			
6	PC_Data_In	PC Out	E14	A2
7	PC_Clk	PC Out	C15	A3
8	PC_Reset	PC Out	C16	A4
9	PC_Ready	PC Out	F16	B3
10	PC_Err	PC In	H16	B4
11	PC_Data_Out	PC In	G14	C3
12	PC_Retry	PC In	H12	C4
13	PC_Ack	PC In	F13	A18
15	(tied to 2)	—	—	—
18-25	GND	—	—	—

Step 6: Programming the Board Using ispVM System

The following are the steps for programming the LatticeXP Evaluation Board:

- Connect your PC to the target board using a Lattice ispDOWNLOAD® Cable
- Launch ispVM System software
- Click **File -> New**
- Click **Edit -> Add Device**

- Click **Select** and select **LatticeXP-ES** for Device Family, **LFXP10E_ES** for Device, **256_ball fpBGA (Standard board) / 388_ball fpBGA (Advanced board)** for Package and then click **OK**
- Click **Browse** under the **Data File** field and point to the JEDEC file previously created by the ispLEVER Project Navigator
- Highlight **Flash Programming Mode** from the Device Access Options field selection list
- Highlight **FLASH Erase, Program, Verify** from the Operation field selection list
- Click on the **OK** button
- Click **GO** to program to LatticeXP device

Step 7: Controlling and Monitoring the Evaluation Board from ORCAstra

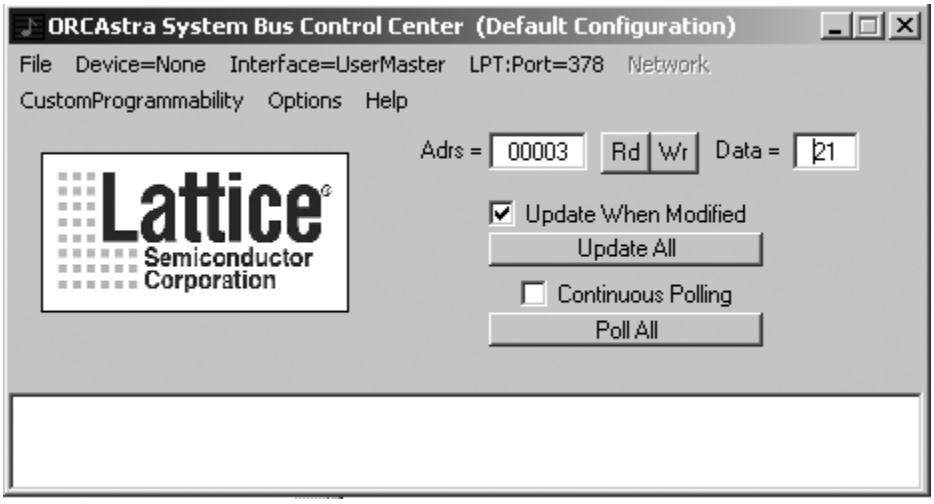
With ORCAstra, users can control and monitor the evaluation board by writing and reading the registers in the test_register module inside the LatticeXP device.

Download the latest ORCAstra software from the Lattice web site at the following address: www.latticesemi.com/products/designsoftware/orcastra.cfm.

After downloading ORCAstra, follow these steps to set it up with the board environment:

- Launch ORCAstra
- On menu bar, click on **Device** and select **Zero No Device**
- In the same menu bar, click on **Interface** and select **FPGA User Master via Parallel Port**
- From the menu bar, click on **LPT:Port** and select **2 0x378**

Figure 2. ORCAstra System Bus Control GUI



The demo files also come with an additional GUI specially created to support LatticeMico8. This GUI enables users to view the value of all the 32 registers together by clicking on the **Read All Registers** button.

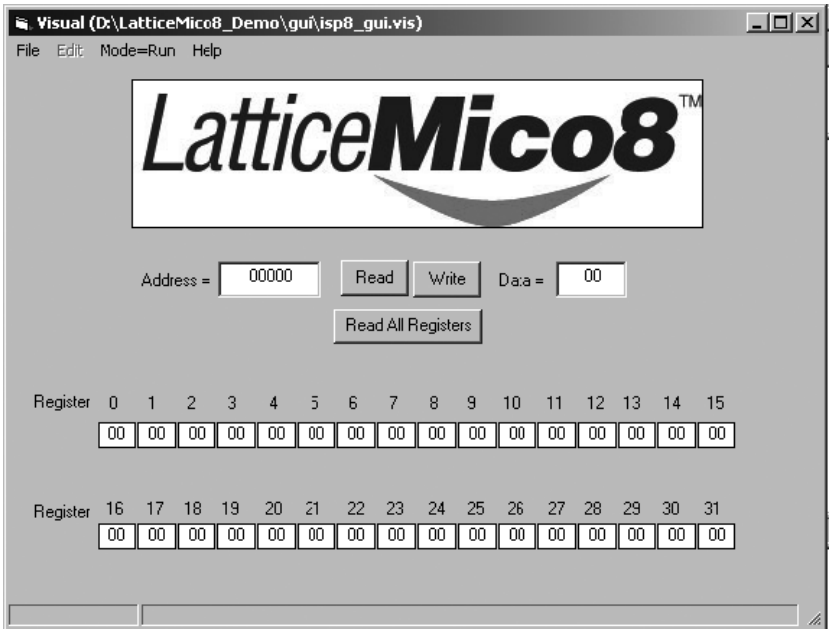
To open the LatticeMico8 GUI from ORCAstra software (v112 or later):

- From the menu bar, select **Custom Programmability --> Visual Window**
- A new smaller window will appear titled Custom Visual Interface
- Select **File --> Open** from its menu bar

- Browse and select **isp8_gui.vis** file located in LatticeMico8_Demo/gui directory

A new window with LatticeMico8 logo will appear.

Figure 3. LatticeMico8 GUI Derived From ORCAstra



The assembly file that was converted to a ROM initialization file in Step 1 contains a program that will determine the Fibonacci number and the Up/down counter based on the command that is given from ORCAstra.

Table 3 lists the addresses and data that are accepted by the program based on the assembly code (isp8_demo.s).

Table 3. List of Demo Commands

Adrs (Address)	Data	Type of Function	Speed
00003	11	Fibonacci	1 (Very Slow)
00003	12	Fibonacci	2 (Slow)
00003	13	Fibonacci	3 (Medium)
00003	14	Fibonacci	4 (Fast)
00003	21	Up/Down Counter	1 (Very Slow)
00003	22	Up/Down Counter	2 (Slow)
00003	23	Up/Down Counter	3 (Medium)
00003	24	Up/Down Counter	4 (Fast)
00002	01	Sending Interrupt	—

The program will act accordingly based on the address and data written by the user from ORCAstra. For example, by typing **Adrs = 00003** and **Data = 11** in the ORCAstra GUI and pressing the **WR** button, the Fibonacci function will be run. The program will generate the first 12 Fibonacci numbers and store the results in Register 0 through Register 12. The value of one of these registers can be read by typing its address. For example, to read Register 0, type **00020** in the **Adrs** field and press the **Rd** button. The value will be shown in the **Data** field.

Below is the list of registers and their corresponding addresses.

Register Name	Reg0	Reg1	Reg2	Reg3	Reg4	Reg5	Reg6	Reg7
Address (hex)	00020	00021	00022	00023	00024	00025	00026	00027

Register Name	Reg8	Reg9	Reg10	Reg11	Reg12	Reg13	Reg14	Reg15
Address (hex)	00028	00029	0002a	0002b	0002c	0002d	0002e	0002f

Register Name	Reg16	Reg17	Reg18	Reg19	Reg20	Reg21	Reg22	Reg23
Address (hex)	00030	00031	00032	00033	00034	00035	00036	00037

Register Name	Reg24	Reg25	Reg26	Reg27	Reg28	Reg29	Reg30	Reg31
Address (hex)	00038	00039	0003a	0003b	0003c	0003d	0003e	0003f

Technical Support Assistance

Hotline: 1-800-LATTICE (North America)
+1-503-268-8001 (Outside North America)
e-mail: techsupport@latticesemi.com
Internet: www.latticesemi.com

Revision History

Date	Version	Change Summary
July 2005	01.0	Initial release.
August 2006	01.2	Updated to include LatticeXP Standard Evaluation Board.
July 2007	01.3	Updated block diagram (changed pc_data_in signal to read pc_datain).
		Updated file names in Step 2: Synthesizing theVerilog Source Codes.
		Clarified some of the menu selections in Step 3: Functional and Timing Simulation.
		Updated information on how to download ORCAstra software in Step 7: Controlling and Monitoring the Evaluation Board from ORCAstra.