A Low-Cost PXE Implementation
Using The LatticeXP FPGA

A Lattice Semiconductor White Paper

April 2005

Lattice Semiconductor
5555 Northeast Moore Ct.
Hillsboro, Oregon 97124 USA
Telephone: (503) 268-8000
www.latticesemi.com

Low Cost PXE Implementation Uses Logic Overlay

What is PXE?

Pre-boot eXecution Enviroment (PXE) is an open standard that can be used by any client system to download its application code from a server. This standard is derived from the well-established standard for PCs called "Wired for Management" (WfM) as defined by Intel and Microsoft. PXE boots the client system from the network by transferring a "boot image file" from a server. This file can be the operating system for the client, or the application code that performs required tasks. Since PXE is not operating system-specific, the image file can load any OS.

The advantages of starting the system through PXE include:

- Easy software upgrades/ maintenance
- Security
- Software is always downloaded into a fully functional system, resulting in fault propagation reduction through the system backplane
- Improved hardware performance by using SDRAM, DDR, etc. for program storage instead of slow non-volatile memory
- Reduced overall cost of ownership
- Since PXE is derived from WfM, embedded systems can directly communicate with commercially available servers

Because PXE is an open standard, many PXE compliant software products have been developed for a number of CPUs . This white paper discusses the implementation of PXE functionality in a CompactPCI system (cPCI) with a DSP CPU. Earlier versions of this system relied on a change of the CPU card for the DSP algorithm update. Now, implementation of PXE support enables an easy DSP algorithm upgrade and reduced system downtime.

cPCI DSP card implementing PXE

Figure 1 shows the block diagram of a cPCI DSP board supporting PXE functionality. This is a dual CPU card with a COM CPU for communication over

Ethernet during the PXE mode (debugged PXE code existed for this CPU and was used for reduced time to market reasons) and the DSP CPU for the card's normal operation. The latest version of the DSP algorithm is stored on the server connected to the cPCI DSP card through the front panel RJ45 connector.

Circuit Description

The COM CPU executes the code in the Flash memory and communicates with the server through the Ethernet MAC IC. The DSP CPU executes the code in the SDRAM and communicates with the backplane through the PCI Bridge IC.

The ispMACH 4256 CPLD implements all high-speed bus interface and state machine logic. The FPGA provides SDRAM interface to COM CPU and DSP CPU, PXE data path and DSP application data path logic, and PCI Bridge and Ethernet controller bus interface logic.

Upon power-on, the COM CPU executes the program stored in the Flash memory to download the DSP application software from the network into the SDRAM memory. After the verification of the code in the SDRAM, the DSP CPU is activated to perform the normal function by executing the code stored in the SDRAM.
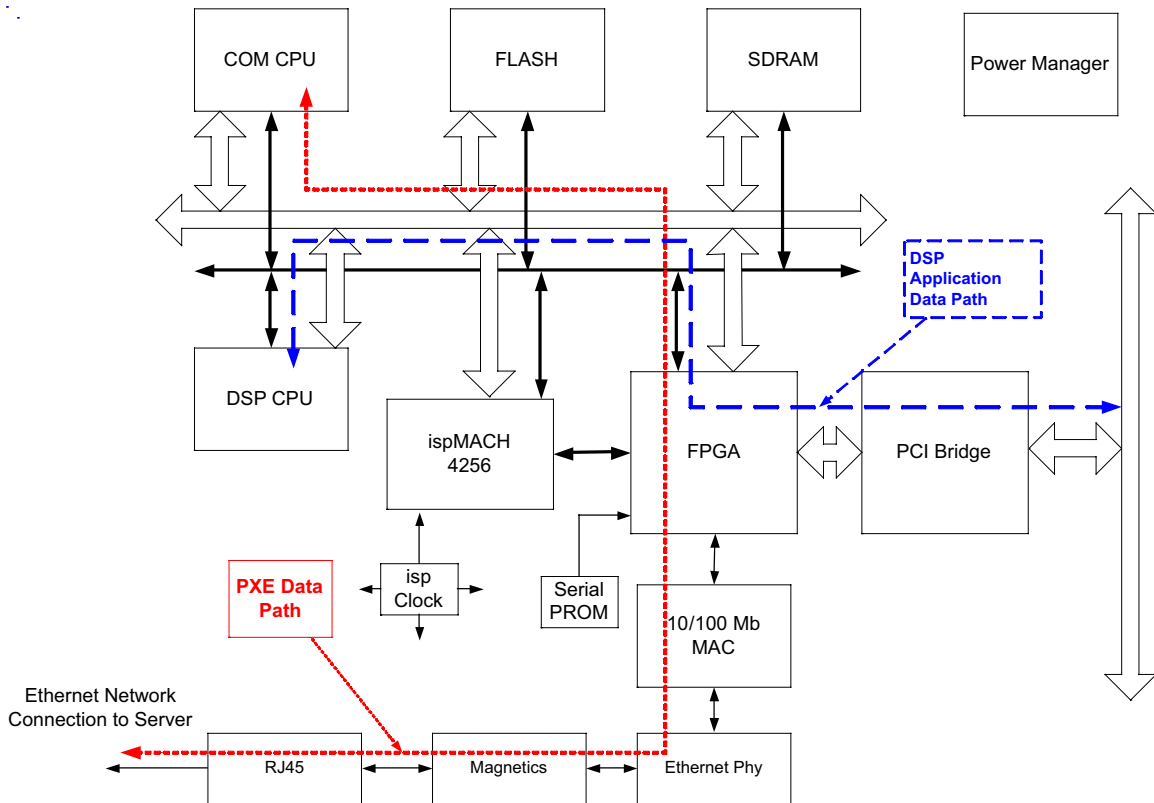
Figure 1- PXE implementation in a cPCI card

Redesigning the cPCI Board Using the Logic Overlay Method

This approach takes advantage of the fact that only one of the data paths is active at any given time. By overlaying one data path logic over the other, the required size of the FPGA is halved. In addition, the Ethernet MAC and PCI Bridge are also overlaid, resulting in fewer ICs. One of the restrictions imposed on the logic overlay method was that the overlaid logic had to be active in microseconds. This was made possible by using the LatticeXP FPGA. .

The LatticeXP FPGA Family

LatticeXP devices add blocks of non-volatile memory to an optimized FPGA architecture to provide a low-cost, non-volatile solution. The devices are manufactured on an economical 130nm Flash process, which provides good intrinsic soft error rate immunity.

A Low-Cost PXE Implementation Using the LatticeXP FPGA
A Lattice Semiconductor White Paper

LatticeXP Architecture

At the core of the LatticeXP devices are Programmable Function Units (PFUs) that allow the implementation of logic and, for 25% of the blocks, distributed memory. Logic is implemented using four input look-up tables (LUT-4s) and register pairs, the *de facto* standard for the FPGA industry and one well understood by system designers and logic synthesis tool suppliers.

 Distributed memory provides an efficient method for designers to implement small blocks of scratch pad memory. Rows of sysMEM Embedded Block RAM (EBR) provide 9kb blocks of memory for use implementing larger memory blocks. At the end of the rows of sysMEM memories are sysCLOCK PLLs that allow clocks to be aligned for improved set-up and clock-to-out times and new clocks to be synthesized.

Around the periphery of the device are sysIO interfaces that allow the device to interconnect to a variety of I/O standards, including LVCMOS, PCI, LVTTL, LVDS, SSTL and HSTL. Additionally, LVPECL, BLVDS and RSDS interface standards can be emulated with the addition of external resistors. DLL calibrated DQS delay blocks, DDR registers and clock transfer circuitry allow the easy implementation of high performance DDR memory interfaces up to 333Mbps. Generic DDR interfaces up to 700Mbps can also be implemented with the device. The various functional blocks of the architecture are interconnected with a routing fabric that provides an optimum balance among speed, flexibility and cost.
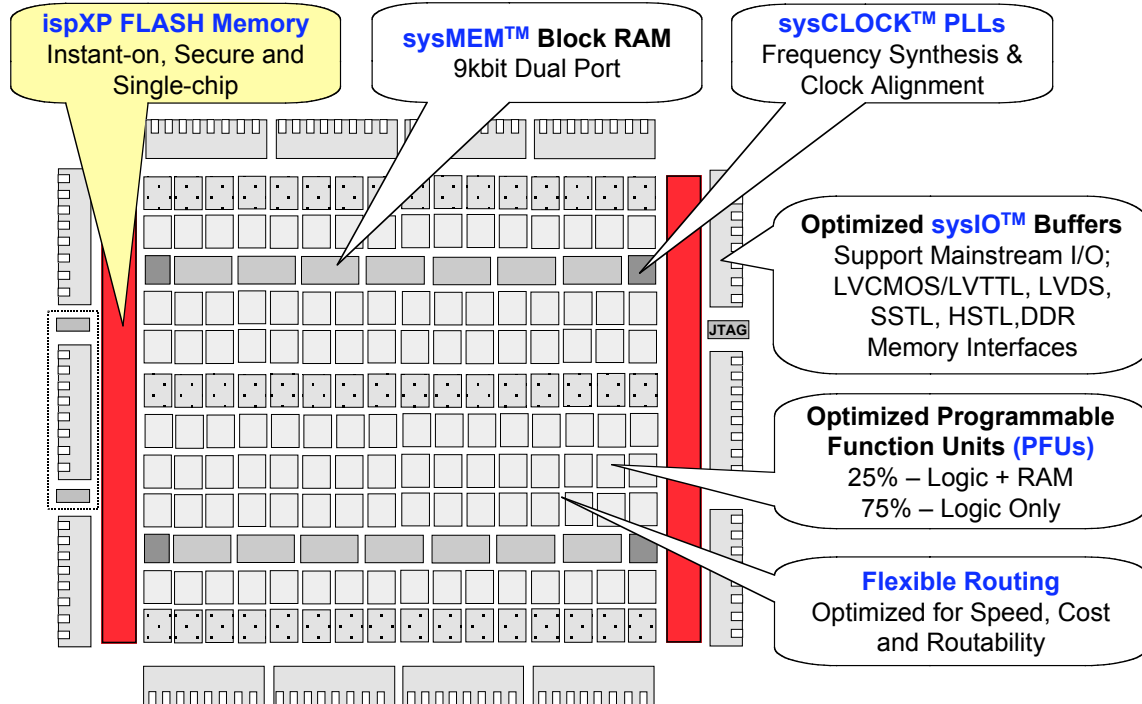
ispXP FLASH Memory
Instant-on, Secure and
Single-chip

sysMEM™ Block RAM
9kbit Dual Port

sysCLOCK™ PLLs
Frequency Synthesis &
Clock Alignment

JTAG

Optimized sysIO™ Buffers
Support Mainstream I/O;
LVCMOS/LVTTL, LVDS,
SSTL, HSTL,DDR
Memory Interfaces

**Optimized Programmable
Function Units (PFUs)**
25% – Logic + RAM
75% – Logic Only

Flexible Routing
Optimized for Speed, Cost
and Routability

Figure 2 – LatticeXP Architecture

Non-Volatile and SRAM Memory

SRAM configuration bits control the operation of the LatticeXP devices. At
power-up, these bits are loaded via the on-chip, non-volatile Flash memory,
resulting in logic availability in less than 1ms after power good. During device
operation the SRAM may be reconfigured from the Flash by toggling a pin or
issuing the correct commands through the device configuration ports. Both the
Flash memory and the SRAM memory can be reprogrammed/reconfigured via
either a JTAG port or a sysCONFIG port (microprocessor style interface).
Configuration of the SRAM via these ports takes between tens and hundreds of
milliseconds, depending on the chosen interface. The Flash memory can be
programmed in as little as 2 seconds. Figure 3 shows the operation of the
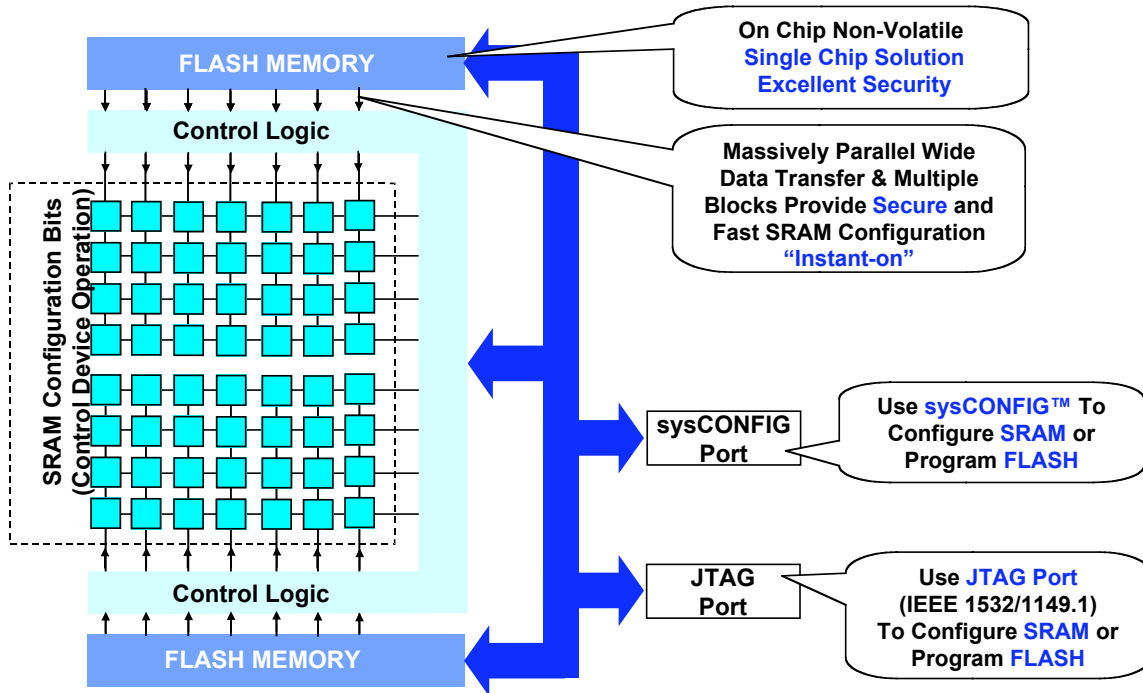different memories within the LatticeXP devices.

Figure 3 – Configuration Memories in the LatticeXP

LatticeXP Device Family

The LatticeXP family supports logic capacities between 3K and 20K Look-up Tables (LUTs) and 62 to 340 I/O in a variety of popular package options. All devices in the same package are pinout compatible, allowing device capacity to be adjusted to match changing design requirements without re-laying out the circuit board. The devices are available in two power supply options, one supporting 3.3/2.5/1.8-volt operation and the other supporting 1.2-volt operation. The higher voltage version allows designers to benefit from the lower power, higher speed and lower cost of 130nm technology without migrating to lower power supply voltages. Table 1 shows the different devices available in the LatticeXP family.

Device	XP3	XP6	XP10	XP15	XP20
LUTs (K)	3.1	5.8	9.7	15.4	19.7
sysMEM Blocks	6	10	24	32	46
sysMEM (Kbits)	54	90	216	288	414
Distributed RAM (Kbits)	12	23	39	61	79
Voltage (V)	1.2/1.8/2.5/3.3V				
PLLs	2	2	4	4	4
Package I/O Combinations					
100-pin TQFP (14x14mm)	62				
144-pin TQFP (20x20mm)	100	100			
208-pin PQFP (28x28mm)	136	142			
256-ball fpBGA (17x17mm)		188	188	188	188
388-ball fpBGA (23x23mm)			244	268	268
484-ball fpBGA (23x23mm)				300	340

Table 1 – LatticeXP Family Members

Description of cPCI circuit using Logic Overlay

The following application reduces the FPGA size by overlaying the logic required by the PXE data path with the logic required for the DSP application data path. The chip count is also reduced because the LatticeXP FPGA (specifically, the LatticeXP10 device) overlays the Ethernet MAC with a PCI interface core.

The PXE mode logic for the LatticeXP10 is stored in the Flash memory and the normal operation mode logic is stored in the LatticeXP10's on-chip Flash configuration memory.

Upon power-on, the ispMACH 4032 CPLD loads the LatticeXP10 with PXE mode logic from the Flash memory using the sysCONFIG port and releases the reset signal on the COM-CPU. The PXE mode logic enables the COM CPU to execute the PXE program from the Flash and load the application image into the SDRAM. Additionally, the PXE mode logic also provides the hardware for performing Power on Self Test, conducted by the COM CPU.

After the DSP algorithm is transferred from the server to the SDRAM, the COM CPU issues a command to the LatticeXP10 to load the configuration required for DSP CPU, called the normal operation mode logic. The PXE mode logic completes one full SDRAM refresh cycle in a burst mode and initiates reconfiguration from on-chip Flash configuration memory through the ispMACH 4032 CPLD.

The circuit passes through three modes of operation and is essentially determined by the LatticeXP10 operation. The modes are:

- ➢ PXE Mode
- ➢ Transitioning from PXE to Normal Operation Mode
- ➢ Normal Operation Mode

PXE Mode
Upon power-on, the LFX is loaded form the Flash memory using the ispMACH 4032 CPLD through the sysCONFIG port of LatticeXP10.

The LatticeXP10 logic in PXE mode (Figure 4):
- ➢ Enables COM CPU to execute code from Flash
- ➢ Ethernet MAC with FIFO for network communication
- ➢ Board self-test assist hardware
- ➢ Generate clock for circuit board operation
- ➢ Drives SDRAM and maps it to COM CPU address space
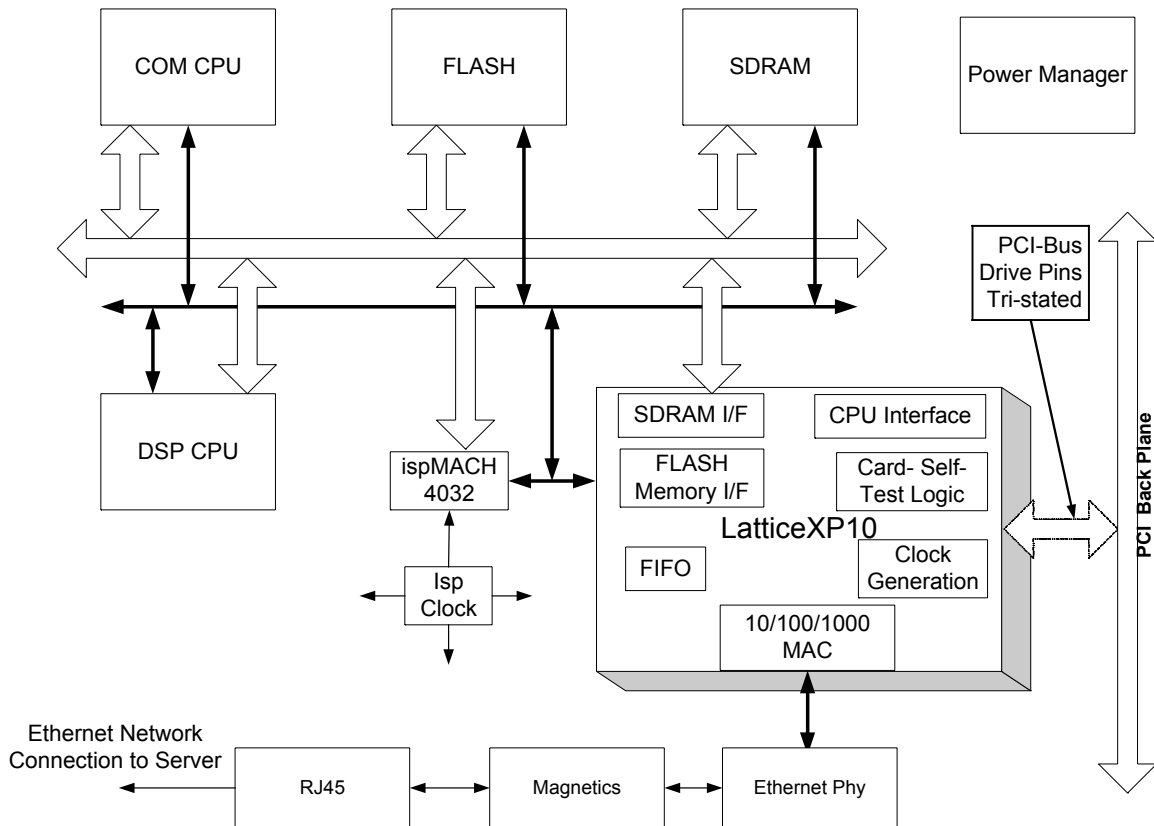- ➢ PCI interface is Tri-stated
- ➢ COM CPU Bus interface

Figure 4 – LatticeXP10 configured with PXE Mode Logic

During this mode, the COM CPU executes the Power On Self Test followed by the downloading of the DSP algorithm code stored in the server into the SDRAM.

The details of using the ispMACH 4032 to load code from Flash memory into the LatticeXP is described in a Technical note (TN1026), ispXP configuration usage guidelines, that can be found on the Lattice semiconductor website, http://www.latticesemi.com/

Transitioning from PXE Mode to Normal Operation Mode

After the application software is loaded into the SDRAM from the network, the COM CPU initiates the process to reconfigure the LatticeXP10 to the normal operation logic. The LatticeXP10 holds the COM CPU, completes the SDRAM refresh cycle and signals the ispMACH 4032 CPLD to initiate the command to load the Flash configuration memory. The ispMACH 4032 CPLD will be the master, maintaining control signals during this transitioning phase. The entire

process of reconfiguring the LatticeXP10 is completed in microseconds, which is well within the SDRAM's refresh time limit.

Normal Operation Mode

During the normal operating mode, the LatticeXP10 will be configured to perform the following functions (Figure 5):

- ➢ Enable DSP CPU execute code from SDRAM
- ➢ PCI interface for driving backplane
- ➢ IPMI (Intelligent Platform Management Interface)
- ➢ Generate clock for circuit board operation
- ➢ Interfaces to SDRAM
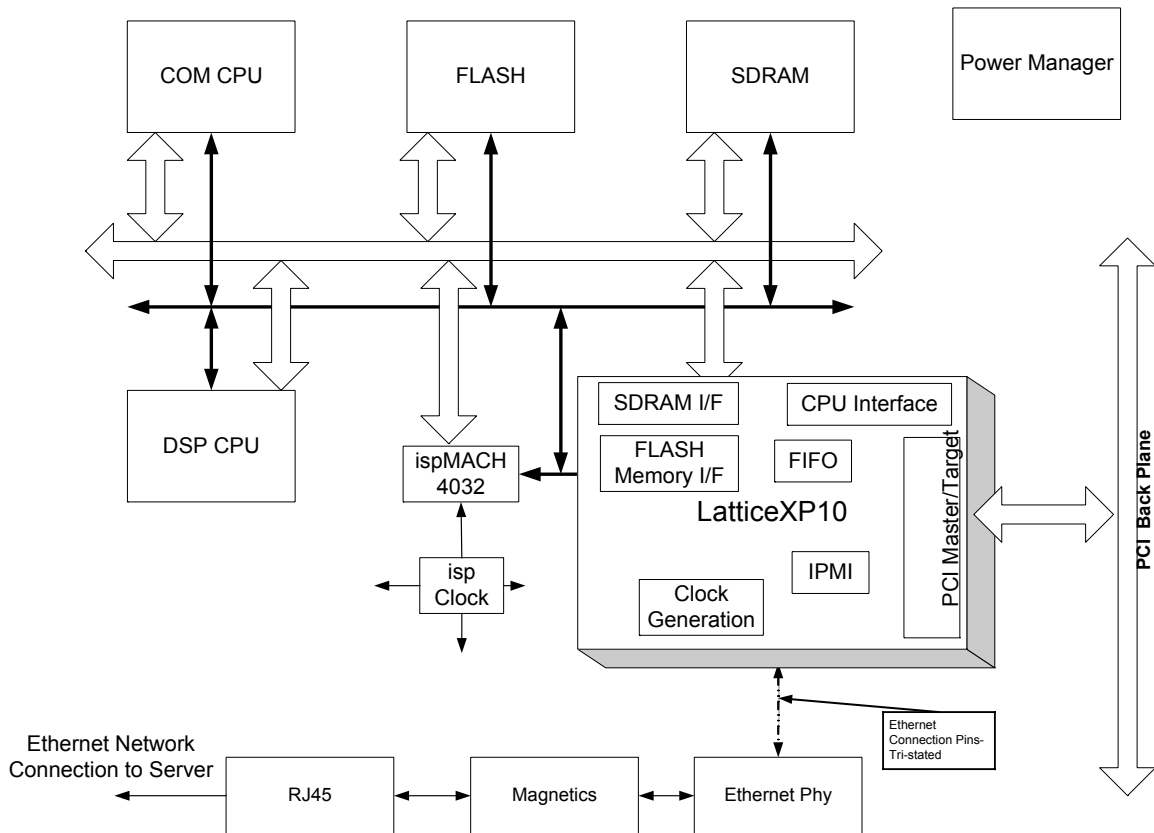- ➢ DSP CPU Bus interface

Figure 5 - Normal operating mode

Advantages of using ispXP technology

Designers have traditionally resorted to memory overlay techniques to reduce hardware memory requirements. Now, with ispXP technology, the same technique can be used to overlay logic and reduce the size of the FPGA required for a given application.

The LatticeXP10 FPGA not only provides a convenient mechanism for storing two configurations, but also provides a mechanism for reconfiguration in microseconds, solving many hardware related time critical requirements (e.g., SDRAM refreshing).

The use of the LatticeXP10 in this design not only reduced the number of ICs used, but also enabled additional functionality: (power-on-self-test with guaranteed disconnection from the backplane), as well as increased card reliability and reduced overall cost.

###