A 155 Mbps ATM Network Interface Controller Using Actel's New 3200DX FPGAs

Given that the asynchronous transmission mode (ATM) peripheral market is highly competitive and time-to-market is critical, logic designers must meet shrinking design cycles. Until recently, designers had to rely on gate arrays to deliver the performance and features required for network peripherals. The current generation of FPGAs, however, allows the designer to achieve the performance and capacity required for ATM applications and to meet the key time-to-market goals of these fast-evolving applications.

Introduction

An ATM network interface card (NIC) requires high-speed system logic functions such as DMA controllers, memory (DRAM, SDRAM, VRAM) controllers, FIFOs, and bus interfaces. This paper describes a 155 Mbps ATM NIC controller that can interface to any standard microprocessor bus. This design, in conjunction with a microprocessor controller and a standard SRAM device, is capable of implementing the segmentation and reassembly (SAR) functions for AAL3/4 communications. This includes the control and interface to the ATM physical layer interface (Utopia), a 1 MB dual-port SRAM, and a host bus. The host bus was chosen as a generic bus for simplicity; however, the discussion applies to any of the popular busing architectures such as PCI, VME, EISA, and SBUS.

Controller Architecture

A block diagram of an ATM NIC is shown in Figure 1. The Utopia interface provides a standard data-path protocol for interfacing to physical layer components. A received cell is written into a 64 x 8 FIFO, which can store an entire 53-byte ATM cell, and is clocked at the 25 MHz Utopia clock rate.

The received ATM cell is read from the FIFO at 66 MHz and the cell is written into system memory (dual-port RAM). The dual-port RAM temporarily stores received data packets while they are being reassembled. The RAM space can be organized such that cells with different virtual channel identifiers (VCI) and virtual path identifiers (VPI) addresses are stored in separate memory areas for reassembly. Thus, the RAM can be accessed based upon the VCI/VPI fields. The addressable contents contain pointers to memory space where reassembly takes place. The DRAM controller reads and writes data to the dual-port RAM device.

The first 5 bytes of an ATM cell contain overhead information, such as destination address, and an error control byte. When reading the ATM cell from the receive FIFO, the first 5 bytes of the cell are directed to the header error control (HEC) block. The HEC is an 8-bit cyclic redundancy code (CRC), which detects transmission errors in the ATM cell header.

ATM Cell Definition

Figure 2 shows the definition of the ATM cell. The cell is composed of 53 bytes, with 5 header bytes and 48 data bytes. The header contains 4 bytes of addressing and 1 byte of error checking information. The addressing information is one of two different types, the user-network interface (UNI) and the network node interface (NNI).

The UNI has one significant difference from the NNI. The first byte of the UNI contains a generic flow control (GFC) field, whereas the GFC is not required in the NNI. The GFC designator is used only for traffic traversing the UNI interface, where the operation, administration, and management (OAM) functions are required.

The remainder of the UNI and NNI fields contain VCIs and VPIs. VCIs and VPIs are used to establish connections within the network and can be thought of as the user's connection address. Unlike Ethernet, these addresses are assigned on a connection-on-demand basis, not a fixed number for each user.

The HEC field is an error check field and can also correct for single bit errors. It is calculated only on the header; the 48-byte payload is not included.

Receive FIFO Design

The receive FIFO is implemented in a dual-port SRAM internal to the FPGA. Data from the Utopia interface is written into the FIFO until the entire cell is available. The FIFO is implemented in the dual-port SRAM and uses the synchronous operation mode. The controller provides the data to the FIFO and activates the write enable signal. On the next write clock, the data is written into the FIFO.

Data can be read out of the FIFO independently from the read port because of the dual-port nature of the SRAM. The read port can operate at the required 66 MHz to achieve the required bandwidth on the 32-bit bus.

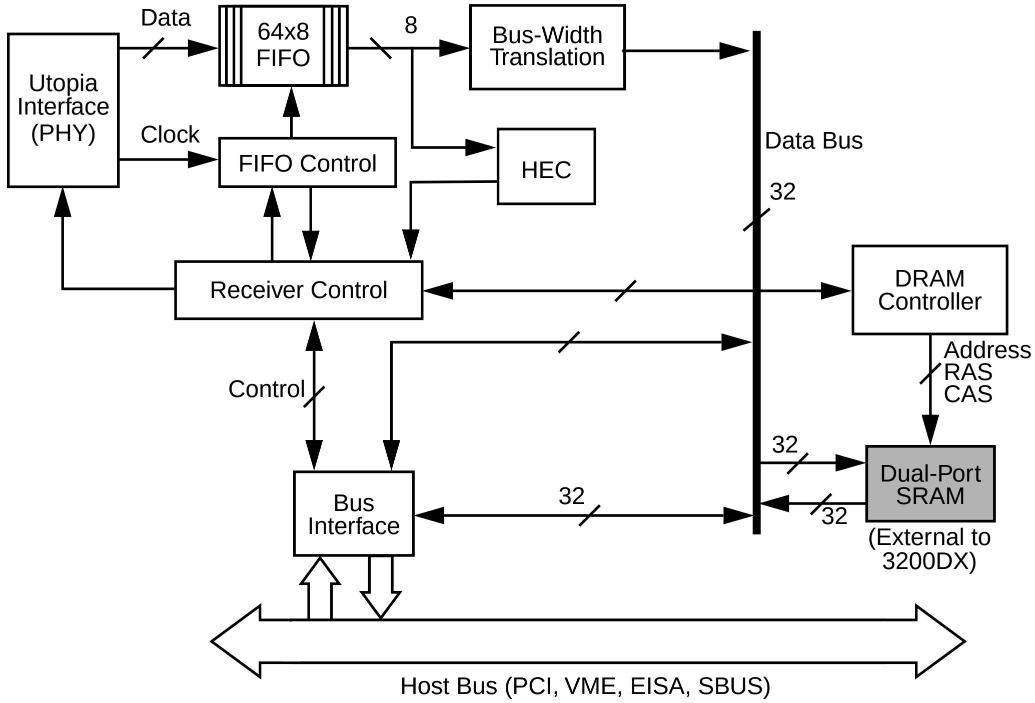

Figure 1 • *Controller Architecture*

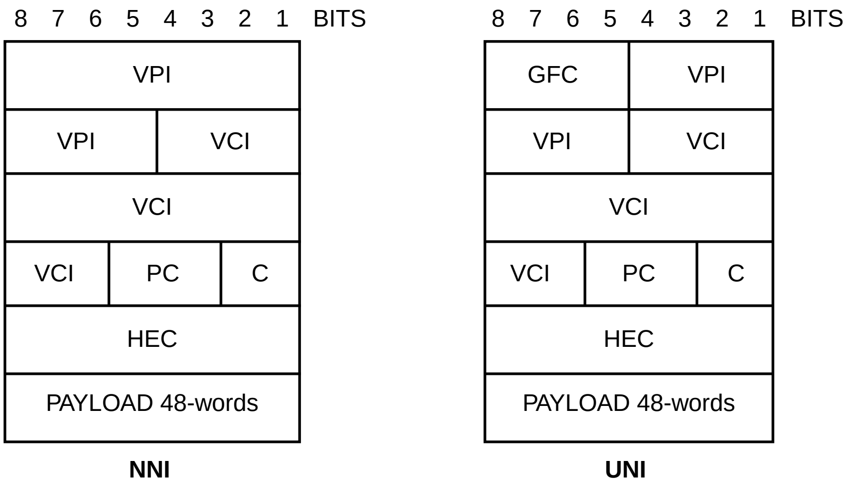

Figure 2 • *ATM Cell Definition*

A variety of FIFO can be automatically generated by the ACTgen Macro Builder tool of the Actel Designer Series software. The user can specify the size, features, and other key functions of the FIFO and ACTgen automatically constructs the requested FIFO “flavor.”

DRAM Controller

The DRAM control circuitry is shown in Figure 3. The key blocks of the design are the address multiplexer, the refresh counters, and the RAS/CAS select logic. Timing on RAS and CAS is preserved by using bidirectional buffers on the RAS and CAS select lines, a common technique when interfacing to DRAMs.

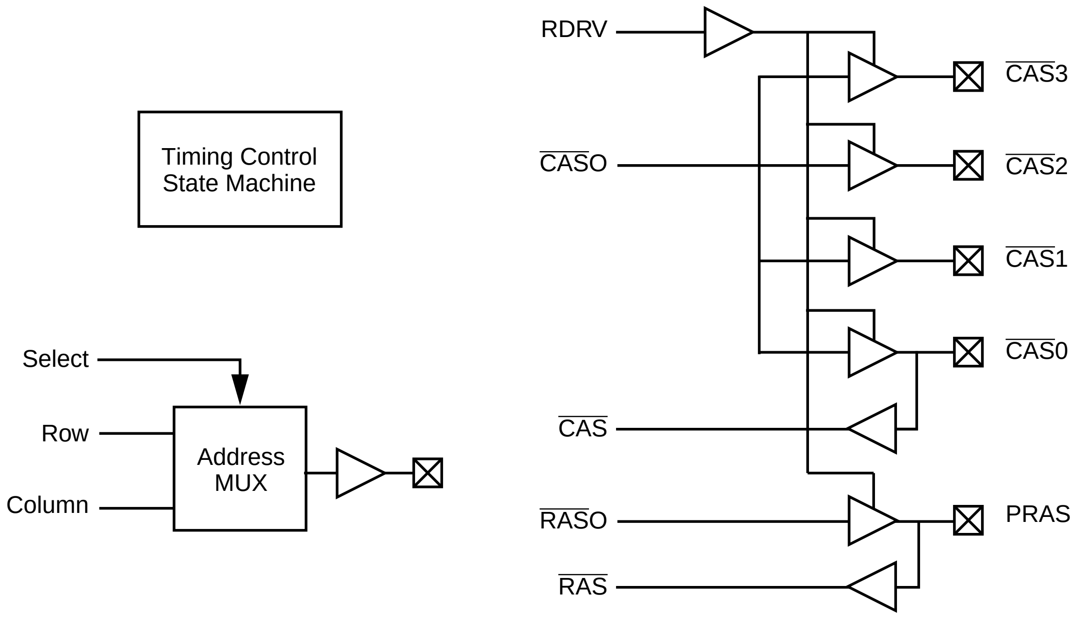

Figure 3 **•** *DRAM Controller Design*

A simple DRAM controller is available from Actel as a predefined function for the 3200DX family.

Packet Assembly

Packet assembly is accomplished in the DRAM. The typical contents of the DRAM are shown in Figure 4. Packets are assembled in the DRAM as the data is emptied from the FIFO. A linked list data structure is used to distribute the data to the required packet. Once a packet is completed, the receive buffer link list is updated. The output DMA controller processes the linked list structure in hardware, keeping processor overhead low. The processor can use its processing power on managing the higher-level protocol algorithms because the low-level data transfers are handled in the FPGA.

A simple DMA controller is available from Actel and can be customized by the user to implement more complex transfers. The design uses VHDL to make it easy to customize. Changes to the state machine make possible additional features, like automatic chaining, burst processing, and bandwidth management.

Header Error Control Design

The header error control logic determines if an error has occurred in the header and can correct for any single bit error. The generator polynomial for the header is $X^8 + X^2 + X + 1$ and is implemented with XOR gates, as shown in Figure 5. A parallel implementation of the CRC is also possible. It uses a larger number of components but can compute the full 8-bit CRC in a single clock cycle. A VHDL code fragment is shown in the following display. The X[7-0] bits are the contents of the CRC register, and the D[7-0] bits are the 8-bit data input. The full design is available from Actel and can be implemented in any Actel FPGA family.

```
X[7] <= D[7] xor D[1] xor D[0] xor X[7];
X[6] <= D[6] xor D[0] xor X[7] xor X[6];
X[5] <= D[5] xor X[7] xor X[6] xor X[5];
X[4] <= D[4] xor X[6] xor X[5] xor X[4];
X[3] <= D[3] xor X[5] xor X[4] xor X[3];
X[2] <= D[2] xor X[4] xor X[3] xor X[2];
X[1] <= D[1] xor X[3] xor X[2] xor X[1];
X[0] <= D[0] xor X[2] xor X[1] xor X[0];
```

Processor Interface Design

The processor interface can be a generic high-speed synchronous interface using a block transfer mechanism to keep bus bandwidth high. Since ATM is a packet-oriented protocol, a block-oriented transfer mechanism to the processor memory is most efficient. The DMA controller can be a complex controller with features like chaining and threading, bus throttling, and other bandwidth-optimizing features—or a simple, fixed-size, block fill controller. The optimum design will depend on the processing needed by the CPU and the performance requirements. FPGAs can be very effective at accelerating processing requirements by getting data set up prior to CPU processing. Many algorithms can be significantly sped up if the data is first organized by a smart DMA controller. This allows the CPU to focus on the processing portion of the algorithm.

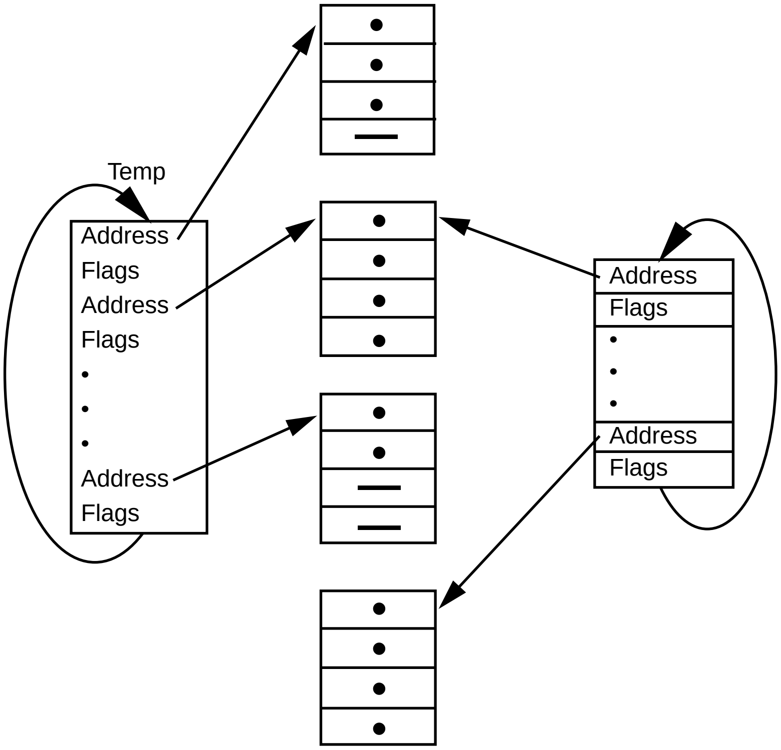

Figure 4 • *Packet Reassembly*

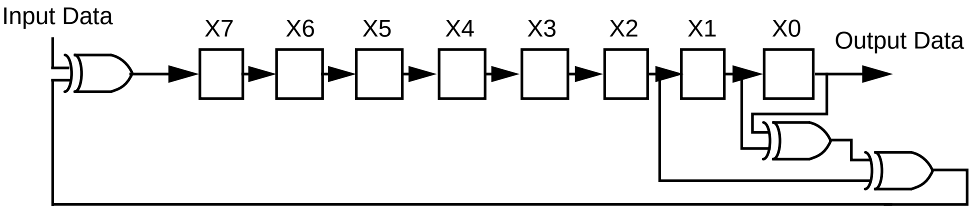

Figure 5 • *Header Error Control Design*

FPGA Implementation

The design can be implemented in an Actel 32200DX FPGA and a single bank of DRAM, and it would operate in excess of 66 MHz internal to the FPGA with a 33 MHz interface to the processor. Actel-provided designs for the FIFO, DRAM controller, and DMA controller help speed the design and ensure that key timing constraints can be met, prior to completion of the entire design.

Applications like this, with complex interactions at the system level, may require additional logic to be designed for use during debugging. The Actel 3200DX family, with its Actionprobe capability, allows the designer to observe all internal signals during device operation, making it easy to identify design errors without needing to change the design.

Summary

ATM network interface cards require high-speed system logic functions such as FIFOs, memory controllers, DMA controllers, and decoders. The 3200DX family, with its system logic integration features of high-capacity, fast dual-port SRAM and wide-decode function, provides just the right mix of capabilities for applications like ATM network interface cards. These system logic functions, along with the predesigned functions for FIFOs, DMA controllers, and DRAM, make the 3200DX family an ideal quick-time-to-market solution.