XAPP944 v1.0 June 14, 2006

Using a Xilinx CoolRunner-II CPLD as a Data Stream Switch

Summary

This application note shows how a Xilinx CoolRunner™-II CPLD can be used as a simple logical switch that can quickly and reliably select between different MPEG video sources. The source code for the design is available on the Xilinx website, and is linked from the "VHDL Code" section. The code can be expanded by the user to perform additional operations using the remaining CPLD resources.

Introductions

As consumer electronics become more complex, we are seeing a significant increase in the number of formats used to transfer data. Video and audio both come in a multitude of different formats, and often from different sources. Managing this data and ensuring the right data arrives at the right destination at the right time is a challenge.

In our example, we use a CoolRunner-II CPLD to select between three MPEG-2 video sources; these could be Satellite, Cable and Terrestrial television. The selected data source can then be sent to a decoder to be streamed to a display, stored on a Hard Disk Drive (HDD), as would happen in a Digital Video Recorder (DVR), or sent over a serial link to another piece of equipment.

MPEG-2 Data Sources

The MPEG-2 encoded data from the different sources arrives in Transport Streams (TS). The Transport Stream consists of 8 bits of data (TS_DATA) and 3 control bits (TS_CLK, TS_SYNC and TS_VAL). This example requires that each data source can be selected with a physical input, so three select inputs are required. If you want to have an electronically generated select signal, only two select signals will be necessary. Finally, the example also requires that the outputs of the multiplexer can be put into a 3-state condition to isolate them from the system. This is easy to achieve in the CoolRunner-II architecture.

CPLD Background

Before implementing even a simple design inside a Xilinx CoolRunner-II CPLD, it is important to understand the architecture. CPLDs are rich in logical resources. The logical array in the CoolRunner-II architecture is a 40x56 PLA – 40 input signals to the logical array can be used to create up to 56 Product (AND) Terms. Product Terms can then be used in any of the 16 Macrocells, containing registers, which are associated with each logical array. The I/O of the CoolRunner-II CPLD can operate at 1.5V, 1.8V, 2.5V and 3.3V by simply changing an attribute when coding the design.

More information on the CoolRunner-II architecture can be found in data sheets and application notes on the **www.xilinx.com** website.

The design requires 37 input pins and 11 output pins. Using Table 1 below, you can see that the appropriate device/package is the XC2C64A-VQ100.

Table 1: **CoolRunner-II Package/I/O Matrix**

	XC2C32A	XC2C64A	XC2C128	XC2C256	XC2C384	XC2C512
I/O Banks	2	2	2	2	4	4
Packages and User I/O	QFG32 (21) VQ44 (33) PC44 (33) CP56 (33)	QFG48 (33) VQ44 (33) PC44 (37) CP56 (45) VQ100 (64)	VQ100 (80) CP132 (100) TQ144 (100)	VQ100 (80) CP132 (106) TQ144 (118) PQ208 (173) FT256 (184)	TQ144 (118) PQ208 (173) FT256 (212) FG324 (240)	PQ208 (173) FT256 (212) FG324 (270)

MPEG-2 Multiplexer Design

Figure 1 shows the block diagram of the multiplexer system.

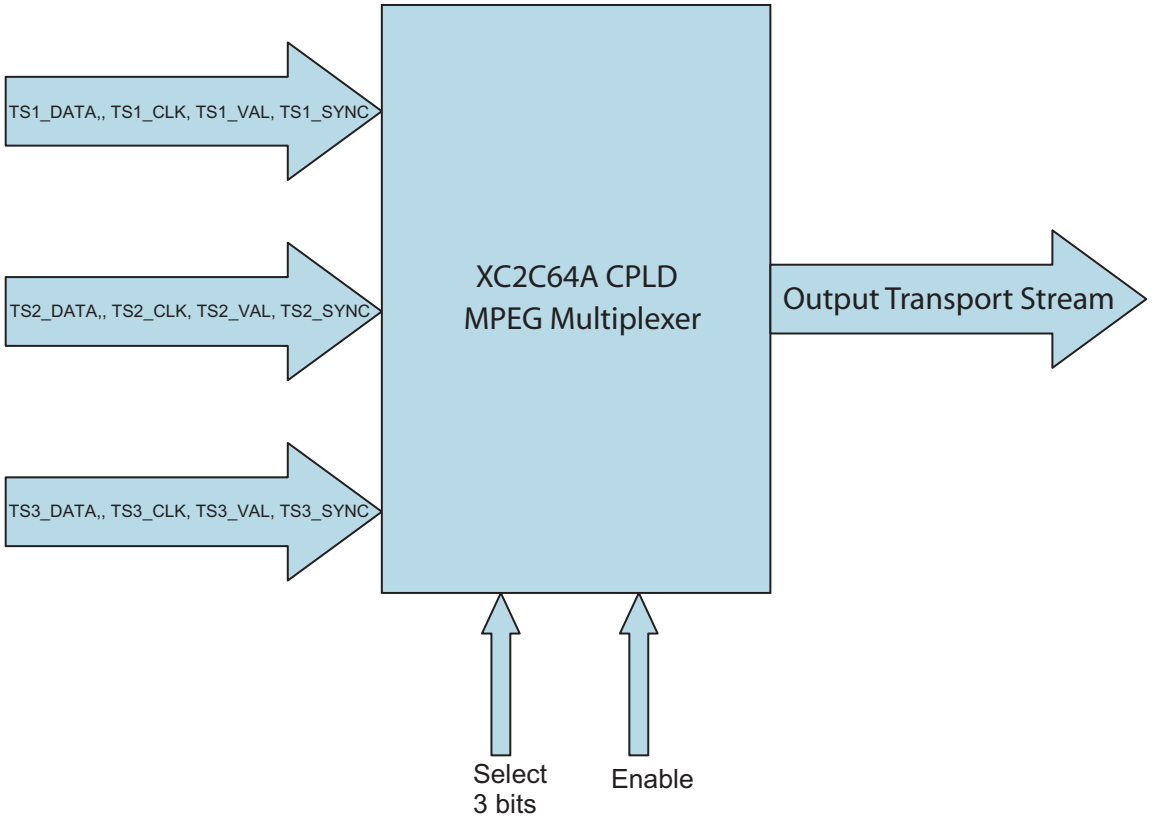

Figure 1: **MPEG Multiplexer**

All the I/O pins in this design are configured to the LVCMOS33 3.3V I/O standard. Depending on the state of the select lines, one of the three Transport Streams, TS1, TS2 and TS3, will be directed to the output of the device. When a logic 0 is applied to the enable signal, the outputs of the CPLD will be put into 3-state mode regardless of the condition of the select signals. This example is intended to supply data to an MPEG-2 receiver and decoder, and then directly to a display. The MPEG-2 Transport Stream uses short (188 byte) packet lengths. As broadcast environments can be prone to error and the loss of one or more packets, receiver devices are usually able to correct small errors. Hence losing one or two bits of data while changing between sources will not matter. If the MPEG-2 sources were to be stored, such as when used in a DVR, the user could use the remaining resources in the CPLD to register the data to ensure that no bits are lost.

Performance and Utilization

This is a simple design that only passes through a small amount of logic. Hence, if implemented in the slowest speed grade, the XC2C64A-7, the pin to pin combinatorial delay is only 14.8ns. As the MPEG-2 standard often has clock and data speeds of 27 MHz, this is easily handled. All similar signals travel through the same path in the CPLD, so they will emerge from the other side with negligible skew, because of to the deterministic nature of the timing model and architecture.

Table 2 shows the resource utilization of the implemented design. It is clear that the limiting factor in choice of device is the number of I/O. It is also evident that there are plenty of logic and register resources available if the user wants to expand the functionality of the design.

Table 2: **Device Utilization**

Macrocells Used/Total	Product Terms Used/Total	Function Block Inputs Used/Total	Registers Used/Total	Pins Used/Total
12/64 (19%)	35/224 (16%)	37/160 (23%)	0/64 (0%)	48/64 (75%)

For the sake of simplicity, we ran the simulation with randomly generated data rather than properly formatted MPEG-2 data.

VHDL Code

To obtain the VHDL code for this application note, please use the following link:

XAPP944 - http://www.xilinx.com/products/xaw/coolvhdlq.htm

Conclusion

Due to their flexibility, Xilinx CoolRunner-II CPLDs are found in a variety of digital consumer products. Multiplexing of MPEG-2 Transport Streams can easily be performed by a CoolRunner-II CPLD with resources to spare for performing further operations. The VHDL design supplied with this application note shows the simplest form of multiplexing between such data sources. If the user requires further functionality, there are many resources available in the CPLD in which to create it.

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
06/14/06	1.0	Initial Xilinx release.