Selecting an I/O Architecture for Your FPGA Design

Application Note 1500

Table of Contents

Introduction

In the not-too-distant past, designers used FPGAs primarily as prototyping vehicles. However, during the last decade FPGA data rates have increased dramatically, and they now rival those of CMOS ASICs. The surge in system performance convinced designers to take advantage of the flexibility inherent in FPGAs to create easy-to-upgrade products for a wide variety of markets, including communications, consumer, industrial, military and automotive applications.

The fast data rates of modern FPGAs allow designers the flexibility to create their own application-specific buses. However, these designers must immediately deal with the challenges of running I/O at high speeds. Factors such as channel-to-channel skew, jitter, and aperture window size limit the theoretical data rates of the FPGA's specifications. To address these issues, FPGA system designers are following the lead of their ASIC-focused predecessors and adopting I/O architectures that inherently reduce the effect of these factors.

This application note describes today's most popular I/O architectures and explores the factors that degrade the I/O performance of each architecture. Finally, we offer some pointers for selecting an appropriate I/O architecture for your application.

Agilent Technologies

I/O Clocking Architectures

Designers using FPGAs typically choose one of the following clocking architectures.

Synchronous clocking

Synchronous clocking is the most widely used clocking scheme in digital systems. In this type of clocking, there is one clock that is distributed to all synchronous logic in the system. All transactions occur at a particular edge of this one clock. This type of clocking is relatively simple to implement. However, to ensure timing accuracy, the clock distribution network must be matched in electrical length to all of the synchronous logic in the system. This becomes increasingly difficult when clock paths enter FPGAs with complex logic and connect multiple FPGAs that are created with various processes and different packaging. The variation in IC process and construction leads to timing error that degrades the performance of this type of clocking. Another limitation is that after a clocking event, the next clock must wait until the data has successfully traveled from one FPGA to the other. Examples of this type of architecture are the Pentium I processor, the PCI bus, and SDRAM memory.

Source synchronous clocking

Source synchronous clocking was developed to address the difficulty that synchronous clocking has with matching the clock distribution paths to each element in the system. In this type of clocking, the circuit that is producing the data pattern will create its own clock that is transferred along with the data. Generating the clock in the same geographical location as the data, in addition to having the clock traverse the same media as the data bus, creates a much tighter timing correlation. The tighter timing means that the data valid windows of the data being latched are better aligned with the clock used to latch them. This reduced channel-to-channel skew allows data transfer rates to exceed that of a synchronous architecture. Examples of this type of architecture are Intel's IA32 front side bus and DDR memory.

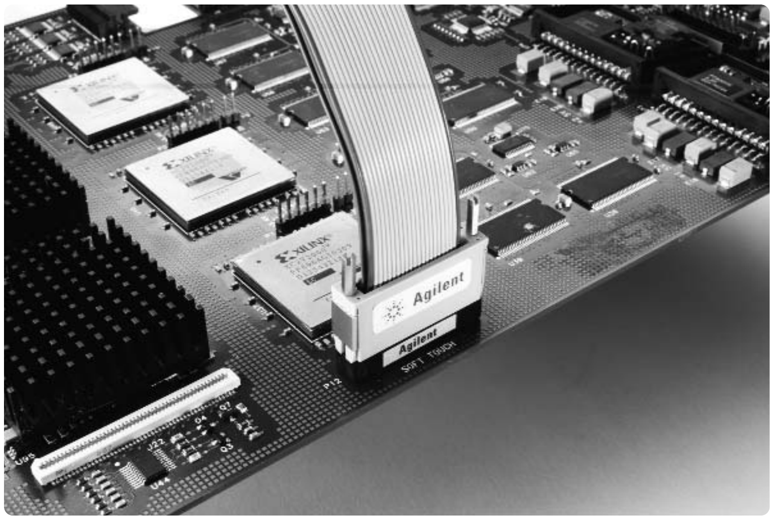


Figure 1. Designers of today's digital systems are using FPGAs because of their flexibility and speed. As a result, designers are facing the challenges of running I/O at high speeds.

Embedded clocking

Source synchronous clocking is prone to the same channel-to-channel skew as synchronous timing, albeit at a much higher data rate. The difference in process variation and interconnect structure between channels within an FPGA will contribute to channel-to-channel skew no matter how tightly the clock is coupled to the multiple signals. Embedded clocking solves this problem. In embedded clocking, the data is encoded in a manner that will guarantee a certain number of transitions per time (i.e., 8B/10B encoding). By ensuring that the data will transition a certain percentage of unit time, a PLL can then lock to the data stream. In embedded clocking, a PLL is used at the receiver that will lock onto the incoming data stream and create a clock based on the transition frequency. This clock is then centered within the data valid window. The drawback of this type of clocking is that the clock centering is only valid for the data channel to which the PLL has locked. However, the data rate for an individual channel is only limited by its jitter margin and aperture window. This type of clocking can achieve extremely high data rates (> 2.5 Gb/s) compared to synchronous and source synchronous designs. This type of design requires more complex circuitry at both the driver and receiver, but the data rates achieved are so much higher than those of previous architectures that the channel counts can be reduced while still delivering an overall throughput increase.

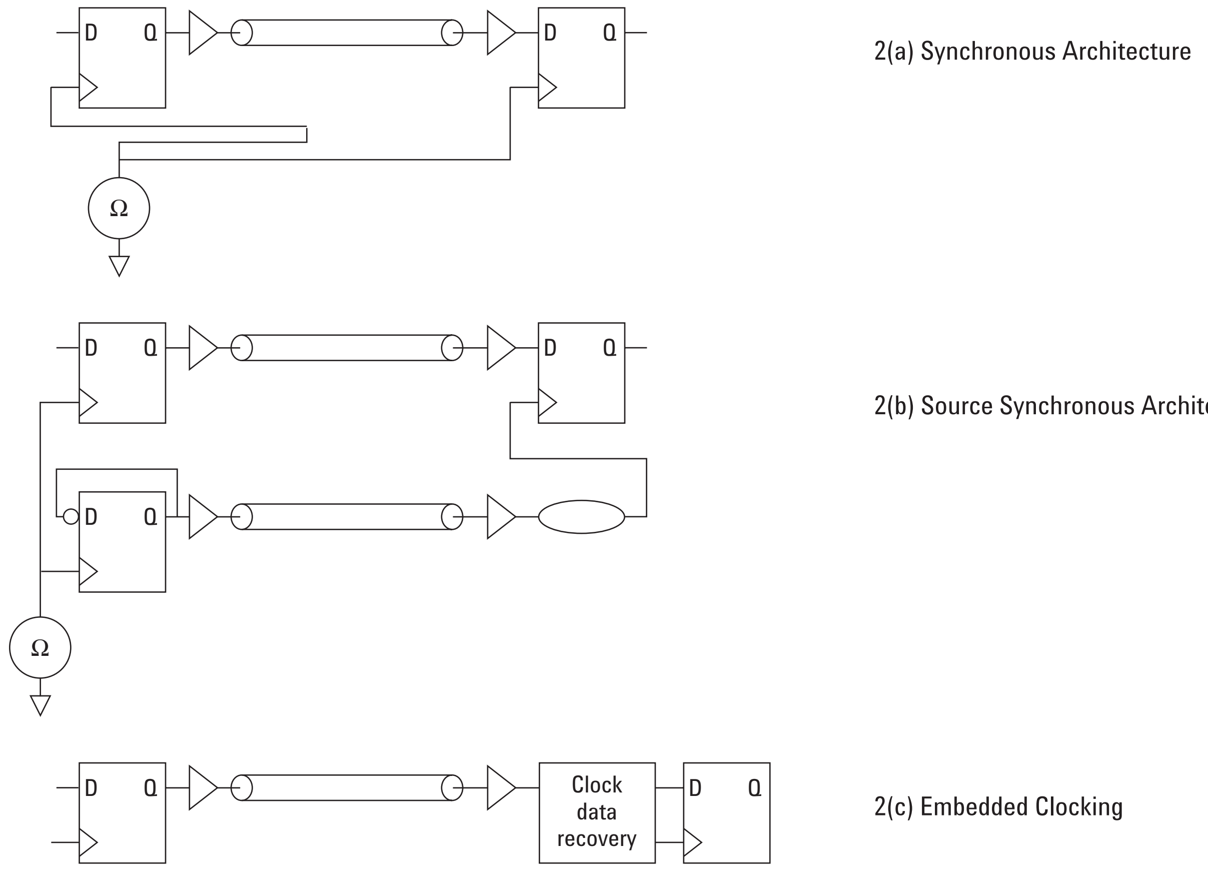

D
Q
Ω
Clock data recovery
2(a) Synchronous Architecture
2(b) Source Synchronous Architecture
2(c) Embedded Clocking

Figure 2. FPGA designers are changing their I/O architectures to overcome the signal integrity issues that arise when running at fast data rates.

Total throughput (or digital bandwidth) is defined as the number of symbols that can be transmitted per second. This depends on the unit interval that can be achieved per channel and the number of channels used in the I/O architecture.

$$UI = t_{symbol}$$

$$DataRate = \frac{1}{UI}$$

$$Throughput = (\#\ of\ channels) \cdot (DataRate)$$

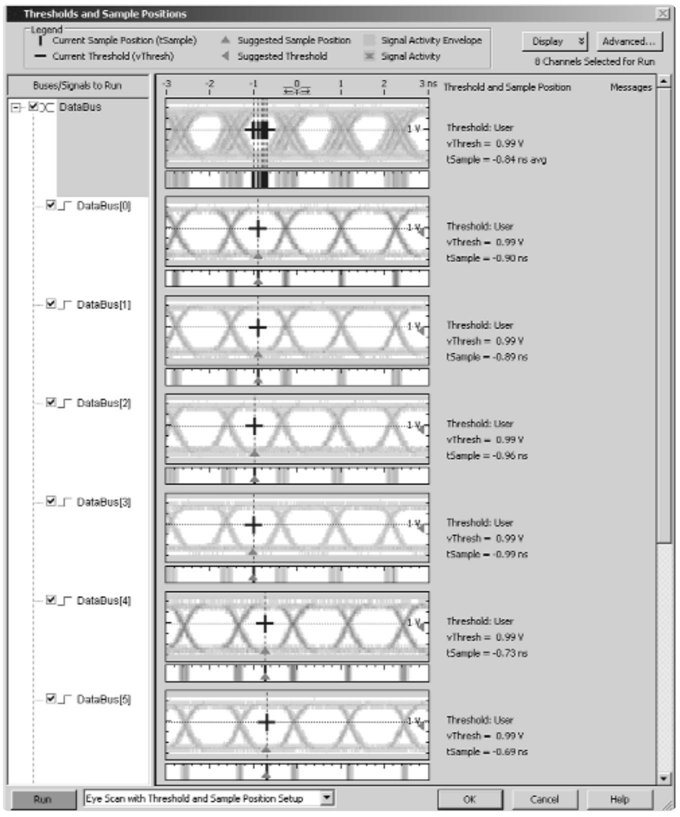
Channel-to-channel skew

Channel-to-channel skew refers to the time difference of the data valid windows between various signal paths. This skew is caused by electrical length differences in the physical signal paths. These paths vary for each channel due to the implementation of the circuit.

The skew between channels limits the data rate of the I/O architecture because the synchronizing clock for the data signals must be placed where it can successfully latch in all of the data channels in the bus. To achieve this placement, the net data valid window of all of the overlaid channels must be large enough for the clock to be placed within it. As you increase the number of channels you use, you increase the chance for channel-to-channel skew. As you use more channels, the net data valid window decreases, thus reducing the maximum transfer rate.

There are two main sources of skew. The first is the electrical length mismatch on the IC. The majority of this mismatch comes from the packaging interconnect. When you use an FPGA, the complex logic can also be a major factor. As CMOS dies approach sizes of 20 mm X 20 mm and packages reach signal counts into the thousands, the difference in channel path length becomes considerable. The on-chip skew is also exacerbated by the fact that signals are being driven from one large FPGA package to another. This can effectively double the skew due to on-chip path length.

The second source of skew is caused by channel mismatch on the PCB. This skew is due either to a physical mismatch of PCB traces on the same layer or to a propagation delay mismatch of PCB traces on different layers. The following figure shows an example of channel-to-channel skew.


Thresholds and Sample Positions
Legend
Current Sample Position (tSample)
Current Threshold (vThresh)
Suggested Sample Position
Suggested Threshold
Signal Activity Envelope
Signal Activity
Display
Advanced...
8 Channels Selected for Run
Buses/Signals to Run
-3
-2
-1
0
1
2
3 ns
Threshold and Sample Position
Messages
DataBus
DataBus[0]
DataBus[1]
DataBus[2]
DataBus[3]
DataBus[4]
DataBus[5]
1 V
Run
Eye Scan with Threshold and Sample Position Setup
OK
Cancel
Help

DataBus: Threshold: User, vThresh = 0.99 V, tSample = -0.84 ns avg
DataBus[0]: Threshold: User, vThresh = 0.99 V, tSample = -0.90 ns
DataBus[1]: Threshold: User, vThresh = 0.99 V, tSample = -0.89 ns
DataBus[2]: Threshold: User, vThresh = 0.99 V, tSample = -0.96 ns
DataBus[3]: Threshold: User, vThresh = 0.99 V, tSample = -0.99 ns
DataBus[4]: Threshold: User, vThresh = 0.99 V, tSample = -0.73 ns
DataBus[5]: Threshold: User, vThresh = 0.99 V, tSample = -0.69 ns

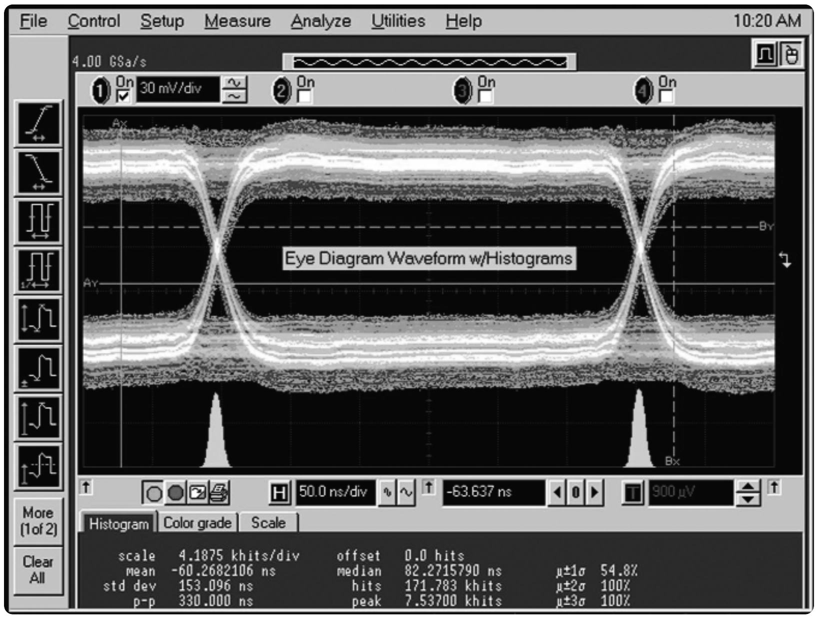
Figure 3. Tools such as logic analyzers can give designers insight into the net channel-to-channel skew that the FPGAs are experiencing. This measurement was taken using eye scan with threshold and sample position setup on an Agilent Technologies logic analyzer. Note the data valid window size shrinkage when all of the signals in the bus are overlaid.

Jitter

Jitter is a term that describes the timing uncertainty within a unit interval (UI). In other words, jitter is the amount of time within a UI in which it cannot be guaranteed that the data is at a stable logic level. This uncertainty region will determine the smallest size UI you can use while still transmitting reliable data. The two major types of jitter are deterministic and non-deterministic jitter.

- **Deterministic jitter –** Deterministic jitter refers to sources of jitter that can be calculated. For example, if a product is specified to operate with 5% power supply variation, then the timing uncertainty due to supply drop needs to be considered in the data rate specification. While the amount of timing uncertainly will vary from application to application, the entire range of timing uncertainty must be accounted for. Some possible sources of deterministic jitter can be process variation, ISI, reflections, simultaneous switching noise, power supply droop, and RC load variation. These sources can all be quantified in the design and added to a timing uncertainly stackup. For a stable design, you must account for the worst-case impact of each of these jitter sources.
- **Non-deterministic jitter –** Non-deterministic jitter refers to statistical sources of timing uncertainty. These are sources in which the noise contribution is modeled by their probability distribution rather than their worst-case value. Traditionally these sources are modeled using a Gaussian distribution. The figures-of-merit for these sources are their RMS value or standard deviation. For non-deterministic sources, you do not need to account for the worst-case impact, since statistically it will happen infrequently. Instead, a bit error rate is used to predict system stability. Examples of these types of noise sources are thermal and shot noise. Multiple sources of statistical noise are accounted for using an RMS summation to find the net contribution of statistical noise.

Jitter can be very difficult to predict. Typically its contribution is found using jitter measurement techniques in an oscilloscope. By allowing an acquisition to accumulate for a relatively large amount of time, you can find the distribution of jitter. If you are using this technique, you need to measure all of the sources of jitter simultaneously. Using jitter analysis tools within the Agilent Infiniium oscilloscopes, the jitter contributions can be isolated and analyzed.



50.0 ns/div   -63.637 ns   900 µV

```
    scale   4.1875 khits/div     offset   0.0 hits
     mean  -60.2682106 ns        median   82.2715790 ns       µ±1σ  54.8%
  std dev   153.096 ns             hits   171.783 khits       µ±2σ  100%
      p-p   330.000 ns             peak   7.53700 khits       µ±3σ  100%
```

Figure 4. Jitter measurement taken on a data signal using an Agilent DSO81304A 13 GHz infiniium oscilloscope. Note the histogram used to find the jitter distribution.

Receiver aperture window

The final contribution to the speed of an I/O design is the aperture window of the receiving element. This is traditionally called “setup and hold” time; it refers to the minimum time that the data must be stable before and after the timing event in order for the receiver to capture the symbol. To ensure successful data collection, the aperture window must be able to fit within the net data valid window of all of the overlaid data channels. FPGA manufacturers specify this value. However, complex logic and data path variation will degrade the ideal specification. In most cases, you need to measure it.

Calculation of throughput

The combination of channel-to-channel skew, jitter, and receiver aperture dictate how fast the I/O design can operate. We define the channel-to-channel skew as the time difference between the transition regions of two channels when operating in the same phase (i.e., intending to transmit a symbol at the same time). The following equations relate the total bit transfer rate per channel to the sources of error described above.

$$UI \geq t_{chan\text{-}to\text{-}chan} + t_{jitter} + t_{aperture}$$

$$Throughput = \left(\frac{\#\ of\ channels}{UI}\right)$$

Selecting an FPGA I/O Architecture

The first step in selecting an I/O architecture is determining your design's data throughput requirement. Once you have defined this requirement, you can evaluate I/O architectures for your design. This is an interactive process requiring measurement data. All of the factors (channel-to-channel skew, jitter, and aperture window) will increase as you add signals to your I/O design. The best way to find these contributions is through empirical data from oscilloscopes and logic analyzers.

Starting with the simple synchronous architecture, you can add signals to increase the overall throughput. As you add signals, observe the degradation factors. At some point, adding signals is no longer practical because of the adverse signal integrity contributions, the main contribution being channel-to-channel skew. If the channel-to-channel skew becomes insurmountable, the synchronous architecture may not be the optimal configuration.

At this point, you can explore the source synchronous architecture. The major advantage of this architecture is that it dramatically reduces the channel-to-channel skew within a data group. At some point the channel-to-channel skew will again become an issue, but at a much higher data rate. The skew in this architecture will be on the same order of magnitude as the jitter contribution. When the jitter and skew contribution become too large for proper operation, you may want to consider the embedded clock architecture.

Finally, examine the embedded clock architecture. With this architecture there is no channel-to-channel skew due to the clock/data recovery design, which is a distinct advantage. In addition, the jitter is reduced because there is no longer a data channel and a clock channel that possess jitter. This architecture will be limited only by the jitter and aperture window contribution.

If industry I/O standards are observed, they typically fall into the following table of data rates.

Individual channel data rate	**Clocking scheme**
DC - 300 Mb/s	Synchronous
300 Mb/s - 2 Gb/s	Source synchronous
> 2 Gb/s	Embedded clock

Table 1. Per channel data rates observed in industry for the clocking architectures presented. This illustrates the evolution of I/O architecture of both ASICs and in FPGAs.

Conclusion

Over the past decade, FPGA speeds have increased, and designers are now using FPGAs as an alternative to ASICs. However, designers are quickly discovering that the increased FPGA speeds are leading to the same signal integrity problems that ASIC designers have been facing for years. As signals are added to an I/O architecture to increase the total throughput, channel-to-channel skew, jitter, and aperture window issues degrade throughput. To address these problems, FPGA designers must turn to different clocking architectures. In order to maintain the trend of increased system data rates, a designer using FPGAs must first understand all of the contributions to the unit interval reduction in order to be successful.

The material in the application note is based on an article "FPGA I/O – When To Go Serial," published in *FPGA and Programmable Logic Journal.* The article was written by Brock J. LaMeres, Agilent Technologies.

Related Literature

Publication Title	Publication Type	Publication Number
Agilent 16900 Series Logic Analyzers	Color brochure	5989-0420EN
Agilent Infiniium 54850 Series Oscilloscopes	Data sheet	5988-7976EN
Logic Analyzer Probing Techniques for High-Speed Digital Systems	Application note 1450	5988-9125EN
B4655A FPGA Dynamic Probe	Data sheet	5989-0423EN
Planning Your Design for Debug: FPGA Dynamic Probe	Design guide	5989-1593EN

Product Web site

For the most up-to-date and complete application and product information, please visit our product Web site at:
www.agilent.com/find/logic

Agilent Technologies' Test and Measurement Support, Services, and Assistance
Agilent Technologies aims to maximize the value you receive, while minimizing your risk and problems. We strive to ensure that you get the test and measurement capabilities you paid for and obtain the support you need. Our extensive support resources and services can help you choose the right Agilent products for your applications and apply them successfully. Every instrument and system we sell has a global warranty. Two concepts underlie Agilent's overall support policy: "Our Promise" and "Your Advantage."

Our Promise
Our Promise means your Agilent test and measurement equipment will meet its advertised performance and functionality. When you are choosing new equipment, we will help you with product information, including realistic performance specifications and practical recommendations from experienced test engineers. When you receive your new Agilent equipment, we can help verify that it works properly and help with initial product operation.

Your Advantage
Your Advantage means that Agilent offers a wide range of additional expert test and measurement services, which you can purchase according to your unique technical and business needs. Solve problems efficiently and gain a competitive edge by contracting with us for calibration, extra-cost upgrades, out-of-warranty repairs, and on-site education and training, as well as design, system integration, project management, and other professional engineering services. Experienced Agilent engineers and technicians worldwide can help you maximize your productivity, optimize the return on investment of your Agilent instruments and systems, and obtain dependable measurement accuracy for the life of those products.

Agilent Email Updates

www.agilent.com/find/emailupdates
Get the latest information on the products and applications you select.

Agilent Direct

www.agilent.com/find/agilentdirect
Quickly choose and use your test equipment solutions with confidence.

Agilent T&M Software and Connectivity
Agilent's Test and Measurement software and connectivity products, solutions and developer network allows you to take time out of connecting your instruments to your computer with tools based on PC standards, so you can focus on your tasks, not on your connections. Visit **www.agilent.com/find/connectivity** for more information.

For more information on Agilent Technologies' products, applications or services, please contact your local Agilent office. The complete list is available at:

www.agilent.com/find/contactus

Phone or Fax
United States:
(tel) 800 829 4444
(fax) 800 829 4433

Canada:
(tel) 877 894 4414
(fax) 800 746 4866

China:
(tel) 800 810 0189
(fax) 800 820 2816

Europe:
(tel) 31 20 547 2111

Japan:
(tel) (81) 426 56 7832
(fax) (81) 426 56 7840

Korea:
(tel) (080) 769 0800
(fax) (080) 769 0900

Latin America:
(tel) (305) 269 7500

Taiwan:
(tel) 0800 047 866
(fax) 0800 286 331

Other Asia Pacific Countries:
(tel) (65) 6375 8100
(fax) (65) 67556 0042
Email: tm_ap@agilent.com
Contacts revised: 1/12/05

Product specifications and descriptions in this document subject to change without notice.

Printed in USA February 25, 2005
5989-1594EN

Agilent Technologies