DesignCon 2006

Why did my chip do that?

A survey of on-chip debug and diagnosis techniques

Carina Chiang
email: carina_chiang@yahoo.com

Brian Bailey, Brian Bailey Consulting
email: brian_bailey@acm.org

Abstract

What techniques are available on-chip to debug and diagnose a circuit when it fails to behave as expected? The use of on-chip silicon techniques to validate, measure, and debug silicon holds great promise but is not yet widely adopted. This paper surveys the benefits and challenges of on-chip measurement techniques, with examples drawn from available commercial offerings and published techniques. This paper concludes by looking at some of the interoperability issues that need to be addressed to enable mainstream adoption.

Author(s) Biography

Carina Chiang has developed ASIC design methodology at Agilent Technologies, Hewlett-Packard, Synopsys, and Apple Computer. Most recently, she has pursued an interest in high-speed I/O testing as well as embedded instrumentation. Ms. Chiang's past DesignCon papers have covered Design-for-Testability analysis of RTL designs and hardware-software co-verification. Ms. Chiang earned degrees in Mathematics from Princeton University and Yale University.

Brian Bailey is an industry and management consultant specializing in the functional verification of electronic systems. He has experience implementing verification tools and solutions for companies of all sizes, and has been involved in some of the most significant advances in the field. He is active in the standards community, currently chairing a committee within Accellera, has published two books in 2005 and is a popular contributor to the press and conferences. He graduated from Brunel University in England with a 1st class honours degree in Electrical and Electronic Engineering.

1 Introduction

In the real world it is difficult to verify a design under all conditions, and when systems operate in the field they can be subjected to conditions never thought about in the design phase or in the lab. When something goes wrong, you need to be able to quickly identify the cause. There can be many reasons for failure including component failure, test escapes, environmental factors, or the design going into a state never considered before. When something fails, manufacturing tests may catch the component failure. However, if manufacturing tests fail to catch the reason for failure, then diagnostic techniques must be used. When that happens you must be able to find out about what happened, what caused the problem and what events led up to the failure.

Traditionally engineers looked to the test and measurement industry to provide them with the necessary tools: probes, oscilloscopes, data capture and logic analysis systems, but many of these methods are becoming more difficult to apply.

Boards now have many layers and getting to the signals required is not always possible unless the board has been designed to bring them to the surface. Even more problematic are the chips themselves and the information that is being moved around the system. It is impossible to bring all of the signals of interest to a pin on the device as in many cases the pins are more precious than the silicon area itself. Shrinking geometries and rising clock speeds make it more likely that applying a probe may perturb the signal being measured. Care must be taken with external probes to ensure that the effects being measured are due to the circuit, rather than due to effects introduced by the probe.

A survey conducted by Collett International in 2002 shows that 62% of chips when first fabricated have functional problems, 51% of them have timing or clocking issues and 14% of them require tuning of the analog components. The trends have shown that we can expect each of these to become bigger problems going forward. In addition, the Collett study also showed that debug accounts for 50% of the total time spent by designers and verification engineers, so any way to reduce that can have a significant impact on time to market. Figures are not available for the typical amount of resources that are consumed by debugging problems once they are in silicon, but this too is likely to be extensive.

This paper examines some of the techniques, such as on-chip instruments to perform in situ measurements, which are emerging to deal with problems found in silicon. It will talk about some of the commercial offerings for IP and tools, as well as the standards that are in place or being worked on to make the solutions easier to adopt and to ensure tool compatibility. This leads to a consideration of the methodology for integrating and using the on-chip instruments, as well as interoperability issues. The objective is to provide an engineer or manager with enough information such that they can quickly gain a basic understanding of the state of the industry and where to find more in depth information. With this information it should also be easier to decide if it makes economic sense to apply some of these techniques in your next chip.

2 The Role of On-Chip Instruments

On-chip instrumentation is an area that will grow in the future in ways we cannot yet imagine. Today there are two areas in particular where on-chip instruments address a real industry need, namely High Speed I/O (HSIO) and functional logic problems.

HSIO is used in an increasing number of systems these days to provide serial connectivity at very high data rates. This creates a set of signal integrity issues such that parts that have been tested and found to be good may not operate correctly in the environment in which they are placed. In order to find these problems, instruments are needed. But as already mentioned accessibility to the necessary signals is difficult, and probing them can cause additional problems. The answer is to move the necessary instruments on-chip.

Logic errors may show up in the lab or field under two sets of circumstances. There is either a manufacturing error in the part, or the design has a functional flaw in it. If the flaw is a 'test escape', then additional tests will need to be developed to avoid flawed devices escaping in the future. This cannot happen until the reason for the failure has been identified. If a design error is identified then a workaround to the problem needs to be identified as quickly as possible. Again, access to signals can make this difficult, especially when the necessary information is embedded deeply within a chip. The answer again is to embed addition logic into the chip that will make it easier to access this information.

For designs based on FPGAs this may not be a major problem if there is spare capacity within the device as additional logic can be compiled to enable the extra visibility required. Most of the FPGA vendors and independent EDA companies that support these devices have tools in place that make this possible. These solutions do not, in some cases, require any additional pins on the FPGA as they utilize capabilities already built into the devices. However a range of solutions exist, and a given solution can be selected based on the circumstances expected for particular designs. If silicon debug is expected to be a concern for an ASIC, the high cost of re-spinning a design makes it essential to weigh the trade-offs in advance to determine the extent to which to incorporate on-chip debug techniques into the ASIC.

When Design for Test (DFT) first made an appearance, it was criticized for consuming too much silicon, but it quickly showed its value in significantly decreased test times. Now we are moving into the need for Design for Debug (DFD), a term coined by a newly formed consortium, which will make a similar demand on silicon area for on-chip instrumentation, but provide a decrease in chip diagnostic and debug times. This in turn leads to faster time to market and faster production ramps.

3 The State of the Industry

Integrated Device Manufacturers (IDMs) such as Intel[1], IBM, HP[2], and Philips[3] have invested the time and resources to implement proprietary on-chip oscilloscopes, on-chip logic analyzers, and other on-chip instruments into their high-end microprocessors and integrated platforms. Just as IDMs often take the lead in advancing EDA methodology to

meet the needs of their complex high-value designs, they are also taking the lead in proving the need and the value of embedded instruments.

Not every engineering team, however, has the resources or schedule to custom develop and implement their own portfolio of proprietary embedded instruments. Embedded instruments are commercially available, especially for FPGAs and for specific processors and bus architectures, but more are needed and as yet there are few standards that allow instruments developed by different companies to work together. EDA support and standardization to enable interoperability of commercial embedded instruments will further accelerate the adoption of these embedded instruments.

3.1 Developing Marketplace for Embedded Instrument Suppliers

The commercial embedded instrument landscape is a developing marketplace. Market forces shape the commercial availability of embedded instruments. Availability includes not only what types of embedded instruments are offered, but also who offers them, and how they are offered. As in any business segment, partnerships and standards play key roles in defining the embedded instrument landscape. These partnerships act to bundle embedded instruments customized for specific processors, for specific bus protocols, and for specific FPGA suppliers, as these each create natural market clusters that offer greater chances for success in implementation, in product offering, and in product support.

The multiplicity of ASIC suppliers and process technologies, along with the analog or timing critical nature of many embedded instruments, may limit the breadth and availability of ASIC embedded instruments to the subset of instruments that either can be delivered as reusable digital synthesizable blocks today, or that are backed by a test chip coupled with a reference methodology.

High-speed serial interfaces (HSIO) create another natural cluster point for embedded instruments that address signal integrity. Suppliers of high-speed serial interfaces may find it desirable to integrate or include embedded instruments to address signal integrity, because their design customers may need to be able to monitor and diagnose any associated signal integrity issues. However, since these embedded signal integrity instruments are apt to be analog or highly layout-dependent, special care must be taken when implementing these instruments. In many cases, the supplier of the HSIO is the one providing (either directly or through partnerships) the necessary instrumentation, and in some cases building in adaptive technology, to ensure that the end customer can get the best results from the use of the IP. This is a trend that is likely to continue.

3.2 Changing Landscape of Product Boundaries

In the discrete external instrument product universe, products keep adding new features or combining features from other instruments. For example, originally only oscilloscopes could generate eye diagrams to display signal integrity information; now digital instruments such as bit-error rate testers (BERT) and even logic analyzers can display eye diagrams. For the most part, though, each external instrument includes data acquisition, data analysis, and display. Some offer software (typically running on an embedded PC) that handles the data analysis and display tasks. In any case, all these capabilities are bundled together by a single instrument supplier. While there is some degree of independent supplier specialization in data acquisition probes and sometimes in

supplementary analysis software, an instrument purchase typically includes data acquisition, data analysis, and information display bundled as a product, and historically housed in a physical box.

In the embedded instrument space, since data acquisition occurs inside the IC, the product segmentation may divide along different boundaries. On chip data acquisition requires a supplier with the ability to design on chip measurement instruments that can be integrated reliably into a target IC. On the other hand, data analysis and information display may not require chip design expertise. That is, as long as the interfaces and protocols that specify the boundaries between data acquisition and data analysis are well defined, supplier separation and specialization may take place.

A similar separation between data acquisition and data analysis/presentation and consequent specialization has occurred for displaying logic simulation data in commercial viewers that do not themselves perform simulation. Such commercial simulation data viewers extend the context for manipulating simulation results, and offer a uniform simulation analysis environment independent of the underlying logic simulator.

It is not difficult to imagine an extension of such viewers to manipulate digital on-chip measurement data instead of logic simulator data.

4 High-Speed I/O and Signal Integrity

In the nascent area of embedded instrumentation for high-speed I/Omany opportunities lie ahead for the creation of new types of embedded instrumentation as well as common methods of analysis, display, calibration, and interoperability.

This section will briefly glance at a few of the commercial solutions that are beginning to become available.

4.1 SerDes and Embedded Instrumentation

Over the past few years, there has been a steady migration in communications standards and protocols. The demand for lower cost portable communications devices, for example, is driving a trend for faster interfaces. As the frequencies for data transfer increase, it becomes more difficult to ensure that all signals in a parallel system are sampled correctly and that the clock is properly transmitted. Almost paradoxically, the fix for this is to go to serial communications, even though it means even higher transfer rates. Replacing wide parallel data transfer with serial data transfer minimizes the interference between parallel lines and eliminates the problem of skew.

SerDes (Serializer/Deserializer) is the core functionality that is common to all of these new standards and protocols, such as SATA, PCI Express, USB and others. The serializer transforms the slower parallel data into a faster serial data stream, embeds the clock and transmits the signal. The other end, the deserializer, then performs the reverse operation recovering the original data. This concept is shown in Figure 1.

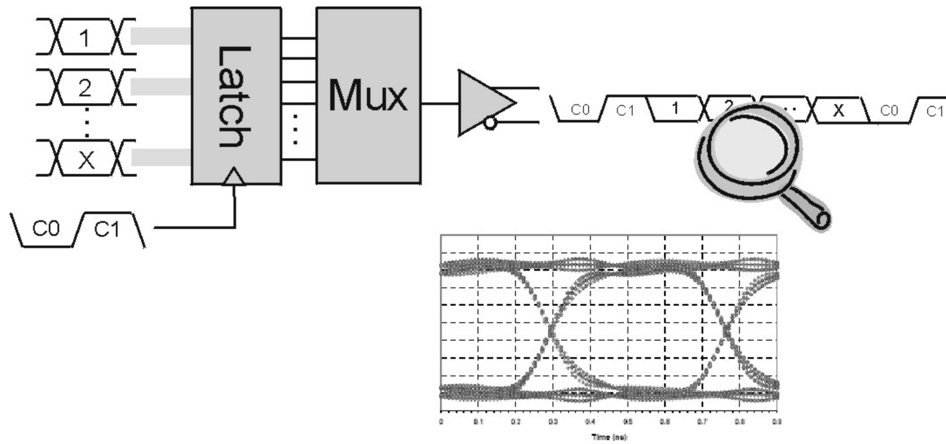

Figure 1: A SerDes Transmitter

Lower cost and lower power consumption may be achieved if the SerDes is integrated on-chip. However, even if the SerDes itself functions properly, when put into a system, it may suffer performance degradation due to signal integrity issues such as transmission-line effects on the board traces. At gigabit frequencies, signal integrity issues may cause an otherwise good device to generate errors due to its environment. Determining whether manufacturing variations or packaging issues are compromising performance or whether the system environment may be introducing signal integrity issues requires the ability to monitor the SerDes in situ, right where the high-speed serial I/O data transfers are taking place.

Traditionally, characterizing and measuring the performance of these devices was achieved with a set of external instruments such as oscilloscopes, Bit Error Rate Testers (BERT), and other instruments for testing picosecond jitter and looking at the signal 'eye' opening. The eye shows the amount of noise, jitter and other intrusions into the system that can be tolerated before errors are observed. Referring back to Figure 1, noise on the received signal and instability in the transmitting or receiving clock will reduce the space in the eye where valid data can be extracted. Measuring these factors is key to ensuring the performance and quality of high-speed I/O circuits, such that the error rate is kept to acceptable levels. However, high-speed I/O circuits are sensitive to external influences such as probes, which can unduly distort signals or load the system.

Integrating measurement instruments on-chip eliminates the problem of perturbation due to the probes and enables measurements to take place at on-chip locations that would otherwise be inaccessible.

Commercial IP suppliers who offer SerDes macrocells and FPGA vendors or chip manufacturers who offer SerDes integrated into their silicon may each offer some degree of proprietary on-chip measurement techniques to monitor the quality of the high-speed I/O. However, in a system, the SerDes transmitter and receiver may reside on components from different vendors. Each vendor's proprietary embedded measurement techniques for measuring, analyzing, and displaying performance may not be interoperable in a multi-vendor SerDes environment.

Commercial IP suppliers such as Synopsys[4] and Rambus[5] offer high-speed I/O IP, such as PCI Express PHY, that have on-chip BERT or on-chip oscilloscope capability to

capture and display eye diagrams. The integration of these embedded signal integrity measurement techniques into commercial IP provides the ability to monitor signal integrity issues during development as well as in manufacturing and possibly even in field operation.

Many FPGA suppliers and ASIC suppliers offer integrated SerDes IP that may include some degree of on-chip BERT features. Alternatively LogicVision's Embedded SerDes Test (EST) can be integrated into SerDes IP blocks from FPGA suppliers such as Xilinx, Altera, and Lattice.

One significant differentiation between suppliers of these embedded measurement features is the degree to which they can perform continuous test on live data streams, which may be more suitable for use during development or for monitoring field operation. Continuous analysis also enables adaptive systems to be built that can continually adjust operating conditions of the transmitter or receiver to meet the demands of their operating environment.

For production test of SerDes, where test times are limited, loopback testing techniques may enable the use of lower cost digital testers, thus lowering costs. LogicVision's Embedded SerDes Test (EST) employs loopback testing (e.g., connecting the output of the transmitter Tx to the input of the receiver Rx) to structurally characterize the parameters that determine signal eye distortion tolerance.

The on-chip instruments use a variety of techniques to measure the jitter tolerance, bit error rate and eye opening, such as clock undersampling, variable reference voltages and controlled current sources.

4.2 Jitter, Power, and Embedded Instrumentation

So far the discussion has centered on embedded instrumentation for SerDes. Many of the techniques used for this can also be applied to other aspects of the chip, such as the power and clock networks.

In the area of on-chip debug and diagnosis, some universities are looking at the next generation of capabilities such as on-chip spectrum analyzers[6]. Another example is work by NEC[7] on frequency domain macros that enable trouble spots in power and clock-networks to be located and analyzed. This works by identifying resonance frequencies in the power networks that can then be corrected using external decoupling capacitors.

A set of extensions to the JTAG standard, known as IEEE 1149.4, adds analog capabilities to boundary scan. This includes such items as switchable voltage references and analog multiplexers. While implementations of this are available, it has not taken off in the same way as its digital counterpart, and there is little evidence of it being extended to enable on-chip analog measurement or instrumentation.

5 Digital Debugging Systems

In the digital world there are a number of different ways to solve the debug problems. The decision about which approach to use is based on the fundamental requirements and

the type of system that is being instrumented. The pivotal question is whether data is captured and analyzed in real time. The general case of debugging a multi-core, multi-clock, or multi-processor system is highly complex, and remains an open problem that a number of companies are trying to solve.

5.1 Real Time Debug

There are times when the analysis of a problem requires viewing detailed timing information or the exact relationships between signals. If this is the case, then real time debug becomes necessary in which the data is captured, analyzed and displayed in a continuous manner. If real time capture and analysis is desired, the data is collected, sent off the chip and displayed in a traditional piece of test equipment, such as an oscilloscope or logic analyzer. This type of solution requires some number of dedicated pins on the chip that need to be propagated through the board or system.

There are two packaged solutions for this type of system, one for each of the major FPGA suppliers. Xilinx partnered with Agilent who supplied the on-chip IP and the external instrument, while Altera partnered with FS2 for the on-chip logic and Tektronix for the instruments. Each system allows for the selection of the internal signals to be probed without recompiling the FPGA. This creates an automated flow and handles all of the interoperability issues making it easy to get started with them. One limitation with these systems is that they do not integrate with the processor debug systems. The Xilinx/Agilent solution is described in more detail later in this paper.

5.2 Non-Real Time Debug, Single Processor

Consider chips that have a single processor and no real time requirements. In this case, you probably do not need to dedicate any additional logic as it is fairly simple to write special purpose debug code that can be executed on the processor to collect the necessary data and to make it available in the standard debug environment connected through the JTAG port. If there are pieces of the hardware that cannot be accessed in this way (write only registers, hardware state machines or other hidden logic) then the easiest course of action is probably to make this information accessible to the software just for debug purposes.

It must be understood though that the addition of this code will alter the execution of the system. Timing will be different, cache operations may change, code or data may be moved around in memory, or many other things will have been perturbed. If this is not an adequate solution, then you have to look at ways to reduce the intrusiveness of the solution. While there are software methods that can be utilized, this paper will concentrate on the addition of logic in the hardware to solve this problem.

5.3 Non Real Time Debug

For most debug situations, real time analysis and display is not necessary, so there is a lot more flexibility in the types of solutions that can be constructed. Figure 2 shows an example of a typical system into which a debug system could be built. The system will consist of a number of boards, each of which may contain several chips including any number of ASICs and FPGAs.

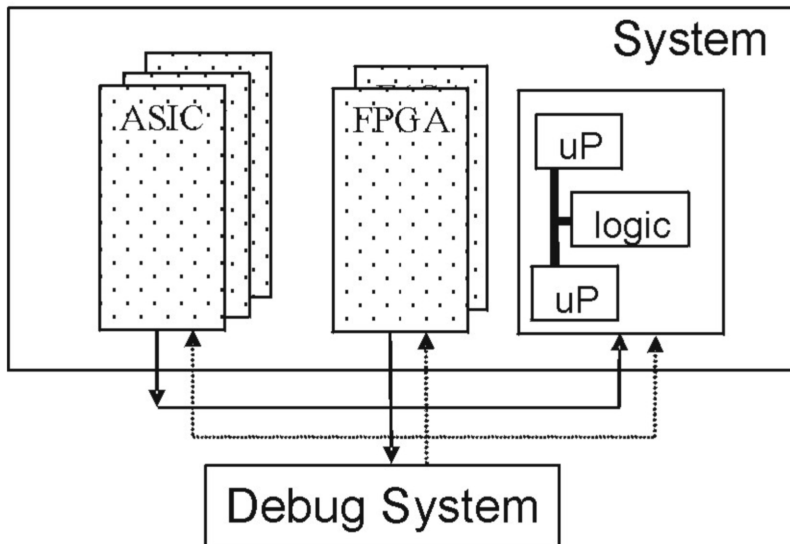

Figure 2: High Level view of debug System

Within any of these chips, there may be one or more processors connected by busses to multiple blocks of logic. Each chip or module within each chip may come from a different vendor. In order to be able to debug such a system, there is clearly a need for interoperability at the physical level and at the information level. For example how does the debug system know what information is available from each chip and what sort of control is available? How is this information related back to the design files so that it can be properly annotated?

Section 5.1 talked about real-time capture of information, but there is another situation when this type of system should be considered. This is a partitioning issue. For any type of logic debug system, it is important to be able to see back in time, and this implies that historical information has been captured and stored. There are two places where the memory to do this storage can be placed. Traditionally it was in the external instruments, and memory there is relatively cheap compared to on-chip memory. However there are costs associated with placing the memory off-chip such as solution scalability when IP blocks, multiple chips, or processors are involved and additional pins on the device are consumed. But if large trace depth is required to analyze the problem, then external instruments will always be able to provide more memory than on-chip solutions.

5.3.1 Components of a debug system

Each of the components in the debug system shown in Figure 2 can itself be broken down into a number of smaller components. Some of these components are shown in Figure 3, with the major components of a logic block and the analysis system expanded. A logic block could also be set up to capture information from one or more busses. Space limitations prevent a detailed discussion of how these component pieces are built or the specific features of each commercial implementation.

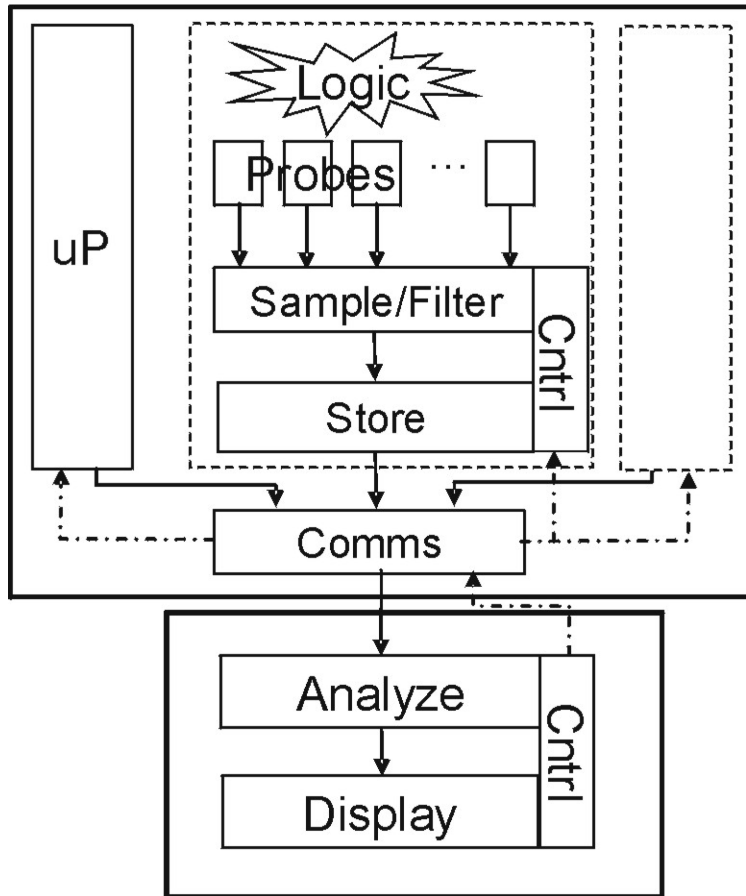

Figure 3: Components of a debug system

In general, there is a flow of information from top to bottom in Figure 3, starting with the data that needs to be captured or probed, a way to sample or filter the data and a means of storage. These steps are performed on-chip, plus there is some additional logic that will be necessary to support these operations. Storage is now assumed to be on-chip. The captured data is then transferred from the chip to an analysis system, where it will be processed and displayed. At the same time control goes from the analysis software back into the components to tell the hardware portions of the debug system what to do. As mentioned previously, there is also an implied knowledge transfer necessary such that the debug system knows what it is talking to in terms of both data and control.

Within this complete flow, there are a lot of places where standards are required for interoperability. In the extreme case, it should be possible to acquire or integrate each of these components from different companies. While this may sound extreme, it is not so unreasonable when considering that each processor vendor may supply pieces of the debug system integrated into their IP, while other logic IP may also be packaged with their pieces of the debug system.

Debug systems for processors have been an absolute requirement ever since the processor migrated on-chip. Without knowing what the processor is doing, it is close to impossible to debug the software. This sub-piece of the puzzle has thus become fairly well developed, but recently, as the complexity of systems has gone up, it has become

necessary to integrate this with logic debug and with the debug systems of other processors. Today we expect to create complex systems out of multiple heterogeneous processors and large numbers of logic blocks, but because of these interoperability issues, it is difficult at best to get a clear picture of the complete state of the system. An example of a way to address this issue will be talked about in another DesignCon paper.[8]

Some parts of the debug system are already guided by existing standards, such as the physical aspects of the communications system, but this is an adaptation of a standard being used for a completely different purpose than originally intended. Even on top of the physical connections between each component, there are other interoperability issues, such as data format, the ways in which time is handled, transfer of information between the design system and the analysis system, and many others. It is because of all of these interoperability issues that a considerable number of the early commercial offerings are based on partnerships between a few companies.

5.3.2 Getting data off the chip

It was stated in the previous section that the physical communications system is the focal point for interoperability today. This is based on the IEEE 1149.1 (JTAG) Test Access Port (TAP) and Boundary Scan[9] standard. This standard, originally targeted at interconnect testing of PC boards, was formally adopted in 1990; since that time it has been adapted for many other purposes, such as IC test, and to gain access to a variety of on-chip test features. For example, the embedded processor debug systems discussed earlier rely on the JTAG TAP port to access the processor's internal debug resources from the I/O pins of the IC, and to provide control functions that tell the processor to stop, run, single step, etc. This extreme level of adaptation is the hallmark of a great standard, in that it has enabled extensibility without compromising the original intent.

At the heart of the JTAG standard is the ability to create a number of 'scan chains' into which data can be placed or values can be read out. The controller defined for this ensures the orderly flow of information through the system and uses a pin efficient four wire system to get all data in and out of the chip. In addition, multiple chips can then be chained together such that the entire system can be controlled from a single piece of control software. Given that the standard has been stretched so far from its original intended usage, it is hardly surprising that it is showing the strain.

The JTAG data transfer rate (100'sKBit/second) shows that it is not viable for extremely large quantities of data, especially when considering the GHz clock rates of many systems today. In addition, getting data in or out of the chip requires that the system be frozen in time so the real time nature of the system may be destroyed while this is happening. Finally, it is a synchronous system requiring that everything be clocked from a single clock source. Many chips today use a number of different clocks, and as soon as separate JTAG chains are created for these, the re-integration of that data for analysis purposes becomes very difficult. There are thus a number of companies and standards bodies looking for ways to correct for these deficiencies.

One such new initiative is an informal group of industry test experts who are looking into internal JTAG (IJTAG). They recently announced the state of their work at the International Test Conference in 2005, although they have been working on the IJTAG initiative since 2004. This effort is focusing on the integration and management of

internal self-test circuitry.

Another set of extensions on top of the JTAG standard called EJTAG was pioneered by MIPS. In this system, a certain address range of the processor memory space is directly connected to the JTAG controller so that data can be written into and out of this memory. The contents of that memory can then be used in real time by the processor. This proposal also contains ways to implement processor breakpoint.

A third group of processor companies formed the Nexus 5001 Forum[10], now officially the IEEE-ISTO 5001[11], whose objective is to define a common set of microcontroller on-chip debug features, protocols, pins and interfaces to external tools which may be used by real-time embedded application developers. This standard, ratified by the IEEE in 2003, extends the JTAG standard by adding an extensible auxiliary port that allows real time transfer of data between the on-chip processor and a host.

Other companies, such as ARM and DAFCA while supporting the existing standards are also developing proprietary standards to deal with providing high speed serial access for debug purposes.

It is too early to tell if any of these new standards or proposals will emerge as the winner, or if features of each of them will be combined into a single unifying standard.

5.3.3 Logic Debug

Within the logic debug category, there are three primary levels at which debug support can be provided. The lowest level provides enhanced visibility, allowing information within a chip to be extracted and displayed. These systems require a minimum of on-chip support and relegate most of the work to the off-chip systems. Moving up in complexity are the systems that permit on-chip storage as well as providing mechanisms for filtering, compressing and triggering the capture of data. And finally, the most complex commercial offerings not only attempt to support these features, but in addition provide the ability to make minor corrections to the chip when the problems are diagnosed. Examples of each of these systems will be described below.

Enhanced visibility systems

Most of the examples of this kind of system are the ones already mentioned in the section on real-time debug. Because there is no on-chip storage, it relies on the external equipment to provide this capability. Xilinx's ChipScope Pro is a good example of this. Xilinx has partnered with Agilent who provides the on-chip logic and the external equipment using a proprietary data format to transfer information between the FPGA and the external equipment. The user can set the number of external pins that are to be used for debug. A special multiplexer core is inserted in the design and connected to the internal signals to be observed with the results of this fed out through the dedicated pins. The multiplexer core is controlled from the external instrument via a JTAG connection so that it can switch between the different groups of internal signals without requiring any recompilation of the FPGA. People used to working in a lab will have the familiarity of the logic analyzer interface to work with.

A slightly different approach is taken by First Silicon Solutions, often called FS2. One of the solutions provided by FS2 is a system that uses minimal on-chip resources, and stores

data off-chip. They provide an external module that has the trace storage built in, and this then connects to a general purpose host computer for data analysis and display. This system can be used for either FPGAs or ASICs and similar to the previous solution utilizes a user-defined number of pins on the device. Trace width and depth is limited with this solution compared to a traditional logic analyzer, but costs are much lower since a generic PC is used. In addition, FS2 has an on-chip storage system that will be described in the next section.

As mentioned previously, FS2 also has a partnership with Altera and Tektronix that puts together a complete packaged solution.

On-chip storage systems

Information storage is a little more complex than just providing a block of memory in which to keep data. Two types of memory could be used, either volatile memory such as DRAM which is the cheapest solution, but can only capture information about one system up-time, or persistent memory such as FLASH that could be used to capture information from several periods of usage. This may be important if you intend to use the logic not just for initial system debug, but also to capture events in the field, such that it can provide information about soft errors, system degradation etc.

As was shown in figure 3 above, the logic or bus blocks have a number of other capabilities:

- The ability to make the contents of the memory available via the chip access mechanism, which in most cases will be JTAG as discussed earlier.
- The ability to time stamp the entries made into the memory. Without this there is not way to determine the timing between captured events.
- Triggers that determine when the data is stored. These could either be one time triggers such that after they have fired, information is captured on successive clocks, or they could act as continuous filters such that data is only stored when the trigger condition is true.
- Source selection. It is possible that the system may have been instrumented with more capture points than the width of the trace memory provided. If this is the case, then multiplexers can be used to select between different banks of signals. These multiplexers are most likely controlled via the JTAG controller, or possibly memory mapped into a processor address space. This is the same as the capability described for real time systems.

If multiple logic capture blocks are going to be used, it is also necessary to consider if they are to be synchronized with each other. For example, does one of them keep the global time counter that is then fed to the others? Also can trigger signals pass between the blocks? These factors influence the overall design of the debug system even if a distributed architecture is selected.

Data Compression

Setting the size of the memory will always be a compromise. Ideally it will be wide enough to capture every signal of interest and deep enough to be able to understand fault

causality. But making it too large will take up more space on the chip than can be afforded. One technique to improve the amount of information that can be stored is to dynamically perform compression on the data. For example, when tracing a processor instruction stream, it is not necessary to store all instructions. Instead it is only necessary to capture those that represent discontinuities caused by branching statements.

A number of commercial systems exist that support on-chip trace storage. Each of these will be discussed briefly.

Synplicity is an independent provider of software for FPGAs. Synplicity provides an RTL debugger product called Identify, which allows the user to define probes and breakpoints from their RTL source. This information is then compiled into the FPGA. When different signals need to be viewed, they use an incremental approach to the place and route so as to reduce the iteration time. On-chip storage is utilized and the results are fed back and displayed on the RTL source. This capability will be discussed more in the analysis section. It does support multiple clock domains. Breakpoints are defined by creating state diagrams.

FS2 was already mentioned in an earlier section as a provider of real time off-chip storage systems, but they also support a full on-chip storage system. This is a highly configurable system where breakpoints, counters, signal banks and other attributes about the desired on-chip system can be defined. From this they generate the necessary IP blocks to take through a traditional synthesis process.

Both Synplicity and FS2 provide triggers. Triggers identify when information should be captured in the trace memory and when to inform the host that its contents should be downloaded. These can be very simple triggers, such as when a selected signal is high, or it can be controlled by state machines. In the case of FS2, counters can also be used. One company, Temento Systems, has made triggering a focus of their on-chip debug system by allowing triggers to be defined as assertions written in the Property Specification Language (PSL), a recently approved IEEE standard[12]. This is a very powerful language that can describe relationships between signals over time. These assertions are compiled into logic that becomes part of the debug system. It is also possible to target this not just for debug purposes but also for the detection and reporting of field errors, or run-time statistics.

Debug and fix systems

The final class of debug system is the one that not only provide all of the features talked about in the previous two sections, but also enable minor fixes to be made in the completed device. While FPGA users have this ability at any time in their development, ASIC designs are not as flexible. One company, DAFCA, embeds small amounts of FPGA-like reconfigurable logic into the system. This is usually done as a wrapper around an IP block such that all of the pins of the block are visible to the reconfigurable shell. These shells are then connected together in an internal scan chain that can transfer data to the on-chip storage system. This reconfigurable logic can thus be used to dynamically select the pins that are captured in the memory, to program the triggers used for capture, but in addition, once the problem has been diagnosed, the same reconfigurable shells can in some cases be programmed to provide on-chip fixes for the problem. While these systems have the highest on-chip costs, they could also save you having to rework a chip,

and saving a chip spin would save large costs in both time and money.

5.3.4 Analysis and Display

In the previous section, the general on-chip capabilities were discussed, but little was said about what happens to the data once it has been transferred off the chip. Analysis is a very important part of the process. The reason why it has been separated for the purposes of this discussion is because this is a logical separation that happens in many of the products talked about so far. For example many of the systems discussed come prepackaged with a display system, however some users may choose to use a display system from a simulator or from an independent display and analysis system such as those provided by Novas.

Having said that, there are a lack of standards that define the data stream associated with the logical connection between the information traced on-chip and the analysis system, and thus most suppliers of the hardware have either needed to partner with analysis and display companies or produce their own. Some of the existing standards are a carry over from previous generations of simulation systems like the Verilog Change Dump format (VCD), but this is far from ideal. To remedy this situation a new consortium has been formed called the Design for Debug Consortium[13]. This is perhaps the first industry effort that has brought together companies from the instrument IP, board test, EDA and DFT industries with the intent of promoting the products, methodologies and standards required to make the complete process more efficient.

In addition to the data transfer, there is also a need to transfer information about what is being probed, and what control the analysis system has over the debug features, such as signal bank selection, or trigger selection. Assuming that the on-chip data has been transferred, there is a lot of complex analysis that can then be done to make the information more useful to an engineer. In the digital world, the engineer would expect all of the same capabilities that they have from a simulation environment, except that the tools do not have the same amount of data. Some of the data can be filled in through a process called data inflation. Given the value of some signals, a number of other signal values can be worked out by regenerating them dynamically. For example if the inputs to a combinatorial block are known, then it is fairly easy to regenerate all of the intermediate and final results of the logic. This assumes that what is on the chip is functionally correct when compared against the logical design.

Another type of data transformation enables the monitoring of a few signals to be translated into a higher-level view of the transactions. This kind of information is much more useful to the engineer than the value on the individual signals. Most analysis systems will map the signal information back to the RTL level, but Novas utilizes some technology from another company, Spiratech, that can take signal values on busses or other standard interfaces and transcribe that into a transaction level view of the operations being performed on those interfaces.

6 Future Directions

The industry has reached a critical mass in terms of the need for these components and in the companies supplying solutions, which made this one of the hot topic areas at the

Design Automation Conference of 2005. However, its adoption is being limited by the lack of standards in the industry. In the preceding sections a number of components were identified, namely, the physical connection mechanism to the chip, the means of interconnecting the component pieces of an on-chip solution and the way in which the data is stored. In addition, there are issues with how the front-end system interacts with those on-board components, the identification of on-chip instruments and the identification of the signals being monitored, plus potential time delays on those signals. The lack of standards in these areas makes it difficult to acquire the best pieces from different companies. What is required is for instruments from different companies to be easily connected on-chip and to be able to independently select the software analysis tool that will use that data, or control the chips instruments.

Today there is little help to the engineer or manager for deciding what instruments should be placed onto a chip. Costs and benefits are still a little hard to quantify. Nobody wants to consider the possibility that a chip will fail, just as you never expect your house to burn down. But there is a certain level of insurance that you are willing to pay for the peace of mind that if it does, someone will help you to recover.

Systems today do not help you decide which are the 'Essential Signals' that need to be captured in order to understand what the system is doing. This will be necessary to help minimize the cost of insertion.

7 Next Steps

Before embarking on a design for debug solution, take a few steps to ensure that you receive the most value from your efforts. Talk to your IP providers to find out what support for on-chip debug they have built in. This is particularly important for processors and high-speed IO blocks; most providers now make this available. It would also be beneficial if they have made their systems extensible so that they can be integrated with other on-chip debug capabilities. At a minimum, look for JTAG support, but also ask about their roadmap for future improvements.

Consider the blocks that you are designing and what debug support you should really be inserting to make sure that you could understand what the system is doing. Don't just think of this as logic necessary to get the system up and running, then to be discarded, but also look at it as a field monitoring system. Can the on-chip instruments improve observability that would enable performance monitoring, or utilization information, usage statistics that could help you decide on how to make improvements in the future? It could provide a means of flagging errors, or even in some cases predicting errors in the future. This would provide a reduction in field support costs by having faulty parts identify themselves rather than having to take them off-line to run diagnostics. It is thus possible to integrate the ideas of Design for Debug and built in self test (BIST).

8 Summary

This paper has provided a very quick overview of some of the on-chip instrumentation and debugging techniques that are being used today. In addition, it has shown some of the benefits that can come from the utilization of these techniques. While there is a lot of

work that has to happen in the industry to ensure that the right standards are in place for complete plug and play, enough commonality exists between them to be able to put effective systems together today.

9 References and Acknowledgements

Companies who provided information that was used in this paper include Agilent, ARM, DAFCA, First Silicon Solutions (MIPS), Novas, Synopsys, and Temento. The authors thank them for their assistance.

---

[1] A. Martin et al., “8GB/s Differential Simultaneous Bidirectional Link with 4mV 9ps Waveform Caapture Diagnostic Capability,” 2003 IEEE International Solid-State Circuits Conference Proceedings.

[2] Josephson et al. “Debug methodology for the McKinley processor”. Test Conference 2001. Proceedings. International. Pages 451-460.

[3] Vermeulen and Goel. “Design for Debug: Catching Design Errors in Digital Chips”. IEEE Design and Test of Computers, May-June 2002, pages 37-45.

[4] DesignWare PHY and SERDES Technologies. Synopsys website http://www.synopsys.com/products/designware/serdes_tech.html

[5] Jared Zerbe et al. "Comparison of adaptive and non-adaptive equalization methods in high-performance backplanes". DesignCon 2005

[6] Mendez-Rivera et al. "An On-Chip Spectrum Analyzer for Analog Built-In Testing" Journal of Electronic Testing: Theory and Applications 21, 205–219, 2005

[7] Takamiya et al. "On-Chip Jitter-Spectrum Analyzer for High Speed Digital Designs". ISSCC 2004 Session 19.5

[8] Stollon, Collins and Stence. “Nexus-Based Multicore Debug”. DesignCon 2006

[9] IEEE Standard Test Access Port and Boundary-Scan Architecture Institute of Electrical and Electronics Engineers 14-Jun-2001 ISBN: 0738129445

[10] Berger and Barr. "Introduction to On-Chip Debug" Embedded Systems Programming, March 2003, pp. 47-48.

[11] http://www.nexus5001.org/

[12] IEEE PSL Standards: http://www.eda.org/ieee-1850/

[13] Design for Debug Consortium: http://www.designfordebug.org/