Implementing Serial Bus Interfaces with General Purpose Digital Instrumentation

Dale Johnson
Geotest- Marvin Test Systems, Inc.
Irvine, California
dalej@geotestinc.com

Abstract— The use of generic digital test instruments for emulating common serial bus protocols can provide benefits over dedicated bus test products, and often presents a trade-off between functionality, flexibility and cost. For example, a dedicated test instrument solution can offer more extensive test capabilities such as protocol support for controlling and analyzing traffic between a bus controller and a device under test. However, a more general-purpose solution that utilizes a digital test instrument can offer the flexibility to adapt to non-standard line rates and timing as well as supporting other digital test needs. Ultimately, the goal is to identify those instances where the clever or novel application of a general-purpose digital test tool is appropriate and provides tangible benefits. This paper presents an overview of how a general-purpose digital I/O instrument can be used to support the widely used serial bus interfaces. By using a general-purpose digital I/O solution, users can potentially realize a lower cost test solution, a more compact test system footprint, multi-site test capability, a common user control interface and expandability for future requirements.

Keywords— digital I/O instrument; JTAG; SPI; I2C; serial bus interface

I. OVERVIEW

As the electronics industry has evolved, so too have the number of methods for transferring information between electronic components. Today there are many protocols for communicating between components, circuit boards, subassemblies, LRU’s and systems. While system level communication has seen a variety of both parallel and serial communication standards evolve; board-level, and especially component level communication has nearly exclusively been relegated to serial data transfer methods. A common choice facing test engineers responsible for testing components using one or more of these serial protocols is whether to purchase test instruments dedicated to a specific bus, or to purchase general-purpose test equipment and program the instrument to provide the required functionality. As with many either/or questions, there are tradeoff’s to be considered, and advantages and disadvantages to be weighed in regards to functionality, flexibility and cost. The use of generic digital test instruments for emulating common serial bus protocols can often provide benefits over dedicated bus test products. This paper explores three serial bus protocols, the Serial Peripheral Interface (SPI) bus, the Inter-Integrated Circuit bus (I²C) bus and a JTAG bus, and discusses how one might use a general-purpose digital instrument to emulate these busses and the benefits of doing so. Following are brief descriptions of these common serial busses.

II. SPI BUS

One of the simplest serial bus protocols is the Serial Peripheral Interface bus, or SPI Bus. The SPI bus is a synchronous serial data transfer standard popularized by Motorola. Devices on the SPI bus transfer information in a full duplex mode and communicate in a master/slave configuration, where the master initiates the data transfer between itself and one or multiple slaves.

The SPI bus uses four signals to affect the transfer of information between devices (Figure 1). These signals are the Serial Clock (CLK), Serial Data Output (SDO), Serial Data Input (SDI) and Chip Select (CS). Data is transferred from the MSB of the Master I/O register to the LSB of the selected Slave I/O register, and from the MSB of the selected Slave I/O register to the LSB of the Master I/O register. As data is shifted out of the master device’s I/O register, the data is being shifted into the slave device’s I/O register. Simultaneously, the slave
probing the internals of complex integrated circuits and evolved from a robust manufacturing test tool to a test bed for component soldering and IC bonding failures. JTAG has connectivity defects, such as PCB shorts and opens, poor test access port for detecting common manufacturing Access Port and Boundary-Scan Architecture, and defines a the more common names for IEEE 1149.1 - Standard Test

A typical message begins by the bus master transmitting a start bit – which is a high-to-low transition on the SDA line while the SCL line is held high. A 7-bit address field that uniquely identifies the slave for which the message is intended follows the start bit. A single bit identifying the message as a read or a write follows the address field. Data is may transition on the SDA line only while the SCL line is held low. Data is transferred by a low-to-high and high-to-low transition on the SCL line. The slave, if present, responds with an ACK bit immediately after the Read/Write bit, and the master then continues with the remainder of the message, either transmitting or receiving data, depending on the type of message. The message is terminated by a low-to-high transition on SDA while SCL is held high. FC is capable of transferring data at 100 Kbps to 3.4 Mbps, although the message protocol overhead reduces the throughput by approximately one-half.

IV. JTAG 1149.1 BUS

JTAG (Joint Test Action Group) and Boundary Scan are the more common names for IEEE 1149.1 - Standard Test Access Port and Boundary-Scan Architecture, and defines a test access port for detecting common manufacturing connectivity defects, such as PCB shorts and opens, poor component soldering and IC bonding failures. JTAG has evolved from a robust manufacturing test tool to a test bed for probing the internals of complex integrated circuits and performing functional tests on logic blocks inside of such devices, at less than full speed. Boundary scan is also commonly used for flashing (programming) PROM’s, and programming CPLD’s, FPGA’s and other embedded components.

III. I²C BUS

The Inter-Integrated Circuit bus (FC) is a serial computer bus invented by Philips. Supporting multiple bus masters, FC is used to communicate between low-speed peripherals and embedded systems. Information is transferred using a simple protocol consisting of three basic message types; a master write to slave message, a master read from slave message, and combination messages that involve multiple reads and/or writes between the master and a slave. I²C uses only two signals to transfer information between components. They are Serial Data (SDA) and Serial Clock (SCL), which are open-drain signals with external pull-up resistors (Figure 2). Using an open-drain signal allows multiple bus masters to exist on the bus, and often the role of Bus Master and Bus Slave can be dynamically switched between devices on the bus.

A basic Boundary Scan system utilizes either four or five signals, depending on whether the optional Reset signal is used. The four remaining signals are the Clock Input (TCK), Test Mode Select (TMS), Test Data Input (TDI) and Test Data Output (TDO) (Figure 3). Clocking a serial bit pattern into the internal state machine (commands) via the TMS pin configures all parts on the same scan chain in parallel. Data is clocked through parts in a Daisy Chain fashion using the TDI and TDO. The TDI signal on the first device in the scan chain is the primary data input. The TDO of the first device connects to the TDI of the second device, and its TDO connects to the TDI of the third device. This continues until the last device of the scan chain, who’s TDO is the primary data output. Typical clock rates for TCK are 10 MHz to 100 MHz, although the slowest device in a scan chain dictates the fastest possible clock speed for the entire chain. Other timing parameters to be considered are the data valid-to-clock setup time and the clock-to-data invalid hold time.

V. SUPPORTING SERIAL BUS APPLICATIONS

It’s common when designing a new device for a manufacturer to design multiple versions of the device, each with a common set of core capabilities, but each also with a unique variation intended to broaden the market potential of the product. For devices employing serial communication busses, wrapping a different interface around the core function is a simple, effective way to address a wide variety of customer needs. Such a scenario is presented here.

Supporting a device that has different interfaces with a common core presents manufacturers, or more specifically, contract manufacturers, with the need to support not just one, but potentially three or more different interfaces. To address a broader market and obtain maximum leverage from the design effort, the customer's product family may include versions of the product with a SPI bus interface, an I²C bus interface, a JTAG interface, and others. Subsequently, to perform production tests on the new design, a test system needs to be capable of interfacing with, and controlling each of the similar, yet separate interfaces. Typically, this would require dedicated bus testers, one for each interface type used in the product family. Each test instrument would typically have a unique interface test adapter to the UUT, and often would include special software tools and programming methodologies that are optimized to the test instrument. Each would also have its own
learning curve as the test engineer became proficient in the use of these various instruments and their tool set.

The capabilities of dedicated bus testers, and the tools provided with them, are typically very robust, providing the options of exercising the bus, injecting errors, validating proper response to error conditions, and fully characterizing compliance with the published specifications for the bus it was designed to test. However, as a contract manufacturer, one is not required to test to this level. The design verification tests have already proven the validity of the design. The manufacturing requirement is to run a less complex subset of the design verification tests in order to provide a high level of confidence that the device, or UUT, was built or assembled correctly. There is no need for high-level, robust instrumentation with all of the software bells and whistles. Consequently, a general-purpose digital instrument can provide the capability to verify operation and support emulation of not one, but all three serial busses. This allows for a single ITA, the opportunity for a common software interface, and the need to learn only one instrument, thus shortening the learning curve.

VI. THE GENERAL PURPOSE SERIAL BUS TESTER

A basic digital test instrument consists of a timing resource, typically a PLL-based programmable clock source, control logic, a pattern sequencer/address generator, and pattern memory. For basic serial bus emulation, the minimum pattern memory required is an Output Memory which stores the output pattern sent to the UUT, a Tristate Control memory which provides dynamic tri-state capability for bi-directional I/O, and a Record Memory for capturing UUT responses. A host computer is connected to the digital instrument for programmatic control of the test hardware. Typical software consists of instrument specific drivers and a high-level test program for generating and loading stimulus patterns, executing the test sequence and analyzing the UUT response captured by the digital instrument. Figure 4 details this basic instrument’s architecture connected via the PXI bus to a host computer.

Controlling or programming a device via a serial link typically involves writing or reading registers. The registers may be byte, word or even double word sizes, but regardless of their widths, the process is similar for accessing the register. A simple command interface (Figure 5) can be designed to provide primitive bus commands for reading or writing to the UUT registers. The bus commands are compiled into serial patterns for downloading to the digital test instrument, with the pattern compilation being specific for the bus interface of interest. For example, if the commands were intended for the I2C bus, then the commands’ bit pattern would include a header containing the seven-bit device address, whereas, an SPI command would just include clock and data patterns for shifting data to the UUT register.

The commands allow high-level configuration and control of the UUT regardless of the interface selected. The test patterns can be loaded directly to the test hardware, or stored to a file for later use in an ATE program. As the need to test devices with new serial interfaces are developed, only the functions for translating the primitive commands to the new serial format need to be developed. The actual commands do not need to be changed. High-level commands to support standard JTAG instructions (BYPASS, EXTEST, SAMPLE/PRELOAD) can also be added, including the ability to import binary files, such as Serial Vector Format files, which supports the programming of embedded EPROMS, CPLD’s, FPGA’s or other devices. The flexibility of the digital test engine allows many scenarios to be incorporated into the tester, or added to the tester as the need arises.

If the digital instrument also provides more advanced capabilities, such as edge timing placement, programmable I/O levels, or Real-Time compare, then the possibility to go beyond the mere exercising / emulating of serial busses exists. Input level sensitivity, validation of setup and hold parameters and real-time validation of the UUT response are examples of what is possible.

Most serial busses are relatively slow, so clock speeds are generally not a problem. However, bus timing could be an issue. For example, if the bus requires adherence to strict data-to-clock setup and hold parameters, then the digital instrument needs to accommodate those requirements. For a digital test instrument that allows a programmable skew offset to be applied between the internal DIO data clock and the DIO data outputs, then meeting these timing specifications is done by applying the setup/hold specifications as a clock to data skew (edge timing) timing parameter. If the digital instrument does not provide this functionality, then the most common approach to testing the setup/hold specifications is to over clock the digital instrument. For example, to represent a data bit, I2C
requires that the SDA signal change state only while the clock is low. Without a clock-to-data skew adjustment feature, the setup and hold time can only be met by over clocking (over sampling) the digital instrument by a factor of 4 (minimum). For example, if the UUT’s I2C bus operates at 400 KHz, then you program the digital instrument to a 1.6 MHz data rate and use four digital instrument states to represent one I2C bus cycle. (Figure 6A & 6B) This method can be employed by virtually any digital test instrument, with two caveats: 1) The 4X clock rate is within the range of the DIO instrument. 2) A deep pattern buffer is required as the test pattern length is reduced by a factor of four (or more).

Most digital test instruments provide many parallel channels, far more than is typically needed for serial bus test applications. The extra channels can be used to provide testing of multiple serial busses (multi-site testing) or for augmenting the capabilities of the test system. For example, if you need to trigger an oscilloscope at a particular time or event within a bus cycle – e.g. when data is valid – then an unused digital channel can become a trigger signal for an oscilloscope. You can program the strobe to occur at every valid bit within the bit stream, only when the UUT is responding, or at any other time of your choosing (Figure 6B).

VII. BUS EMULATION EXAMPLES

Figures 7, 8 and 9 detail how a general-purpose digital test instrument can be used to support the SPI, I2C, and JTAG busses, including support for multiple busses.
VIII. CONCLUSION

While a dedicated test instrument can offer more targeted test capabilities for serial bus interfaces, a solution built around a general-purpose digital test instrument can offer flexibility that is unmatched when compared to a dedicated bus test tool. These features include the flexibility to adapt to multiple bus environments, adapt to changing test conditions, support non-standard protocols (data rates, timing, message sizes...), provide synchronization for external equipment to time-specific bus events, as well as support other static and dynamic digital test needs. A general purpose solution can reduce the footprint of the test system and reduce the test equipment learning curve as the test engineer is working with one instrument and learning one programming language. Supporting multiple serial busses with one general-purpose instrument also means that the test engineer needs to interface with only one vendor should technical support be required. Finally, significant cost savings can be realized since a variety of different bus protocols can be supported without incurring incremental hardware investment each time a new bus protocol is added to a test setup. Ultimately, the goal is to identify those instances where the clever or novel application of a general-purpose digital test tool is appropriate and provides tangible benefits in time, cost flexibility and effective utilization of resources.

References

http://en.wikipedia.org/wiki/I2c
http://en.wikipedia.org/wiki/JTAG