An SBus Monitor Board

H. A. Xie, K. E. Forward, K. M. Adams and D. Leask.
h.xie@ee.mu.OZ.AU. k.forward@ee.mu.OZ.AU. k.adams@ee.mu.OZ.AU. d.leask@ee.mu.OZ.AU.

The University of Melbourne
Parkville, Victoria, 3052
AUSTRALIA.

Abstract

During the development of computer peripherals which interface to the processor via the system bus it is often necessary to acquire the signals on the bus at the hardware level. It is difficult to attach general-purpose logic analysers and in-circuit emulators to a multiple pin bus connector and hence it is not practical to catch all the bus data required
to ensure that such signals are in accordance with the bus specification. Hence a given connector specific bus monitor board is a necessary instrument to attach to the system motherboard in order to monitor all bus activities. A connector specific bus monitor board provides an efficient resource with which to study the internal philosophy of system software, the software implementation process for different communication layers, and to provide debugging for hardware developers. The bus monitor board described here is designed to attach to a SUN SBus and is similar to the Transformable Computer, which appeared recently, in that its architecture is reconfigurable via the use of a Field Programmable Gate Array (FPGA). It can be programmed to customise it to various users' specific needs. It differs from the Transformable Computer in that although it can be programmed to function as a coprocessor its primary function is dedicated SBus Monitoring. It is less costly than a Transformable Computer.

In this article we describe the prototype of an SBus monitor's architecture and functions, and present the experimental results obtained from a Sun SPARC work station and an Aurora SBox Expansion Chassis, which demonstrate its ability to capture and display data communication and bus activity. Since this prototype board is programmable, it has the potential to provide many special purpose SBus monitors, but also function as a programmable coprocessor.

1. Introduction

The SBus Monitor Board (SMB) is a single slot SBus board which can be programmed to function as a special SBus logic analyser. In contrast an ordinary logic analyser [1], although more versatile, is difficult to connect and use. The SMB is both a hardware board on which complex digital circuits can be implemented [2] and a single width SBus analyser [3]. But it is not like a hardware tracer system [4], which includes logic analysers, controllers, recording devices, and another host computer. An FPGA has been used to provide programmable hardware which enables the user to select, the data to be stored and the trigger which freezes the data. The FPGA enables the board to meet various demands raised by different users which can be totally unknown to the designers of the board and which may only become known to the user during the debugging session and therefore impossible to build in beforehand. This striking feature of programmability is shared by the Virtual Computer(EVC) or the Anyboard [5], [6], which comprise a complete computer system including motherboard and baby board, and cost nearly as much as a Work station. However although the SMB is specially suitable for SBus monitoring it can also be programmed to function as a coprocessor, but boards such as the EVC cannot be used as bus monitors as they do not connect to all of the SBus signals. Another difference is that the SMB has a single FPGA and costs only a fraction of the cost of a normal workstation. This SBus instrument has been programmed to provide two working modes. One is the data acquisition mode, which is used to capture SBus data from the host machine and store the trace in the memory on the SMB. The other is the data display mode, in which the acquired data is displayed in a window on the same host machine.

In the data acquisition mode the SMB is an SBus pure slave, which does not request or occupy any SBus lines during the monitoring period. In this sense it acts exactly as an oscilloscope used to monitor electrical signals. It can be plugged into any slot of the SBus system without causing any disturbances to the normal operation of the system. The SMB stores the data it has gathered in memory which it addresses with a counter which is incremented each time a new data set is to be stored. When the address reaches its maximum value it steps back

to zero and overwrites the existing data. The storage of data in this fashion continues until the incoming data matches a trigger pattern. To ensure that data before and after the trigger are available for display and analysis data, tracing continues for a preset period after the trigger. Whilst all pins of the SBus are observed, only 32 of them are selected to be stored in the trace in the memory which is written to 4 bytes at a time. In this way the SMB looks like a Logic State Analyser, but is much easier to connect into the system. The signals to be traced are determined by the user.

In data display mode, all the captured data from the trace memory are read back by the host processor, via the SBus, and tabularly or graphically displayed on its screen. The SMB provides the first level of the SBus data processing, and direct access to bus signals in each clock period during the operation of the host computer. The timing relationship of a signal relative to the other signals is available to the user, who can then examine the captured waveforms off-line. Such analysis is the only means of determining the real-time functioning of a processor and the various peripherals connected to it. An instrument such as this is essential for the performance optimisation of any hardware interfaces.

The SMB utilises Field Programmable Gate Array FPGA [7] technology, which offers reconfiguration with multiple functions within the same system at different times by loading different configuration programs. This leads to fewer packages and increased reliability of the SMB. The large number of logic functions, registers, and I/O interfaces made possible by the FPGA make it ideal for implementing comparators triggers, timing and control functions. Moreover, the SMB can be reconfigured as an SBus coprocessor since it has access to all of the SBus signals, a large number of programmable gates and its own memory. 10 of the 160 programmable I/O blocks in the FPGA perimeter are devoted to configuration pins, 82 are I/Os connected to the SBus socket , 52 are connected to the SMB memory, and 16 are connected to buffered LED displays which can be used to observe the operation of the SMB.

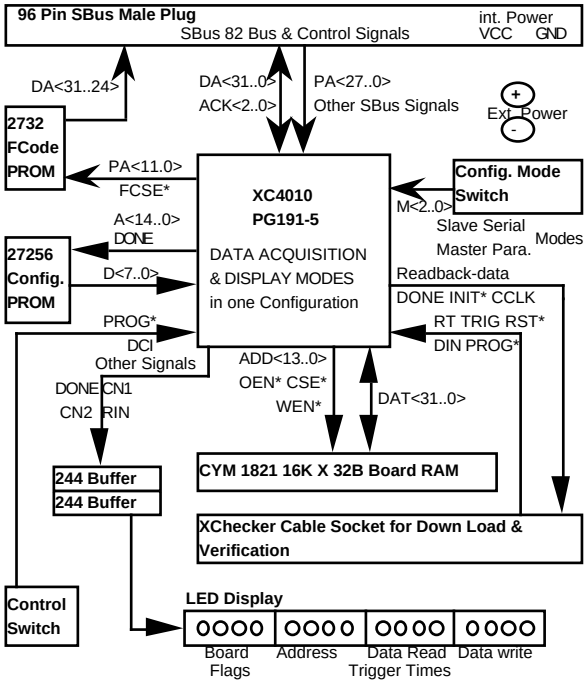
2. Board Architecture and Function

The board was designed and manufactured using Cadence software [8]. It is a four-layer single-SBus-slot board (SPARC Station 1, or 1 plus). Its block diagram is shown in Figure 1.

The board is composed of the following eight parts.

(1). SBus Male Connector

This is a 96-pin male expansion connector compatible with the SBus specification. The maximum SBus clock frequency is 25 MHz and the board has been designed to operate at this frequency. There are 82 bits for the SBus bus and control signals on the connector and the remainder are power pins.



SBus Monitor Board Block Diagram

Figure 1

(2). The FPGA

This is a 4010 PG 191-5 Xilinx programmable Gate Array and it accommodates all logic functions, control registers, and I/O interfaces required to perform the functions of the monitor. A configuration file is used to structure the FPGA so that it performs any logic functions which the user desires. In Figure 2 the FPGA is located in the centre of diagram. The SBus data bus, DA<31..0> and acknowledgment bits, ACK<2..0>, are bidirectional while the physical address, PA<27..0>, and all other signals are unidirectional from the SBus connector into the FPGA. The FPGA outputs an address bus, ADD<13..0>, and bidirectional data bus, DAT<31..0>, chip select, CSE*, output enable OEN* and write enable, WENN*, to the board memory. It also provides the SBus low address lines, PA<11..0>, and chip select, FCSE*, to the board FCode PROM, and 16 internal signals to the board LED display via two buffers. The FCode PROM is required to ensure that the SMB functions correctly in its interactions with the host processor. Its configuration modes are controlled by a switch through its mode bits M0, M1, M2. There are two configuration modes available on it. One is the master parallel up or down mode which loads the configuration file from the board configuration PROM. The other is the slave serial mode which operates through the board download and readback connector. The FPGA can be powered externally or via internal SBus lines, and the direct controls of the reconfiguration and internal logic level are selected through another switch.

(3). 27256 PROM

A 27256 built-in PROM is used to configure the FPGA.

(4). CYM 1821 PZ-12C RAM
A fast CYM1821 (16 K x 32 bits) RAM with a maximum access time of 12 ns for 32 bit data records is mounted on the board. The SBus system can read and write this through the FPGA.

(5). 2732 FCode PROM
A 2732 FCode PROM is used during SBus booting to ensure the SMB is recognised as a legitimate SBus board by the host processor and other SBus drivers [9,10]. Its 8-bit data bus is directly connected to the SBus connector while its 12-bit address bus, chip select, and acknowledge bits are controlled by the FPGA.

(6). 74ACT244PC Buffer
Two 74ACT244PC buffers are used to buffer up to 16 internal FPGA signals and to provide LED displays on the board. If they are not used in the users' design, the relevant 16-pins on the FPGA can be reconfigured in other I/O interfaces.

(7). Download and Readback Connector
A dynamic download and readback connector is installed for the serial configuration mode. It can be plugged into the Xilinx Xchecker cable, which can then be connected to the machine which is to provide the configuration file.

(8). External Power Connector
An external power connector is used to power the board before the target machine is switched on. After the board is configured, an internal power source from the SBus connector can be used as well.

3. FPGA Related Aspects of Implementation

In order to function as the logic component of the SMB the FPGA needs to be:

- Connected to every SBus signal.
- Able to clock selected signals into the RAM.
- Able to stop the collection of the data when a user determined trigger sequence occurs.
- Programmable by the user.

There are 82 SBus signals, the memory requires 16 address, 32 data lines and 3 control signals. This is a total of 133 signals and as the FPGA has 160 available IO pins these are easily accommodated. All 82 SBus signals are connected to the FPGA including data DA<31..0>, acknowledgment bits ACK<2..0>, physical address PA<27..0>, interrupt request INTREQ<6..0>, transfer size SIZ<2..0>, later error LERR*, data parity DTAPAR, reset RESET*, slave select SEL*, transfer direction Read, address strobe AS*, bus grant BG*, bus request BR*, and system clock CLK.

The Xilinx 4010 chip has adequate CLB's to enable the user to program a finite state machine which will load a word comprising selected signals from the SBus into the RAM each time a clock occurs. The clock can be the system clock or in fact any other signal the user selects.

In addition to the logic required for data collection the Xilinx 4010 chip has adequate CLB's to enable the user to program a finite state machine which will detect the most complex trigger sequence. For example a user could program an FSM which would detect that a sequence of address or data patterns had occurred. When the trigger occurs data collection is suspended and the data collected prior to the trigger is stored in the RAM.

The SMB has a 27256 PROM for master parallel configuration on power up. Through Xchecker cable any other configuration file can be dynamically loaded from the host system to the FPGA at any time.

The FPGA provides 51 onboard signals including data, DAT<31..0>, address ADD<15..0>, memory chip W/R control signals WEN*, OEN*, and CSE* ; 12 multi trigger count signals, i.e. address AN<3..0>, data write WN<3..0>, and data read RN<3..0>; 2 board flag signals, i.e. board working mode CN1 (data trace or display) and trigger mode CN2 (overwriting or reserve); and board FCode PROM control signal FCSE*. The rest belongs to the internal signals.

By default an application is mapped to the FPGA via 17 functional blocks programmed inside the FPGA. This application enables the user to select the 32 bus signals to be monitored together with the trigger. The trigger components which the user can specify include: Address or Data pattern, multiple or single, trace length, and qualifiers. The user can also select the length of time for which the monitoring process will continue before a timeout occurs.

4. Modes of Operation

Since the FPGA is programmable to the gate level this board can be configured to perform a wide variety of operations but it has two basic modes of operation: (1) the configuration mode and (2) the operational mode. When used as an SMB the operational mode can be further subdivided into (2a) the data acquisition mode and (2b) the data display mode.

4.1 Configuration Mode

The SMB has two configuration modes, which are decided by the mode selection switch M0, M1, and M2 in Figure 1. One is a master parallel up or down configuration mode, which is activated through a built-in 27256 PROM whenever the board is powered-up. The whole configuration sequence including clear, initiation, configuration, startup, will finish within a fraction of second before the SBus system is booted. Then the board will allow the SBus system to read back its FCode 2732 PROM with the size of word or byte from the slot address 0, where it is sitting. The board will issue byte acknowledge to each of these accesses until all its ID numbers are read. After the SBus system booting, the SMB becomes a pure slave, and all its attributes appear on

the device information list of the host machine. If the board FPGA is forced to clear its configuration memory by setting its PROG* pin to low level, it will automatically reload its configuration program after this pin goes high again. Through the system software the SMB FCode ID numbers can be read back at any time.

Alternatively the slave serial configuration mode can be used. This is initiated through the Xilinx Xchecker download cable from the host machine. Any suitable configuration programs can be downloaded, as desired, provided they have been well tested by simulation. On the Xchecker menu any configuration program in the library directory can be dynamically be loaded within several seconds before the FPGA's DONE pin goes high to enter the users' transformable hardware architecture. By providing both configuration modes the SMB offers a convenient tool to different users.

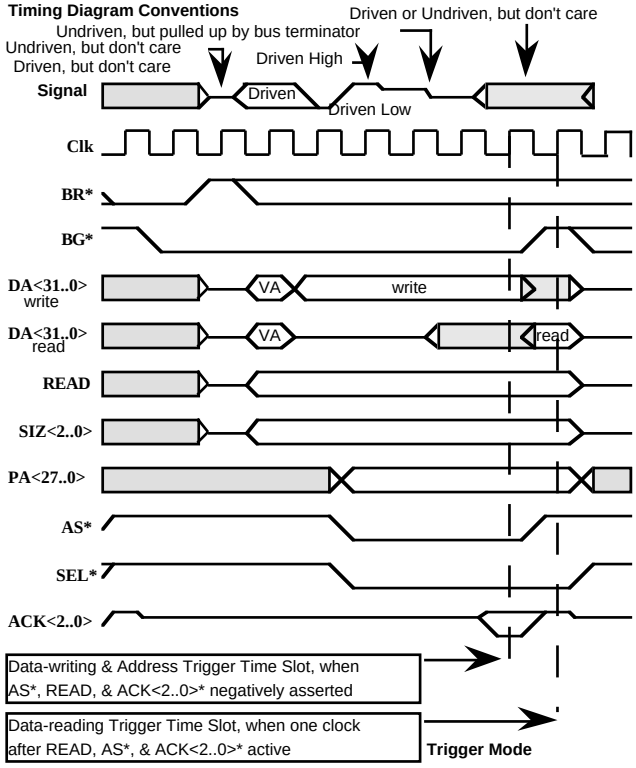


Figure 2

4.2 Data Acquisition Mode

The configuration used to test the SMB provides two working modes, which are interchangeable. One is the data acquisition mode, in which the board selects 32-bits of different combinations of 82-bit SBus signals, and keeps recording the SBus signals in the board memory. The board generates a set of periodical addresses to its own memory. In this way the data stored in the board memory is continually being overwritten as the data is updated. During this data acquisition process comparators in the FPGA search for a preset data or address pattern until a trigger point occurs or a preset timeout is reached. The design of the comparators and all other internal and I/O signals is compatible with the specified SBus AC parameters [11,12].

Data or address words can be used to trigger the SMB and freeze the data collection and these triggers can be qualified by signals such as read and write. Here qualification means that the trigger does not occur when the data or address trigger appears unless the qualifying signal is also asserted. Figure 2, which is an SBus timing diagram illustrates the timing of read and write actions on the bus. In order to ensure that either a read or a write is occurring, the trigger is generated according to the notes on the diagram.

(1) Data-Writing or Address Trigger Mode
If the data from the SBus system are to be written on an SBus slave, they will be kept on the SBus until the system receives a suitable acknowledgment ACK<2..0> from that slave within 255 clock cycles. When a slave generates an acknowledgment during a particular clock cycle of a write, it is acknowledging that the data are on the data lines during that clock cycle. That is, if address strobe, ASI*, at least one bit of transfer acknowledgment ACK<2..0>, and transfer direction, READ, are all negatively asserted, then it is the right time slot in which to compare the preset trigger pattern to see if there is a match.

(2) Data-Reading Trigger Mode
If the data from a slave are being read back to the SBus system, they have to be driven onto the SBus data lines for exactly one clock cycle during the clock cycle immediately following the data acknowledgment. That is, if both address strobe, AS*, and at least one bit of transfer acknowledgment, ACK<2..0>, are negatively asserted, and transfer direction, READ, is positively asserted, then after a one clock cycle delay it will be the right time slot in which to compare the preset trigger pattern to see if there is a match.

(3) Multiple Trigger Mode
Like a logic state analyser the SMB has an arm mode, in which it cannot accept a trigger until it has been armed by multiple trigger cases, which are dependent on previously specified conditions (sequential or parallel address, data-writing, and data-reading triggers). The SMB also has a multiple trigger mode. Of course, any further modifications can easily be made on this reconfigurable board.

If any of the above trigger patterns, together with their preset occurrence times are matched, a trigger flag bit will be set and displayed on the board LED. The SMB will keep its tracing records for a preset number of clock cycles on the SBus before it stops the address counter. Then, the board quits the data acquisition mode, and enters the data display mode. The data then in the on- board memory are

data which occurred on the bus before and after the trigger point.

The SMB is ready for monitoring after its configuration but it is in a standby mode until the multiple control bits are assigned to an addressable header register inside the FPGA. These must be set properly, in accordance with users' requirements, before a startup command can be issued. A suitable header format includes the start, trigger source selection, data multiplexer selection, transfer acknowledgment, size control, timeout, trace length, and address or data patterns with their qualifying bits. On receiving a startup command from the keyboard of the SBus system, these register bits are directly taken out and shifted into the relevant working units one by one, and then the data acquisition mode is activated.

4.3 Data Display Mode

In the data display mode the SMB is idle and it remains so until it receives an SBus system command to read back the captured data for display from its board memory. Alternatively the processor may read back or rewrite to the header register bits inside the FPGA to make changes to the header register bits before another startup command begins another monitoring cycle. System software has been developed to implement board memory and FCode PROM reading, or reading and writing the internal header register.

Thus two working modes reside in one configuration program and the board can multiply enter the data acquisition mode under commands carried to it via the SBus , and automatically return to the data display mode after a trigger point is found or a timeout is reached. There are four buffered LED displays on the SMB to indicate the status of the board. One is on each of the DONE pin, the board working mode, the trigger flag, and the readback after serial download of its configuration program flag. They have been found to be useful to the user as a check on the board's status.

5. Simulation

Simulation has been used to ensure that the FPGA configuration achieves the correct functions and timing before it is downloaded to the SMB. The FPGA is the main component of the SMB and its I/O pins are connected to the 82 signals of the SBus connector within a 2 inch net distance according to the SBus specifications All logic control circuits are configured inside the FPGA, the outputs of which are fully buffered. Simulation is concentrated on the FPGA as the remainder of the board is relatively straight forward. By taking care with the layout of the printed circuit board its correct operation has been assured.

All circuit blocks have been simulated individually and as a whole system. The results meet all preset requirements. The logic work bench RAPIDSIM provides a complete environment and various tools for both functional and time simulations. The test data are created in accordance with the SBus timing diagram. The test system frequency is set at 20 and 25 MHz. At 40 MHz, the SMB can still work under some conditions, but some of its circuit blocks need minor modifications. All simulations are compatible to the SBus specifications, and the data are correctly recorded on the board.

5.1 Unit Delay Simulation

Circuits are divided according to their functions. After each circuit block was designed, its logic output was tested first by using the unit delay simulation, but the effects of the delay can only be estimated at this first stage. When all circuit blocks were finished, they were connected together, and the whole system was tested. It was extremely important to make all the interior signals mutually compatible at this stage as when actual delays, instead of the unit delays were inserted, not all interior signals could be matched.

5.2 Time Delay Simulation

Many changes in the circuits were made before simulation showed that the FPGA and the board were functioning correctly. The structure of some blocks had to be changed several times before they could be routed to meet the timing constraints required for operation on the SBus. For example, a trigger enable signal is created on the precise combination of pattern comparators' outputs, the SBus control signals AS* & RD, and acknowledgment bits ACK<2..0> which lasts only one clock cycle. Because they are routed on different CLBs inside the FPGA, which are of different delays and mismatched, circuit modifications must be made in order to guarantee a logic correctness after routing delays are added during time simulation. All the FPGA pin-outs were suitably assigned, so that connections to the different packages of the board could be short; and net distances as short as possible within a group of chip signals. Nets inside the FPGA were carefully classified as different weight nets. Some are critical; some are not. Every effort was made to ensure that the nets were assigned weighings which reflected their level of criticality with the highest weighting assigned to the most critical nets. In this way for example we could ensure that the SBus signals were correctly stored inside the board memory when they were clocked in by the edge of the system clock.

6. Tests on the SBus system

A test system is created. An SBox expansion chassis [13] is introduced through its connection board onto the SBus system. There are two kinds of tests. One test is on the SBox. In this case an SBox cable is plugged into slot one on the SBus SPARC Workstation. On the SBox chassis the SMB sits on slot 4 on the left side of its motherboard, and a target board is in slot 1 on the right side. According to the SUN SPARC workstation

specification, a 32 Mbyte address space is assigned to each SBus slot. The SBox allocates this space in 16, 8, 4, 4 Mbytes from slot 4 to slot 1 respectively when its jumper JP1 is removed. The startup location is 0xa000000 for the SMB, and 0xbc00000 for the target board.

The other test is on the SBus system. In this case the SMB is plugged into slot three (pure slave) on the SBus SPARC work station 1, while the SBox together with the target board remains in the same positions. Thus, the SMB startup location on the SBus is 0xe000000.

A test software package was developed to implement different functions on the SMB and target board. Two test examples are reported on here. The first uses an address pattern qualified by a read to trigger the freezing of the data and the second uses a data pattern qualified by a write to trigger the freezing of the data. After each test the data was displayed and analysed.

6.1 Triggered by an Address Pattern qualified by a read

Figure 3 shows the result obtained when the trigger was an address word qualified by a read. In this diagram, which is taken from the screen display provided by the software associated with the SMB, the cursor is set to the trigger point. The display then shows what has happened on the bus after the trigger. The address 0xbc00000 is the trigger and the read is asserted so when AS* the address strobe and ACK the acknowledgment are asserted the data in the memory has been frozen. As the user is interested in the activity on the bus after the trigger data has been collected for several clock cycles after the trigger and these samples have been used to provide the display shown in the diagram. The particular read is a 4 byte word where from a single instruction the processor reads four consecutive bytes from the target board memory. The first byte 0xfd is read at 0xbc00000, the second 0x00 at 0xbc00001, the third 0x10 at 0xbc00002, and the fourth 0xb7 at 0xbc00003. The address strobes and acknowledgments for each byte read can be seen in the diagram. The SBox from which these traces were taken then takes 86 clock cycles to pullup all of its data lines.

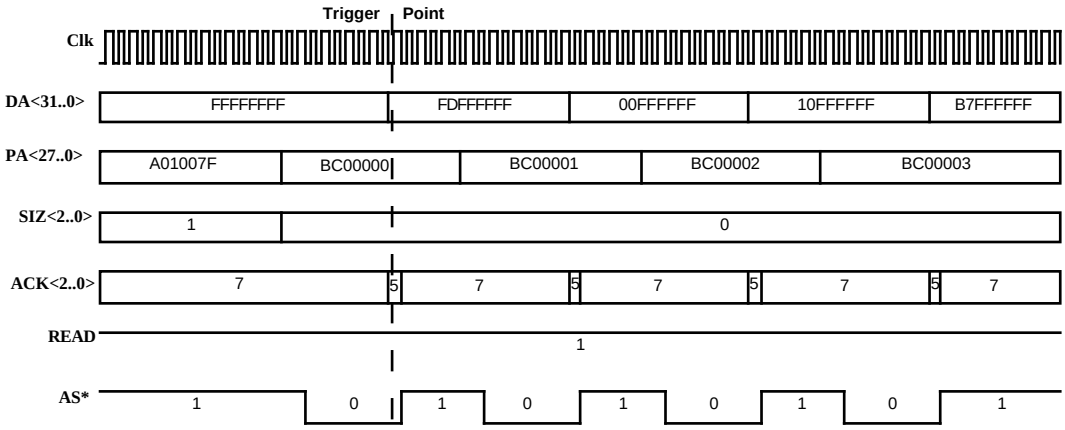


Figure 3

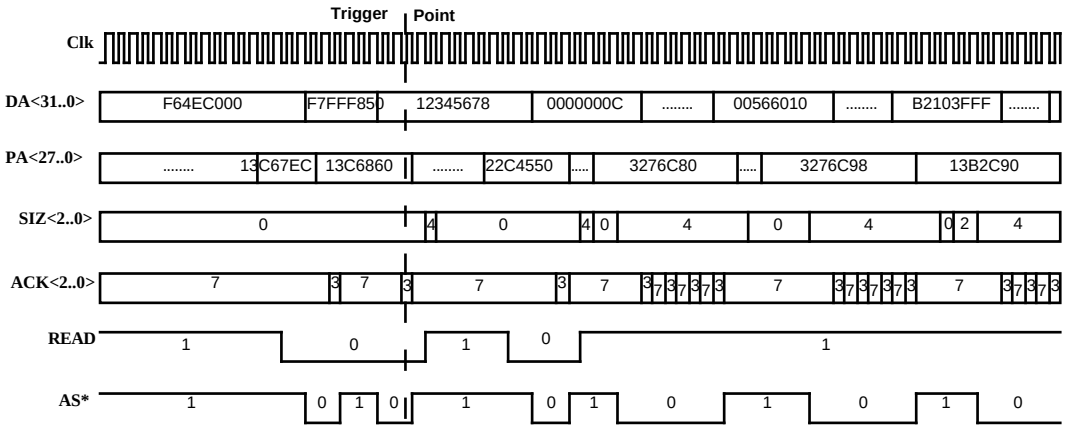


Figure 4

6.2 Triggered by a Data Pattern qualified by a write

In Figure 4 the cursor again shows the trigger point which is at the beginning of the time slot the data appears in because it is qualified by the write, rather than at the end of it as in the previous case where it was qualified by the read signal. The pattern of the trigger is DA(31..0) = 0x12345678 and it can be seen from the diagram that this is written into address location 0x13c6860 which is in the main memory of the host. It can be seen that the trigger and the write to the host memory is followed by a four word burst read which is unrelated to the data transfer to the SBox.

6.3 Results and Discussion

These are quite simple examples, deliberately chosen to be so to enable them to be easily understood, but they illustrate the point that the SMB can be used to determine in intimate detail what is happening on the bus.

In the first test example the SBox control signal, such as transfer direction Read reflects the global activities in the SBus system while the others, such as data DA(31..0), physical address PA(27..0), acknowledgment bits ACK(2..0), transfer size SIZ(2..0), and address strobe AS*, are only related to the SBox system in some sense. The SBox bus activity is dedicated to its own, not to the whole SBus system. After each data transfer the SBox data lines are pulled up one after another while address lines are held for quite long periods of time before another data transfer occurs.

From the second test example it is observed that each data transfer on the SBus is not an isolated incident, but is associated with other data transfer activities. This is characteristic of SBus operations which carry the signal, generated by the operating system in addition to the application programs' signals. In fact further test found that the operation of writing a word to the SBox did not complete until more than 8000 clock cycles after the operation started. Other tests also showed that each word-reading or word-writing on the target board is accompanied by other SBus operations with the same data.

7. Further Studies

Now that we have established the operation of this device, it is our intention to extend its applications into the following areas.

7.1 Single Process SBus Monitor

The data recorded on the SMB in the example above show the multiple processes which are time shared by the single processor. Although the SBus system dynamically allocates the RAM physical address for different processes, it is still possible to find and trace this kind of dynamically allocated address to follow a particular process by appropriate qualifiers. Thus the board will only register the data related to a single preset process and the activities belonging to other processes will be excluded. This is a very effective way to expand the length of the trace without increasing any memory capacity of the board.

7.2 A Timing Study on Software & Hardware Implementations of ATM Adaptation and Physical Layers in Multimedia Terminals

Various measurement-specific circuits on this FPGA-based SMB can be configured to meet ATM protocol and transfer requirements. Normally the protocol header and data are measured through the system software, which is slow, and will affect the SBus system. The SMB now makes it possible to register and examine ATM packets as they appear on the bus through the SMB hardware. This leads to improvements in the speed of operation of ATM networks.

7.3 Digital Signal Processing

There are many systems where a DSP chip and a microprocessor operate on the same bus to optimise the overall operation. It is possible to configure the SMB as a coprocessor which can perform the same functions as a DSP chip and apply it to applications in the digital signal processing area. Digital filters is but one of the applications currently under investigation.

8. Summary and Conclusion

An SMB has been described which can be used to monitor the operations on a host computer's bus. The host computer is used to develop the configuration file downloaded to the FPGA in the monitor board, control the operation of the SMB and display the results. Examples which illustrate the board's application have been shown and brief descriptions of further projects which will use this board have been given.

9. References

[1] 1607A Logic State Analyser Hewlett Packard 1976
1600A Logic State Analyser Hewlett Packard 1975
[2] C. S. Choy, et al., "A Hardware Prototyping Board for Complex Digital Design", IEEE Region 10 Conference Tencon 92 pp.191-195, Nov. 1992
[3] SBT-225 Single Width SBus Analyzer, Vmetro, US, 1992
[4] T. Horikawa, "TOPAZ Hardware-Tracer Based Computer Performance Measurement and Evaluation System", NEC Res. & Develop, Vol. 33 No.4, pp. 638-647, Oct. 1992
[5] S. Casselman, "Virtual Computing and The Virtual Computer", IEEE Workshop on FPGAs for Custom Computing Machines, Napa, Ca. pp. 43-48, Apr. 1993

[6] E. David, et al., “AnyBoard: An FPGA-Based, Reconfigurable System”, IEEE Design & Test of Computers, pp.21-29, Sep. 1992
[7] The Programmable Logic Data Book, Xilinx Inc., 1994
[8] User Guide, Reference Cadence, 1990-1993
[9] Open Boot PROM Toolkit User’s Guide, Sun Microsystems, Inc., 1990
[10] Writing FCode Programs for SBus Cards, Sun Microsystems, Inc., 1990
[11] SBus Specification A2, Writing SBus Device Drivers, Sun Microsystems, Inc., 1990
[12] J. D. Lyle, “SBus Information, Applications, and Experience”, Springer-Verlag, 1993
[13] SBox Expansion Chassis User’s Manual, Aurora Technologies, Inc., 1992