(12) UK Patent Application (19) GB (11) 2 399 908 (13) A

(43) Date of A Publication 29.09.2004

(21) Application No: 0405429.2

(22) Date of Filing: 10.03.2004

(30) Priority Data:
(31) 10402092 (32) 28.03.2003 (33) US

(71) Applicant(s):
Hewlett-Packard Development Company L.P.
20555 S.H. 249, Houston, Texas 77070,
United States of America

(72) Inventor(s):
Tyler James Johnson
Theodore Carter Briggs

(74) Agent and/or Address for Service:
Carpmaels & Ransford
43 Bloomsbury Square, LONDON,
WC1A 2RA, United Kingdom

(51) INT CL[7]:
G06F 11/22 , G01R 31/317 , G06F 13/40

(52) UK CL (Edition W):
G4A AFMG AFMW

(56) Documents Cited:
EP 0892352 A1 EP 0727747 A2

(58) Field of Search:
UK CL (Edition W) G4A
INT CL[7] G01R, G06F
Other: ONLINE: EPODOC, WPI, JAPIO

(54) Abstract Title: **A bus interface module for a debug bus**

(57) A bus interface module (BIM) 104 connectable to a debug bus 100 is described. In one embodiment, the BIM 104 comprises a plurality of bus segments 102(0)-102(4) connected in a ring such that an output of each BIM 104 is connected to an input of a next BIM 104 via the debug bus 100, the BIM (104) comprising logic 210 for receiving data from a previous BIM 104, logic 206 for receiving data from local logic associated with the BIM 104, and logic 212 for combining the previous BIM 104 data with local logic data and transmitting the combined data to a next BIM 104.

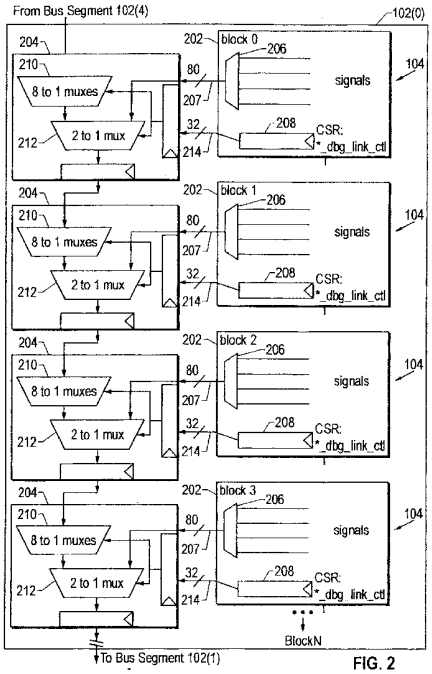

FIG. 2

GB 2 399 908 A

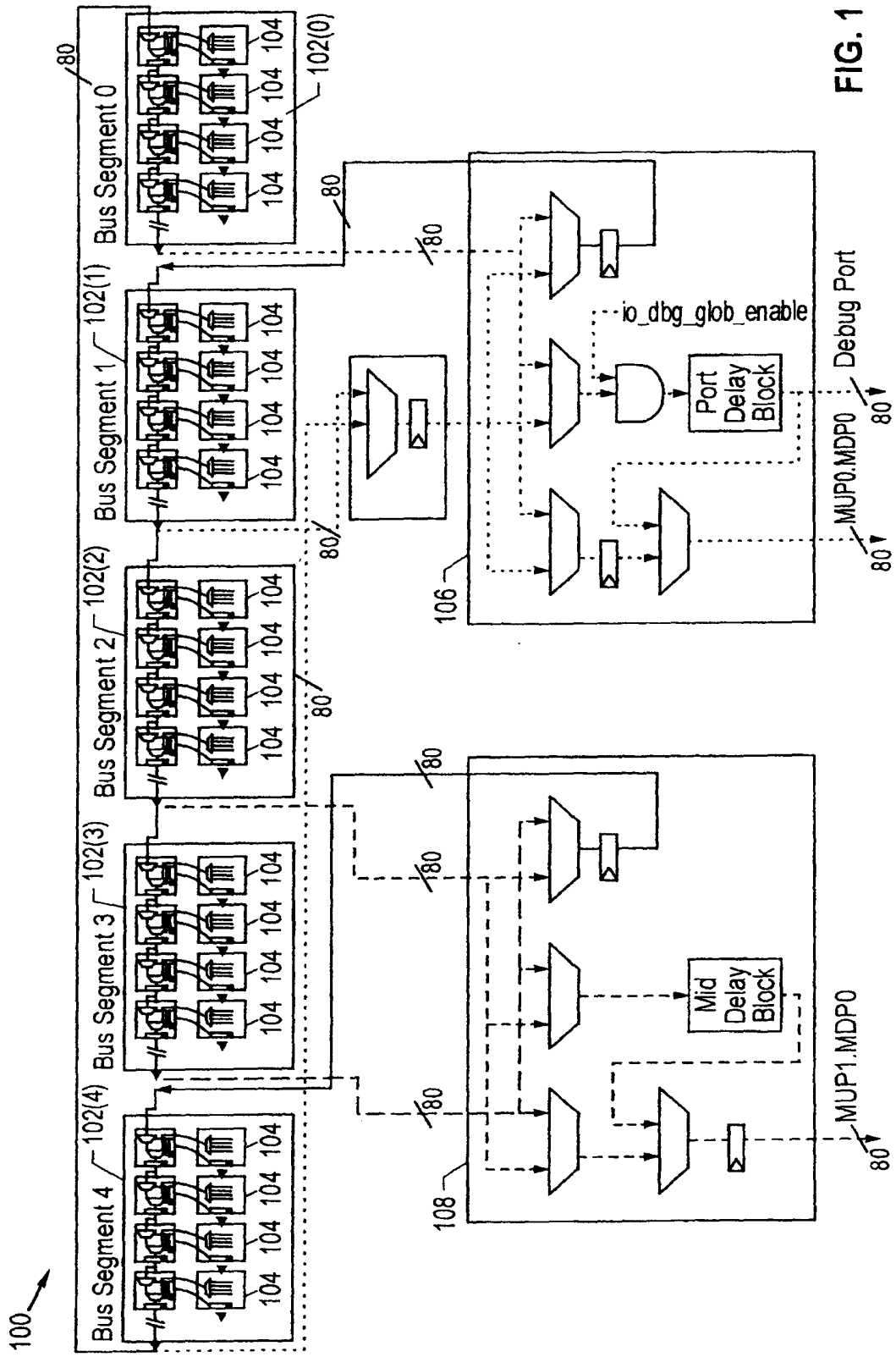
100
Bus Segment 0
Bus Segment 1
Bus Segment 2
Bus Segment 3
Bus Segment 4
102(0)
102(1)
102(2)
102(3)
102(4)
104
80
106
108
io_dbg_glob_enable
Port Delay Block
Mid Delay Block
Debug Port
MUP0.MDP0
MUP1.MDP0

FIG. 1

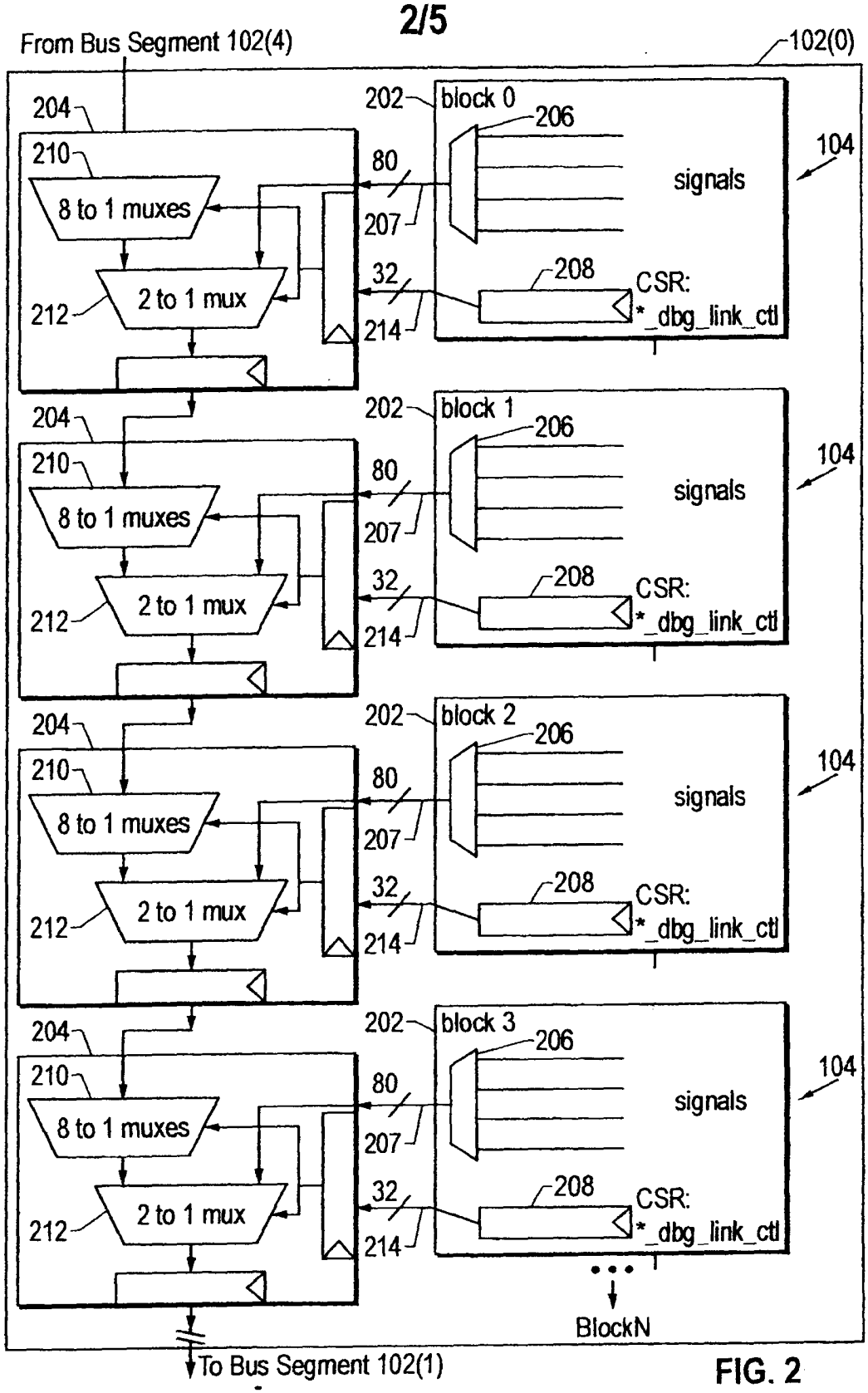
From Bus Segment 102(4)
102(0)
204
210
8 to 1 muxes
212
2 to 1 mux
202
block 0
206
80
207
signals
104
32
214
208
CSR:
*_dbg_link_ctl
block 1
block 2
block 3
BlockN
To Bus Segment 102(1)

FIG. 2

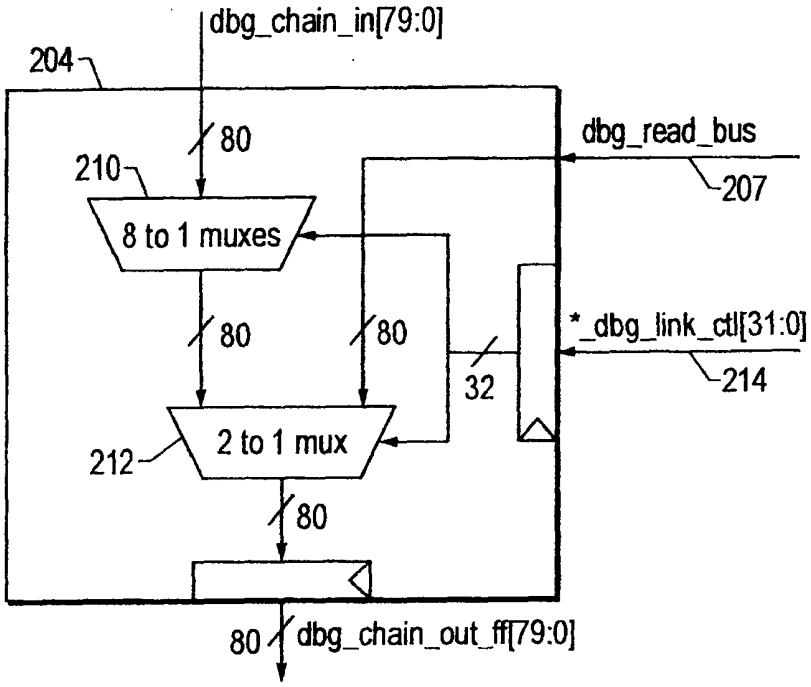
dbg_chain_in[79:0]
204
80
210
8 to 1 muxes
dbg_read_bus
207
80
80
*_dbg_link_ctl[31:0]
214
32
212
2 to 1 mux
80
80
dbg_chain_out_ff[79:0]

FIG. 3

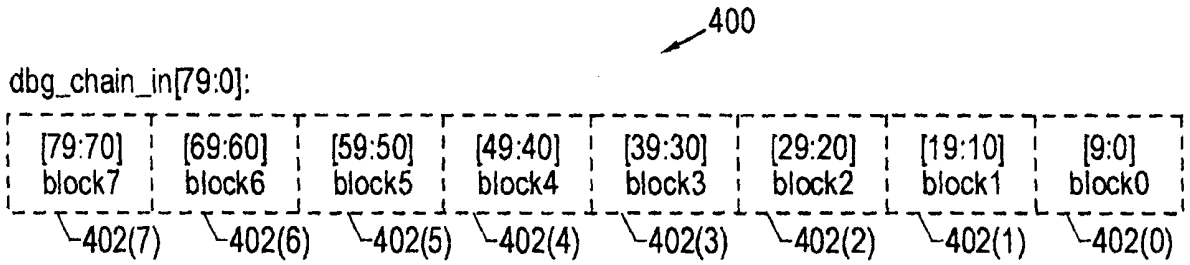
400
dbg_chain_in[79:0]:
[79:70] block7
[69:60] block6
[59:50] block5
[49:40] block4
[39:30] block3
[29:20] block2
[19:10] block1
[9:0] block0
402(7)
402(6)
402(5)
402(4)
402(3)
402(2)
402(1)
402(0)

FIG. 4A

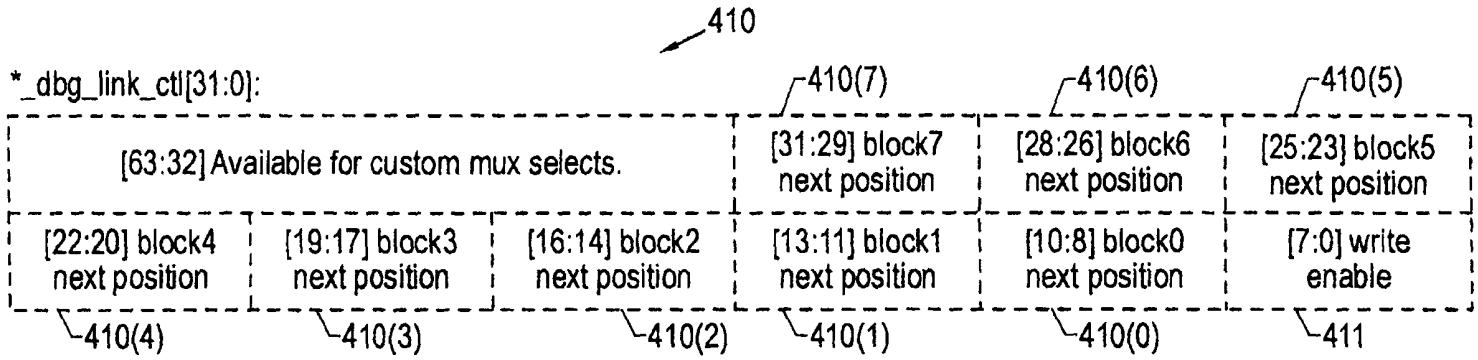
410
*_dbg_link_ctl[31:0]:
410(7)
410(6)
410(5)
[63:32] Available for custom mux selects.
[31:29] block7 next position
[28:26] block6 next position
[25:23] block5 next position
[22:20] block4 next position
[19:17] block3 next position
[16:14] block2 next position
[13:11] block1 next position
[10:8] block0 next position
[7:0] write enable
410(4)
410(3)
410(2)
410(1)
410(0)
411

FIG. 4B

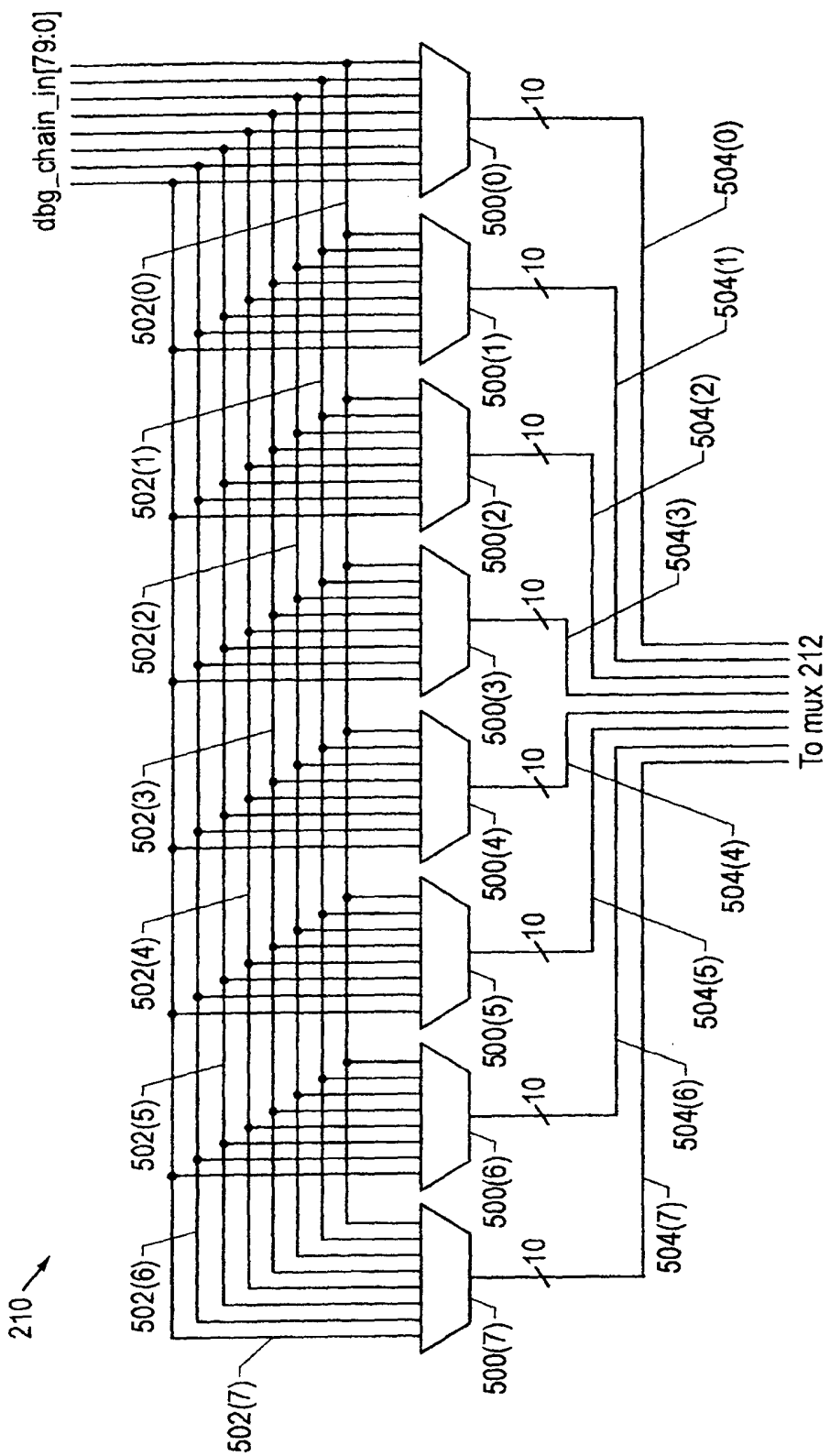
dbg_chain_in[79:0]
210
502(0)
502(1)
502(2)
502(3)
502(4)
502(5)
502(6)
502(7)
500(0)
500(1)
500(2)
500(3)
500(4)
500(5)
500(6)
500(7)
10
504(0)
504(1)
504(2)
504(3)
504(4)
504(5)
504(6)
504(7)
To mux 212

FIG. 5

A BUS INTERFACE MODULE

CROSS-REFERENCE TO RELATED APPLICATIONS

[0001] This application is related to U.S. Patent Application Serial No. 10/402,034, filed 28 March 2003 entitled AN INTEGRATED CIRCUIT (Docket No. 200209004-1); U.S. Patent Application Serial No. 10/402,628 filed 28 March 2003 entitled SYSTEM AND METHOD FOR USING A DEBUG BUS AS A CAPTURE BUFFER (Docket No. 200208677-1); and U.S. Patent Application Serial No. _____, filed _____ entitled SYSTEM AND METHOD FOR VERIFYING HDL EVENTS (Docket No. 200208679-1), all of which are hereby incorporated by reference in their entirety.

BACKGROUND

[0002] The increasing complexity of system designs, increased investment required due to this complexity, and shortened product cycles have presented significant challenges to post-silicon design verification of chipsets. This is especially true with respect to high-end cache coherent non-uniform memory access ("ccNUMA") chipsets where systems can be extremely large and complex. Processor post-silicon verification is typically focused on electrical verification at least as much as functional verification due

to the large amount of full custom design. Chipsets present a different challenge due to the large number of cells of which they are comprised. Additionally, due to the sheer number of buses, internal bus arbitration, cache coherency control, queue arbitration, etc., in a large ccNUMA server, post-silicon functional verification of such a chipset consumes a greater amount of resources with respect to electrical verification than processors typically consume. Internal observability, while relatively simple in pre-silicon verification, poses a major obstacle to debug and functional test coverage.

[0003] Determining when system verification is complete is a second major obstacle to completing post-silicon verification in a time-effective manner. While pre-silicon simulation-based testing depends significantly on labor intensive directed and pseudo-random testing, post-silicon testing has historically depended on observing system operations that imply correct behavior.

[0004] Performing post-silicon design verification is an industry standard practice that facilitates exposure of bugs not typically uncovered in pre-silicon verification. Typical post-silicon bugs discovered include those that are manifested after long or at-speed operation of the system, those resulting due to incorrect modeling of hardware and firmware interfaces, those resulting from Register-Transfer Language ("RTL") errors that escaped pre-silicon detection, and those resulting from incorrect mapping of RTL-to-silicon (synthesis/physical bugs). Accepted methods of exercising systems to expose post-silicon bugs include running operating systems and software applications targeted for the final

system, creating specific directed software tests that stress different portions of the system, and running software tests that create random system operations.

[0005] Real-time observability ("RTO") refers to the ability to monitor and capture internal signals in real time either on- or off-chip. While internal signal observability features have been available in some field programmable gate array ("FPGA") architectures and application specific integrated circuits ("ASICs"), they have typically been of limited scope. Limiting factors have been silicon area, wiring constraints, and I/O limitations. In addition, observability features have traditionally been used for debug and not functional test coverage.

SUMMARY

[0006] A bus interface module ("BIM") connectable to a debug bus is described. In one embodiment, the BIM comprises a plurality of BIM segments connected in a ring such that an output of each BIM is connected to an input of a next BIM via the debug bus, the BIM comprising logic for receiving data from a previous BIM, logic for receiving data from local logic associated with the BIM, and logic for combining the previous BIM data with local logic data and transmitting the combined data to a next BIM.

BRIEF DESCRIPTION OF THE DRAWINGS

[0007] FIG. 1 is a block diagram of a debug bus of one embodiment;

[0008] FIG. 2 is a block diagram of a bus segment of the debug bus of FIG. 1;

[0009] FIG. 3 is a block diagram of a standard logic block used to implement the bus segment of FIG. 2;

[0010] FIG. 4A illustrates the format of the dbg_chain_in signal of the debug bus of FIG. 1;

[0011] FIG. 4B illustrates the format of the *_dbg_link_ctl signal of the debug bus of FIG. 1; and

[0012] FIG. 5 is a block diagram of logic structure implemented in the standard logic block of FIG. 3 for swapping blocks of data.

DETAILED DESCRIPTION OF THE DRAWINGS

[0013] In the drawings, like or similar elements are designated with identical reference numerals throughout the several views thereof, and the various elements depicted are not necessarily drawn to scale.

[0014] As illustrated in FIG. 1, in accordance with one embodiment, a debug bus 100 comprises a plurality of bus segments 102(0)-102(4) interconnected in a serial ring that runs at the core clock speed of an IC, e.g., an ASIC, in which the bus is implemented. In one implementation, the debug bus 100 is 80-bits wide; however, in general, the width of the debug bus is consistent with device pin constraints. Moreover, although the illustrated embodiment employs only five bus segments 102(0)-102(4), it will be appreciated that greater or fewer than five bus segments may be implemented as necessary for providing appropriate logical and physical partitioning.

[0015] Each bus segment 102(0)-102(4) comprises several access points 104 at which data from surrounding logic is MUXed onto the debug bus 100. As will be described in

greater detail below with reference to FIGs. 3 and 4, each access point 104 comprises a standard logic block with a proprietary MUX structure that drives debug data into the access point, which subsequently drives the data onto the debug bus 100.

[0016] As illustrated in FIG. 1, two observability ports 106, 108 are defined. In one embodiment, one of the ports, i.e., port 106, is a dedicated debug port. The other port, i.e., port 108, is loaded with functional signals. The debug bus 100 contains debug data that drives both of these ports 106, 108. In one implementation, the debug port 106 has 80 data pins, plus four strobe pins that are single pumped, with the intention that the port 106 be connected directly to a logic analyzer (not shown).

[0017] As previously indicated, the debug port 106 is fed directly from the debug bus 100, which runs at core clock speed and connects the bus segments 106 in a serial ring. The debug bus 100 is segmented so that for any of a plurality of functional areas of an IC in which the bus is implemented, packets to and from the area can be observed in addition to 80 bits of internal state data. Additional details regarding implementation and operation of the debug bus 100 and ports 102, 104 are provided in commonly-assigned, co-pending U.S. Patent Application Serial No. 10/402,034, filed 28 March 2003, entitled AN INTEGRATED CIRCUIT (Docket No. 200209004-1), which has been incorporated by reference in its entirety hereinabove.

[0018] FIG. 2 is a more detailed block diagram of the bus segment 102(0) of the debug bus 100 illustrated in FIG. 1. As illustrated in FIG. 2, the bus segment 102(0) includes a

plurality of access points 104. It should be noted that although only four access points 104 are shown, each bus segment 102(0)-102(4) may comprise greater or fewer access points as necessitated by the number of signals that must be handled by the bus segment.

[0019] As shown in FIG. 2, each access point 104 includes a local data intake section 202 and a corresponding debug bus interface block ("DBIB") 204 connected thereto. The DBIB structure may also be referred to as a logic module structure or a Bus Interface Module ("BIM") segment. At each access point 104, up to 80 bits of data from surrounding logic ("dbug_read_bus") is provided to the DBIB 204 thereof via a MUX 206 along a bus 207. A control and status register ("CSR") 208 provides a 32-bit MUX select signal ("*_dbg_link_ctl") to MUXes 210, 212 of the corresponding DBIB 204 via a bus 214 for purposes that will be described in greater detail below.

[0020] FIG. 3 is a more detailed block diagram of one of the DBIBs 204 of FIG. 2. In one embodiment, the debug bus 100 is logically divided into eight 10-bit blocks. Each DBIB 204 can move and/or replicate incoming debug bus data ("dbg_chain_in") from the previous DBIB in the chain in these 10-bit blocks to make room for incoming data ("dbg_read_bus") from the corresponding local data intake section 202, if necessary, and pass the newly configured data ("dbg_chain_out") to the next DBIB 204 in the chain. Generally, each DBIB 204 performs the following three functions: (1) it passes data on from the previous access point, (2) it moves 10-bit blocks of data from the previous access point to other ranges of the debug bus, allowing for

more efficient bandwidth utilization; and (3) it MUXes in data from surrounding logic in 10-bit chunks.

[0021] As previously indicated, to make MUXing of data manageable, the debug bus 100 is logically divided into eight 10-bit blocks, each of which can be individually manipulated. Further, each access point 104 registers data from the previous access point, rearranges the data from previous access point in 10-bit blocks as specified by the corresponding CSR signal("*_dbg_link_ctl"), and MUXes in local data to be passed on to the next access point.

[0022] FIG. 4A illustrates the format of the dbg_chain_in signal, designated in FIG. 4A by reference numeral 400. As previously described, in one embodiment, the signal 400 comprises eight 10-bit data blocks 402(0)-402(7) which can be individually manipulated as will be described in greater detail below. Although not shown, it will be recognized that the formats of the dbg_chain_out and dbg_read_bus signals are substantially identical to that of the dbg_chain_in signal 400. FIG. 4B illustrates the format of the *_dbg_link_ctl signal, designated in FIG. 4B by reference numeral 410. In one embodiment, the first eight bits of the signal 410, designated by reference numeral 411, form a write enable signal for the MUXes 210, 212. Eight three-bit fields 412(0)-412(7) are used to specify the block position of each of the eight blocks of data of the dbg_chain_in signal in the dbg_data_out chain. In particular, the contents of the field 412(0) (bits 8-10) specify the position in the dbg_chain_out signal of the first block 402(0) of the dbg_chain_in signal 400; the contents of the field 412(1) (bits 11-13) specify

the position in the dbg_chain_out signal of the second block 402(1) of the dbg_chain_in signal 400; and so on.

[0023] FIG. 5 is a more detailed block diagram of the swap structure of the MUX 210. As illustrated in FIG. 5, data comprising each of the eight 10-bit blocks 402(0)-402(7) of the dbg_chain_in signal 400 are input via eight 10-bit wide lines 502(0)-502(7), respectively, to all of *N* *N*x1 MUXes 500(0)-500(7). In the illustrated embodiment, the value of *N* is eight; however, it will be understood that the value of *N* will be dependent upon the width of the debug bus and the granularity with which the bus is subdivided. Each of the MUXes 500(0)-500(7) outputs via a 10-bit line 504(0)-504(7), respectively, any one of the data blocks 402(0)-402(7) input as specified by the value of the control signal *_dbg_link_ctl.

[0024] For example, the first block (block 402(0)) of the signal 400 could be shifted to the fourth block position of the dbg_chain_out signal by outputting it from the MUX 500(3). Additionally, a selected block of data may be replicated by outputting it from multiple ones of the MUXes 500(0)-500(7). In the foregoing manner, the structure illustrated in FIG. 5 enables any 10-bit block of data at the input of the MUXes 500 to be moved to any other block at the MUX outputs 502(0)-502(7) simply by selecting one or more of the MUXes 500(0)-500(7) to output that data block.

[0025] Accordingly, it will be recognized that the disclosed logic structure for moving data involves a plurality of MUXes, where the number of MUXes is equal to the width of the debug bus divided by the width of each of the data blocks. In general, the embodiment described herein

defines a standardized logic block that can be used to implement data bus that is then routed to internal and external debug observability equipment. Internal equipment can be a logic analyzer intellectual property macro or equivalently any logic that processes data and can be read/controlled by software. The standardized logic block includes staging registers for timing closure and MUXes that enable data to be combined in order to allow users to extract data in compact blocks that effectively facilitate debug, performance analysis, and post-silicon test coverage. Any data can be moved anywhere else on the bus, with the caveat that data is moved with a granularity such that the MUX implementation is physically small enough to be reasonable. Advantages of the embodiments described herein include that, due to standardization, the design is easier to configure and use and fewer MUX designs mean that the proper operation of the overall embodiment is easier to verify.

[0026] An implementation of the invention described herein thus provides a bus interface module for enabling real-time observability in an IC. The embodiments shown and described have been characterized as being illustrative only; it should therefore be readily understood that various changes and modifications could be made therein without departing from the scope of the present invention as set forth in the following claims. For example, while the embodiments are described with reference to an ASIC, it will be appreciated that the embodiments may be implemented in other types of ICs, such as custom chipsets, Field Programmable Gate Arrays ("FPGAs"), programmable logic devices ("PLDs"), generic array logic ("GAL") modules, and the like. Furthermore, while the embodiments shown are implemented using CSRs, it will be

appreciated that control signals may also be applied in a variety of other manners, including, for example, directly or may be applied via scan registers or Model Specific Registers ("MSRs").

[0027] Additionally, although the embodiments described herein anticipate that the data from the previous DBIB will be shifted to allow data from local logic to be placed on the bus, it will be recognized that the local debug data could be moved, e.g., using a structure such as that illustrated in FIG. 5, with the data from the previous DBIB being left as is.

[0028] Accordingly, all such modifications, extensions, variations, amendments, additions, deletions, combinations, and the like are deemed to be within the ambit of the present invention whose scope is defined solely by the claims set forth hereinbelow.

CLAIMS

1. A bus interface module ("BIM") (104) connectable to a debug bus (100) comprising a plurality of BIM segments (102(0)-102(4)) connected in a ring such that an output of each BIM (104) is connected to an input of a next BIM (104) via the debug bus (100), the BIM (102) comprising:

logic (210) for receiving data from a previous BIM (102);

logic (206) for receiving data from local logic associated with the BIM (104); and

logic (212) for combining the previous BIM (104) data with local logic data and transmitting the combined data to a next BIM (104).

2. The BIM (104) of claim 1 wherein the logic (210) for receiving data from a previous BIM (104), logic (206) for receiving data from local logic, and logic (212) for combining each comprise a plurality of multiplexers (206, 210, 212).

3. The BIM (104) of claim 1 or claim 2 wherein the data received from a previous BIM (104) and the data received from local logic are logically subdivided into individually-manipulable data blocks.

4. The BIM (104) of claim 3 further comprising logic (204) for replicating a selected block of data.

5. The BIM (104) of claim 3 or claim 4 further comprising logic (204) for moving data received from the previous BIM (104) in a first block position to a second block position.

6. The BIM (104) of claim 5 wherein the logic (204) for moving data comprises *N* *N*x1 multiplexors (210), each input of the *N* *N*x1 multiplexors (210) operating to accept one block of the data received from a previous BIM (104).

7. The BIM (104) of claim 6 wherein the value of *N* is equal to the width of the debug bus (100) divided by the width of each of the data blocks.

8. The BIM (104) of claim 5, claim 6, or claim 7, further comprising circuitry (208) for providing a control signal specifying a block position of each block of data received from the previous BIM (104) to the logic (204) for moving data.

9. The BIM (104) of claim 8 wherein the circuitry (208) for providing a control signal comprises a control status register ("CSR").

10. The BIM (104) of claim 5, claim 6, claim 7, claim 8, or claim 9, further comprising logic (204) for transmitting at least a portion of the data received from local logic in the first one of the block positions.

Application No:	GB0405429.2	**Examiner:**	Dr Stephen Richardson
Claims searched:	All	**Date of search:**	15 July 2004

Patents Act 1977: Search Report under Section 17

Documents considered to be relevant:

Category	Relevant to claims	Identity of document and passage or figure of particular reference
X	1 at least	EP 0727747 A2 (SUN MICROSYSTEMS) see Figure 3A and abstract.
A	-	EP 0892352 A1 (BULL HN INFORMATION SYSTEMS) see Figure 1 and abstract.

Categories:

X	Document indicating lack of novelty or inventive step	A	Document indicating technological background and/or state of the art.
Y	Document indicating lack of inventive step if combined with one or more other documents of same category.	P	Document published on or after the declared priority date but before the filing date of this invention.
&	Member of the same patent family	E	Patent document published on or after, but with priority date earlier than, the filing date of this application.

Field of Search:

Search of GB, EP, WO & US patent documents classified in the following areas of the UKC[W] :

G4A

Worldwide search of patent documents classified in the following areas of the IPC[07]

G01R; G06F

The following online and other databases have been used in the preparation of this search report

ONLINE: EPODOC, WPI, JAPIO