CHAPTER 6

J-Bus Interface

This chapter contains the following topics about the J-Bus interface (JBI) functional block:

6.1 Functional Description

For a detailed description on the external J-Bus interface, refer to *OpenSPARC T1 Processor External Interface Specification*. The OpenSPARC T1 J-Bus interface (JBI) block generates J-Bus transactions and responds to external J-Bus transactions.

The JBI block:

- Interfaces with following blocks in an OpenSPARC T1 processor:
  - L2-cache (scbuf and sctag) to read and write data to L2-cache
  - I/O Bridge (IOB) - for programmed input/output (PIO), interrupts, and debug port
  - J-Bus I/O pads
- Most of the JBI sub-blocks use the J-Bus clock, and remaining part runs at the CPU core clock or *cmp clk*. The data transfer between the two clock domains is by way of queues within the two clock domains, these are the Request header queues and the Return data queues. The interface to the L2-cache is through the direct memory access (DMA) reads and DMA writes.
- The IOB debug port data is stored in the debug FIFOs and then it is sent out to the external J-Bus.
- IOB PIO requests are stored in the PIO queue and the return data is stored in the PIO return queue. Similarly, there is an interrupt queue and an interrupt ACK/NACK queues in the JBI in order to interface to the IOB.

- There are only two sub-blocks in the JBI (J-Bus parser and J-Bus transaction issue) specific to J-Bus. All of the other blocks are J-Bus independent. J-Bus independent blocks can be used for any other external bus interface implementation.

FIGURE 6-1 displays the JBI block diagram.

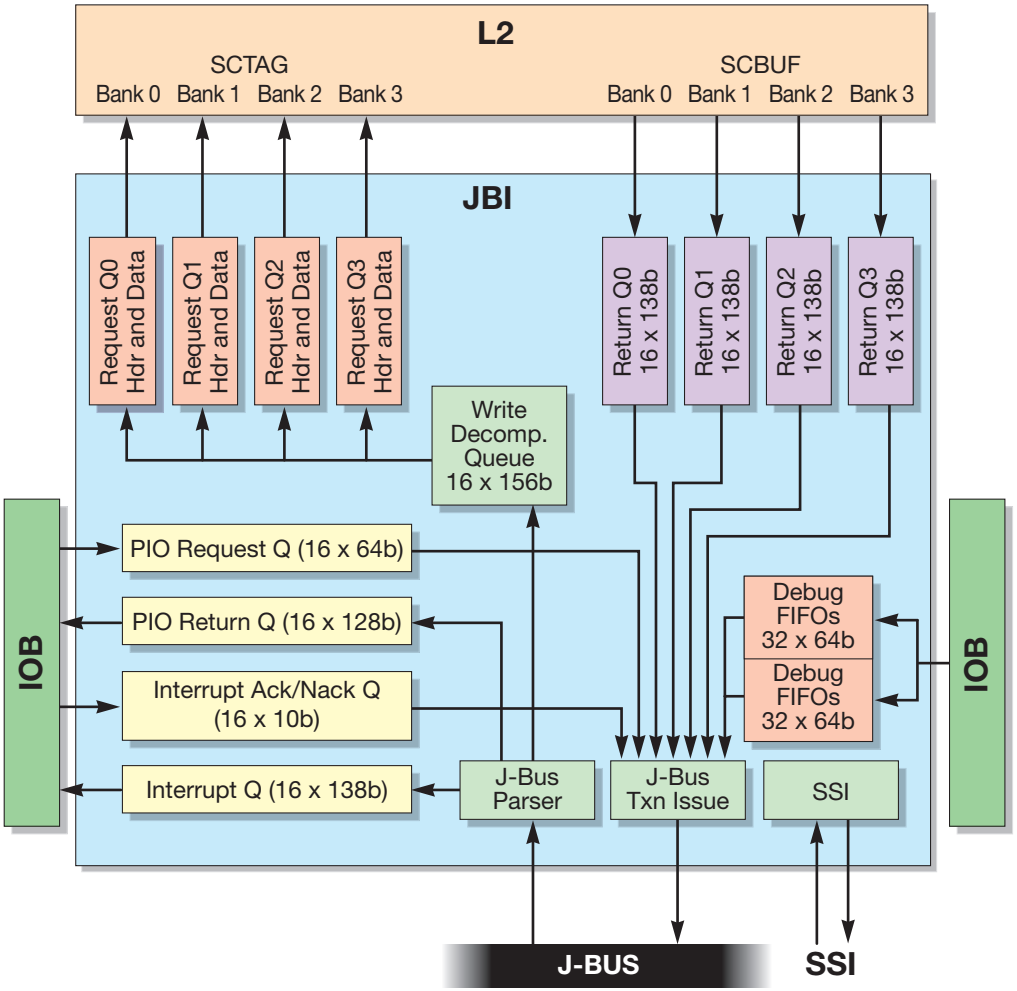

FIGURE 6-1 JBI Functional Block Diagram

The following sub-sections describe the various JBI transactions and interfaces from the JBI to the other functional blocks.

6.1.1 J-Bus Requests to the L2-Cache

There are two types of requests from J-Bus to L2 – read and write.

6.1.1.1 Write Requests to the L2-Cache

DMA write request from J-Bus is parsed by J-Bus parser and then it passes the information to the Write Decomposition Queue, which will then send Request Header and Data to sctag of L2-cache.

- The following types of writes are supported (refer to the *OpenSPARC T1 External Interface Specification* for details of the transaction types):

1. WriteInvalidate (WRI), WriteInvalidateSelf (WRIS), NonCachedWriteComressible (NCBWR) are treated as 64-byte writes
2. NCWR is treated as 8-byte write
3. WriteMerge (WRM):
   - WRM is similar to WRI but with 64-bit Byte enables, supporting 0 to 64-byte writes.
   - Multiple 8-byte write requests (WR8) to the L2-cache

- Write decomposition
  - WRM is broken into 8-byte write requests (WR8) and sent to the L2-cache at the head of the write decomposition queue (WDQ)
  - Number of requests is dependent on the WRM byte enable pattern
  - Each WR8 request writes 1 to 8 contiguous bytes
  - If a run of contiguous bytes crosses an 8-byte address boundary, two WR8s are generated
  - A WRM transaction can generate up to 32 WR8s to the L2-cache
- Writes to the L2-cache may observe strict ordering with respect to the other writes to the L2-cache (software programmable)

6.1.1.2 Read Requests to the L2-Cache

A DMA read request from the J-Bus is parsed by the J-Bus parser and then the information is passed to the write decomposition queue (WDQ), which will then send the request header to the sctag of the L2-cache. Data returned from the L2-cache scbuf is then passed from the return queues to the J-Bus transaction issue, and then to the J-Bus.

- Type of reads supported:
  - ReadToDiscard (RDD), ReadToShare (RDS), ReadToShareAlways (RDSA), NonCachedBlockRead (NCBRD) translates to 64-byte RDDs to the L2-cache
  - NonCachedRead (NCRD) translates to 8-byte RDD to the L2-cache
  - There is a maximum of 4 outstanding reads to each L2-cache bank
- Reads to the L2-cache may observe strict ordering with respect to writes to the L2-cache (software programmable)

6.1.1.3 Flow Control

WDQ gives backward pressure to the J-Bus when the programmable high watermark has been reached. Credit based flow control exists between the JBI and the L2-cache, arising from the L2-cache's two-entry snoop input buffer and the four-entry RDMA write buffer.

6.1.2 I/O Buffer Requests to the J-Bus

Write requests (NCWR) can be 1, 2, 4, or 8-byte writes and those writes are aligned to size. Write request comes from the I/O buffer (IOB), gets stored in the PIO request queue, and then goes out on the J-Bus.

Read requests comes from IOB, gets stored in the PIO request queue, and then goes out on the J-Bus. The data read from J-Bus is then parsed by J-Bus parser, and then the data is stored in the PIO return queue which is sent to the IOB.

The Read transactions (NCRD) can be 1, 2, 4, 8, 16-byte reads and are aligned to size. There is a maximum support for 1 to 4 pending reads to the J-Bus (software programmable). Read returns to the IOB may observe strict ordering with respect to the writes to the L2-cache (software programmable).

6.1.3 J-Bus Interrupt Requests to the IOB

- A J-Bus interrupt in the mondo vector format is received by the J-Bus parser and then it is stored in the interrupt queue before being sent to the IOB.
- A modified mondo interrupt transaction is where only the first data cycle is forwarded to the CPU.
- The mondo interrupt queue is maximally sized to 16 entries, and there is no flow control on queue.
- Interrupts to the IOB may observe strict ordering with respect to the writes to the L2-cache (software programmable).
- An interrupt ACK/NACK received from the IOB is first stored in the interrupt ACK/NACK queue, and then it is sent out on the J-Bus.

6.1.4 J-Bus Interface Details

The J-Bus interface has the following characteristics:

- JBI Requests the J-Bus as agent 0
- Masters transaction using agent ID 0 to 3
- 16 transaction IDs (TIDs) assigned in the least recently used order
- A read TID becomes available when the read data is returned
- A write TID is never marked *unavailable*
- Responds to the addresses corresponding to agent ID 0 to 3
- External J-Bus arbitration:
  - Adheres to the J-Bus arbitration protocol
  - May arbitrate to maximize its time as the default owner, in order to opportunistically drive the debug port data even when it has nothing to issue (software controlled)
  - JBI starts up in multi-segment arb mode, which can be change by way of software
- Flow control - address OK (AOK) and data OK (DOK)
  - Uses only AOK-off to flow control the J-Bus when WDQ reaches its high watermark
  - DOK-off is not used for flow control
  - Follows J-Bus protocol when other agents assert their AOKs/DOKs

6.1.5 Debug Port to the J-Bus

- There are two debug first-in/first-outs (FIFOs), each with 32 entries
- Programmable to fill and dump from one or both FIFOs. If both FIFOs are programmed, then each FIFO is alternately filled, but they are dumped both in parallel, thus using half as many cycles.
- Arbitration models for using the J-Bus to report debug data include:
  - Default - when the J-Bus is idle, and the JBI has no other transactions available to issue on to the J-Bus, the JBI opportunistically dumps debug data if it is the default owner
  - DATA_ARB - JBI will arbitrate (arb) whenever the FIFOs are higher than the low watermark, and the JBI is not the bus owner
  - AGGR_ARB - JBI arbs whenever it does not own the bus, so the bus behavior does not change based on the quantity of the debug output
- Debug data appears on the J-Bus as a Read16 return cycle to the AID4 with debug data payload on J_AD[127:0]
- Fake DMA range (0x80_1000_0000 to 0x80_FFFF_FFFF) is used for the debug data
- Error injection is supported in outbound and inbound J-Bus traffic
- BI debug info, when enabled, is placed in the transaction headers (the JBI queues info in the upper 64 bits of the AD)

6.1.6 J-Bus Internal Arbitration

- There are seven agents for internal arbitration:
  - Four read return queues
  - PIO request queues
  - Mondo interrupt ACK/NACK queues
  - Debug FIFO
- In the default arbitration, the debug FIFO has the lowest priority, and there is round-robin arbitration between the other six agents
- Until the FIFO is flushed, the debug FIFO has the highest priority when the HI_WATER or MAX_WAIT limits are reached

6.1.7 Error Handling in JBI

- There are 19 different fatal and not-correctable errors, each with a log enable, signal enable, error detected bit, and error overflow detected bit. (Refer to the *UltraSPARC T1 Supplement to UltraSPARC Architecture 2005 Specification* for details on programming control bits and reading status registers.)
- J-Bus snapshot registers contain address, data, control, parity bits.
- J-Bus requests to non-existent memory causes a read to address 0 before the JBI issues an error cycle on the J-Bus.
- Fatal error asserts DOK-on for 4 cycles, which instructs the external J-Bus to PCI-Express ASIC to perform a warm reset.

6.1.8 Performance Counters

- There are two performance counters in the JBI, which are 31-bits wide each.
- The software can select one of the 12 events to be counted:
  - J-Bus cycles
  - DMA read transactions (inbound)
  - Total DMA read latency
  - DMA write transactions
  - DMA WR8 transactions
  - Ordering waits (number of jbi->l2 queues blocked each cycle)
  - PIO read transactions
  - Total PIO read latency
  - PIO write transactions
  - AOK off or DOK off seen
  - AOK off seen
  - DOK off seen