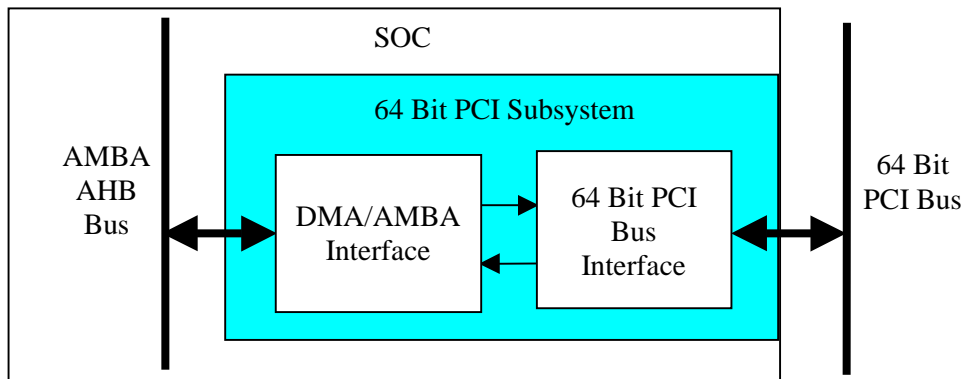


## **AU-PB8064: 64 Bit PCI AMBA Subsystem Core** **AMBA AHB Bus 64 Bit PCI Bus Interface with DMA**

The AU-PB8064 64 Bit PCI AMBA Subsystem provides a 64 Bit PCI peripheral subsystem for AMBA based SOCs. It contains a 64 Bit PCI Bus Interface with PCI master and slave functionality. The 64 bit PCI AMBA Subsystem connects seamlessly to the AMBA AHB Bus. A DMA Engine is included to move data of PCI master memory and I/O transactions. PCI master configuration transaction data is transferred by direct AMBA Bus reads and writes. PCI slave transactions are translated into AMBA Bus transactions by the 64 Bit PCI AMBA Subsystem and driven onto the AMBA Bus. The figure below shows the 64 Bit PCI AMBA Subsystem within an SOC. The 64 Bit PCI AMBA Subsystem Core is available as a synthesizable Verilog model from Aurora VLSI, Inc. Contact [CustomerService@auroravlsi.com](mailto:CustomerService@auroravlsi.com).



PCI Bus speeds of 33MHz and 66MHz are supported. The 64 Bit PCI Bus Interface block of the PCI Subsystem provides 64 bit PCI Bus master and slave functionality. An AMBA Bus master can initiate a PCI transaction through this PCI Bus Interface block in master mode. In slave mode, the PCI Bus data is written to or read from an AMBA Bus slave. The PCI Bus Interface block contains the standard PCI configuration registers including four base address registers.

Internal to an SOC, the PCI Subsystem is a peripheral on the AMBA AHB Bus. Direct Memory Access- DMA, between the DMA/AMBA Interface block of the PCI Subsystem, and AMBA Bus targets, is used to transfer data of PCI master memory and I/O transactions over the AMBA Bus to/from the PCI Subsystem. When doing DMA, the PCI Subsystem is an AMBA Bus master.

When the PCI Subsystem is functioning as a PCI slave, the target address is provided by the PCI Bus transaction. This address is mapped into an AMBA Bus address by the PCI Subsystem. PCI data is driven onto the AMBA Bus for PCI slave writes and read from the AMBA Bus for PCI slave reads. When doing PCI slave transactions, the PCI Subsystem is an AMBA Bus master.

The PCI Subsystem is an AMBA Bus slave for control and status register accesses, and for PCI master configuration transactions. All control and status register accesses and PCI master configuration transactions originate in an AMBA Bus master outside of the PCI Subsystem, such as a host processor.

64 Bit PCI AMBA Subsystem features are summarized:

64 Bit PCI Bus Interface

- PCI 2.1 and 2.2 compliant
- Master and slave support 64 bit address and data transfers
- Supports variable burst size transfers
- Performs zero wait state transfers
- Master is capable of performing I/O, Memory, and Configuration types of PCI transfers
- Master supports byte mode operation
- Master is capable of performing Memory Write Invalidate and Memory Read Line operations
- Performs back to back transfers
- PCI master transaction status captured in eight entry PCI Master Status FIFO

DMA/AMBA Interface

- AMBA AHB Bus interface
- Single channel DMA Engine for PCI master memory and I/O transactions
- Physical DMA addresses
- Programmable DMA starting address
- Programmable DMA transfer count- up to 64 Kbytes
- Programmable DMA AMBA Bus interface transaction size- 8 to 1024 bytes
- Programmable DMA AMBA Bus data transfer size- 4 or 8 bytes
- Locked DMA operation optional (software programmable)
- Direct software writes or information extracted from descriptors in memory, to program DMA control information
- Dedicated AMBA Bus master interface for the DMA channel
- Dedicated AMBA Bus master interface for PCI slave transactions
- Unique PCI slave address to AMBA Bus address mapping for each PCI slave address space
- AMBA Bus slave interface for register reads and writes and PCI master configuration transactions
- Interrupts:
  - PCI master write data to PCI Bus Interface DMA completed
  - PCI master read data from PCI Bus Interface DMA completed
  - PCI master transaction status available
  - PCI Bus interrupt

The core is delivered as a synthesizable RTL Verilog model. Deliverables include:

- RTL Verilog source code model of the core
- Verilog testbench and test cases
- Synthesis scripts examples
- Complete detailed documentation and training class notes

## **64 Bit PCI Bus Interface**

The PCI Subsystem includes the Stargate SSP8064 64 Bit PCI Bus Interface Core. Additional logic at the application interface of the 64 Bit PCI Bus Interface provides a DMA Engine, control and status registers, AMBA Bus interface, and host processor interrupts for the 64 Bit PCI Subsystem.

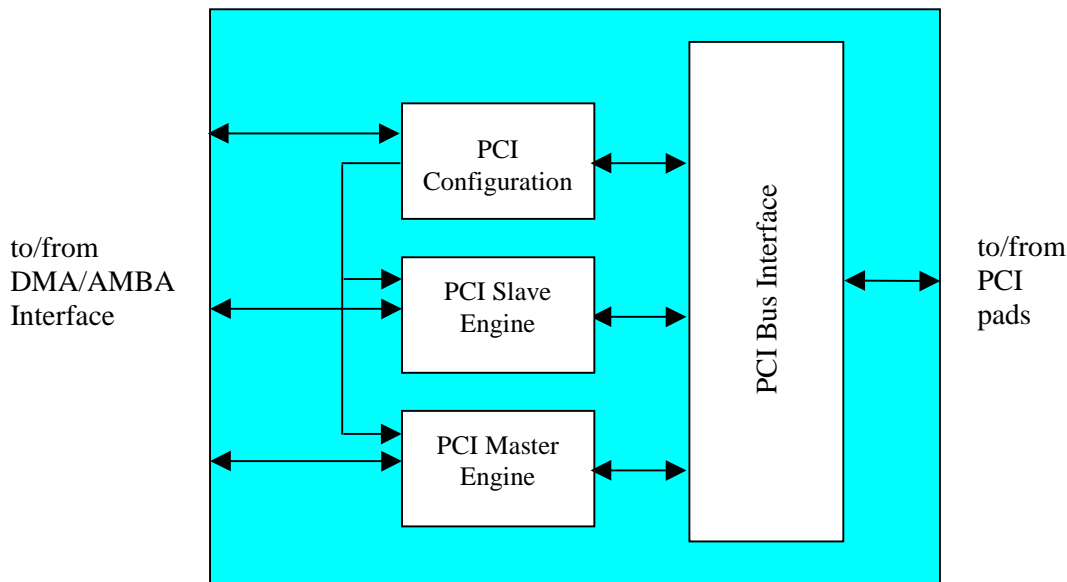
A block diagram of the PCI Bus Interface is shown below.

The PCI Master Engine handles master cycles, retries, command generation, and data transfers. It also handles Memory Write Invalidate (MWI) and Memory Read Line (MRL) transfers. It generates handshake signals to communicate with the DMA/AMBA Interface block.

The PCI Slave Engine handles address decode, command decode, and generation of slave PCI cycles. It is capable of performing burst transfers.

The PCI Configuration block has the configuration address space of the PCI Bus Interface. It is programmable to accommodate multiple base address registers. It is accessed from the DMA/AMBA Interface block and by PCI slave transactions from the PCI Interface block.

The PCI Bus Interface communicates with the slave, configuration, and master data paths and address paths to generate appropriate transfers on the PCI bus. Pad control, data multiplexing, and parity generation and detection are performed in this block.



## **DMA/AMBA Interface**

The DMA/AMBA Interface includes a single channel DMA Engine. The DMA Engine is used to transfer PCI master memory and I/O write data from the data's source, over the AMBA Bus, to the PCI Master Engine in the PCI Bus Interface block. PCI master memory and I/O read data is sent from the PCI Master Engine, over the AMBA Bus, to the received data's destination, by the DMA Engine.

Block moves of up to 64K bytes are supported by the DMA Engine. The exact transfer count of each DMA operation is the size of the data block that is being transferred, and is set by software. DMA operations are done as a series of data transfers to/from the PCI Master Engine, and AMBA Bus transactions to move the data block. The length of each AMBA Bus transaction is software programmable so that it can be optimized according to system characteristics. Each individual AMBA Bus transaction is from 8 bytes to 1024 bytes according to a value programmed into a DMA channel's control register. Additionally, the data size of each data transfer on the AMBA Bus is software programmable to be four or eight bytes.

The series of accesses that make up a complete DMA operation may be locked together so that no other device gets the AMBA Bus until the DMA operation is finished. This is under software control.

The DMA starting address is set by software. This is the write data source starting address in memory for write data, and the read data destination starting address in memory for read data. These starting addresses are incremented by the DMA Engine to form the AMBA Bus transaction addresses as the DMA operation progresses. All addresses are physical addresses.

The DMA control information that is set by software- starting address, transfer count, AMBA Bus transaction size, data transfer size, and lock flag, can be set by direct software writes to PCI Subsystem registers that hold this DMA control information. Alternatively, this DMA control information can be set from descriptors in memory that hold the DMA control information. Scatter/gather DMA is done using a chained descriptor list as the DMA control information source. When using descriptors to set the DMA control information, the descriptors are initialized by software. To support DMA configuration from descriptors, the DMA/AMBA Interface contains logic to read the descriptors from memory, and load the appropriate DMA/AMBA Interface registers with the DMA control information.

A DMA operation begins when the DMA channel is enabled, after the starting address, transfer count, AMBA Bus transaction size, data transfer size, and lock flag are configured. The DMA channel moves the data block, and the DMA operation ends when the entire data block has been moved.

The PCI Subsystem can also function as a PCI master in direct access (non-DMA) mode. To do a direct access PCI master write, the PCI address, byte write enables, and write data are written to registers within the PCI Subsystem. A direct access PCI master read is done by writing the PCI address to a PCI Subsystem register, and then reading the PCI read data and byte read enables from PCI Subsystem registers. Optionally, the direct access address register can be automatically incremented upon completion of each PCI data transfer. There are six sets of PCI master direct access registers; one set for each PCI master bus transaction type/direction- PCI memory read, memory write, I/O read, I/O write, configuration read, and configuration write. PCI master direct access mode can be used to provide a bus master AMBA/PCI bridge.

Status is captured for each PCI master transaction. It is held in an eight entry Master Status FIFO. The host processor may read this FIFO or optionally in DMA mode, the PCI Subsystem automatically reports the status by sending it out over the AMBA Bus.

As a PCI slave, the PCI Subsystem accepts PCI transactions, translates the PCI address into an AMBA Bus address, and generates the appropriate read or write AMBA Bus transaction. On PCI slave reads, when the read data is returned to the PCI Subsystem over the AMBA Bus, the PCI Subsystem provides it to the PCI Interface block that then drives the read data onto the PCI Bus. The PCI Subsystem supports four PCI slave address spaces. Each PCI slave address space has a unique software programmable AMBA Bus base address.

The PCI Subsystem contains several registers. These registers are accessed using AMBA Bus single transactions initiated by host processor reads and writes. In addition to the SSP8064 registers, there are registers that hold:

- General control and status information
- PCI slave mode control and status information
- PCI slave mode AMBA Bus base addresses
- PCI master DMA addresses, control, and status information
- PCI master direct access control and status information
- PCI master direct access addresses, data, and byte enables

The DMA/AMBA Interface generates interrupts to notify the host processor of events that are important to driver software. These interrupts include:

- Interrupt upon a completed PCI master DMA operation or chain of PCI master DMA operations
- Software interrupt
- Interrupt upon a PCI reset

The PCI Subsystem can assert the PCI interrupt INTA through software.