Enhanced Security and Energy Efficiency of Microcontrollers and SoCs

Joseph Yiu
Senior Embedded Technology Manager,
CPU Group, ARM
Cambridge, United Kingdom
Version 1.0. Jan-2016

***Abstract*— The on-chip system design of microcontrollers and SoCs can have a significant impact on the overall security, power efficiency and responsiveness of the device. This paper outlines the key aspects of the system design and looks at new technology developments that promise to improve the security of these SoCs while still enabling high efficiency and low latency designs.**

***Keywords*— AMBA 5 AHB5; Low Power Interface Specification; Q channel; Cortex-M processors**

1 BACKGROUND

Low power microcontrollers and SoCs with connectivity play a significant role in the adoption of the IoT (Internet of Things). Most of the IoT end-point devices, such as sensors and wearable devices, are powered by low power chip designs such as microcontrollers powered by ARM® Cortex®-M processors. In addition to the processor technologies, a range of other technologies are needed for the processor systems in these devices to ensure they have low power and secure solutions. This includes the on–chip bus systems and power management solutions.

Traditionally, most ARM Cortex-M processors use the AMBA® 3 (Advanced Microcontroller Bus Architecture) AHB Lite bus protocol for the main system buses and the AMBA 3 APB for connecting to debug components and peripherals. Although the ARM Cortex-M7 processor uses AMBA 4 AXI for the main system bus for performance reasons, it also supports AHB Lite for the peripheral bus, for TCM accesses (using an AHB Lite slave bus for DMA operations) and for debug interface connections. The Cortex-M processors also export status signals for architectural sleep modes, and provide additional interface signals for advanced low power features such as optional SRPG (State Retention Power Gating) and WIC (Wakeup Interrupt Controller) support.

As the need for low power secure solutions become critical for emerging applications, an update is required within the system design to enable future low power and secure designs. In November 2015, ARM announced the AMBA 5 AHB5 specification (reference 1). This bus protocol extended the capability and features of the AHB protocol, including adding specific support for security, and is introduced in section 4 of this document. In addition, this paper also covers the Q channel, one of the on-chip signaling protocols covered by the AMBA 4 Low Power Interface Specification. These two on-chip signaling protocols will be used in a range of future ARM based processors, including some of the next generation ARM Cortex-M processors.

2 INTRODUCTION TO AHB LITE

The AMBA rev 2.0 specification was released in 1999 and has been used in ARM processors for many years. This specification covered the AHB (Advanced High-performance Bus), ASB (Advanced System Bus, which is replaced by AHB) and APB (Advanced Peripheral Bus) protocols. The AHB and APB protocols have been used in a wide range of ARM and third party IP products such as low power processors, memory controllers and peripherals. Today, AMBA protocols have become the de facto standard on-chip bus interconnect for low power processor systems, supported by a wide range of semiconductor IP companies.

The AHB protocol was updated in 2006 to become the AHB Lite protocol - a simplified version of AHB. The AHB and AHB Lite protocols have the following key characteristics:

- Supports pipelined operations with wait state support
- Supports transfer sizes of 8-bit, 16-bit, 32-bit, … up to 1024-bit (typical AHB data bus width are 32-bit and 64-bit)
- Supports various types of burst transfers (fix length and variable length increment bursts)
- Inclusion of side band signals to provide various protection information.
- Supports error response from bus slaves.
- Supports multiple bus masters using multi-layer AHB arrangement.
- Supports locked transfers.

The AHB Lite specification is intended to provide a minimal set of bus protocol features for low power processor systems. In the AHB to AHB Lite update, the following features are removed:

- Multi-master hand-shaking signals like Bus Request (HBUSREQx) and Bus Granted (HGRANTx) are removed and replaced by a multi-layer AHB approach.
- Split and Retry response types are removed from slave responses (these responses are no longer needed due to the multi-layer AHB technology).
- The HMASTER signal (which indicates bus master ID for the transaction) is not covered by the AHB Lite specification. However, many IP products and bus systems continue to support this signal.

In order to provide all the required functionality in the ARMv7-M architecture, the Cortex-M processors provide an AHB Lite bus interface and a range of additional sideband signals. These side band signals are needed because the simple nature of the AHB Lite protocol does not provide all the features required in advanced system designs. As a result, the AHB5 specification released in 2015 extended the AHB protocol to cover the additional features and to make system integration for advanced low power systems easier.

3 Introduction to AHB system design

Before looking at the details of AHB5 features, we will first provide a very basic introduction to AHB. An AHB system runs on a single clock speed (HCLK) and has an active low reset signal called “HRESETn” – the ‘n’ suffix indicates that this signal is active low. All functional signal transitions take place at the rising edges of HCLK.

In a simple system, there might be a single bus master (e.g. the CPU) connected to multiple AHB slaves. The connection involves two groups of signals:

- Address phase signals (address, control information), and
- Data phase signals (data, ready, response).

A simplified bus connection diagram is shown in Figure 1. An address decoder component is needed for the generation of slave select signals decoded from the address (HADDR) in the same clock cycle using combinatorial logic.

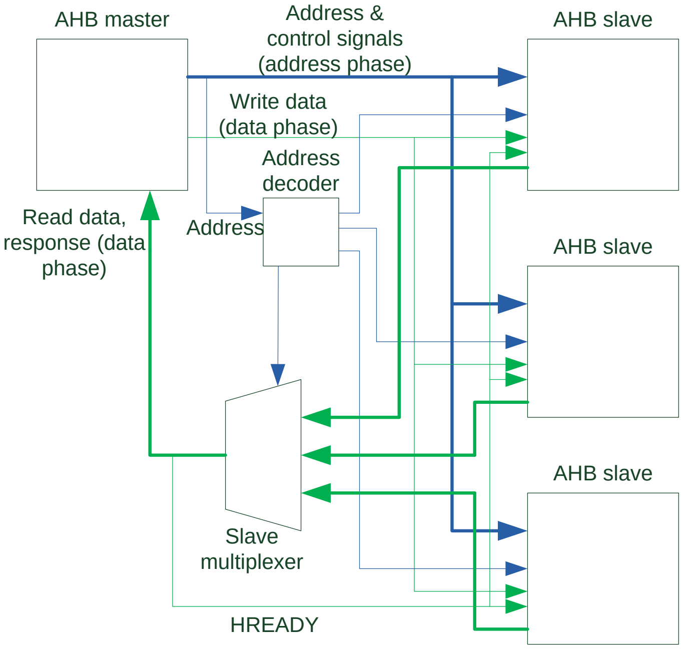


Figure 1: A simple single master AHB system

The bus operation is pipelined (Figure 2). The data phase of a transfer is one phase behind the address phase. And the phase boundary is determined by a HREADY signal, which is provided from the currently selected bus slave at data phase.

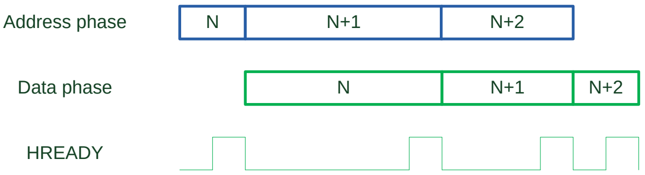


Figure 2: AHB protocol use pipelined operations

In the AHB Lite protocol, the address phase signals consist of:

Signal	Functions	Details
HADDR[31:0]	Address	Address of transfer
HTRANS[1:0]	Transfer type	IDLE(2’b00) – Idle BUSY(2’b01) – Busy – indicates the master is not ready for the next transfer in the middle of a burst NSEQ (2’b10) – Non-sequential. For single transfers and first transfer in a burst. SEQ (2’b11) – Sequential. For burst transfers except the first one.
HWRITE	Write / Read	When 1, indicates that the transfer direction is write. Otherwise it is read.
HSIZE[2:0]	Transfer size	3’b000 – byte 3’b001 – Half word (16-bit) 3’b010 – word
HPROT[3:0]	Protection information	Bit 0 – 1 indicates data transfer. 0 indicates instruction fetches. Bit 1 – 1 indicates privileged level Bit 2 – 1 indicates the transfer is Bufferable Bit 3 – 1 indicates the transfer is Cacheable
HBURST[2:0]	Burst type	3’b000 – SINGLE 3’b001 – INCR (incremental burst without a fixed length) 3’b010 – WRAP4 burst 3’b011 – INCR4 burst 3’b100 – WRAP8 burst 3’b101 – INCR8 burst 3’b110 – WRAP16 burst 3’b111 – INCR16 burst
HMASTLOCK	Lock transfer	When 1, indicates the transfer is part of a locked transfer sequence
HSELx	Slave select	Slave select signals from address decoder. Typically one for each slave.

Table 1: AHB Lite address phase signals

All these signals, with the exception of HSELx, are generated from the AHB bus master.

The data phase signals include:

Signal	Functions	Details
HWDATA[n-1:0]	Write data	Write data from AHB master to bus slaves. It can be 32-bit/64-bit/etc.
HRDATA[n-1:0]	Read data	Read data from AHB slave to bus master. It can be 32-bit/64-bit/etc.
HRESP	Response type	0 – OKAY, 1- ERROR.
HREADYOUT	Slave ready	Ready signal from bus slave.
HREADY	Bus ready	Combined ready signal from AHB slave multiplexer to bus master, and also route back to bus slaves. It indicates end of a bus phase.

Table 2: AHB Lite data phase signals

It is possible to build an AHB system with multiple bus masters using multi-layer AHB arrangements. This is typically handled using a bus component called a Bus Matrix (Figure 3), which contains internal registering stages to hold off transfers when the targeted downstream AHB is used by a different bus master.

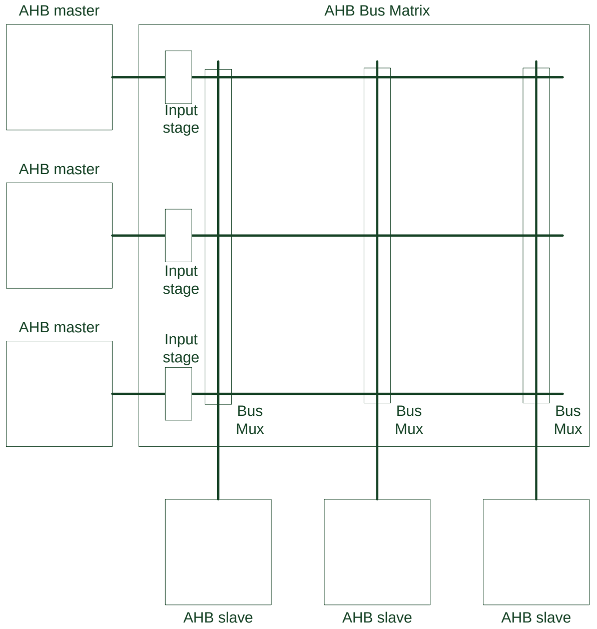


Figure 3: AHB Bus Matrix is a common component to support multi-master system designs

In multi-master systems, an additional HMASTER signal can be generated by the bus matrix component to indicate which bus master generated the transfer that is being propagated to an AHB slave. In Cortex-M processors, HMASTER signals can also be used to indicate transfer type (e.g. transfers generated from the debugger).

The simple nature of AHB has a number of advantages:

- Low gate count, which is therefore well suited for low power processor systems.
- Low latency in terms of number of clock cycles required to process a bus transfer.
- All registers trigger at the clock rising edge, making chip implementation and testing easier.

On the commercial side, the AHB specifications are open access, and third parties are free to access AHB specification to design components without paying a royalty fee.

4 KEY ENHANCEMENTS FROM THE UPDATE OF AHB LITE TO AHB5

4.1 Summary

The development of the AHB5 specification has several key objectives

- Supports ARM TrustZone® operation. TrustZone technology is an extension of the processor architecture which is designed to provide an additional level of security.
- Supports system designs which utilize advanced processor architecture features like exclusive accesses and memory attributes.
- Easy design migration from legacy designs based on AHB Lite.
- Consistency with AMBA AXI on-chip bus protocol

The key enhancements are summarized in the list below:

- Addition of a new sideband signal to support TrustZone operations.
- Extension of HPROT signal for additional memory attributes.
- Addition of several new sideband signals to support exclusive accesses, including HMASTER signals.
- Addition of user defined signals (HxUSER)
- A number of clarifications in the wording of the specification.
- Addition of a number “properties” to allow better capabilities in certain chip design tools.

4.2 TrustZone support

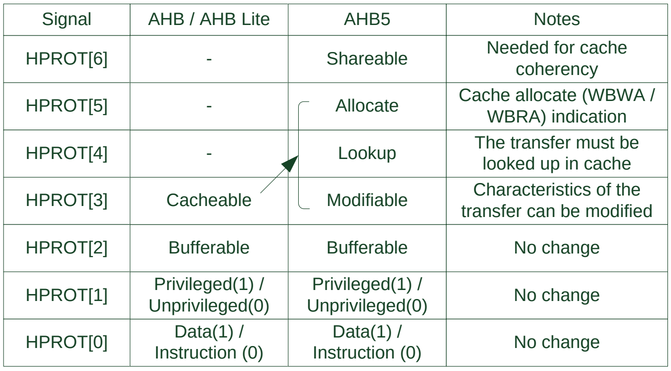
AHB5 adds a new signal called HNONSEC. It is an address phase signal:

- When this signal is 0, it indicates that the transfer is Secure.
- When this signal is 1, it indicates that the transfer is Non-Secure.

The bus system and memory controller can then utilize this signal to reject transfers if a Non-Secure transfer is attempting to access a Secure memory location. This enhancement also allows the security attribute of a transfer to propagate between AHB and AXI systems. It means TrustZone operations can span across AXI-AHB boundaries. In AXI, the equivalent signals are ARPROT[1] (for read)/ AWPROT[1] (for write).

4.3 Extension of protection information (HPROT)

Due to the increasing complexity of memory systems, additional memory attribution signals are needed. For example, a single-bit cacheable information signal is inadequate to cover multiple types of cache policies. As a result, HPROT is expanded from 4 bits in AHB Lite to 7 bits in AHB5 (Figure 4).



Signal	AHB / AHB Lite	AHB5	Notes
HPROT[6]	-	Shareable	Needed for cache coherency
HPROT[5]	-	Allocate	Cache allocate (WBWA / WBRA) indication
HPROT[4]	-	Lookup	The transfer must be looked up in cache
HPROT[3]	Cacheable	Modifiable	Characteristics of the transfer can be modified
HPROT[2]	Bufferable	Bufferable	No change
HPROT[1]	Privileged(1) / Unprivileged(0)	Privileged(1) / Unprivileged(0)	No change
HPROT[0]	Data(1) / Instruction (0)	Data(1) / Instruction (0)	No change

Figure 4: Extension of HPROT signals in AHB5

4.4 Addition of Exclusive Access support

Exclusive accesses are memory read and write instructions designed to support semaphore operations. Typically, semaphore operations are implemented using variables shared between multiple processes (these processes can be running on the same processor or on multiple processors in parallel), and the modification of such variables is carried out via a read-modify-write sequence with exclusive access operations. Different from normal memory store instructions, exclusive store instructions return success/fail status to the software, where an exclusive fail means the data could have been modified by a different process and the read-modify-write sequence should be re-issued. For multi-processor systems, the memory system can include an exclusive access monitor (Figure 5) to detect access conflicts of semaphore variables. This exclusive access monitor needs to have access to exclusive access sideband signals which identify exclusive load and exclusive store operations, and to return exclusive store status to the processor that issued the exclusive store.

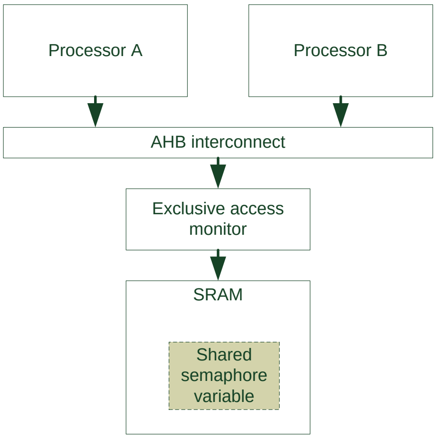


Figure 5: Exclusive access monitor requires exclusive access support signals to monitor exclusive accesses

For example, if another processor has updated the same variable between the exclusive read and exclusive write operations, the exclusive access monitor return an exclusive fail response to an exclusive store operation and at the same time block the update of the variable from reaching the memory. In this scenario, the processor that issued the semaphore read-modify-write sequence must re-issue the whole sequence to reattempt the semaphore variable update (Figure 6). To allow the exclusive sideband signals to be triggered correctly, the processor (such as the Cortex-M3 and Cortex-M4 processors) needs to use special exclusive access load store instructions when handling the semaphores variables.

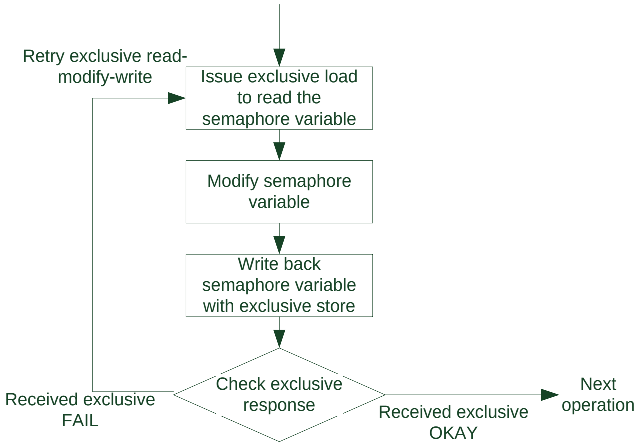

Retry exclusive read-modify-write
Issue exclusive load to read the semaphore variable
Modify semaphore variable
Write back semaphore variable with exclusive store
Check exclusive response
Received exclusive FAIL
Received exclusive OKAY
Next operation

Figure 6: Exclusive access flow

AHB5 added three signals to support exclusive access operations:

Signal	Descriptions	Direction
HEXCL	Address phase signal to indicate the read/write transfer is an exclusive access	From bus masters to exclusive access monitor
HEXOKAY	Data phase response signal to indicate exclusive store is success (unlike HRESP, this is single cycle)	From exclusive access monitor to bus masters
HMASTER[*n*-1:0]	Identify the bus master that issued the transfer	From interconnect (e.g. bus matrix) to exclusive access monitor.

Table 3: AHB5 signals for exclusive access support

When an exclusive load or exclusive store instruction is executed, and the memory location accessed is shared, the associated transfer has the HEXCL signal asserted. In the data phase of exclusive store, the exclusive access monitor can return the exclusive access status using a signal called HEXOKAY.

4.5 *Addition of User Defined signals*

Three user define signals have been added. They are optional, and the widths of these signals are implementation defined:

Signal	Descriptions
HAUSER[*x*-1:0]	User signal associated with address phase signals from bus masters to slaves
HWUSER[*y*-1:0]	User signal associated with data phase signals for write operations (from AHB masters to AHB slaves)
HRUSER[*z*-1:0]	User signal associated with data phase signals for read operations from (from AHB slaves to AHB masters)

Table 4: AHB5 user defined signals

The usage of these user defined signals is system specific. Typical uses include:

- To propagate ECC (Error Correction Code) alongside address phase signals, and data signals.
- To propagate HMASTER information.

The user signals are not new to some chip designers. However, these signals were not part of the official AHB specification until now.

4.6 *Clarifications*

Several new clarifications have been introduced in the AHB5 specification documentation. The most significant are:

- Signal stability – The AHB protocol requires that during wait states, some signals (e.g. control information for the next active transfer) must be stable unless an error response is received. In the AHB5 specification, it is clarified that the word "stable" does not imply that the signal must not glitch between clock rising edges, unless a property of an AHB5 system called "Stable_Between_Clock" is set. If this property is not set, the signals can glitch in the middle of a clock cycle but the values at clock rising edges must remain the same. This clarification is essential as modern logic synthesis tools can utilize various optimizations which can restructure digital circuits and thus chip designers may not be able to guarantee true glitch free operations in the bus signals.
- Lock transfer sequence – in AHB5, it is explicitly required that a lock transfer sequence cannot spread across multiple bus slaves. In theory, lock transfer sequences going across multiple slaves could have resulted in deadlock cases in multi-master systems.
- Multiple HSEL – AHB5 clarifies that an AHB slave can have multiple HSEL inputs. This is useful when a bus slave has multiple interfaces (e.g. a memory controller could have memory space and configuration registers).

4.7 Properties

A selection of properties is defined in the AHB5 specification to assist design tool software to determine the functions and feature set of AHB fabric components or AHB systems. These properties are not signals (they do not exist in real hardware). However, they can help ESL (Electronics System Level) design tools in various ways. For example, features implemented in an AHB5 component can be described in an IP-XACT file so that ESL tools know what side band signals are essential.

The value of properties can be TRUE or FALSE (if a property is not set, it is considered to be FALSE). The properties defined include:

Properties	Descriptions
Extended_Memory_Types	If true, the interface supports HPROT[6:0], otherwise the interface supports HPROT[3:0] as in AHB Lite.
Secure_Transfers	If true, the interface supports HNONSEC signal.
Endian	If true, the interface is in Big Endian
Stable_Between_Clock	If true, the signals remain stable between clock edges.
Exclusive_Transfers	If true, the interface supports signals for exclusive accesses.
Multi_Copy_Atomicity	If true, a system provides multi-copy atomicity, which means all observers of the system see consistent information such as transfer ordering.

Table 5: Properties defined in the AHB5 specification

5 Compatibility with AHB Lite

Most AHB5 signals have the same behaviour as in AHB Lite. So in many instances, AHB bus masters and bus slaves can be reused as they are. If a TrustZone enabled environment is needed, then additional system components will be required for memory partitioning and security management. This is covered in the next section.

In some cases, additional signals might need to be tied off or glue logic utilized when connecting a legacy AHB Lite bus master to an AHB5 system.

Signals	Handling of the signals
HNONSEC	If the legacy bus master is fixed as always Non-secure, then this signal can be tied to zero. If the legacy bus master is fixed as always Secure, then a bus wrapper should be used to generate HNONSEC based on address range. See next section.
HPROT	When connecting AHB Lite master (HPROT[3:0]) to AHB5 (HPROT[6:0]) HPROT[3:0] →HPROT[3:0] HPROT[3] →HPROT[4] HPROT[6:5] can be tied to a fixed value as required, or generated from the address
HEXCL, HEXOKAY	AHB Lite does not have these signals. Legacy master can have HEXCL tied to 0, and HEXOKAY ignored.

Table 6: Handling of AHB5 signals when connecting a legacy AHB Lite bus master to an AHB5 system

In some instances, a simple bus wrapper can be added. For example, Cortex-M3 and Cortex-M4 processors support a range of sideband signals that were not part of the previous AHB specification, but these signals can nevertheless map into new signals specified in AHB5. When using a Cortex-M3 or

Cortex-M4 processor with an AHB5 system, a simple bus wrapper can be used to handle HPROT signals as follows:

- MEMATTRx[1] →HPROT[6]
- MEMATTRx[0] →HPROT[5]
- HPROTx[3] →HPROT[4]
- HPROTx[3:0] →HPROT[3:0]

The handling of the exclusive signals on the Cortex-M3, Cortex-M4 and Cortex-M7 processors require additional glue logic. The exclusive bus signal extensions for these processors provide the EXREQ (exclusive request) and EXRESP (exclusive fail response) signals. While the EXREQ can map to the new HEXCL directly, EXRESP has inverted polarity compared to HEXOKAY. Since the EXRESP signals in these Cortex-M processors can accept a single cycle response, the glue logic required can be designed as in Figure 7:

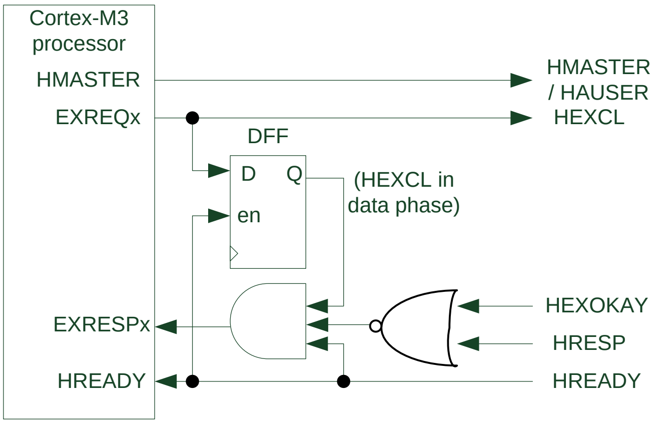


Figure 7: Glue logic for connecting exclusive signals on Cortex-M3/M4/M7 processors to AHB5

The HMASTER signals on the current Cortex-M processors can connect to the HMASTER signal on AHB5, or propagate to downstream AHB using the HAUSER signal. These signals are used to indicate if a bus transfer is generated by the software operations on the processor, from a debugger, or from other mechanisms inside the processor.

Legacy bus slaves connected to AHB5 can ignore some of the new signals (e.g. HPROT, HNONSEC) if such features are not required. If a bus slave never stores any semaphore variables that require exclusive access, then the HEXOKAY output signal can be tied low. If the bus slave is a memory block that might contain semaphore data, we should use one of the following arrangements:

- Add an exclusive access monitor to the system, or
- Update the bus slave to support exclusive access, or
- Add glue logic to respond with HEXOKAY for each exclusive store, provided it is known that only one bus master will access the semaphore data in this memory, or
- Move the semaphore data into another memory that supports exclusive accesses, or use other semaphore mechanisms (e.g. hardware semaphore peripherals: this is system specific).

6 System design with TrustZone for ARMv8-M

6.1 *System design concept*

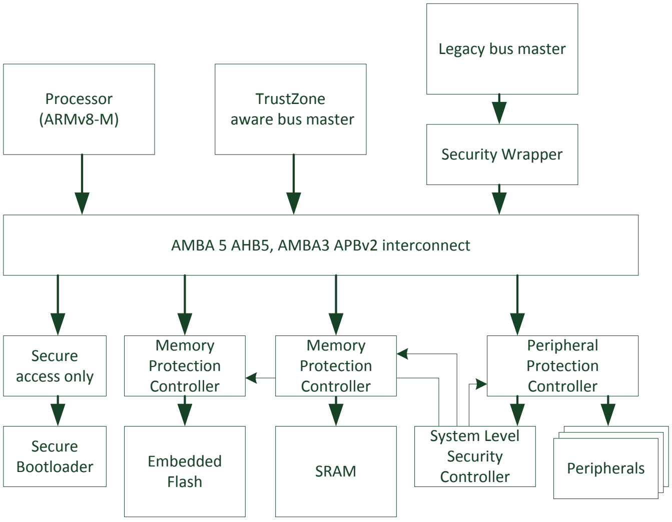
One of the most import objectives of AHB5 is to enable TrustZone technology in low power microcontrollers and SoC designs. TrustZone technology has been added to the ARMv8-M architecture and will be available in the next generation of Cortex-M processors. In many aspects, TrustZone for ARMv8-M is similar to the TrustZone technology used in the ARM Cortex-A Processors, and will enable many next generation microcontrollers to implement advanced security features.

System designs for the next generation of Cortex-M processors will utilise a range of new features from the AHB5 specification. The HNONSEC signal is used to indicate if a transfer is Secure or Non-Secure, and based on this information, bus fabric components can block Non-Secure transactions into Secure memories. Similarly, peripherals can also be assigned as Secure or Non-Secure, and the bus system can ensure that Secure peripherals can only be accessed by Secure transfers.

While it is possible to have separate memory blocks for Secure and Non-Secure memories, it is expected that it will be more common to partition a memory block into Secure and Non-Secure spaces under software control to allow higher flexibility and to simplify chip designs. Assignment of security domains in peripherals will need to be programmable in many applications. As a result, it is expected that several new AHB5 system components will be used in chip designs. Some examples of such components are:

- A Memory Protection Controller for partitioning of a memory block into Secure and Non-Secure spaces.
- A Peripheral Protection Controller for assigning peripherals into Secure and Non-Secure domains.
- Bus security wrappers may be needed for legacy bus masters.

Figure 8 shows a simple system design based on a representative ARMv8-M processor.



Legacy bus master
Processor (ARMv8-M)
TrustZone aware bus master
Security Wrapper
AMBA 5 AHB5, AMBA3 APBv2 interconnect
Secure access only
Memory Protection Controller
Memory Protection Controller
Peripheral Protection Controller
Secure Bootloader
Embedded Flash
SRAM
System Level Security Controller
Peripherals

Figure 8: Simple system design for TrustZone for ARMv8-M

Such a system design would also likely include some form of system level security controller (accessible in the secure state only) to manage security settings in various parts of the design.

6.2 *Address partitioning and region ID values*

In order to allow software running on ARMv8-M processors to determine if a memory range is in a Secure or Non-Secure space, each continuous memory range with the same security attribute is given a region ID value by the Security Attribution Unit (SAU) inside the processor, or by the Implementation Defined Attribution Unit (IDAU) closely coupled to the processor. The region ID value of an address, together with the security attribute, can be read by an instruction called TT (Test Target). Secure software can then utilize the TT instruction to check the starting and ending addresses of a data structure or array. If, for example, both address are in the Non-Secure domain and have the same ID value, then the software can tell that this data structure or array is in a continuous Non-Secure memory space.

In order for this to work, the region ID values for each of the regions must be unique. Since the ID values are only 8-bit, there can only be 256 regions defined by the IDAU and 256 regions defined by the SAU. This restriction means that there might not be enough region ID numbers for each of the peripherals if there are a large number of peripherals and each of them is assigned their own unique region ID. In some software environments, the OS might also need to manage a memory range as memory pages and then assign security attributes to each page dynamically. With this arrangement, large numbers of ID values could be required. Furthermore, the algorithm for assigning region numbers can be complicated because adjoining memory pages with the same security attributes should get the same region number. Otherwise, the starting and ending addresses of a data structure spanning across memory page boundaries will get different region numbers, causing incorrect results in the security attribute checks.

In order to solve this potential problem, a memory alias approach can be used (Figure 9). With this arrangement, a memory block can be mapped into a Secure address range (with ID region n) and a Non-Secure address range (with ID region n+1). The Secure memory pages can then be made visible in the Secure address space with a single ID value, and the Non-secure memory pages made visible in the Non-Secure address space with another ID value. In this way, only two ID values are needed.

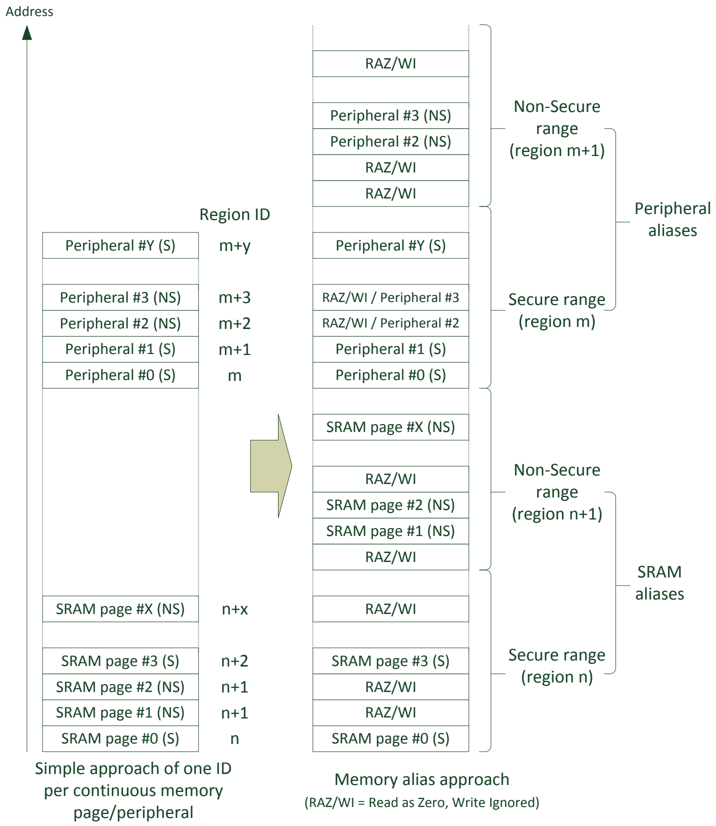


Figure 9: Memory alias scheme in system design for ARMv8-M

Peripherals can also be managed in the same way. A chip designer can optionally make a Non-Secure peripheral visible via the Secure address space. By doing so the peripheral driver code for Secure software can use the same peripheral base address, no matter whether the peripheral is assigned as Secure or Non-Secure.

6.3 *Adding legacy bus masters in an ARMv8-M system with TrustZone*

Legacy bus masters can be used in a TrustZone enabled ARMv8-M system. The bus master can be connected to the AHB5 bus inter-connect components via a bus wrapper, in one of two configurations:

- Always in the Non-Secure domain – in this case HNONSEC can be tied high so that all transfers generated are Non-Secure. The bus wrapper can optionally block Non-Secure transfers to Secure addresses (in such cases, address lookup hardware similar to an IDAU is required in the bus wrapper, or can be directly coupled to it).
- Always in the Secure domain – in this case the bus wrapper can generate HNONSEC according to the ARMv8-M architectural requirements: transfers to Secure addresses are marked as Secure (HNONSEC=0) and transfers to Non-Secure addresses are marked as Non-Secure (HNONSEC=1). Such operations require address lookup hardware similar to an IDAU.

7 Q CHANNEL (LOW POWER INTERFACE)

The Low Power Interface Specification (reference 2) was introduced as a part of the AMBA 4 specification series. This specification contains two interface protocols:

- Q channel for power on/off management control in simple subsystems.
- P channel for advanced subsystems that support multiple power profiles/states.

For low power microcontrollers and IoT Endpoints, the Q channel provides a standardized interface suitable for most typical power management arrangements; such as power gating, state retention power gating (SRPG) for logic and state retention for memories.

The Q channel connects between a design unit (e.g. a processor) and a power management unit (device specific). The interface operates with 4 signals, as shown in Table 7. The polarity of these signals must be chosen so that if the interface signals get clamped to 0 in a low power state, it will not affect the signal levels:

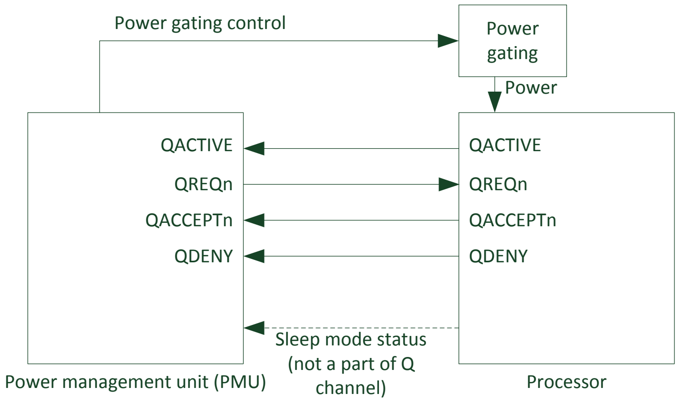


Figure 10: Q channel arrangement

Signal	Descriptions
QACTIVE	Indicates the design unit has an outstanding action to performance
QREQn	Active low signal to request the low power state
QACCEPTn	Active low signal to acknowledge that the low power request is accepted
QDENY	Active high signal to indicate that the low power request is denied

Table 7: Q channel signals

If a design unit (such as a processor) provides multiple power domains, then it may also provide multiple Q channel interfaces. For example, a Cortex-M processor can have a system power domain and a debug power domain, and separate Q channel interfaces can be used for managing the power control activity.

The following example shows a power management scenario for a processor’s system power domain. At the starting stage of the power up sequence, the power gating control applies power to the processor, and in this scenario both the QACTIVE and QREQn signals are high before the processor starts (Figure 11).

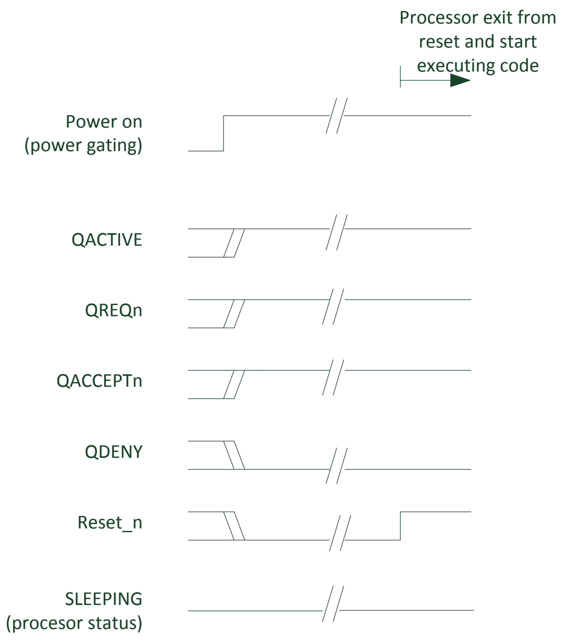


Figure 11: Example Q channel activity for system domain when a processor boot up

When the processor enters sleep mode, the power management unit detects the sleep operation, and then requests to change the processor to low power state (e.g. SRPG) by asserting QREQn to 0. The processor can then drive QACCEPTn low to indicate that the low power state request is accepted. After the processor accepts the low power mode request, it then puts the processor in the targeted low power state (i.e. SRPG), as shown in Figure 12.

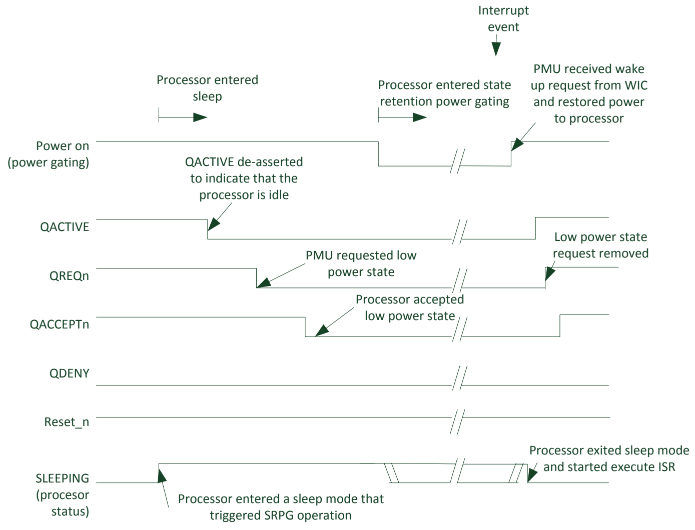


Figure 12: Example Q channel activity for system domain when a processor enter a sleep mode that uses SRPG

Assuming that the processor system has a Wakeup Interrupt Controller (WIC) in an always-on power domain, or similar hardware features, then a peripheral activity triggering an interrupt request can wake up the system via a separate connection between the WIC and the PMU. In this scenario the PMU can restore the power and clock signal activity to the processor system, and then the Q channel can complete its handshaking sequence (right hand side of Figure 12).

If an interrupt request arrives just after the processor has entered sleep mode, then it is possible for the processor to reject the low power state request using the QDENY signal. In such cases the PMU must not power down the processor in order to allow it to continue its operations, as shown in Figure 13.

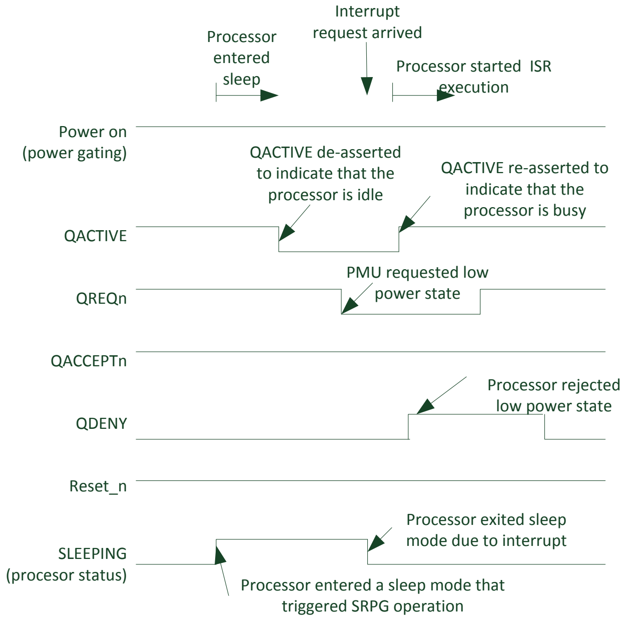


Figure 13: Example Q channel activity for system domain when a processor rejects a low power state request

It is possible that some power domains of a processor can start up in a low power state. For example, the debug power domain of a processor can be in an OFF state when the system starts, and turned on only when a debugger is connected. In such cases the QACTIVE and QREQn signals will have a start-up level of zero instead of one.

In addition to processor systems, the Q channel can also be used in other system components. Since the handshake protocol is fairly simple and is very generic, it can be deployed for many components of a low power microcontroller or System-on-Chip design.

8 SUMMARY

In this paper we introduced two on-chip protocols for low power microcontrollers and SoC designs:

- AMBA 5 AHB5 is a bus protocol for on-chip bus systems. It is an extension of AMBA 3 AHB Lite and provides support for ARM TrustZone technology, as well as improved support for a range of processor features.

- Q Channel is a protocol for power control. It is part of the AMBA 4 Low Power Interface Specification and enables a standardized control interface for a wide range of power management operations.

These two new interface protocols, combined with the upcoming processors based on the ARMv8-M architecture, will enable the next generation of chip designs to address security and low power requirements in a wide range of applications, including IoT devices and ultra-low power microcontrollers.

9 References

The following AMBA specification documents are referenced in this paper. This is where you can find the full details of the AHB5 protocol and Q channel interface.

#	Documentation
1	ARM AMBA 5 AHB specification http://infocenter.arm.com/help/topic/com.arm.doc.ihi0033/index.html
2	Low Power Interface Specification http://infocenter.arm.com/help/topic/com.arm.doc.ihi0068b/index.html