Cortex-M23 processor ARMv8-M IoT Kit User Guide

Version 1.0

Revision Information

The following revisions have been made to this document.

Date	Version	Confidentiality	Change
13 January 2017	1.0	Non-Confidential	First release

Proprietary Notice

Confidentiality Status

This document is Non-Confidential. The right to use, copy and disclose this document may be subject to license restrictions in accordance with the terms of the agreement entered into by ARM and the party that ARM delivered this document to.

Unrestricted Access is an ARM internal classification.

Product Status

The information in this document is final, that is for a developed product.

Web Address

http://www.arm.com

Contents

1 Introduction

This document provides a description of a reference system that uses TrustZone® technology to implement a secure subsystem. The system integrates a generic *Internet of Things* (IoT) subsystem that is referred to as a Kit, and includes extra system peripherals to form a full system. This system is intended to be implemented on the *Cortex-M Prototyping System* (MPS2+)-based system *Field Programmable Gate Array* (FPGA) and on Fast Models *Fixed Virtual Platform* (FVP).

1.1 IoT Kit MPS2+ system

The IoT Kit MPS2 system integrates the following two key parts:

- An IoT subsystem.
- An FPGA expansion subsystem.

Note

Only the components that are described in the IoT Kit subsystem are considered to be part of the Secure system. Other peripherals that may be present in the FPGA or FVP implementations are not within the scope of the Secure system.

1.2 IoT subsystem

The IoT subsystem integrates key components available from ARM to create a subsystem that can remain largely unmodified when integrated into different systems. The IoT subsystem integrates the following parts:

- An ARMv8-M processor core with an *Embedded Trace Macrocell* (ETM).

  Note

  The FVP does not support the ETM.
- A single bank of SRAM.
- CoreLink™ SIE-200 System IP for Embedded components including:
  - AHB5 *Memory Protection Controller* (MPC).
  - AHB5 *Peripheral Protection Controller* (PPC).
  - AHB5 *Exclusive Access Monitor* (EAM).
  - AHB5 to APB Bridge.
  - AHB5 Fabric.
- An *Implementation Defined Attribution Unit* (IDAU).
- A subsystem security controller.
- CMSDK components including.

- Timers and Dual Timers.
- Watchdog timer.

The IoT subsystem described here is the first-generation IoT subsystem Kit and does not implement the following:

- Power control.

  The subsystem is expected to be always ON.

- Clock control.

  The main system runs from a single clock source that is generated from outside the subsystem.

- Comprehensive reset generation.

  The subsystem contains a basic reset control. Power-on reset is generated externally for the system, and the processor is able to request and therefore cause a system reset that does not affect the debug logic within the processor.

1.3 FPGA expansion subsystem

The FPGA expansion subsystem extends the IoT subsystem by integrating more components to form a full example system. The FPGA expansion subsystem integrates the following:

- ZBT SRAM controllers provide access to on board ZBT SRAMs. These function as the main code memory and also as extra data storage memory.
- PL081 DMA controllers, primarily to acts as other masters within the system.
- An SRAM controller, to provide access to external devices with asynchronous SRAM like interfaces.
- An FPGA system controller.
- An *Implementation Defined Attribution Unit* (IDAU).
- CoreLink SIE-200 System IP for Embedded and CMSDK components including:
  - GPIOs.
  - UARTs.
  - AHB5 *Memory Protection Controller* (MPC).
  - AHB5 *Master Security Controller* (MSC).
  - AHB5 *Peripheral Protection Controller* (PPC).
  - AHB5 *Exclusive Access Monitor* (EAM).
  - AHB5 to APB Bridge.
  - AHB5 Fabric.

- PL022 SPI.
- I2S Controller.

2 System requirements and compliance

The IoT Kit MPS2 system is designed to meet requirements as follows:

- Generic TrustZone technology for ARMv8-M requirements.
- ARM IoT system design requirements.

2.1 Generic TrustZone technology for ARMv8-M requirements

The following requirements form a basic set that all systems that support TrustZone technology for ARMv8-M must have:

- Memory spaces (program and data) that are already, or can be partitioned, into Secure and Non-secure memory space.
- Peripherals that are already in, or can be placed into, Secure and Non-secure memory space.
- No access to Secure assets from the Non-secure world, which includes software running on processors in Non-secure mode, and peripherals that are Non-secure.
- Secure assets that can program code or data memory space, or any operating hardware and program state. This includes avoiding cases where states are inadvertently shared by using peripherals that are able to operate both in Secure and Non-secure mode concurrently.
- Boot-up from a memory space that is Secure, and optionally execute Non-secure firmware after Secure world initialization.
- Support for debug authentication signals that allow or disallow Secure debug and trace. This includes the ability to set the default reset values, and extra capability at boot either automatically or by Secure boot firmware to override the reset values after extra certificate authentication. The certificate authentication scheme is not defined in this document.

In an IoT SoC, Secure code and data memory are often implemented on-chip. Since this system is targeting the MPS2+ FPGA platform, most of the memory system is implemented with ZBT SRAMs on the circuit board. These SRAMs are also used to store Secure code, which means the Secure memory content is observable by probing the PCB board traces or pins. An attacker with physical access to the board might compromise the security of the system by viewing Secure memory content, modifying it, and so change the behavior of the program.

On an FPGA implementation of this platform, extra back doors might be provided to allow the designer to override some security measures. These include an access path from external on board *Microcontroller Unit* (MCU) to access the *Advanced High Performance Bus* (AHB) system using a *Serial Peripheral Interface* (SPI).

This path provides access to Secure peripherals and allows the MCU to control **SPIDEN** and **SPNIDEN** settings. Debug tool vendors can make use of this back door for tool testing.

2.2 ARM IoT system requirements

The IoT Kit MPS2 system is also designed to meet a set of ARM requirements. These requirements are selected with the aim of:

- Demonstrating the creation of a Secure hardware platform using ARMv8-M processors, bus, and peripheral components.
- Providing an example implementation of key blocks in the system.
- Creating a degree of standardization in parts of the system design to ease software and firmware developers as they move between different generations of the IoT subsystem.

These requirements are as follows:

- The system fabric is primarily AHB5 based.
- The system is partitioned into a IoT Kit subsystem and the expansion subsystem, where:
    - The IoT Kit subsystem contains the processor, and other key bus, peripheral and CMSDK components that are expected to be common among many IoT systems.
    - The expansion subsystem, where extra masters and peripherals are added to form an example system. For example, to target the FPGA on the MPS2 board.
- This partitioning also extends to memory and interrupt maps, where memory regions and interrupts signals specific to the IoT subsystem Kit are defined and reserved only for its own use, and in other areas that are provided as extension.
- The system uses ARMv8-M TrustZone technology and separates the system into two worlds, where:
    - The IDAU defines the main system partitioning between Secure and Non-secure world where there are strict association of addresses spaces with security.
    - Aliasing with extra security filters is used top map shared resources between the Secure and Non-secure worlds.
    - The *Secure Attribution Unit* (SAU) allows trusted firmware to map more memory for Secure use only, and define memory areas as Non-secure Callable.
- Basic Timers and Watchdog are placed within the IoT Kit subsystem to provide a standard set for use by software.

3 IoT Kit subsystem architecture

The IoT Kit subsystem integrates key components available from ARM to create a subsystem that can be integrated into different systems. The following figure shows the structure of the IoT Kit subsystem.

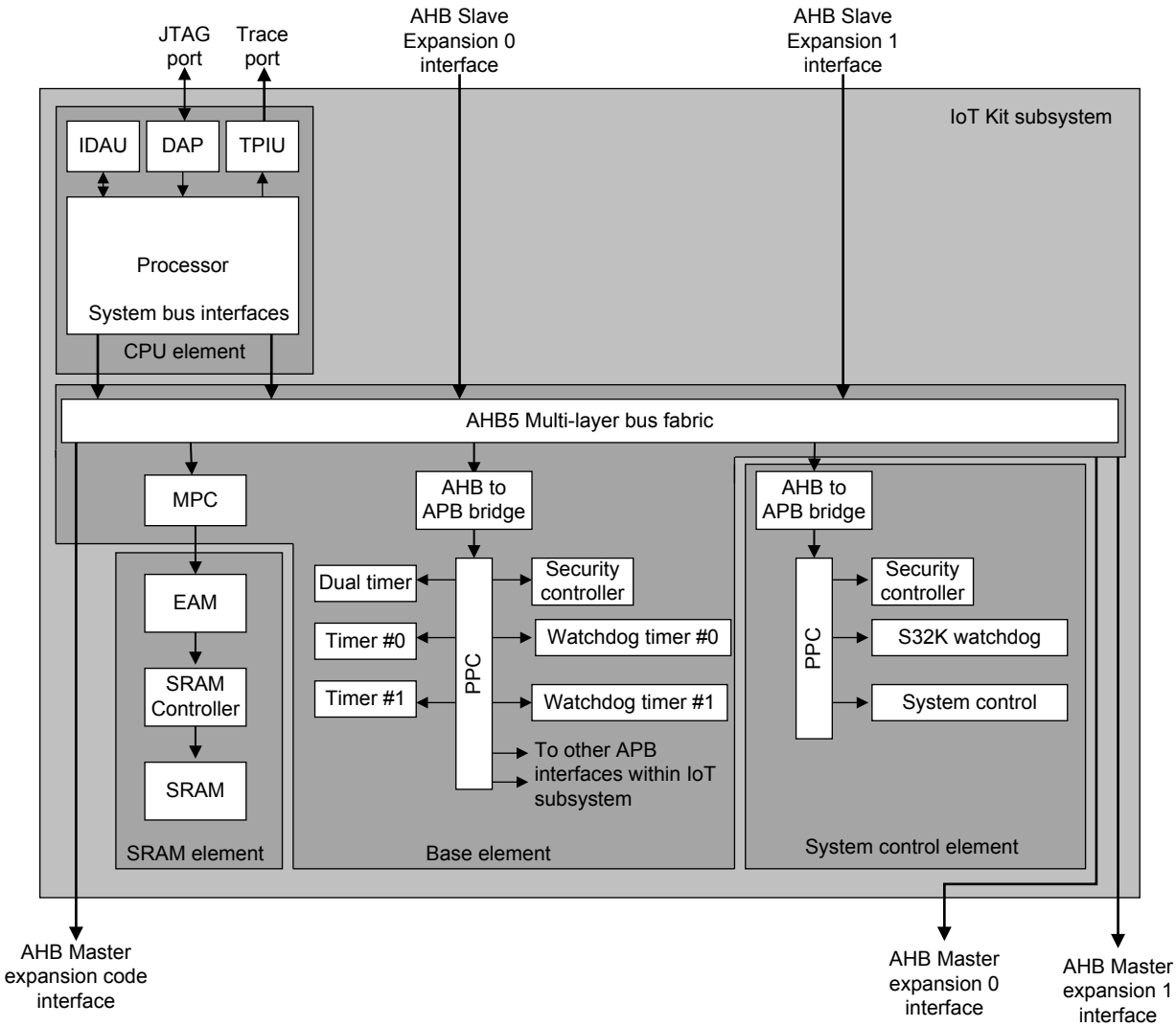

Figure 1 IoT Kit subsystem structure

The subsystem is divided into several elements:

- CPU element.

  This element contains an ARMv8-M processor.

- Base element.

  This element implements the main AHB5 and APB interconnect. In addition, it also contains several timers and watchdog timers. The *Memory Protection Controller* (MPC) and *Peripheral Protection Controller* (PPC) are also on paths to memory and peripherals respectively to provide security gating.

- The SRAM element implements the main storage RAM for the system which is primarily intended for data storage, though code can also reside in the memory. *Exclusive*

Access Monitors (EAMs) are implemented in this element to support exclusive access that is now supported on AMBA AHB5.

- The System control element contains logic to provide software controls for power, clocks, resets, and general system control. In addition, a slow watchdog timer and a simple dual timer running on a separate slow clock are implemented within this element. The PPC is also included to provide Security gating to all peripheral in the element.

3.1 System top-level interfaces

The top-level system provides the following interfaces:

- Functional clock and reset interfaces.
- AHB slave expansion interfaces.
- AHB master expansion interfaces.
- Interrupt interface.
- JTAG and SWO interface.
- Trace port.
- Debug authentication interface.
- Top-level static configuration signals.
- Security control expansion signals.

These interfaces are not visible in the FVP.

Functional clock and reset interfaces

These interfaces are not visible in the FVP.

The IoT Kit subsystem has the following clock and reset signals[1].

Signal Name	Width	IN/OUT	Description
MAINCLK	1	IN	Main Clock Input. This clock is used by the subsystem to generate most other clocks that are used within the system.
nPORESET	1	IN	Active low power-on reset input signal.
SYSCLK	1	OUT	Main system clock. This clock is used by the subsystem.

[1] FPGA Users should refer to Application Note AN519 Example IoT Kit Subsystem design for V2M-MPS2 (ARM DAI 0519)

S32KCLK	1	IN	Slow clock. A 32KHz clock input and is asynchronous to the other clocks in the subsystem.
SWCLKTCK	1	IN	SW/JTAG clock. Typically driven by an external debugger and is asynchronous to the other clocks in the subsystem.
nTRST	1	IN	Active low JTAG test logic reset signal.
TRACECLKIN	1	IN	Trace port clock input. Typically driven by the external Trace Port Analyzer.
TRACECLK	1	OUT	Exported trace port clock.
TRESETn	1	IN	Active low trace port asynchronous reset. Typically driven by the external Trace Port Analyzer.
nPORESET_OUT	1	OUT	Active low power-on reset for the expansion subsystem
nSYSRESET_OUT	1	OUT	Active low system reset for the expansion subsystem.
WARMRESETREQ	1	IN	Active high request to perform a warm system reset.

Table 1 Functional clock and reset related interfaces

The IoT subsystem uses a main input clock, **MAINCLK**, that is used to derive most other clocks within the subsystem. This includes **SYSCLK**, which is synchronous and at the same frequency as **MAINCLK**.

The **nPORESET** signal is the power-on reset signal for the system. This signal must be held for at least four **S32KCLK** cycles at power-on of the system.

The **S32KCLK** clock input is an asynchronous clock input that is used to drive the **S32KWATCHDOG** and **S32KTIMER** signals. The **SWCLKTCK** input clock and **nTRST** reset input are associated with the JTAG or *Serial Wire Out* (SWO) debug port. The **TRACECLKIN** input clock, **TRACECLK** output clock, and **TRESETn** reset inputs are associated with the Trace output port.

The **nPORESET_OUT** reset output is the power-on reset signal that is used by the expansion logic to the subsystem. The reset output merges other reset sources that are required to cause power-on reset. The **nSYSRESET_OUT** reset output is the system reset signal that is generated by the subsystem to perform system reset. This reset signal must not be used to reset any debug related logic. This reset output is also asserted if **nPORESET_OUT** is asserted. Both reset outputs are synchronous to **SYSCLK**, and will be sufficiently long enough in duration to reset logic on the S32KCLK domain after resynchronization to **S32KCLK**.

If the expansion logic to the subsystem requires system reset to be asserted, it can raise the request by asserting the **WARMRESETREQ** signal. This signal, when asserted, but is held high until the reset occurs on **nSYSRESET_OUT** and must be cleared as a result of the reset being asserted.

AHB Slave expansion interfaces

The IoT Kit subsystem provides two AMBA AHB5 slave expansion interfaces[2] to allow the system integrator to add extra bus masters to the system. These interfaces are named:

- AHB5 slave expansion 0 interface.
- AHB5 slave expansion 1 interface.

Each of these interfaces supports the following features:

- A full 32-bit address to allow access to the entire system memory map.
- 32-bit data width.
- TrustZone technology for ARMv8-M support, with the **HNONSEC** signal.
- Exclusive access support to SRAM memory.

AHB Master expansion interfaces

The IoT Kit subsystem provides two AMBA AHB5 master expansion interfaces[2] to allow the system integrator to add extra slave peripherals to the system. These interfaces are named:

- AHB5 Master expansion 0 interface.
- AHB5 Master expansion 1 interface.

In addition, a separate AHB5 Master interface is provided primarily to provide access to code memory. This interface is called the AHB5 Master expansion code interface.

Each of these interfaces supports the following features:

- A 32-bit address bus, with each access providing access to the full 32-bit address space.
- 32-bit data width.
- TrustZone technology for ARMv8-M support, with the inclusion of the **HNONSEC** signal.
- Exclusive access support, though the slave memory device in the expansion system that is expected to support exclusive access accessible must implement exclusive access monitoring in order for exclusive accesses to function correctly.

[2] FPGA Users should refer to Application Note AN519 Example IoT Kit Subsystem design for V2M-MPS2 (ARM DAI 0519)

Note

In the expansion system, a *Memory Protection Controller* (MPC) must be placed on the path to code memory on the AHB5 Master expansion code interface to provide security access gating for the aliased memory region that this interface supports.

Interrupt interface

The following table lists the interrupt signals for use by the expansion subsystem. These connect to the interrupt controller of the processor within the IoT Kit subsystem.

Signal Name	Width	IN/OUT	Description
EXP_IRQ[EXP_NUMIRQ-1:0]	EXP_NUMIRQ[3]	IN	These are interrupt inputs from the expansion subsystem to the interrupt controller of the processor core within the subsystem. The ARMv8-M core in the subsystem implements with 32 reserved interrupt lines for internal use and the remaining (EXP_NUMIRQ) made available for expansion on this interface. Each bit EXP_IRQ[n] is connected to IRQ[32+n] of the processor NVIC.
EXP_NMI	1	IN	This provides a *Non-maskable Interrupt* (NMI) input from the expansion subsystem to the interrupt controller of the processor core within the subsystem. This input is merged with other NMI interrupt sources with the IoT Kit subsystem before being seen by the NVIC of the processor core.

Table 2 Interrupt interface

[3] Defined in Table 12

JTAG and SWO interface

The processor JTAG interface is made available to the top level of the IoT Kit subsystem. The interrupt signals are listed in the following table. Refer to the *Integration and Implementation Manual* for the ARMv8-M processor for more information on this interface.

Signal Name	Width	IN/OUT	Description
TDI	1	IN	JTAG data in.
TDO	1	OUT	JTAG data out.
nTDOEN	1	OUT	JTAG TDO output enable.
SWDITMS	1	IN	Serial wire data input and JTAG TMS.
SWDO	1	OUT	Serial wire data output.
SWDOEN	1	OUT	Serial wire data output enable
SWSEL	1	OUT	Serial wire protocol active signal.
JTAGSEL	1	OUT	JTAG protocol active signal.

Table 3 JTAG and SWO interface

This interface is synchronous to the **SWCLKTCK** clock and the **nTRST** reset input resets it.

Trace port

The ARMv8-M processor trace port is made available to the top level of the IoT Kit subsystem. The interrupt signals are listed in the following table. See *the Integration and Implementation Manual* for the processor for more information on this interface.

Signal Name	Width	IN/OUT	Description
TRACECLK	1	OUT	Exported trace port clock.
TRACEDATA	4	OUT	Trace port data.
TRACESWO	1	OUT	Serial Wire Viewer data

Table 4 Trace port

This interface is synchronous to the **TRACECLK** clock output.

Debug authentication interface[4]

The following table lists the debug authentication signals of the IoT Kit subsystem.

The inputs signals define the debug authentication signal values when not overridden by the internal Debug Authentication Registers. The final debug authentication signals are then made available as outputs to the rest of the system.

Signal Name	Width	IN/OUT	Description
DBGEN_IN	1	IN	Debug enable input
NIDEN_IN	1	IN	Non-invasive Debug Enable Input
SPIDEN_IN	1	IN	Secure privilege invasive Debug Enable Input
SPNIDEN_IN	1	IN	Secure privilege Non-invasive Debug Enable Input
DBGEN	1	OUT	Merged Debug Enable Output
NIDEN	1	OUT	Merged Non-invasive Debug Enable Output
SPIDEN	1	OUT	Merged Secure privilege invasive Debug Enable Output
SPNIDEN	1	OUT	Merged Secure privilege Non-invasive Debug Enable Output

Table 5 Debug authentication interface

Note

The **DEVICEEN** input of the *Debug Access Port* (DAP) is connected to **DBGEN**. Therefore to begin JTAG-based Debug, **DBGEN** must be set to high.

Top-level static configuration signals.

There are currently no static configuration signals that are defined at the top level of the IoT Kit subsystem.

Security control expansion signals.

The IoT Kit subsystem provides extra status and control signals to handle more *Master Security Controllers* (MSCs), *Memory Protection Controllers* (MPCs), *Peripheral Protection* Controllers (PPCs,) and bridges with write buffers in the expansion system. These signals allow all to be controlled using the set of security control registers already implemented in the IoT Kit subsystem.

[4] FPGA Users should refer to Application Note AN519 Example IoT Kit Subsystem design for V2M-MPS2 (ARM DAI 0519)

Memory Protection Controller expansion

The IoT Kit can support up to 16 MPCs in the expansion system. The following signals allow the interrupts of the MPCs to be merged to the single MPC combined interrupt.

Signal Name	Width	IN/OUT	Description
S_MPCEXP_STATUS	16	IN	Interrupt status inputs from all expansion MPCs. These are visible to the programmer using the S_MPCEXP_STATUS register in the Secure Privilege Control Register block and are used to raise an interrupt using the MPC combined interrupt.

Table 6 MPC Expansion interrupt status input

APB Peripheral Protection Controller expansion

The IoT Kit can support up to four extra APB PPCs in the expansion system. The following signals are provided to control the PPCs.

Signal Name	Width	IN/OUT	Description
S_APBPPCEXP_STATUS	4	IN	APB PPC interrupt status input. Each bit 'N' is connected to a single APB PPC <N> where N is 0-3. These are associated to the S_APBPPCEXP_STATUS[3:0] field in the SECPPCINTSTAT register.
S_APBPPCEXP_CLEAR	4	OUT	APB PPC interrupt clear output. Each bit 'N' is connected to a single APB PPC<N> where N is 0-3.These are associated to the S_APBPPCEXP_CLR[3:0] field in the SECPPCINTCLR register.
APB_NS_PPCEXP0	16	OUT	APB PPC Non-secure gating control. These are a set of four 16-bit interfaces, and each interface connects to a PPC. When each bit 'm' of an interface is HIGH, it defines the APB<m> interface that the target PPC controls as Non-secure access only. Each 16-bit signal **APB_NS_PPCEXP<N>** is driven by the APBNSPPCECP<N> register
APB_NS_PPCEXP1	16	OUT	
APB_NS_PPCEXP2	16	OUT	
APB_NS_PPCEXP3	16	OUT	
APB_P_PPCEXP0	16	OUT	APB PPC Privilege gating control. These are a set of four 16-bit interfaces. When each bit 'm' of an interface is HIGH, it defines the APB<m> interface that the target PPC controls as privilege access only. Each signal is selected from either
APB_P_PPCEXP1	16	OUT	
APB_P_PPCEXP2	16	OUT	
APB_P_PPCEXP3	16	OUT	

			APBSPPPCEXP<N>[m] if APB_NS_PPCEXP<N>[m] is 0 or **APBNSPPPCEXP<N>[m]** otherwise.

Table 7 APB PPC expansion interface

AHB protection controller expansion

The IoT Kit can support up to four extra AHB PPCs in the expansion system. The following signals are provided to control each PPC.

Signal Name	Width	IN/OUT	Description
S_AHBPPCEXP_STATUS	4	IN	AHB PPC interrupt status input. Each bit 'N' is connected to a single AHB PPC <N> where N is 0-3. These are associated to the S_AHBPPCEXP_STATUS[3:0] field in the SECPPCINTSTAT register.
S_AHBPPCEXP_CLEAR	4	OUT	AHB PPC interrupt clear output. Each bit 'N' is connected to a single AHB PPC<N> where N is 0-3. These are associated to the S_AHBPPCEXP_CLR[3:0] field in the SECPPCINTCLR register.
AHB_NS_PPCEXP0	16	OUT	AHB PPC Non-secure gating control. These are a set of four 16-bit interfaces, and each interface connects to a PPC. When each bit 'm' of an interface is HIGH, it defines the AHB<m> interface that the target PPC controls as Non-secure access only. Each 16-bit signal **AHB_NS_PPCEXP<N>** is driven by the AHBNSPPCECP<N> register
AHB_NS_PPCEXP1	16	OUT	
AHB_NS_PPCEXP2	16	OUT	
AHB_NS_PPCEXP3	16	OUT	
AHB_P_PPCEXP0	16	OUT	AHB PPC Privilege gating control. These are a set of four 16-bit interfaces. When each bit 'm' of an interface is HIGH, it defines the AHB<m> interface that the target PPC controls as privilege access only. Each signal is selected from either **AHBSPPPCEXP<N>[m]** if AHB_NS_PPCEXP<N>[m] = '0' or **AHBNSPPPCEXP<N>[m]** otherwise.
AHB_P_PPCEXP1	16	OUT	
AHB_P_PPCEXP2	16	OUT	
AHB_P_PPCEXP3	16	OUT	

Table 8 AHB PPC expansion interface

Master Security Controller expansion

The IoT Kit supports up to 16 extra MSCs to be added to the expansion system. The following signals are provided to control each PPC.

Signal Name	Width	IN/OUT	Description
S_MSCEXP_STATUS	16	IN	MSC interrupt status input. Each bit 'N' is connected to a single MSC <N> where N is 0-15. These are associated with the S_MSCEXP_STATUS[15:0] field in the SECMSCINTSTAT register.
S_MSCEXP_CLEAR	16	OUT	MSC interrupt clear output. Each bit 'N' is connected to a single MSC <N> where N is 0-15. These are associated with the S_MSCEXP_CLR[15:0] field in the SECMSCINTCLR register.
NS_MSCEXP	16	OUT	MSC Non-secure configuration. Each bit 'N' is connected to a single MSC <N> where N is 0-15. Set to HIGH to configure a master as Non-secure. These are associated with the NS_MSCEXP[15:0] field in the NSMSCECP register

Table 9 MSC Expansion interface

Bridge buffer error expansion

Some bridges in the system can contain write buffers. To avoid slowing down bus interfaces, buffered write access arriving at these bridges can be completed in advance without error. If any bus error occurs downstream, interrupts are used to notify the processor of the error. The IoT Kit supports up to 16 extra bridges with buffer error signaling to be added to the expansion system.

Signal Name	Width	IN/OUT	Description
BRGEXP_STATUS	16	IN	Bridge error interrupt status input. Each bit 'N' is connected to a single bridge <N> where N is 0-15. These are associated with the BRGEXP_STATUS[15:0] field in the BRGINSTAT Register
BRGEXP_CLEAR	16	OUT	Bridge error interrupt clear output. Each bit 'N' is connected to a single bridge <N> where N is 0-15. These are associated with the BRGEXP _CLR[15:0] field in the BRGINSTAT Register

Table 10 Bridge error interrupt expansion Interface

Other security expansion signals

The following table lists other signals related to security that are required by PPCs and MSCs in the expansion system.

Signal Name	Width	IN/OUT	Description
SEC_RESP_CFG	1	OUT	This signal configures how to respond to an access when a security violation occurs. 0 Read-Zero Write Ignore 1 Bus error This signal is controlled by the SECRESPCFG register.

Table 11 Other security expansion signals

3.2 Top-level system parameters

The following table lists all the parameters that are available at the top level of the IoT Kit subsystem for user configuration. These parameters are not directly visible to the FVP.

Parameter	Default values	Description
INITSVTOR0_RST[31:0]	`0x1000_0000`	Reset value of the Secure vector table offset address register in the System Control Register.
INITNSVTOR0[31:0]	`0x0000_0000`	Reset value of the Non-secure vector table offset address at the processor core.
CPU0WAIT_RST	`0`	CPU wait at boot: 0 boot normally 1 wait at boot.
EXP_NUMIRQ	64	Specifies the number of expansion interrupts. Set to 64. This means that the NVIC has 64+32 = 96 interrupts. Minimum value of 2. **Note** The FPGA implementation changes this value to 92.
EXP_IRQ_DIS_0[EXP_NUMIRQ -1:0]	EXP_IRQ_DIS_ 0[EXP_NUMIR Q-1:0] all set to high.	Disables support for individual expansion interrupts on the primary processor core, allowing a range of non-contiguous interrupts IRQDIS[i] = 1 indicates that IRQ[i] is not present
EXP_SYS_ID_PRESENT[31:16]	`0xFFFF`	Each bit n of this vector defines if an AHB master with HMASTERID = n exist in the system. Bit 15 down to 0 are all IDs reserved for internal use and not available on this interface.
CPU0_FPU	1	*Floating Point Unit* (FPU) is present on the processor
CPU0_DSP	1	DSP extension instructions are included on the processor.
CPU0_MPU_NS	8	Number of Non-secure MPU entries on the processor
CPU0_MPU_S	8	Number of Secure MPU entries on the processor
CPU0_SAU	8	Number of SAU entries on the

		processor
CPU0_IRQ_LVL	4	Number of interrupt priorities that are implemented in the NVIC, equal to $2^{CPU0\_IRQ\_LVL}$. Supports 3 to 8 bits. Currently at four. This provides 16 levels of interrupt priority.

Table 12 Top-level user configurable parameters

3.3 System memory map

The following table shows the high-level view of the memory map of the IoT Kit subsystem. This memory map is divided into Secure and Non-secure regions. In general, the memory alternates between Secure and Non-secure regions in 256MB steps, with only a few address areas that are exempted from security mapping because they are related to debug functionality.

To provide memory and peripherals that can be mapped as Secure or Non-secure using software, several address regions are aliased as shown in the table. *The Implementation Defined Attribution Unit* (IDAU) region ID, and the *Non-secure Callable* (NSC) setting for each region are also shown in the table.

Note

> For a row where the column **Alias with row ID** is not empty, the column indicates which other row entry it is aliased to in the memory map within the same table for that entry.

In general, except when stated, all access to unmapped regions of the memory result in a bus-error response. An exception to that is when accessing unmapped address space within a 4KB region of a peripheral area that did not result in a security violation. In this case, the access is Read Zero or Write Ignored (RZWI). Any accesses that result in security violations will either RZWI or result in a bus error response depending on the SECRESPCFG register setting.

Some regions of memory map are reserved to maintain compatibility as more features are added into future subsystem. Other areas are mapped to AHB Master expansion 0 interface and AHB Master expansion 1 interface.

Row ID	Address		Size	Region name	Description	Alias with row ID	IDAU region values		
	From	To					Security[5]	IDAU ID	NSC
1	0x0000_0000	0x0DFF_FFFF	224MB	Code memory	Maps to AHB5 Master expansion code Interface	3[6]	NS	0	0
2	0x0E00_0000	0x0FFF_FFFF	32MB	Reserved	Reserved				
3	0x1000_0000	0x1DFF_FFFF	224MB	Code Memory	Maps to AHB5 Master expansion code interface	1[2]	S	1	CODE NSC[7]

[5] This column does not define Privileged or Unprivileged accessibility. These are defined by the MPC, PPC, or by the register blocks that are mapped to each area.
[6] Even though they actually not aliased at the interface, these areas are expected to be aliased by customers for Non-secure and Secure shared code memory. In addition you should use Memory Protection Controller(s) externally to selectively map each block of memory between secure and non-secure memory regions.
[7] The NSC values are currently defined in the FVP model using parameters only.

4	0x1E00_0000	0x1FFF_FFFF	32MB	Reserved	Reserved				
5	0x2000_0000	0x20FF_FFFF	16MB	Internal SRAM	Internal SRAM area.	8	NS	2	0
6	0x2100_0000	0x27FF_FFFF	112MB	Reserved	Reserved				
7	0x2800_0000	0x2FFF_FFFF	128MB	Expansion 0	Maps to AHB5 master expansion 0 interface				
8	0x3000_0000	0x30FF_FFFF	16MB	Internal SRAM	Internal SRAM area.	5	S	3	RAM NSC2
9	0x3100_0000	0x37FF_FFFF	112MB	Reserved	Reserved				
10	0x3800_0000	0x3FFF_FFFF	128MB	Expansion 0	Maps to AHB5 Master expansion 0 interface				
11	0x4000_0000	0x4000_FFFF	64KB	Base peripheral	Base element peripheral region.		NS	4	0
12	0x4001_0000	0x4001_FFFF	64KB	Private CPU	CPU element peripheral region.	18			
13	0x4002_0000	0x4002_FFFF	64KB	System Control	System Control element peripheral region.	19			
14	0x4003_0000	0x4007_FFFF		Reserved	Reserved				
15	0x4008_0000	0x400F_FFFF	512KB	Base peripheral	Base element peripheral region.				
16	0x4010_0000	0x4FFF_FFFF	255MB	Expansion 1	Maps to AHB5 Master expansion 1 interface				
17	0x5000_0000	0x5000_FFFF	64KB	Base peripheral	Base element peripheral region.		S	5	0
18	0x5001_0000	0x5001_FFFF	64KB	Private CPU	CPU element peripheral region.	12			
19	0x5002_0000	0x5002_FFFF	64KB	System Control	System Control element peripheral region.	13			
20	0x5003_0000	0x5007_FFFF		Reserved	Reserved				
21	0x5008_0000	0x500F_FFFF	512KB	Base peripheral	Base element peripheral region.				
22	0x5010_0000	0x5FFF_FFFF	255MB	Expansion 1	Maps to AHB5				

					Master expansion 1 interface			
23	0x6000_0000	0x6FFF_FFFF	256MB	Expansion 0	Maps to AHB5 Master expansion 0 interface	NS	6	0
24	0x7000_0000	0x7FFF_FFFF	256MB	Expansion 0	Maps to AHB5 Master expansion 0 interface	S	7	0
25	0x8000_0000	0x8FFF_FFFF	256MB	Expansion 1	Maps to AHB5 Master expansion 1 interface	NS	8	0
26	0x9000_0000	0x9FFF_FFFF	256MB	Expansion 1	Maps to AHB5 Master expansion 1 interface	S	9	0
27	0xA000_0000	0xAFFF_FFFF	256MB	Expansion 1	Maps to AHB5 Master expansion 1 interface	NS	A	0
28	0xB000_0000	0xBFFF_FFFF	256MB	Expansion 1	Maps to AHB5 Master expansion 1 interface	S	B	0
29	0xC000_0000	0xCFFF_FFFF	256MB	Expansion 1	Maps to AHB5 Master expansion 1 interface	NS	C	0
30	0xD000_0000	0xDFFF_FFFF	256MB	Expansion 1	Maps to AHB5 Master expansion 1 interface	S	D	0
31	0xE000_0000	0XE00F_FFFF	1MB	PPB	Private Peripheral Bus. Local to each core.	Exempt		
32	0xE010_0000	0XEFFF_FFFF	255MB	Expansion 1	Maps to AHB5 Master expansion 1 interface	NS	E	0
33	0xF000_0000	0XF00F_FFFF	1MB	System Debug	System Debug.	Exempt		
34	0xF010_0000	0XFFFF_FFFF	255MB	Expansion 1	maps to AHB5 Master expansion 1 interface	S	F	0

Table 13 High-level system address map

Internal SRAM regions

The internal SRAM regions are the location of SRAM within the subsystem. Currently, only a single SRAM is mapped into both the Secure and Non-secure regions. The remaining regions are reserved. An MPC then determines how the memory locations within the SRAM are mapped to the Secure and Non-secure regions.

ROW ID	Address		Size	Region Name	Description	Alias With Row ID	Security[8]
	From	To					
1	0x2000_0000	0x2000_7FFF	32KB	SRAM Bank 0	Maps to internal SRAM Bank 0	3	NS-MPC
2	0x2000_8000	0x20FF_FFFF		Reserved	Reserved		
3	0x3000_0000	0x3000_7FFF	32KB	SRAM Bank 0	Maps to internal SRAM Bank 0	1	S-MPC
4	0x3000_8000	0x30FF_FFFF		Reserved	Reserved		

Table 14 SRAM Region Address Map

Base peripheral regions

The Base peripheral regions are where peripherals of the Base element reside. There are four regions in total, two Secure and two Non-secure regions. Some peripherals are aliased to both Secure and Non-secure regions. The final mapping to both the Secure or Non-secure world, and Privileged or Non-privileged world is determined by PPCs.

Row ID	Address		Size	Region name	Description	Alias with row ID	Security[9]
	From	To					
1	0x4000_0000	0x4000_0FFF	4KB	Timer 0	CMSDK timer	8	NS-PPC
2	0x4000_1000	0x4000_1FFF	4KB	Timer 1	CMSDK timer	9	NS-PPC
3	0x4000_2000	0x4000_2FFF	4KB	Dual Timer	CMSDK dual timer	10	NS-PPC
4	0x4000_3000	0x4000_FFFF		Reserved	Reserved		
5	0x4008_0000	0x4008_0FFF	4KB	NSPCTRL	Non-secure privilege control block.		NSP
6	0x4008_1000	0x4008_1FFF	4KB	Non-secure Watchdog	Non-secure CMSDK watchdog		NSP
7	0x4008_2000	0x400F_FFFF		Reserved	Reserved		
8	0x5000_0000	0x5000_0FFF	4KB	Timer 0	CMSDK timer	1	S-PPC
9	0x5000_1000	0x5000_1FFF	4KB	Timer 1	CMSDK timer	2	S-PPC
10	0x5000_2000	0x5000_2FFF	4KB	Dual Timer	CMSDK dual timer	3	S-PPC
11	0x5000_3000	0x5000_FFFF		Reserved	Reserved		

[8] NS_MPC: Non-secure access only gated by an MPC. S_MPC: Secure access only gated by an MPC.
[9] NS_MPC: Non-secure access only gated by a PPC. S_PPC: Secure access only gated by a PPC. NSP: Non-secure privilege access only. SP: Secure privilege access only.

12	0x5008_0000	0x5008_0FFF	4KB	SPCTRL	Secure privilege control block	
13	0x5008_1000	0x5008_1FFF	4KB	Secure Watchdog	Secure CMSDK watchdog	SP
14	0x5008_2000	0x5008_2FFF		Reserved	Reserved	
15	0x5008_3000	0x5008_3FFF	4KB	SRAM0MPC	SRAM 0 Memory Protection Controller.	SP
16	0x5008_4000	0x500F_FFFF		Reserved	Reserved	

Table 15 Base peripheral regions address map

Private CPU region

The Private CPU region is privately visible to each processor element. If there are multiple processor elements in the subsystem, each only sees its own implementation of this region. Currently, no peripherals are implemented within this block and this area is reserved for future use.

System control

The System control region is where peripherals in the System control element reside.

Row ID	Address		Size	Region name	Description	Alias with row ID	Security[10]
	From	To-					
1	0x4002_0000	0x4002_1FFF	4KB	SYSINFO	System Information block.	4	NS
	0x4002_1000	0x4002_EFFF		Reserved	Reserved		
2	0x4002_F000	0x4001_FFFF	4KB	S32KTIMER	CMSDK timer running on S32KCLK	6	NS-PPC
3	0x5002_0000	0x5002_0FFF	4KB	SYSINFO	System Information block.	1	S
4	0x5002_1000	0x5002_1FFF	4KB	SYSCONTROL	System Control block.	1	SP
	0x5002_1000	0x5001_FFFF	-	Reserved	Reserved	-	-
5	0x5002_E000	0x5002_EFFF	4KB	S32KWATCHDOG	CMSDK Watchdog running on S32KCLK		SP
6	0x5002_F000	0x5001_FFFF	4KB	S32KTIMER	CMSDK Timer running on S32KCLK	2	S-PPC

Table 16 System control region address map

[10] NS_MPC: Non-secure access only gated by a PPC. S_PPC: Secure access only gated by a PPC. NSP: Non-secure privilege access only. SP: Secure privilege access only.

PPB region

The PPB memory region provides access to internal and external processor resources. This includes the following:

- The *Instrumentation Trace Macrocell* (ITM).
- The *Data Watchpoint and Trace* (DWT).
- The *Flashpatch and Breakpoint* (FPB).
- The *System Control Space* (SCS), including the *Memory Protection Unit* (MPU) and the *Nested Vectored Interrupt Controller* (NVIC).
- The *Embedded Trace Macrocell* (ETM).
- *Cross Trigger Interface* (CTI), if included.
- The Debug ROM table.

This region is as defined in the *ARMv8-M Architecture Reference Manual* and the *ARM Integration and Implementation Manual* for the Processor.

Note

The FVP does not implement the ITM, ETM, or CTI.

System debug region

This region is reserved.

3.4 CPU element

The IoT Kit subsystem contains a single ARMv8-M processor core. It is configured with the following features:

Parameter	Configuration	Description
SECEXT	1	ARMv8-M Security Extension is included.
MPU_NS	CPU0_MPU_NS	8 Non-secure *Memory Protection Unit* (MPU) regions included.
MPU_S	CPU0_MPU_S	8 Secure MPU regions included.
SAU	CPU0_SAU	8 *Secure Attribution Unit* (SAU) regions included.
NUMIRQ	EXP_NUMIRQ + 32	Number of user interrupts implemented.
IRQLVL	4	Defines the number of bits of interrupt priority that is implemented in the NVIC, which therefore provides $2^{CPU0\_IRQ\_LVL}$ levels of interrupt priority.
IRQLATENCY	All zeros	Set all interrupts to not low latency.
IRQDIS	All zeros	Disable support for individual interrupt.
DBGLVL	2	Full debug resources included, which includes four watchpoint and eight breakpoint comparators.
ITM	Build dependent	Refer to Application Note AN519.
ETM	Build dependent	Refer to Application Note AN519.
MTB	Build dependent	Refer to Application Note AN519.
MTBWIDTH	12	12 bits MTB RAM interface address width. But not used.
WIC	1	WIC included.
WICLINES	EXP_NUMIRQ + 35	All interrupts are sensitive to WIC.
CTI	1	CTI not included.
RAR	1	Only reset the architecturally

	required state.

Table 17 IoT Kit subsystem processor configuration settings

The processor has several static configuration input signals. These are tied as shown in the following table.

Signal Name	Tie value	Description
CFGSSTCALIB[25:0]	`0x200_0000`	Secure SysTick calibration configuration indicating that no alternative reference clock is provided, and the frequency of clock arriving at the processor is not computable in hardware.
CFGSSTCALIB[23:0]	TENMS	= `0x00_0000`
CFGSSTCALIB[24]	SKEW	= LOW
CFGSSTCALIB[25]	NOREF	= HIGH
CFGNSSTCALIB[25:0]	`0x200_0000`	Non-secure SysTick calibration configuration indicating that no alternative reference clock is provided, and the frequency of clock arriving at the processor is not computable in hardware.
CFGNSSTCALIB[23:0]	TENMS	= `0x00_0000`
CFGNSSTCALIB[24]	SKEW	= LOW
CFGNSSTCALIB[25]	NOREF	= HIGH
CFGFPU	1	FPU not present.
CFGDSP	1	DSP not present.
CFGSECEXT	1	ARMv8-M security support enabled.
MPUNSDISABLE	0	Disable support for the Non-secure MPU not enabled.
MPUSDISABLE	0	Disable support for the Secure MPU not enabled.
SAUDISABLE	0	Disable support for the SAU not enabled.

Table 18

IoT Kit subsystem processor static configuration input signals settings

The CPU element also integrates the *Debug Access Port* (DAP) and the *Trace Port Interface Unit* (TPIU) provided with the processor. Their respective JTAG and Trace interfaces are available as the interfaces on the top level of the IoT subsystem.

Both Code and System AHB interfaces are connected to the AHB5 Interconnect in the Base element

The FVP does not implement DAP or TPIU.

CPU_IDENTITY

The CPU element also implements a CPU_IDENTITY register block that is only visible to accesses on the system interface from the processor.

This base address of this register is `0x4001_F000` in a Non-secure region and `0x5001_F000` in the Secure region. Both areas are always read accessible. Write accesses to this register block are always ignored.

Note

In the single processor system defined in this document, CPU_IDENTITY is not needed. Therefore this register block is not implemented in this IoT Kit subsystem and reserved. Access to this register block results in a bus error response, or RAZ/WI in the FVP.

3.5 Base element

The Base element integrates the following CoreLink SIE-200 components:

- AHB5 interconnect
- *Memory Protection Controllers* (MPC)
- AHB5 to APB bus converters
- APB *Peripheral Protection Controller* (PPC)
- *Exclusive Access Monitors* (EAM)

The Base element also integrates the following from the *Cortex-M System Design Kit* (CMSDK)

- Timers and Dual Timers,
- Watchdogs

The IoT Kit subsystem also includes a security controller that is unique to the subsystem.

Secure privilege control block

The Secure privilege control block is part of the security controller, and implements program visible states that allow software to control security gating units within the design.

This register block base address is `0x5008_0000`.

This register is Secure privileged access only and supports 32-bit read/write accesses.

The following table lists the registers within this unit. For write access to these registers, only 32-bit writes are supported. Any byte and halfword writes result in the write data being ignored.

Offset	Name	Access	Reset value	Description
0x000-0x008	Reserved		0x0000_0000	Reserved
0x010	SECRESPCFG	Read/write	0x0000_0000	Security Violation Response Configuration Register
0x014	NSCCFG	Read/write	0x0000_0000	Non Secure Callable Configuration for IDAU
0x018	Reserved		0x0000_0000	Reserved
0x01C	SECMPCINTSTATUS	Read-only	0x0000_0000	Secure MPC Interrupt Status
0x020	SECPPCINTSTAT	Read-only	0x0000_0000	Secure PPC Interrupt Status
0x024	SECPPCINTCLR	Write-only	0x0000_0000	Secure PPC Interrupt Clear
0x028	SECPPCINTEN	Read/write	0x0000_0000	Secure PPC Interrupt Enable
0x02C	Reserved		0x0000_0000	Reserved
0x030	SECMSCINTSTAT	Read-only	0x0000_0000	Secure MSC Interrupt Status
0x034	SECMSCINTCLR	Write-only	0x0000_0000	Secure MSC Interrupt Clear
0x038	SECMSCINTEN	Read/write	0x0000_0000	Secure MSC Interrupt Enable
0x03C	Reserved		0x0000_0000	Reserved

0x040	BRGINTSTAT	Read-only	0x0000_0000	Bridge Buffer Error Interrupt Status
0x044	BRGINTCLR	Write-only	0x0000_0000	Bridge Buffer Error Interrupt Clear
0x048	BRGINTEN	Read/write	0x0000_0000	Bridge Buffer Error Interrupt Enable
0x04C	Reserved		0x0000_0000	Reserved
0x050	AHBNSPPC0	Read/write	0x0000_0000	Non-secure access AHB slave peripheral Protection Control #0. Defines the Non-secure access settings for peripherals in the Base element
0x054-0x05C	Reserved for AHBNSPPC1 to AHBNSPPC3		0x0000_0000	Reserved for Future Non-secure access AHB Slave peripheral Protection Control
0x060	AHBNSPPCEXP0	Read/write	0x0000_0000	Expansion 0 Non-secure access AHB slave peripheral Protection Control. Each field defines the Non-secure access settings for an associated peripheral: 1 Allow Non-secure access 0 Disallow Non-secure access Resets to 0
0x064	AHBNSPPCEXP1	Read/write	0x0000_0000	Expansion 1 Non-secure access AHB slave peripheral Protection Control. Each field defines the Non-secure access settings for an associated peripheral: 1 Allow Non-secure access 0 Disallow Non-secure access Resets to 0
0x068	AHBNSPPCEXP2	Read/write	0x0000_0000	Expansion 2 Non-secure Access AHB slave peripheral Protection Control. Each field defines the Non-secure access settings for an associated peripheral: 1 Allow Non-secure access 0 Disallow Non-secure access Resets to 0
0x06C	AHBNSPPCEXP3	Read/write	0x0000_0000	Expansion 3 Non-secure access AHB slave Peripheral Protection Control. Each field defines the Non-secure access settings for an associated peripheral: 1 Allow Non-secure access 0 Disallow Non-secure access Resets to 0
0x070	APBNSPPC0	Read/write	0x0000_0000	Non-secure access APB slave Peripheral Protection Control #0.
0x074	APBNSPPC1	Read/write	0x0000_0000	Non-secure access APB slave Peripheral Protection Control #1. This register controls the PPC within the System Control element.
0x078-0x07C	Reserved for APBNSPPC2 to APBNSPPC3		0x0000_0000	Non-secure Access APB slave Peripheral Protection Control [3:1]
0x080	APBNSPPCEXP0	Read/write	0x0000_0000	Expansion 0 Non-secure access APB slave Peripheral

				Protection Control. Each field defines the Non-secure access settings for an associated peripheral: 1 Allow Non-secure access 0 Disallow Non-secure access Resets to 0
0x084	APBNSPPCEXP1	Read/write	0x0000_0000	Expansion 1 Non_Secure access APB slave Peripheral Protection Control. Each field defines the Non-secure access settings for an associated peripheral: 1 Allow Non-secure access 0 Disallow Non-secure access Resets to 0
0x088	APBNSPPCEXP2	Read/write	0x0000_0000	Expansion 2 Non-secure access APB slave Peripheral Protection Control. Each field defines the Non-secure access settings for an associated peripheral: 1 Allow Non-secure access 0 Disallow Non-secure access Resets to 0
0x08C	APBNSPPCEXP3	Read/write	0x0000_0000	Expansion 3 Non-secure access APB slave Peripheral Protection Control. Each field defines the Non-secure access settings for an associated peripheral: 1 Allow Non-secure access 0 Disallow Non-secure access Resets to 0
0x090	AHBSPPPC0	Read-only	0x0000_0000	Secure Unprivileged access AHB slave Peripheral Protection Control #0.
0x094-0x09C	Reserved for AHBSPPPC1 to AHBSPPPC3		0x0000_0000	Reserved for Future Secure Unprivileged access AHB slave Peripheral Protection Control
0x0A0	AHBSPPPCEXP0	Read/write	0x0000_0000	Expansion 0 Secure Unprivileged access AHB slave Peripheral Protection Control. Each field defines the Secure Unprivileged access settings for an associated peripheral: 1 Allow Non-secure access 0 Disallow Non-secure access Resets to 0
0x0A4	AHBSPPPCEXP1	Read/write	0x0000_0000	Expansion 1 Secure Unprivileged access AHB slave Peripheral Protection Control. Each field defines the Secure Unprivileged access settings for an associated peripheral: 1 Allow Non-secure access 0 Disallow Non-secure access Resets to 0
0x0A8	AHBSPPPCEXP2	Read/write	0x0000_0000	Expansion 2 Secure Unprivileged access AHB slave

				Peripheral Protection Control. Each field defines the Secure Unprivileged access settings for an associated peripheral: 1 Allow Non-secure access 0 Disallow Non-secure access Resets to 0
0x0AC	AHBSPPPCEXP3	Read/write	0x0000_0000	Expansion 3 Secure Unprivileged access AHB slave Peripheral Protection Control. Each field defines the Secure Unprivileged access settings for an associated peripheral: 1 Allow Non-secure access 0 Disallow Non-secure access Resets to 0
0x0B0	APBSPPPC0	Read/write	0x0000_0000	Secure Unprivileged access APB slave Peripheral. Protection Control #0. This register control the PPC within the Base element
0x0B4	APBSPPPC1	Read/write	0x0000_0000	Secure Unprivileged Access APB slave Peripheral. Protection Control #1. This register controls the PPC within the System Control element.
0x0B8-0x0BC	Reserved for APBSPPPC2 to APBSPPPC3		0x0000_0000	Reserved for Future Secure Unprivileged Access APB slave Peripheral Protection Control
0x0C0	APBSPPPCEXP0	Read/write	0x0000_0000	Expansion 0 Secure Unprivileged Access APB slave Peripheral Protection Control. Each field defines the Secure Unprivileged access settings for an associated peripheral: 1 Allow Non-secure access 0 Disallow Non-secure access Resets to 0
0x0C4	APBSPPPCEXP1	Read/write	0x0000_0000	Expansion 1 Secure Unprivileged access APB slave Peripheral Protection Control. Each field defines the Secure Unprivileged access settings for an associated peripheral: 1 Allow Non-secure access 0 Disallow Non-secure access Resets to 0
0x0C8	APBSPPPCEXP2	Read/write	0x0000_0000	Expansion 2 Secure Unprivileged access APB slave Peripheral Protection Control. Each field defines the Secure Unprivileged access settings for an associated peripheral: 1 Allow Non-secure access 0 Disallow Non-secure access Resets to 0
0x0CC	APBSPPPCEXP3	Read/write	0x0000_0000	Expansion 3 Secure Unprivileged access APB slave Peripheral Protection Control. Each field defines the Secure Unprivileged access settings for an associated peripheral: 1 Allow Non-secure access

				0 Disallow Non-secure access Resets to 0
0x0D0	NSMSCEXP	Read-only	0x0000_0000	Expansion MSC Non-secure Configuration. Each field defines if a Master connected to an Expansion Master Security Controller is Secure or Non-secure: 1 Master is Non-secure, 0 Master is Secure.
0x0D4 – 0xFCC	Reserved		0x0000_0000	Reserved
0xFD0	PID4	Read-only	0x0000_0004	Peripheral ID 4
0xFD4	PID5	Read-only	0x0000_0000	Reserved
0xFD8	PID6	Read-only	0x0000_0000	Reserved
0xFDC	PID7	Read-only	0x0000_0000	Reserved
0xFE0	PID0	Read-only	0x0000_0052	Peripheral ID 0
0xFE4	PID1	Read-only	0x0000_00B8	Peripheral ID 1
0xFE8	PID2	Read-only	0x0000_000B	Peripheral ID 2
0xFEC	PID3	Read-only	0x0000_0000	Peripheral ID 3
0xFF0	CID0	Read-only	0x0000_000D	Component ID 0
0xFF4	CID1	Read-only	0x0000_00F0	Component ID 1
0xFF8	CID2	Read-only	0x0000_0005	Component ID 2
0xFFC	CID3	Read-only	0x0000_00B1	Component ID 3

Table 19 Secure Privilege Control Register Map

SECRESPCFG

The Security Violation Slave Response Configuration Register is used to define the slave response to an access that causes security violation on the bus fabric.

Bits	Name	Access	Width	Reset value	Description
0	SECRESPCFG	Read/write	1	`0`	This field configures the slave response if there was a security violation: 0: Read-Zero Write Ignore 1: bus error
31:1	Reserved	Read-only	31	`0x00000000`	Reserved

Table 20 SECRESPCFG register

NSCCFG

The Non-secure Callable Configuration Register allows software to define if the Secure Code region `0x1000_0000` to `0x1FFF_FFFF` and the Secure SRAM region `0x3000_0000` to `0x3FFF_FFFF` are Non-secure Callable regions of memory.

Bits	Name	Access	Width	Reset value	Description
0	CODENSC	Read/write	1	`0`	Configures if the CODE region

					(0x1000_0000 to 0x1FFF_FFFF) is Non-secure Callable: 0 Not Non-secure Callable 1 Non-secure Callable
1	RAMNSC	Read/write	1	0	Configures if the RAM region (0x3000_0000 to 0x3FFF_FFFF) is Non-secure Callable: 0 Not Non-secure Callable 1 Non-secure Callable.
31:2	Reserved	Read-only	30	0x00000000	Reserved

Table 21 NSCCFG register

SECMPCINTSTATUS

External interrupt signals, and the interrupt signals from all *Memory Protection Controllers* (MPCs) within the kit that drive the S_MPCEXP_STATUS[n] pins are merged and sent to the processor NVIC on a single Interrupt signal.

The Secure MPC Interrupt Status Register provides Secure software with the ability to check which one of the MPCs is causing the interrupt. When the source of the interrupt is identified, the interrupt must be cleared using the register interface of the source MPC.

Bits	Name	Access	Width	Reset value	Description
0	S_MPCSRAM0_STATUS	Read-only	1	0	Interrupt status for Memory Protection Controller of SRAM BANK 0
15:1	Reserved	Read-only	15	0	Reserved
31:16	S_MPCEXP_STATUS	Read-only	16	0x0000	Interrupt status for Expansion Memory Protection Controller. Each bit 'n' shows the status of input signal S_MPCEXP_STATUS[n]

Table 22 SECMPCINTSTATUS register

SECPPCINTSTAT, SECPPCINTCLR, and SECPPCINTEN

When access violations occur on any PPC within or external to the kit (communicated using the S_APBPPCEXP_STATUS[3:0] and S_AHBPPCEXP_STATUS[3:0] inputs), a level interrupt is raised using a combined interrupt to the processor NVIC.

The PPC Secure PPC Interrupt Status, Clear, and Enable Registers allow software to determine source of the interrupt, clear the interrupt, and enable or disable (Mask) the interrupt.

Bits	Name	Access	Width	Reset value	Description
0	**S_APBPPC0PERIP_STATUS**	Read-only	1	0	Interrupt Status of Peripheral Protection Controller for APB slaves within the Base element
1	**S_APBPPC1PERIP_STATUS**	Read-only	1	0	Interrupt Status of Peripheral Protection Controller for APB slaves within the System Control element
3:2	Reserved	Read-only	2	0x0	Reserved
7:4	**S_APBPPCEXP_STATUS**	Read-only	4	0x00	Interrupt Status of expansion Peripheral Protection Controller for APB slaves. Each bit 'n' shows the status of input signal **S_APBPPCEXP_STATUS[n]** that connects to an external APB PPC
19:8	Reserved	Read-only	12	0x000	Reserved
23:20	**S_AHBPPCEXP_STATUS**	Read-only	4	0x00	Interrupt Status of expansion Peripheral Protection Controller for AHB slaves. Each bit 'n' shows the status of input signal **S_AHBPPCEXP_STATUS[n]** that connects to an external APB PPC
31:24	Reserved	Read-only	8	0x00	Reserved

Table 23 SECPPCINTSTAT register

Bits	Name	Access	Width	Reset value	Description
0	**S_APBPPC0PERIP_CLR**	Write-only	1	0	Interrupt Clear of Peripheral Protection Controller for APB slaves within the Base element. Write '1' to clear.
1	**S_APBPPC1PERIP_CLR**	Write-only	1	0	Interrupt Clear of Peripheral Protection Controller for APB slaves within the System Control element. Write '1' to clear.
3:2	Reserved	Read-only	2	0x0	Reserved
7:4	**S_APBPPCEXP_CLR**	Write-only	4	0x00	Interrupt Clear of expansion Peripheral Protection Controller for APB slaves. Each bit 'n' clears the interrupt source of an external APB PPC by driving the output signal **S_APBPPCEXP_CLEAR[n]** being merged into the single PPC interrupt.
19:8	Reserved	Read-only	12	0x000	Reserved
23:20	**S_AHBPPCEXP_CLR**	Write-only	4	0x00	Interrupt Clear of expansion Peripheral Protection Controller for AHB slaves. Each bit 'n' clears the interrupt source of an external APB PPC by driving the output signal **S_AHBPPCEXP_CLEAR[n]** being merged into the single PPC interrupt.
31:24	Reserved	Read-only	8	0x00	Reserved

Table 24 SECPPCINTCLR register

Bits	Name	Access	Width	Reset value	Description
0	**S_APBPPC0PERIP_EN**	Read/write	1	0	Interrupt Enable of Peripheral Protection Controller for APB slaves within the Base element. Write '1' to enable and '0' to mask this interrupt source.

1	**S_APBPPC1PERIP_EN**	Read/write	1	0	Interrupt Enable of Peripheral Protection Controller for APB slaves within the System Control element. Write '1' to enable and '0' to mask this interrupt source.
3:2	Reserved	Read-only	2	0x0	Reserved
7:4	**S_APBPPCEXP_EN**	Write-only	4	0x00	Interrupt Enable of expansion Peripheral Protection Controller for APB slaves. Each bit 'n' Enables or disable an interrupt from **S_APBPPCEXP_STATUS[n]**
19:8	Reserved	Read-only	12	0x000	Reserved
23:20	**S_AHBPPCEXP_EN**	Write-only	4	0x00	Interrupt Enable of expansion Peripheral Protection Controller for AHB slaves. Each bit 'n' Enables or disable an interrupt from **S_AHBPPCEXP_STATUS[n]**
31:24	Reserved	Read-only	8	0x00	Reserved

Table 25 SECPPCINTEN register

SECMSCINTSTAT, SECMSCINTCLR, and SECMSCINTEN

When a security violation occurs at any MSC in the kit or external to the kit communicated using **S_MSCEXP_STATUS[n]** inputs, an interrupt is raised using a combined interrupt to the processor NVIC.

The Secure MSC Interrupt Status Clear Register and Enable Register enables software to determine source of the interrupt, clear the interrupt, and enable or disable (mask) the interrupt.

Bits	Name	Access	Width	Reset value	Description
15:0	Reserved	Read-only	16	0x0000	Reserved
31:16	**S_MSCEXP_STATUS**	Read-only	16	0x0000	Interrupt status for expansion MSC. Each bit 'n' shows the status of input signal **S_MSCEXP_STATUS[n]**

Table 26 SECMSCINTSTAT register

Bits	Name	Access	Width	Reset value	Description
15:0	Reserved	Read-only	16	0x0000	Reserved
31:16	**S_MSCEXP_CLR**	Write-only	16	0x0000	Interrupt clear for expansion MSC. Each bit 'n' drives the output signal **S_MSCEXP_CLEAR[n]**

Table 27 SECMSCINTCLR register

Bits	Name	Access	Width	Reset value	Description
15:0	Reserved	Read-only	16	0x0000	Reserved
31:16	**S_MSCEXP_EN**	Read/write	16	0x0000	Interrupt enable for expansion MSC. Each bit 'n' enables or disables the use of the input interrupt signal **S_MSCEXP_STATUS[n]**

Table 28 SECMSCINTEN register

BRGINTSTAT, BRGINTCLR, and BRGINTEN

The IoT can contain AHB bus bridges. These bridges can be necessary to handle clock and power domain crossing.

To improve system performance, some of these bridges can buffer write data, and complete a write access on its slave interface before any potential error response is received for the associated write access issued on its master interface.

When this occurs, these bridges can raise a combined interrupt to the processor NVIC to inform that such a failure has occurred.

The Bridge Buffer Error Interrupt Status, Clear and Enable Register allow software to determine source of the interrupt, clear the interrupt, and enable or disable (mask) the interrupt.

Bits	Name	Access	Width	Reset value	Description
15:0	Reserved	Read-only	16	0x0000	Reserved
31:16	BRGEXP_STATUS	Read-only	16	0x0000	Interrupt clear of expansion bridge buffer error interrupts. Each bit 'n' shows the status of BRGEXP_STATUS[n]

Table 29 BRGINTSTAT register

Bits	Name	Access	Width	Reset value	Description
15:0	Reserved	Read-only	16	0x0000	Reserved
31:16	BRGEXP_CLR	Write-only	16	0x0000	Interrupt status of expansion bridge buffer error interrupts. Each bit 'n' shows the status of BRGEXP_CLEAR[n]

Table 30 BRGINTCLR register

Bits	Name	Access	Width	Reset value	Description
15:0	Reserved	Read-only	16	`0x0000`	Reserved
31:16	BRGEXP_EN	Read/write	16	`0x0000`	Interrupt enable of expansion bridge buffer error interrupts. Each bit 'n' enables the use of the input interrupt BRGEXP_STATUS[n]

Table 31 BRGINTEN register

AHBNSPPC0

The Non-secure access AHB Slave Peripheral Protection Controller Register allows software to configure if each AHB peripheral that it controls using an AHB PPC is Secure access only, or is Non-secure access only. Each field defines the Secure or Non-secure access setting for an associated peripheral, as follows:

1 Allow Non-secure access only

0 Allow Secure access only

There is no AHB PPC within the IoT Kit subsystem. This register is empty and is reserved for future use.

Bits	Name	Access	Width	Reset value	Description
31:0	Reserved	Read-only	32	`0x0000_0000`	Reserved

Table 32 AHBNSPPC0 register

AHBNSPPCEXP0, AHBNSPPCEXP1, AHBNSPPCEXP2, and AHBNSPPCEXP3

The expansion Non-secure Access AHB Slave Peripheral Protection Controller Registers, 0, 1, 2 and 3 allow software to configure each AHB peripheral that they control using AHB PPCs that reside in the expansion subsystem outside the IoT Kit subsystem.

Up to 4 external AHB PPCs are supported, with each AHBNSPPCEXP<N> associated with an external AHB PPC <N>.

Each bit of each register then defines the Secure or Non-secure access setting for an associated peripheral that is controlled using the associated AHB PPC, as follows:

0 Allow Non-secure access only.

1 Allow Secure access only.

These controls directly drive the expansion signals on the Security control expansion interface. All four registers are similar and each register, N where N is from 0 to 3, is as follows:

Bits	Name	Access	Width	Reset value	Description
15:0	AHBNSPPCEXPN	Read/write	16	`0x0000`	Expansion N Non-secure access AHB slave Peripheral Protection Control. Each bit 'n' drives the output signal **AHB_NS_PPCEXPN[n]**
31:16	Reserved	Read-only	16	`0x0000`	Reserved

Table 33 AHBNSPPCEXP*N* register

APBNSPPC0 and APBNSPPC1

A Non-secure Access APB slave Peripheral Protection Control Register allows software to configure if each APB peripheral that it controls using an APB PPC is Secure access only or Non-secure access only.

Each field defines the Secure or Non-secure access setting for an associated peripheral, as follows:

1 Allow Non-secure access only. Secure access is not allowed

0 Allow Secure access only. Non-secure access is not allowed

The APBNSPPC0 register controls peripherals that are in the Base element, while the APBNSPPC1 register controls peripherals that are in the System control element.

Bits	Name	Access	Width	Reset value	Description
0	NS_TIMER0	Read/write	1	`0`	APB access security setting for Timer 0
1	NS_TIMER1	Read/write	1	`0`	APB access security setting for Timer 1
2	NS_DTIMER	Read/write	1	`0`	APB access security setting for Dual Timer
31:3	Reserved	Read-only	29	`0x000_0000`	Reserved

Table 34 APBSPPPC0 register

Bits	Name	Access	Width	Reset value	Description
0	NS_S32K Timer	Read/write	1	`0x00`	S32K Timer
31:1	Reserved	Read-only	31	`0x0000_0000`	Reserved

Table 35 APBSPPPC1 register

APBNSPPCEXP0, APBNSPPCEXP1, APBNSPPCEXP2, and APBNSPPCEXP3

The Expansion Non-secure Access APB Slave Peripheral Protection Controller Register (0, 1, 2 or 3) allows software to configure each APB peripheral that it controls using an APB PPC in the expansion subsystem outside the IoT Kit subsystem.

Up to four external APB PPCs are supported, with **APBNSPPCEXP<N>** associated with an external APB PPC <N>.

Each bit of each register defines the Secure or Non-secure access setting for an associated peripheral that is controlled using the associated APB PPC, as follows:

1 Allow Non-secure access only

0 Allow Secure access only

These controls directly drive the expansion signals on the Security control expansion interface. All four registers are similar and each register N, where N is from 0 to 3, is as follows:

Bits	Name	Access	Width	Reset value	Description
15:0	APBNSPPCEXPN	Read/write	16	`0x0000`	Expansion N Non-secure access APB slave Peripheral Protection Control. Each bit 'n' drives the output signal **APB_NS_PPCEXPN[n]**
31:16	Reserved	Read-only	16	`0x0000`	Reserved

Table 36 APBNSPPCEXPN register

AHBSPPPC0

The Secure Unprivileged Access AHB Slave Peripheral Protection Controller Register allows software to configure if each AHB peripheral that it controls using an AHB PPC is only Secure privileged Secure access only, or is also allowed to be Secure unprivileged access.

Each field defines this for an associated peripheral, with the following settings:

1 Allow Secure Unprivileged and privileged access.

0 Allow Secure privileged access only.

There is no AHB Peripheral Protection Controller in the IoT Kit. This register is empty and is reserved for future use.

Bits	Name	Access	Width	Reset value	Description
31:0	Reserved	Read-only	32	`0x0000_0000`	Reserved

Table 37 AHBSPPPC0 register

AHBSPPPCEXP0, AHBSPPPCEXP1, AHBSPPPCEXP2, and AHBSPPPCEXP3

The Expansion Secure Privilege Access AHB Slave Peripheral Protection Controller Register 0, 1, 2 and 3 allows software to configure each AHB peripheral in the Kit, that it controls using an AHB PPC in the expansion subsystem outside the IoT Kit subsystem, is only Secure privileged Secure access only or is also allowed to be Secure unprivileged access.

Each field defines this for an associated peripheral, with the following settings:

1 Allow Secure unprivileged and privileged access.

0 Allow Secure privileged access only.

These controls directly control the expansion signals on the Security control expansion interface. All four registers are similar and each register N, where N is from 0 to 3, is as follows:

Bits	Name	Access	Width	Reset value	Description
15:0	AHBSPPPCEXPN	Read/write	16	`0x0000`	Expansion N Secure privilege access AHB slave Peripheral Protection Control. Each bit 'n' drives the output signal **AHB_SP_PPCEXP<N>[n]** where N is 0-3.
31:16	Reserved	Read-only	16	`0x0000`	Reserved

Table 38 AHBSPPPCEXPN Register

APBSPPPC0 and APBSPPPC1

The Secure Unprivileged Access APB Slave Peripheral Protection Controller Register allows software to configure if each APB peripheral in the kit that it controls using an AHB PPC is only Secure privileged Secure access only or is allowed to be Secure unprivileged access as well.

Each field defines this for an associated peripheral, with the following settings:

1 Allow Secure unprivileged and privileged access.

0 Allow Secure privileged access only.

Bits	Name	Access	Width	Reset value	Description
0	SP_TIMER0	Read/write	1	`0x00`	APB access secure privileged setting for Timer 0
1	SP_TIMER1	Read/write	1	`0x00`	APB access secure privileged setting for Timer 1
2	SP_DTIMER	Read/write	1	`0x00`	APB access secure

					privileged setting for Dual Timer
31:3	Reserved	Read-only	29	`0x000_0000`	Reserved

Table 39 APBSPPPC0 Register

Bits	Name	Access	Width	Reset value	Description
0	SP_S32KTIMER	Read/write	1	`0x00`	APB access secure privileged setting for S32KCLK Timer
31:1	Reserved	Read-only	31	`0x0000_0000`	Reserved

Table 40 APBSPPPC1 register

APBSPPPCEXP0, APBSPPPCEXP1, APBSPPPCEXP2, and APBSPPPCEXP3

The Expansion Secure Privilege Access APB Slave Peripheral Protection Controller Registers, (0, 1, 2 or 3), allows software to configure each APB peripheral that it controls using an APB PPC, in the expansion subsystem outside the IoT Kit subsystem, to be only Secure privileged Secure access only, or to be Secure Unprivileged access as well.

Up to four external AHB PPCs are supported, with AHBSPPPCEXP<N> associated with an external AHB PPC <N>.

Each bit of each register defines this for an associated peripheral, with the following settings:

1 Allow Secure Unprivileged and privileged access.

0 Allow Secure privileged access only.

These controls directly drive the expansion signals on the Security control expansion interface. All four registers are similar and each register N, where N is from 0-3, is as follows:

Bits	Name	Access	Width	Reset value	Description
15:0	APBSPPPCEXPN	Read/write	16	`0x0000`	Expansion N Secure privilege access APB slave Peripheral Protection Control. Each bit 'n' drives the output signal **APB_SP_PPCEXP<N>[n]** where N is 0-3.
31:16	Reserved	Read-only	16	`0x0000`	Reserved

Table 41 APBSPPPCEXPN register

NSMSCEXP

The Non-secure Expansion Master Security Controller Register allows software to configure whether the master located behind each MSC in the expansion subsystem is a Secure or Non-secure device.

Bits	Name	Access	Width	Reset value	Description
15:0	Reserved	Read-only	16	`0x0000`	Reserved
31:16	NS_MSCEXP	Read/write	16	`0x0000`	Expansion MSC Non-secure configuration. Each bit 'n' controls the Non-secure configuration of each MSC and drives the signals NS_MSCEXP[n]. Set to HIGH to define a Master as Non-secure. Otherwise it is Secure.

Table 42 NSMSCEXP Register

Non-secure privilege control block

The Non-secure privilege control block is part of the security controller. It implements program visible states that allow software to control various security gating units within the design. This register block base address is `0x4008_0000`. This register is Non-secure Privileged access only and supports 32-bit read/write accesses.

For write access to these registers, only 32-bit writes are supported. Any byte or halfword write results in its write data being ignored. The following table lists the registers within this unit. Details of each register are described in the following subsections.

Offset	Name	Access	Reset value	Description
0c000-0x06C	Reserved			Reserved
0x090	AHBNSPPPC0	Read/write	0x0000_0000	Non-secure Unprivileged access AHB slave Peripheral Protection Control #0. Each field defines the Non-secure Unprivileged access settings for an associated AHB slave peripheral.
0x094-0x09C	Reserved for AHBNSPPPC0 to AHBNSPPPC3	Read-only	0x0000_0000	Reserved for Future Secure Unprivileged access AHB slave Peripheral Protection Control
0x0A0	AHBNSPPPCEXP0	Read/write	0x0000_0000	Expansion 0 Non-secure Unprivileged access AHB slave Peripheral Protection Control. Each field defines the Non-secure Unprivileged access settings for an associated peripheral: 1 Allow Non-secure Unprivileged access 0 Disallow Non-secure Unprivileged access Resets to 0
0x0A4	AHBNSPPPCEXP1	Read/write	0x0000_0000	Expansion 1 Non-secure Unprivileged access AHB slave Peripheral Protection Control. Each field defines the Non-secure Unprivileged access settings for an associated peripheral:

				1 Allow Non-secure Unprivileged access 0 Disallow Non-secure Unprivileged access Resets to 0
0x0A8	AHBNSPPPCEXP2	Read/write	0x0000_0000	Expansion 2 Non-secure Unprivileged access AHB slave Peripheral Protection Control. Each field defines the Non-secure Unprivileged access settings for an associated peripheral: 1 Allow Non-secure Unprivileged access 0 Disallow Non-secure Unprivileged access Resets to 0
0x0AC	AHBNSPPPCEXP3	Read/write	0x0000_0000	Expansion 3 Non-secure Unprivileged access AHB slave Peripheral Protection Control. Each field defines the Non-secure Unprivileged access settings for an associated peripheral: 1 Allow Non-secure Unprivileged access 0 Disallow Non-secure Unprivileged access Resets to 0
0x0B0	APBNSPPPC0	Read/write	0x0000_0000	Non-secure Unprivileged access APB slave Peripheral Protection Control #0. Each field defines the Non-secure Unprivileged access settings for an associated peripheral: 1 Allow Non-secure Unprivileged access 0 Disallow Non-secure Unprivileged access Resets to 0
0x0B4	APBNSPPPC1	Read/write	0x0000_0000	"Non-secure Unprivileged access APB slave Peripheral Protection Control #1. Each field defines the Non-secure Unprivileged access settings for an associated peripheral: 1 Allow Non-secure Unprivileged access 0 Disallow Non-secure Unprivileged access Resets to 0
0x0B8-0x0BC	Reserved for APBNSPPPC2 to APBNSPPPC3		0x0000_0000	Reserved for Future Non-secure Unprivileged access APB slave Peripheral Protection Control
0x0C0	APBNSPPPCEXP0	Read/write	0x0000_0000	Expansion 0 Non-secure Unprivileged access APB slave Peripheral Protection Control. Each field defines the Non-secure Unprivileged access settings for an associated peripheral: 1 Allow Non-secure Unprivileged access 0 Disallow Non-secure Unprivileged access Resets to 0
0x0C4	APBNSPPPCEXP1	Read/write	0x0000_0000	Expansion 1 Non-secure Unprivileged access APB slave Peripheral Protection Control. Each field defines the Non-secure Unprivileged access settings for an associated peripheral:

				1 Allow Non-secure Unprivileged access 0 Disallow Non-secure Unprivileged access Resets to 0
0x0C8	APBNSPPPCEXP2	Read/write	0x0000_0000	Expansion 2 Non-secure Unprivileged access APB slave Peripheral Protection Control. Each field defines the Non-secure Unprivileged access settings for an associated peripheral: 1 Allow Non-secure Unprivileged access 0 Disallow Non-secure Unprivileged access Resets to 0
0x0CC	APBNSPPPCEXP3	Read/write	0x0000_0000	Expansion 3 Non-secure Unprivileged access APB slave Peripheral Protection Control. Each field defines the Non-secure Unprivileged access settings for an associated peripheral: 1 Allow Non-secure Unprivileged access 0 Disallow Non-secure Unprivileged access Resets to 0
0x0D0 – 0xFCC	Reserved		0x0000_0000	Reserved
0xFD0	PIDR4	Read-only	0x0000_0004	Peripheral ID 4
0xFD4	PIDR5	Read-only	0x0000_0000	Reserved
0xFD8	PIDR6	Read-only	0x0000_0000	Reserved
0xFDC	PIDR7	Read-only	0x0000_0000	Reserved
0xFE0	PIDR0	Read-only	0x0000_0053	Peripheral ID 0
0xFE4	PIDR1	Read-only	0x0000_00B8	Peripheral ID 1
0xFE8	PIDR2	Read-only	0x0000_000B	Peripheral ID 2
0xFEC	PIDR3	Read-only	0x0000_0000	Peripheral ID 3
0xFF0	CIDR0	Read-only	0x0000_000D	Component ID 0
0xFF4	CIDR1	Read-only	0x0000_00F0	Component ID 1
0xFF8	CIDR2	Read-only	0x0000_0005	Component ID 2
0xFFC	CIDR3	Read-only	0x0000_00B1	Component ID 3

Table 43 Non-secure privilege control register map

AHBNSPPPC0

The Non-secure Unprivileged Access AHB Slave Peripheral Protection Controller Register enables software to configure whether each AHB peripheral that it controls using an AHB PPC is only Non-secure privileged access only, or is allowed Non-secure Unprivileged access as well.

Each field defines this for an associated peripheral, with the following settings:

1 Allow Non-secure Unprivileged and privileged access.

0 Allow Non-secure privileged access only.

There is no AHB PPC in the IoT Kit and therefore this register is empty and is reserved for future use.

Bits	Name	Access	Width	Reset value	Description
31:0	Reserved	Read-only	32	`0x0000_0000`	Reserved

Table 44 AHBNSPPPC0 register

APBNSPPPC0 and APBNSPPPC1

The Non-secure Unprivileged access APB Slave Peripheral Protection Controller Register allows software to configure whether each APB peripheral that it controls using an AHB PPC is only Non-secure privileged access only, or is also allowed to be Non-secure Unprivileged access.

Each field defines this for an associated peripheral, with the following settings:

1 Allow Non-secure unprivileged and privileged access.

0 Allow Non-secure privileged access only.

Bits	Name	Access	Width	Reset value	Description
0	NSP_TIMER0	Read/write	1	`0x00`	APB access Non-secure privileged setting for Timer 0
1	NSP_TIMER1	Read/write	1	`0x00`	APB access Non-secure privileged setting for Timer 1
2	NSP_DTIMER	Read/write	1	`0x00`	APB access Non-secure privileged setting for Dual Timer
31:3	Reserved	Read-only	29	`0x000_0000`	Reserved

Table 45 APBNSPPPC0 register

Bits	Name	Access	Width	Reset value	Description
0	NSP_S32KTIMER	Read/write	1	`0x00`	APB access Non-secure privileged setting for **S32KCLK** Timer
31:1	Reserved	Read-only	31	`0x0_0000`	Reserved

Table 46 APBNSPPPC1 register

APBNSPPPCEXP0, APBNSPPPCEXP1, APBNSPPPCEXP2 and APBNSPPPCEXP3

The Expansion Non-Secure Privilege Access APB Slave Peripheral Protection Controller Register (0, 1, 2 or 3) allows software to configure each APB peripheral that it controls using the APB PPC in the expansion subsystem outside the subsystem. The peripheral can be Non-secure privileged access only or it can also be allowed to be Non-secure Unprivileged access.

Each field defines this for an associated peripheral, with the following settings:

1 Allow Non-Secure Unprivileged and privileged access.

0 Allow Non-Secure privileged access only.

These controls directly drive the expansion signals on the security control expansion interface.

All four registers are similar and each register N, where N is from 0 to 3, is as follows:

Bits	Name	Access	Width	Reset value	Description
15:0	APBNSPPPCEXP<N>	read-write	16	0x0000	Expansion N Non-Secure Privilege Access APB slave Peripheral Protection Control. Each bit 'n' will drive the output signal **APBPPPCEXP<N>[n]** if **APBNSPPCEXP<N>[n]** is HIGH, where N is 0 to 3. The parameter APBPPCEXP_DIS<N> defines if each bit within this register is actually implement such that if APBPPCEXP_DIS<N>[i] = `0b1` then APBNSPPPCEXP<N>[i] is disabled, reads as zeros and any writes to it is ignored.
31:16	Reserved	read-only	16	0x0000	Reserved

Table 47 APBSPPPCEXPN register

AHBNSPPPCEXP0, AHBNSPPPCEXP1, AHBNSPPPCEXP2 and AHBNSPPPCEXP3

The Expansion Non-Secure Privilege Access AHB Slave Peripheral Protection Controller Register (0, 1, 2 or 3) allows software to configure each AHB peripheral that it controls using the AHB PPC in the expansion subsystem outside the Subsystem. The peripheral can be Non-secure privileged Secure access only or it can also be allowed to be Non-secure unprivileged access.

Each field defines this for an associated peripheral, with the following settings:

1 Allow non-secure unprivileged and privileged access.

0 Allow non-secure privileged access only.

These directly control the expansion signals on the Security control expansion interface.

All four registers are similar and each register N, where N is from 0 to 3, is as follows:

Bits	Name	Access	Width	Reset value	Description
15:0	AHBNSPPPCEXP<N>	read-write	16	0x0000	Expansion N Non-Secure Privilege Access AHB slave Peripheral Protection Control. Each bit 'n' will drive the output signal

					AHBPPPCEXP<N>[n] if **AHBNSPPCEXP<N>[n]** is HIGH, where N is 0 to 3. The parameter AHBPPCEXP_DIS<N> defines if each bit within this register is actually implement such that if AHBPPCEXP_DIS<N>[i] = `0b1` then AHBNSPPPCEXP<N>[i] is disabled, reads as zeros and any writes to it is ignored.
31:16	Reserved	read-only	16	0x0000	Reserved

Table 48 AHBNSPPPCEXPN register

SRAM Memory Protection Controller

An MPC is included on the path to each SRAM block so that accesses can be blocked when a security violation occurs. Each SRAM block is implemented within an SRAM element. Each MPC APB configuration interface is mapped to the following base address:

- `0x5008_3000` for SRAM Bank 0.

All SRAM MPCs are identical and have the following configuration settings.

Parameter	Configuration	Description
ADDR_WIDTH	15	Address Width. 15 bits for 32Kbyte of SRAM
BLK_SIZE	8	Block size: (1 << BLK_SIZE) bytes. Set at 8 for 256byte block size.

Table 49 SRAM MPC configuration settings

At boot, the SRAM is Secure access only. Software must change or restore the settings in the MPC to release memory for Non-secure world use.

APB Peripheral Protection Controller

The Base element contains a single APB PPC. Control of this PPC resides in the Secure privilege control block and Non-secure privilege control block.

3.6 SRAM element

A single SRAM element is implemented in the IoT Kit subsystem. This element implements the following:

- A 32Kbyte SRAM.
- An SRAM Interface.
- An *Exclusive Access Monitor* (EAM).

Note

Each SRAM element has an MPC that is associated with it and is implemented within the Base element.

Exclusive Access Monitor

An *Exclusive Access Monitor* (EAM) is included on the path to the SRAM block. The EAM has the following configuration:

Parameter	Configuration	Description
ID_PRESENT	SYS_ID_PRESENT[31:0]	Each bit n of this vector defines if a Master associated with IF HMASTERID = n exist in the system. The configuration is defined by the top-level system parameter SYS_ID_PRESENT.

Table 50 SRAM EAM configuration settings

Note

This parameter is not accessible to software or the FVP.

The EAM performs the task of monitoring and handling exclusive accesses that are supported on AHB5. It does not have its own software accessible programming interface.

3.7 System control element

The System control element implements the following:

- System information block.
- System Control Register block.
- Secure debug authentication control block.
- Watchdog.

  This CMSDK Watchdog timer runs on the **S32KCLK**.
- Timer.

  This CMSDK timer runs on the **S32KCLK**.

System information block

The System Information Register block provides information on the system configuration and identity. This register block is read-only and can be accessed by any security attributes. This module resides at base address `0x5002_0000` in the Secure area, and `0x4002_0000` in the Non-secure area of the Base peripheral region.

Offset	Name	Access	Reset value	Description	Security[11]
0x000	SYS_VERSION	Read-only	`0x0004_1743`	System Version register	All[12]
0x004	SYS_CONFIG	Read-only	`0x0000_0031`	System Hardware Configuration register	All
0x010 – 0xFCC	Reserved				
0xFD0	PIDR4	Read-only	`0x0000_0004`	Peripheral ID 4	All
0xFD4	PIDR5	Read-only	`0x0000_0000`	Reserved	
0xFD8	PIDR6	Read-only	`0x0000_0000`	Reserved	
0xFDC	PIDR7	Read-only	`0x0000_0000`	Reserved	
0xFE0	PIDR0	Read-only	`0x0000_0058`	Peripheral ID 0	All
0xFE4	PIDR1	Read-only	`0x0000_00B8`	Peripheral ID 1	All
0xFE8	PIDR2	Read-only	`0x0000_000B`	Peripheral ID 2	All

[11] This column does not define Privileged or Unprivileged accessibility. These are defined by the MPC, PPC, or by the register blocks that are mapped to each area. See the lower level details of each area for details. S – Secure Access, NS – Non-secure
[12] All: Allows all types of security access SP: Secure privilege access only.

0xFEC	PIDR3	Read-only	0x0000_0000	Peripheral ID 3	All
0xFF0	CIDR0	Read-only	0x0000_000D	Component ID 0	All
0xFF4	CIDR1	Read-only	0x0000_00F0	Component ID 1	All
0xFF8	CIDR2	Read-only	0x0000_0005	Component ID 2	All
0xFFC	CIDR3	Read-only	0x0000_00B1	Component ID 3	All

Table 51 System information block register map

SYS_VERSION

The System Version Register provides an area where software can determine the system part number and revision.

Bits	Name	Access	Width	Reset value	Description
11:0	PART_NUMBER	Read-only	12	0x743	Part Number for the subsystem Product
19:12	DESIGNER_ID	Read-only	8	0x41	ARM Product with designer code 0x41
23:20	MINOR_REVISION	Read-only	4	0x0	Set to 0x0
27:24	MAJOR_REVISION	Read-only	4	0x0	Set to 0x0
31:28	CONFIGURATION	Read-only	4	0x0	Set as 0x0 for IoT Kit

Table 52 SYS_VERSION register

SYS_CONFIG

The System Hardware Configuration Register provides an area where software can determine the configuration of the system.

Bits	Name	Access	Width	Reset value	Description
3:0	SRAM_NUM_BANK	Read-only	4	0x1	SRAM Number of Banks.
7:4	SRAM_BANK_SIZE	Read-only	4	0x3	SRAM Bank Sizes. 0 = 4Kbytes, 1 = 8Kbytes 2 = 16Kbytes 3 = 32Kbytes 4 = 64Kbytes 5 = 128Kbytes 6= 256kbytes

					Others are reserved.
8	CPU0_HAS_TCM	Read-only	1	0	CPU 0 has Data TCM. 0 = No 1 = Yes
9	CPU1_HAS_TCM	Read-only	1	0	CPU 1 has Data TCM 0 = No 1 = Yes
10	HAS_CORDIO	Read-only	1	0	Cordio RF Core Included 0 = No 1 = Yes
11	HAS_CRYPTO	Read-only	1	0	CryptoCell Included. 0 = No 1 = Yes
12	CPU0_TCM_BANK_NUM	Read-only	4	0	The SRAM Bank that maps CPU0 Data TCM
16	CPU1_TCM_BANK_NUM	Read-only	4	0	The SRAM Bank that maps CPU1 Data TCM
31:20	Reserved	Read-only	12	0xXXX	Reserved

Table 53 SYS_CONFIG register

System Control Register block

The System Control Register block implements registers for power, clocks, resets, and other general system control. This module is at base address `0x5002_1000` in the Secure region of the Base peripheral region. The System Control Register blocks are Secure Privilege access only. For write access to these registers, only 32-bit writes are supported. Any byte or halfword writes result in the write data being ignored.

The following table shows the details of this register block.

Offset	Name	Access	Reset value	Description	Security [13]
0x000	SECDBGSTAT	Read-only	`0x0000_0000`	Secure Debug Configuration Status Register	SP

[13] The System Control Register block implements registers for Power, resets and other general system control. This module resides at base address 0x5002_1000 in the Secure region of the Base peripheral region. The System Control Register block is Secure privilege access only. For write access to these registers, only 32 bit writes are supported. Any byte and halfword writes will result in its write detail ignored.

0x004	SECDBGSET	Write-only	0x0000_0000	Secure Debug Configuration Set Register	SP
0x008	SECDBGCLR	Write-only	0x0000_0000	Secure Debug Configuration Clear Register	SP
0x00C -0x100	Reserved	Read/write	0x0000_0000		
0x100	RESET_SYNDROME	Read/write	0x0000_0001	Reset syndrome	SP
0x104	RESET_MASK	Read/write	0x0000_0000	Reset MASK	SP
0x108	SWRESET	Write-only	0x0000_0000	Software Reset	SP
0x10C	GRETREG	Read/write	0x0000_0000	General Purpose Retention Register	SP
0x110	INITSVRTOR0	Read/write	Parameterized	Initial Secure Reset Vector Register For CPU 0	SP
0x114	Reserved				
0x118	CPUWAIT	Read/write	Parameterized	CPU Boot wait control after reset	SP
0x11C	BUSWAIT	Read/write	Parameterized	Bus Access wait control after reset	SP
0x120	WICCTRL	Read/write	0x0000_0000	CPU WIC Request and Acknowledgement	SP
0x124 – 0xFCC	Reserved				
0xFD0	PIDR4	Read-only	0x0000_0004	Peripheral ID 4	SP
0xFD4	PIDR5	Read-only	0x0000_0000	Reserved	
0xFD8	PIDR6	Read-only	0x0000_0000	Reserved	
0xFDC	PIDR7	Read-only	0x0000_0000	Reserved	
0xFE0	PIDR0	Read-only	0x0000_0054	Peripheral ID 0	SP
0xFE4	PIDR1	Read-only	0x0000_00B8	Peripheral ID 1	SP
0xFE8	PIDR2	Read-only	0x0000_000B	Peripheral ID 2	SP
0xFEC	PIDR3	Read-only	0x0000_0000	Peripheral ID 3	SP
0xFF0	CIDR0	Read-only	0x0000_000D	Component ID 0	SP
0xFF4	CIDR1	Read-only	0x0000_00F0	Component ID 1	SP
0xFF8	CIDR2	Read-only	0x0000_0005	Component ID 2	SP

0xFFC	CIDR3	Read-only	`0x0000_00B1`	Component ID 3	SP

Table 54 System Control Register block register map

Secure debug authentication control block

The Secure Debug Configuration Registers are used to select the source value for the combined Secure Debug Authentication Signals, **DBGEN**, **NIDEN**, **SPIDEN**, and **SPNIDEN**. For each signal, a selector is provided to select between an internal register value and the value on the boundary of the IoT Kit subsystem.

Secure software can set or clear each value by setting the associated bit in the **SECDBGSET** register or the **SECDBGCLR** register respectively. Secure software can read the system-wide value by reading the associated **SECDBGSTAT** register bit.

For example, the source of **DBGEN** value that is used in the system is selected by the **DBGEN_SEL** where:

- If **DBGEN_SEL** is LOW, the Input **DBGEN_IN** signal is used to define the system-wide **DBGEN** value.
- If **DBGEN_SEL** is HIGH the internal register value **DBGEN_I** is used to define the system-wide **DBGEN** value.

Write to the **SECDBGSET** register with **DBGEN_I_SET** or **DBGEN_SEL_SET** set to set HIGH to set the **DBGEN_I** and **DBGEN_SEL** value respectively.

Write to the SECDBGCLR register with **DBGEN_I_CLR** or **DBGEN_SEL_CLR** set to set LOW to set the **DBGEN_I** and **DBGEN_SEL** values respectively.

To read the value of **DBGEN**, read the SECDBGSTAT register for the **DBGEN_SEL_STAT** value.

The DGBEN value is also made available to external expansion logic using the **DBGEN** output signal of the IoT Kit subsystem.

These registers are reset by the internal equivalent of power-on reset, **nPORESET_OUT** only.

Bits	Name	Access	Width	Reset value	Description
0	**DBGEN_STATUS**	Read-only	1	**DBGEN_IN**	Active high debug enable value.
1	**DBGEN_SEL_STATUS**	Read-only	1	`0`	Active high debug enable selector value.
2	**NIDEN_STATUS**	Read-only	1	**NIDEN_IN**[14]	Active high non-invasive debug enable value.

[14] **DBGEN_IN**, **NIDEN_IN**, **SPIDEN_IN** and **SPNIDEN_IN** are input signals on the Debug Authentication Interface.

3	**NIDEN_SEL_STATUS**	Read-only	1	`0`	Active high Non-invasive debug enable selector value.
4	**SPIDEN_STATUS**	Read-only	1	**SPIDEN_IN**[10]	Active high Secure privilege invasive debug enable value.
5	**SPIDEN_SEL_STATUS**	Read-only	1	`0x00`	Active high Secure privilege invasive debug enable selector value.
6	**SPNIDEN_STATUS**	Read-only	1	**SPNIDEN_IN**[10]	Active high Secure privilege Non-invasive debug enable value.
7	**SPNIDEN_SEL_STATUS**	Read-only	1	`0x00`	Active high Secure privilege Non-invasive debug enable selector value.
31:8	Reserved	Read-only	24	`0x00000000`	Reserved

Table 55 SECDBGSTAT register

Bits	Name	Access	Width	Reset value	Description
0	DBGEN_I_SET	Write-only	1	`0`	Active high Internal debug enable Set control.
1	DBGEN_SEL_SET	Write-only	1	`0`	Active high debug enable selector Set control.
2	NIDEN_I_SET	Write-only	1	`0`	Active high Internal Non-invasive debug enable Set control.
3	NIDEN_SEL_SET	Write-only	1	`0`	Active high Non-invasive debug enable selector Set control.
4	SPIDEN_I_SET	Write-only	1	`0`	Active high Internal Secure privilege invasive debug enable Set control.
5	SPIDEN_SEL_SET	Write-only	1	`0`	Active high Secure privilege invasive debug enable selector

					Set control.
6	SPNIDEN_I_SET	Write-only	1	`0`	Active high Internal Secure privilege Non-invasive debug enable Set control.
7	SPNIDEN_SEL_SET	Write-only	1	`0`	Active high Secure privilege Non-invasive debug enable selector Set control.
31:8	Reserved	Read-only	24	`0x00000000`	Reserved

Table 56 SECDBGSET register

Bits	Name	Access	Width	Reset value	Description
0	DBGEN_CLR	Write-only	1	`0`	Active high Internal debug enable Clear control.
1	DBGEN_SEL_CLR	Write-only	1	`0`	Active high debug enable selector Clear control.
2	NIDEN_CLR	Write-only	1	`0`	Active high Internal Non-invasive debug enable Clear control.
3	NIDEN_SEL_CLR	Write-only	1	`0`	Active high Non-invasive debug enable selector Clear control.
4	SPIDEN_CLR	Write-only	1	`0`	Active high Internal Secure privilege invasive debug enable Clear control.
5	SPIDEN_SEL_CLR	Write-only	1	`0`	Active high Secure privilege invasive debug enable selector Clear control.
6	SPNIDEN_CLR	Write-only	1	`0`	Active high Internal Secure privilege Non-invasive debug enable Clear control.
7	SPNIDEN_SEL_CLR	Write-only	1	`0`	Active high Secure privilege Non-invasive debug enable selector Clear control.
31:8	Reserved	Read-only	24	`0x00000000`	Reserved

Table 57 SECDBGCLR register

RESET_SYNDROME

This register stores the reason for the last system reset using the **nSYSRESET** signal. These registers are only cleared by **nPORESET** input or by software writing zero to clear it.

Bits	Name	Access	Width	Reset value	Description
0	PoR	Read/write	1	`0x1`	Power-On
1	NSWD	Read/write	1	`0x0`	Non-secure Watchdog
2	SWD	Read/write	1	`0x0`	Secure Watchdog
3	S32KWD	Read/write	1	`0x0`	Watchdog on the **S32KCLK** clock
4	SYSRSTREQ0	Read/write	1	`0x0`	CPU 0 System Reset Request
5	Reserved		1	`0x0`	
6	LOCKUP0	Read/write	1	`0x0`	CPU 0 Lock-up Status
7	Reserved		1	`0x0`	
8	WARMRESETINPUT	Read/write	1	`0x0`	External Warm reset Request
9	SWRESETREQ	Read/write	1	`0x0`	System Warm reset Request
31:10	Reserved	Read-only	22	`0x0`	

Table 58 RESET_SYNDROME register

RESET_MASK

The RESET_MASK register allows software to control which reset sources are to be merged to generate the system-wide **nSYSRESET** or the internal **nPORESET_OUT** signals. Set each bit to HIGH to enable each source. This register is reset by the internal equivalent of **nPORESET_OUT**.

Bits	Name	Access	Width	Reset value	Description
0	Reserved	Read-only	1	`0x0`	Reserved
1	NSWD_EN	Read/write	1	`0x0`	Enable Non-secure Watchdog Reset
3:2	Reserved	Read-only	2	`0x0`	Reserved
4	SYSRSTREQ0_EN	Read/write	1	`0x0`	Enable Merging CPU 0 System Reset Request
5	Reserved	Read/write	1	`0x0`	
6	LOCKUP0_EN	Read/write	1	`0x0`	Enable Merging CPU 0 Lock-up Status
31:7	Reserved	Read-only	24	`0x000_0000`	Reserved

Table 59 RESET_MASK register

SWRESET

The SWRESET register allows software to request a system reset. This register is reset by the internal equivalent of **nPORESET_OUT and nSYSRESET_OUT**.

Bits	Name	Access	Width	Reset value	Description
8:0	Reserved	Read-only	9	`0x00`	Reserved
9	SWRESETREQ	Write-only	1	`0x0`	Software Reset Request. Set to HIGH to request a system reset.
31:10	Reserved	Read-only	22	`0x00_0000`	Reserved

Table 60 SWRESET register

GRETREG

The General Purpose Retention Register provides 16 bits of retention register for generate storage, especially through power down of the reset of the system. This register is reset by the internal equivalent of **nPORESET_OUT** only.

Bits	Name	Access	Width	Reset value	Description
15:0	GRETREG	Read/write	16	`0x0000`	General Purpose Retention Register.
31:16	Reserved	Read-only	16		Reserved

Table 61 GRETREG register

INITSVRTOR0

This register is used to define the Initial Secure Vector table offset (VTOR_S.TBLOFF[31:7]) out of reset. This register is reset by the internal equivalent of **nPORESET_OUT** only.

Bits	Name	Access	Width	Reset value	Description
6:0	Reserved	Read-only	7		Reserved
31:7	INITSVRTOR0	Read/write	25	INITSVTOR0_RST[31:7]	Default Secure Vector table offset at reset.

Table 62 INITSVRTOR0 register

CPUWAIT

This register provides controls to force the core boot to wait after reset. Another processor in the expansion system or the debugger can then allow the processor to start later as required. This register is reset by the internal equivalent of **nPORESET_OUT** only and its reset value is defined by the CPU0WAIT_RST parameter.

Bits	Name	Access	Width	Reset value	Description
0	CPU0WAIT	Read/write	1	CPU0WAIT_RST	CPU 0 waits at boot. 0 boot normally. 1 wait at boot.
31:1	Reserved	Read-only	31		Reserved

Table 63 CPUWAIT register

BUSWAIT

The BUSWAIT register allows software to gate access entering the Base element from specific masters in the system, causing them to stall so that the processor can complete the configuration of the MPCs or other security registers in the system before the stalled accesses begin. This register is reset by the internal equivalent of **nPORESET_OUT** only.

Accesses in the IoT Kit subsystem are not gated because all devices in the system, other than the processor, are expected to be controlled by the main processor, and do not start autonomously, but wait for the system to wake from the OFF state.

Bits	Name	Access	Width	Reset value	Description
31:0	Reserved	Read-only	32	Reserved	

Table 64 BUSWAIT register

WICCTRL

The WIC Control register allows software to perform the WIC enable handshake for each individual core. This register is reset by the internal equivalent of **nPORESET_OUT** only

Bits	Name	Access	Width	Reset value	Description
0	CPU0WICEN	Read/write	1	`0x0000`	CPU 0 WIC Enable Request
15:1	Reserved	Read-only	15		Reserved
16	CPU0WICRDY	Read-only	1	`0x0000`	CPU 0 WIC Enable Acknowledge
31:17	Reserved	Read-only	15		Reserved

Table 65 WICCTRL register

3.8 Interrupt Map

The following table describes the interrupt map of the processor core in the IoT Kit subsystem.

Interrupt Input	Interrupt Source
NMI	Combined Secure Watchdog, S32K Watchdog, and NMI_expansion
IRQ[0]	Non-secure watchdog reset request
IRQ[1]	Non-secure watchdog interrupt
IRQ[2]	S32K Timer
IRQ[3]	Timer 0
IRQ[4]	Timer 1
IRQ[5]	Dual timer
IRQ[8:6]	Reserved
IRQ[9]	MPC combined (Secure)
IRQ[10]	PPC combined (Secure)
IRQ[11]	MSC combined (Secure)
IRQ[12]	Bridge error combined interrupt (Secure)
IRQ[31:13]	Reserved
IRQ[EXP_NUMIRQ+32:32]	Expansion interrupt inputs[15]

Table 66 Interrupt map

Note

The interrupt signal and the reset request signal of the Non-secure watchdog are both used to generate interrupts to the processor. The reset request interrupt must be handled as a Secure interrupt by the *Trusted Execution Environment* (TEE) so that it does not directly reset the system.

If you want to allow the Non-secure watchdog to be able to reset the system, set the NSWD_EN field of the RESET_MASK register to HIGH.

The Secure watchdog interrupt request, along with the S32K watchdog interrupt request are merged to generate the NMI. This interrupt must also be handled as a Secure interrupt.

[15] See Table 80

3.9 Clocking infrastructure

Other than the clock sources entering the system, the current IoT subsystem does not generate any additional clock signal in the system other than the **TRACECLK** output. **SYSCLK** is the same as **MAINCLK**.

All logic in the system is running on **SYSCLK**.

3.10 Reset infrastructure

The input signal **nPORESET** is the power-on reset input for the IoT Kit subsystem. This input resets the Reset Syndrome register directly. The **nPORESET** signal is then combined with the masked reset inputs from Watchdog Timers to generate the internal combined power-on reset signal, which is available as **nPORESET_OUT**. Therefore **nPORESET_OUT** resets all logic except the Reset Syndrome register.

System Reset is performed by the **nSYSRESET** signal. This signal is generated by merging reset requests from the following sources, with some using a mask defined in the RESET_MASK Register in the System Control Register block:

- **nPORESET_OUT**
- SYSTEM Reset Request from the processor.
- Warm reset Request from the external expansion logic on the **WARMRESETINPUT** signal.
- Software Reset Request using the SWRESET register.

nSYSRESET does not reset debug related logic or the Reset Syndrome register. This reset is available for use by the expansion logic using the **nSYSRESET_OUT** output signal.

Both **nPORESET_OUT** and **nSYSRESET_OUT** are both synchronous to the **MAINCLK** clock. Therefore before using the reset signal on any other clock domain, you must resynchronize the reset.

Power control infrastructure

The IoT Kit subsystem does not implement any power control.

4 MPS2+ system expansion

The FPGA expansion subsystem that is presented here describes how the IoT Kit subsystem is extended by integrating more components to form a full FPGA system targeting the MPS2+ FPGA platform. The subsystem cannot be expanded on the FVP.

The proposed FPGA expansion subsystem integrates the following:

- ZBT SRAM controllers provide access to on board ZBT SRAMs, and these function as the main code memory and also as extra data storage memory.
- PL081 DMA controllers, primarily to acts as other masters within the system.
- An SRAM controller, to provide access to external devices with asynchronous SRAM like interfaces. This supports PSRAM and the ethernet in the MPS2+ FPGA platform.
- Expansion security controller. This provides security control for all peripherals that reside outside the IoT Kit subsystem but are within the FPGA subsystem.
- An FPGA System Controller.
- An *Implementation Defined Attribution Unit* (IDAU).
- CMSDK Components including:
    - *General-purpose Input/Output* (GPIOs) used to drive the I/O pins of the FPGA.
    - UARTs.
- CoreLink SIE-200 System IP for Embedded components including:
    - AHB5 *Memory Protection Controller* (MPC).
    - AHB5 *Master Security Controller* (MSC).
    - AHB5 *Peripheral Protection Controller* (PPC).
    - APB *Peripheral Protection Controller* (PPC).
    - AHB5 *Exclusive Access Monitor* (EAM).
    - AHB5 to APB Bridge.
    - AHB5 Fabric.
- Other IO components:
    - PL022 SPI.
    - I2S Controller.
- Color VGA interface.

The following section provides a high-level block diagram of the System, and provides an address map of all key peripherals in the MPS platform. It also provides more details on how the expansion security control signals are deployed.

4.1 MPS2+ FPGA based on IoT Kit subsystem

The following diagram shows the high-level structure of the full MPS2+ FPGA System.

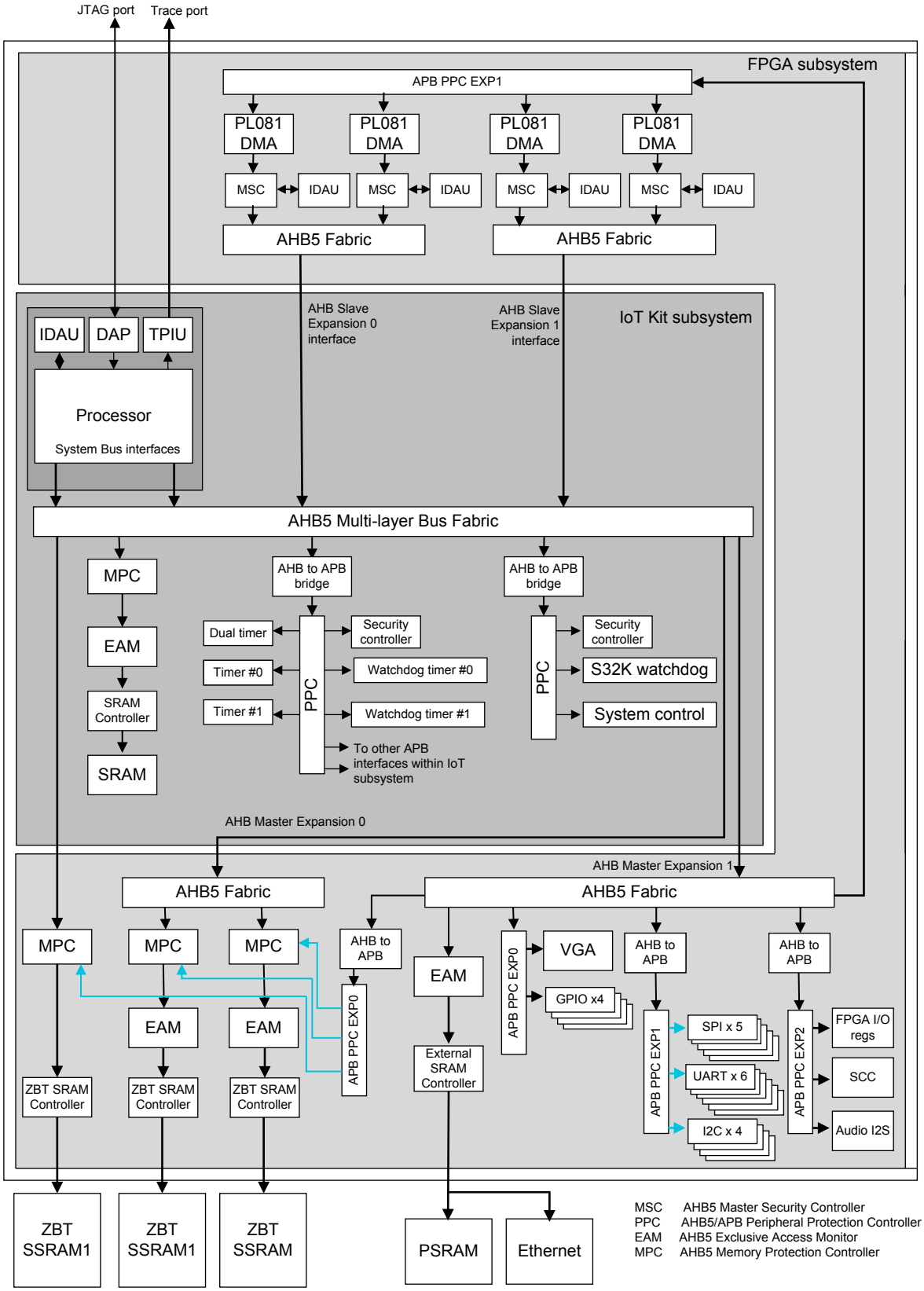

Table 67 Structure of MPS2 FPGA system using the IoT Kit subsystem

Note

The FPGA subsystem extends the IoT Kit subsystem by adding to the expansion interfaces of the IoT subsystem.

4.2 Memory Maps

External ZBT SRAMs synchronous SRAM for code (SSRAM1)

The MPS2+ FPGA prototyping board provides a ZBT SRAM bank for code storage, designated SSRAM1. This SSRAM1 bank has two external 32-bit ZBT SSRAM in parallel, forming a 64-bit ZBT SSRAM, with 8MB of SRAM space in total, but only 4MB is populated. This 8MB memory is mapped to both to the Non-secure and Secure Code Memory region as shown in the following table.

To provide security gating, an MPC is placed in the path to these memories. This MPC is called SSRAM1MPC. Its configuration interface is at `0x5800_7000` and its interrupt signal is connected to S_MPCEXP_STATUS[0]. All unused region in the Code Memory Space must return a bus error response when accessed.

Row ID	Address		Size	Region name	Description	Alias with row ID	IDAU region values		
	From	To					Security	IDAU ID	NSC
1	0x0000_0000	0x007F_FFFF	8MB	Code Memory	ZBT SRAM (SSRAM1)	4	NS	0	0
2	0x0080_0000	0x0DFF_FFFF	116MB		Not used and Return Bus Errors when accessed.	-			
3	0x0E00_0000	0x0FFF_FFFF	32MB	Reserved	Reserved	-			
4	0x1000_0000	0x107F_FFFF	8MB	Code Memory	ZBT SRAM (SSRAM1)	1	S	1	CODE NSC
5	0x1080_0000	0x1DFF_FFFF	116MB		Not used and Return Bus Errors when accessed.	-			
6	0x1E00_0000	0x1FFF_FFFF	32MB	Reserved	Reserved	-			

Table 68 External SSRAM1 mapping to code memory

Because only 4MB out of the 8MB is populated in the SSRAM1 BANK, the top 4MB region is aliased with the lower 4MB region. As a result both share security settings. This ensures that there are no security holes that allow Secure and Non-secure access to the same physical location on the ZBT SSRAM at the same time.

The SSRAM1MPC has the configuration setting as listed below.

DATA_WIDTH	32-bits	Data Width: 32-bits
ADDR_WIDTH	21	Address Width. Set at 22bits to support 4Mbyte of memory space.
MASTER_WIDTH	5	HMASTER signal width. 5 bit for 32 masters

USER_WIDTH	0	User signal width parameter, default: 1, ports tied if 0
BLK_SIZE	8	Block size: (1 << BLK_SIZE) bytes, min. value: 5, max. value: 20. Set at 8 for 256 bytes blocks.
GATE_RESP	0	Response on data AHB when accessed during programming lock 0 Add wait states until lock is released (default) 1 Drive bus error

Table 69 SSRAM1MPC configuration settings

External ZBT SRAMs synchronous SRAM (SSRAM2 and SSRAM3)

The MPS2+ FPGA prototyping board provides another two fast ZBT SRAM banks for general-purpose data and code storage, designated SSRAM2 and SSRAM3. Each of these memories is 4MB in size and has 32-bit wide data interfaces. Both ZBT SRAMs are accessed using the AHB5 Master expansion 0 Interface and are mapped as a contiguous 4MB of memory as shown in the following table. All unused regions that are shown in the table must return bus error response when accessed.

ROW ID	Address		Size	Region Name	Description	Alias With Row ID	IDAU Region values		
	From	To					Security	IDAU ID	NSC
1	0x2000_0000	0x20FF_FFFF	16MB	Internal SRAM	Internal SRAM Area.	6	NS	2	0
2	0x2100_0000	0x27FF_FFFF	112MB	Reserved	Reserved				
3	0x2800_0000	0x281F_FFFF	2MB	Expansion 0	ZBT SRAM (SSRAM2)	8			
4	0x2820_0000	0x283F_FFFF	2MB		ZBT SRAM (SSRAM3)	9			
5	0x2840_0000	0x2FFF_FFFF	124MB		Not used and Return Bus Errors when accessed.				
6	0x3000_0000	0x30FF_FFFF	16MB	Internal SRAM	Internal SRAM Area.	1	S	3	RAMNSC2
7	0x3100_0000	0x37FF_FFFF	112MB	Reserved	Reserved				
8	0x3800_0000	0x381F_FFFF	2MB	Expansion 0	ZBT SRAM (SSRAM2)	3			
9	0x3820_0000	0x383F_FFFF	2MB		ZBT SRAM (SSRAM3)	4			
10	0x3840_0000	0x3FFF_FFFF	124MB		Maps to AHB5 Master expansion 0 Interface				

Table 70 SSRAM2 and SSRAM3 address mapping

To support exclusive access and also support Security gating so that blocks of aliased memory can be assigned individually to Secure or Non-secure regions, an *Exclusive Access Monitor* (EAM) and an MPC are added on the path of each ZBT SRAM.

The MPCs are as follows:

- The name of the MPC to SSRAM2 is SSRAM2MPC. Its APB interface is mapped to address `0x5800_8000`. The interrupt signal is connected to S_MPCEXP_STATUS[1].
- The name of the MPC to SSRAM3 is SSRAM3MPC. Its APB interface is mapped to address `0x5800_9000`. The interrupt signal is connected to S_MPCEXP_STATUS[2].

Both SSRAM1MPC and SSRAM2MPC have the same configuration setting as listed in the following table.

Parameter	Configuration	Description
DATA_WIDTH	32bits	Data Width: 32bits
ADDR_WIDTH	21	Address Width. Set at 21bits to support 2Mbyte of memory space.
MASTER_WIDTH	5	HMASTER signal width. 5 bit for 32 masters
USER_WIDTH	0	User signal width parameter, default: 1, ports tied if 0
BLK_SIZE	8	Block size: (1 << BLK_SIZE) bytes, min. value: 5, max. value: 20. Set at 8 for 256 bytes blocks.
GATE_RESP	0	Response on data AHB when accessed during programming lock: 0 – Add wait states until lock is released (default) 1 – Drive bus error

Table 71 SSRAM2MPC and SSRAM3MPC configuration settings

PSRAM

The MPS2+ FPGA prototyping board provides a 16-bit PSRAM interfaces supporting two banks of parallel SRAMs, each up to 8MB in size, providing a maximum of 16MB of SRAM. These memories are mapped to Non-secure SRAM space as follows:

ROW ID	Address		Size	Region Name	Description	Alias With Row ID	IDAU Region values		
	From	To					Security	IDAU ID	NSC
1	0x8000_0000	0x80FF_FFFF	16MB	AHB Master expansion 1 Interface Area	PSRAM		NS	8	0
2	0x8100_0001	0x8FFF_FFFF	246MB		Not used and Return Bus Errors when accessed.				

Table 72 External PSRAM mapping to code memory

Expansion system peripherals

Other than the SSRAMs, PSRAMs, and the ethernet MAC and PHY, all FPGA peripherals that are extensions to the IoT Kit are mapped into two key areas of the memory map:

- `0x4010_0000` to `0x4FFF_FFFF`.

  Non-secure region which maps to AHB Master Expansion 1 interface.

- `0x5010_0000` to `0x5FFF_FFFF`.

  Secure region which maps to AHB Master Expansion 1 interface.

The following table shows how these peripherals are mapped. Most are components that are available from CMSDK and from currently available Cortex-M FPGA system that targets the MPS2+ board. In addition, extra masters, for example, two PL081 Direct Memory Access (DMA) units are also added into the system to provide support for DMA.

To support the ARMv8-M Security Extension and allow software to map these peripherals to a Secure or Non-secure address space, many peripherals are mapped twice and either an APB PPC or an AHB PPC is used, which allow secure software to define which security domain each peripheral will reside in. An FPGA Secure Privilege Control block and a Non-secure privilege Control block then provide controls to these PPCs.

For expansion AHB Masters within the system, an MSC is added to each master with an associated IDAU which mirrors that already used in the kit. There are four PL081 DMA Engines. Each DMA can be mapped either as a Secure or a Non-secure master by configuring the PPC and the MSC. The intention is to allow one to be mapped as a secure DMA engine with another as Non-secure DMA engines on each expansion slave interface.

Row ID	Address		Size	Description	Alias with row ID	IDAU region values	
	From	To				Security	ID
1	`0x4010_0000`	`0x4010_0FFF`	4KB	GPIO 0	37	NS	4
2	`0x4010_1000`	`0x4010_1FFF`	4KB	GPIO 1	38		
3	`0x4010_2000`	`0x4010_2FFF`	4KB	GPIO 2	39		
4	`0x4010_3000`	`0x4010_3FFF`	4KB	GPIO 3	40		
5	`0x4010_7000`	`0x4010_FFFF`		Not used and return bus errors when accessed.			
6	`0x4011_0000`	`0x4011_0FFF`	4KB	DMA 0	42		
7	`0x4011_1000`	`0x4011_1FFF`	4KB	DMA 1	43		
8	`0x4011_2000`	`0x4011_2FFF`	4KB	DMA 2	44		
9	`0x4011_3000`	`0x4011_3FFF`	4KB	DMA 3	45		
10	`0x4011_4000`	`0x401F_FFFF`		Not used and			

				return bus errors when accessed.	
11	0x4020_0000	0x4020_0FFF	4KB	UART 0	47
12	0x4020_1000	0x4020_1FFF	4K	UART 1	48
13	0x4020_2000	0x4020_2FFF	4KB	UART 2	49
14	0x4020_3000	0x4020_3FFF	4KB	UART 3	50
15	0x4020_4000	0x4020_4FFF	4KB	UART 4	51
16	0x4020_5000	0x4020_5FFF	4KB	FPGA - PL022 (SPI)	52
17	0x4020_6000	0x4020_6FFF	4KB	FPGA - PL022 (SPI for LCD)	53
18	0x4020_7000	0x4020_7FFF	4KB	FPGA - SBCon I2C (Touch)	54
19	0x4020_8000	0x4020_8FFF	4KB	FPGA - SBCon I2C (Audio Configuration)	55
20	0x4020_9000	0x4020_9FFF	4KB	FPGA - PL022 (SPI ADC)	56
21	0x4020_A000	0x4020_AFFF	4KB	FPGA - PL022 (SPI Shield0)	57
22	0x4020_B000	0x4020_BFFF	4KB	FPGA - PL022 (SPI Shield1)	58
23	0x4020_C000	0x4020_CFFF	4KB	SBCon (Shiled0)	59
24	0x4020_D000	0x4020_DFFF	4KB	SBCon (Shiled1)	60
25	0x4020_E000	0x402F_FFFF		Not used and Return Bus Errors when accessed.	
26	0x4030_0000	0x4030_0FFF	4KB	FPGA - SCC registers	62
27	0x4030_1000	0x4030_1FFF	4KB	FPGA - I2S (Audio)	63
28	0x4030_2000	0x4030_2FFF	4KB	FPGA - GPIO (System Ctrl + I/O)	64
29	0x4030_3000	0x40FF_FFFF		Not used and Return Bus Errors when accessed.	

30	0x4100_0000	0x4100_FFFF	64KB	VGA Console	66		
31	0x4110_0000	0x4113_FFFF	256KB	VGA Image	67		
32	0x4114_0000	0x41FF_FFFF		Not used and Return Bus Errors when accessed.			
33	0x4200_0000	0x420F_FFFF	1MB	Ethernet	69		
34	0x4210_0000	0x4800_6FFF		Not used and Return Bus Errors when accessed.			
35	0x4800_7000	0x4800_7FFF	4KB	FPGA Non-secure Privilege Control			
36	0x4800_8000	0x4FFFF_FFFF		Not used and Return Bus Errors when accessed.			
37	0x5010_0000	0x5010_0FFF	4KB	GPIO 0	1	S	5
38	0x5010_1000	0x5010_1FFF	4KB	GPIO 1	2		
39	0x5010_2000	0x5010_2FFF	4KB	GPIO 2	3		
40	0x5010_3000	0x5010_3FFF	4KB	GPIO 3	4		
41	0x5010_7000	0x5010_FFFF		Not used and Return Bus Errors when accessed.			
42	0x5011_0000	0x5011_0FFF	4KB	DMA 0	6		
43	0x5011_1000	0x5011_1FFF	4KB	DMA 1	7		
44	0x5011_2000	0x5010_2FFF	4KB	DMA 2			
44	0x5011_3000	0x5010_3FFF	4KB	DMA 3			
46	0x5011_4000	0x501F_FFFF		Not used and Return Bus Errors when accessed.			
47	0x5020_0000	0x5020_0FFF	4KB	UART 0	11		
48	0x5020_1000	0x5020_1FFF	4KB	UART 1	12		
49	0x5020_2000	0x5020_2FFF	4KB	UART 2	13		
50	0x5020_3000	0x5020_3FFF	4KB	UART 3	14		
51	0x5020_4000	0x5020_4FFF	4KB	UART 4	15		

52	0x5020_5000	0x5020_5FFF	4KB	FPGA - PL022 (SPI)	16
53	0x5020_6000	0x5020_6FFF	4KB	FPGA - PL022 (SPI for LCD)	17
54	0x5020_7000	0x5020_7FFF	4KB	FPGA - SBCon I2C (Touch)	18
55	0x5020_8000	0x5020_8FFF	4KB	FPGA - SBCon I2C (Audio Configuration)	19
56	0x5020_9000	0x5020_9FFF	4KB	FPGA - PL022 (SPI ADC)	20
57	0x5020_A000	0x5020_AFFF	4KB	FPGA - PL022 (SPI Shield0)	21
58	0x5020_B000	0x5020_BFFF	4KB	FPGA - PL022 (SPI Shield1)	22
59	0x5020_C000	0x5020_CFFF	4KB	SBCon (Shiled0)	23
60	0x5020_D000	0x5020_DFFF	4KB	SBCon (Shiled1)	24
61	0x5020_E000	0x502F_FFFF		Not used and Return Bus Errors when accessed.	
62	0x5030_0000	0x5030_0FFF	4KB	FPGA - SCC registers	26
63	0x5030_1000	0x5030_1FFF	4KB	FPGA - I2S (Audio)	27
64	0x5030_2000	0x5030_2FFF	4KB	FPGA - GPIO (System Ctrl + I/O)	28
65	0x5030_3000	0x50FF_FFFF		Not used and Return Bus Errors when accessed.	
66	0x5100_0000	0x5100_FFFF	64KB	VGA Console	30
67	0x5110_0000	0x5113_FFFF	256KB	VGA Image	31
68	0x5114_0000	0x51FF_FFFF		Not used and Return Bus Errors when accessed.	
69	0x5200_0000	0x520F_FFFF	1MB	Ethernet	33
70	0x5210_0000	0x5800_6FFF		Not used and Return Bus	

				Errors when accessed.
71	0x5800_7000	0x5800_7FFF	4KB	SSRAM1 Memory Protection Controller (MPC)
72	0x5800_8000	0x5800_8FFF	4KB	SSRAM2 Memory Protection Controller (MPC)
73	0x5800_9000	0x5800_9FFF	4KB	SSRAM3 Memory Protection Controller (MPC)
74	0x5800_A000	0x5FFF_FFFF		Not used and Return Bus Errors when accessed.

Table 73 FPGA expansion peripheral map

FPGA Secure privilege control

The Secure privilege control and Non-secure privilege blocks of the IoT Kit subsystem provide expansion security control signals to control the various security gating units within the subsystem. These signals are driven by the register in the Secure privilege control block and the Non-Secure privilege control block. The following table lists the signals that each PPC, MSC, and MPC is connected to.

Components name	Components signals	Security expansion signals
DMA 0 MSC	msc_irq	S_MSCEXP_STATUS[0]
	msc_irq_clear	S_MSCEXP_CLEAR[0]
	cfg_nonsec	NS_MSCEXP[0]
DMA 1 MSC	msc_irq	S_MSCEXP_STATUS[1]
	msc_irq_clear	S_MSCEXP_CLEAR[1]
	cfg_nonsec	NS_MSCEXP[1]
DMA 2 MSC	msc_irq	S_MSCEXP_STATUS[2]
	msc_irq_clear	S_MSCEXP_CLEAR[2]
	cfg_nonsec	NS_MSCEXP[2]
DMA 3 MSC	msc_irq	S_MSCEXP_STATUS[3]
	msc_irq_clear	S_MSCEXP_CLEAR[3]
	cfg_nonsec	NS_MSCEXP[3]

	apb_ppc_irq	S_APBPPCEXP_STATUS[0]
	apb_ppc_clear	S_APBPPCEXP_CLEAR[0]
APB PPC EXP 0	cfg_sec_resp	SEC_RESP_CFG
	cfg_non_sec	APB_NS_PPCEXP0[15:0]
	cfg_ap	APB_P_PPCEXP0[15:0]
APB PPC EXP 1	apb_ppc_irq	S_APBPPCEXP_STATUS[1]
	apb_ppc_clear	S_APBPPCEXP_CLEAR[1]
	cfg_sec_resp	SEC_RESP_CFG
	cfg_non_sec	APB_NS_PPCEXP1[15:0]
	cfg_ap	APB_P_PPCEXP1[15:0]
APB PPC EXP 2	apb_ppc_irq	S_APBPPCEXP_STATUS[2]
	apb_ppc_clear	S_APBPPCEXP_CLEAR[2]
	cfg_sec_resp	SEC_RESP_CFG
	cfg_non_sec	APB_NS_PPCEXP2[15:0]
	cfg_ap	APB_P_PPCEXP2[15:0]
AHB PPC EXP 0	ahb_ppc_irq	S_AHBPPCEXP_STATUS[0]
	ahb_ppc_clear	S_AHBPPCEXP_CLEAR[0]
	cfg_sec_resp	SEC_RESP_CFG
	cfg_non_sec	AHB_NS_PPCEXP0[15:0]
	chg_ap	AHB_P_PPCEXP0[15:0]
AHB PPC EXP 1	ahb_ppc_irq	S_AHBPPCEXP_STATUS[1]
	ahb_ppc_clear	S_AHBPPCEXP_CLEAR[1]
	cfg_sec_resp	SEC_RESP_CFG
	cfg_non_sec	AHB_NS_PPCEXP1[15:0]
	chg_ap	AHB_P_PPCEXP1[15:0]
MPC SSRAM0	secure_error_irq	S_MPCEXP_STATUS[0]
MPC SSRAM1	secure_error_irq	S_MPCEXP_STATUS[1]
MPC SSRAM2	secure_error_irq	S_MPCEXP_STATUS[2]

Table 74 Security expansion signals connectivity

The following table lists the peripherals that are controlled by APB PPC EXP 0. Each APB <n> interface is controlled by APB_NS_PPCEXP0[n] and APB_P_PPCEXP0[n].

APB PPC EXP 0 Interface Number <n>	Name
0	SSRAM1 Memory Protection Controller (MPC)
1	SSRAM2 Memory Protection Controller (MPC)
2	SSRAM3 Memory Protection Controller (MPC)
15:3	Reserved

Table 75 Peripherals mapping of APB PPC EXP 0

The following table lists the peripherals that are controlled by APB PPC EXP 1. Each APB <n> interface is controlled by APB_NS_PPCEXP1[n] and APB_P_PPCEXP1[n].

APB PPC EXP 1 Interface Number <n>	Name
0	SPI_0
1	SPI_1
2	SPI_2
3	SPI_3
4	SPI_4
5	UART_0
6	UART_1
7	UART_2
8	UART_3
9	UART_4
10	I2C_0
11	I2C_1
12	I2C_2
13	I2C_3
15:14	Reserved

Table 76 Peripherals mapping of APB PPC EXP 1

The following table lists the peripherals that are controlled by APB PPC EXP 2. Each APB <n> interface is controlled by APB_NS_PPCEXP2[n] and APB_P_PPCEXP2[n].

APB PPC EXP 0 Interface Number <n>	Name
0	SCC
1	AUDIO
2	FPGAIO
15:3	Reserved

Table 77 Peripherals mapping of APB PPC EXP 2

The following table lists the peripherals that are controlled by AHB PPC EXP 0. Each APB <n> interface controlled by the AHB_NS_PPCEXP0[n] and AHB_P_PPCEXP0[n].

AHB PPC EXP 0 Interface Number <n>	Name
0	VGA
1	GPIO_0
2	GPIO_1
3	GPIO_2
4	GPIO_3
15:5	Reserved

Table 78 Peripherals mapping of AHB PPC EXP 0

The following table lists the peripherals that are controlled by AHB PPC EXP 1. Each APB <n> interface is controlled by AHB_NS_PPCEXP1[n] and AHB_P_PPCEXP0[n].

AHB PPC EXP 0 Interface Number <n>	Name
0	DMA0
1	DMA1
2	DMA2
3	DMA3
15:4	Reserved

Table 79 Peripherals mapping of AHB PPC EXP1

4.3 Interrupt map

The interrupts within the FPGA subsystem extend the IoT Kit interrupt map by adding to the expansion area as follows:

Interrupt Input	Interrupt Source
IRQ[32]	UART 0 Receive Interrupt
IRQ[33]	UART 0 Transmit Interrupt
IRQ[34]	UART 1 Receive Interrupt
IRQ[35]	UART 1 Transmit Interrupt
IRQ[36]	UART 2 Receive Interrupt
IRQ[37]	UART 2 Transmit Interrupt
IRQ[38]	UART 3 Receive Interrupt
IRQ[39]	UART 3 Transmit Interrupt
IRQ[40]	UART 4 Receive Interrupt
IRQ[41]	UART 4 Transmit Interrupt
IRQ[42]	UART 0 Combined Interrupt
IRQ[43]	UART 1 Combined Interrupt
IRQ[44]	UART 2 Combined Interrupt
IRQ[45]	UART 3 Combined Interrupt
IRQ[46]	UART 4 Combined Interrupt
IRQ[47]	UART Overflow (0, 1, 2, 3 and 4)
IRQ[48]	Ethernet
IRQ[49]	Audio I2S
IRQ[50]	Touch Screen
IRQ[51]	SPI #0
IRQ[52]	SPI #1
IRQ[53]	SPI #2
IRQ[54]	SPI #3
IRQ[55]	SPI #4
IRQ[56]	Secure DMA #0 Error Interrupt Request
IRQ[57]	Secure DMA #0 Terminal Count Interrupt Request
IRQ[58]	Secure DMA #0 Combined Interrupt Request
IRQ[59]	Non-secure DMA #0 Error Interrupt Request
IRQ[60]	Non-secure DMA #0 Terminal Count Interrupt Request
IRQ[61]	Non-secure DMA #0 Combined Interrupt Request

IRQ[62]	Secure DMA #1 Error Interrupt Request
IRQ[63]	Secure DMA #1 Terminal Count Interrupt Request
IRQ[64]	Secure DMA #1 Combined Interrupt Request
IRQ[65]	Non-secure DMA #1 Error Interrupt Request
IRQ[66]	Non-secure DMA #1 Terminal Count Interrupt Request
IRQ[67]	Non-secure DMA #1 Combined Interrupt Request
IRQ[68]	GPIO 0 Combined Interrupt
IRQ[69]	GPIO 1 Combined Interrupt
IRQ[70]	GPIO 2 Combined Interrupt
IRQ[71]	GPIO 3 Combined Interrupt
IRQ[87:72][16]	GPIO 0 individual interrupts
IRQ[103:88][16]	GPIO 1 individual interrupts
IRQ[119:104][16]	GPIO 2 individual interrupts
IRQ[123:120][16]	GPIO 3 individual interrupts

Table 80 FPGA expansion interrupt map

[16] These interrupts are not supported in the FVP.

5 AHB5 TrustZone memory protection controller

The AMBA 5 High Performance Bus (AHB5) TrustZone *Memory Protection Controller* (MPC) gates transactions towards a memory interface when a security violation occurs.

Each MPC is normally instantiated on the path to a non-security aware AHB5 memory, and is configured by the main secure core in the System.

The MPC divides the memory that it protects into equal size blocks. For each block, it provides a single software configurable bit to define if the block is mapped to a Secure region or Non-secure region. The MPC, using these configuration bits, then performs the task of gating accesses to the memory depending on the AHB access security attribute, so that an access can only target a region that is defined to have the same security settings as the access attributes.

All the security configuration bits in the memory block are stored in a Look-Up-Table (LUT), packed into multiple 32-bit entries that are accessed using indexes using the configuration registers.

The configuration registers can only be written by the processor in the system using Secure accesses (PPRTO[1]==0). Any type of access can read the identification registers. Only 32-bit reads and write are supported.

The configuration registers are listed in the table below.

OFFSET	NAME	TYPE	RESET	DESCRIPTION
0x000	CTRL	RW	0x00000000	bit[2:0] Reserved. bit[4] Security error response configuration. This defines how a security violation is handled on the AHB bus: 0 Access read zeros and writes are masked out. No bus error is returned. 1 Access is blocked and responded with Bus Error. bit[5] Reserved bit[6] Interface gating request. When set to high, the MPC will hold-off AHB accesses trying to get through the MPC. This can be used to prevent access to the SRAM prior to or while configuring the security attributes of the MPC. However, use this with care since gating the MPC can sometimes cause system deadlock. bit[7] Interface gating acknowledge. This indicates if the gating, requested using the gating request is active. bit[8] Autoincrement. This bit allows index, when accessing the security settings in the Look-up-table to be automatically incremented after each access. bit[30:9] Reserved. bit[31] Security lockdown. When set to HIGH, this will prevent the configuration of the following register from being modified: - CTRL - BLK_LUT - INT_EN

				Once set to HIGH, this register cannot be cleared without a reset being applied to the entire MPC.
0x004-0x00C	RSVD	RO	0x0	Reserved
0x010	BLK_MAX	RO	-	Maximum value of block based index.
0x014	BLK_CFG	RO	-	Bit[3:0] – block size 0-32 Bytes 1-64 Bytes … Block size = 1 << (BLK_CFG+5) Bit[30:4] – Reserved Bit[31] – Init in progress
0x018	BLK_IDX	RW	0	Index value for accessing block based look up table
0x01C	BLK_LUT[n]	RW	- IMPLEMENTATION DEFINED	Block based gating Look Up Table (LUT): Access to block based look up configuration space pointed to by BLK_IDX. Bit[31:0] – each bit indicate one block: If BLK_IDX is 0, bit[0] is block #0, bit[31] is block #31. If BLK_IDX is 1, bit[0] is block #32, bit[31] is block #63. … For each configuration bit, 0 indicates Secure, 1 indicates Non-secure. A full word write or read to this register automatically increments the BLK_IDX by one
0x020	INT_STAT	RO	0x00000000	bit[0] – mpc_irq triggered. This indicates if a security violation has occurred. bit[31:1] – Reserved
0x024	INT_CLEAR	WO	0x00000000	bit[0] – mpc_irq clear (cleared automatically) bit[31:1] – Reserved
0x028	INT_EN	RW	0x00000000	bit[0] – mpc_irq enable Bits are valid when mpc_irq triggered is set
0x02C	INT_INFO1	RO	0x00000000	Access address of the first mpc_irq triggered Bits are valid when mpc_irq triggered is set.
0x030	INT_INFO2	RO	0x00000000	Other access attribute of the failing access. The first mpc_irq triggered Bit [15:0] hamster Bit [16] – hnonsec Bit [17] – cfg_ns Bit [31:18] – Reserved Bits are valid when mpc_irq triggered is set
0x034	INT_SET	WO	0x00000000	bit[0] – mpc_irq set. When set to HIGH, enables this MPC to raise an interrupt. This bit is meant for debug purpose only. bit[31:1] – Reserved
0x038 – 0xFCC	RSVD	RO	0x0	Reserved

0xFD0	PIDR4	RO	0x04	Peripheral ID 4 ([7:4] block count, [3:0] jep106_c_code)
0xFD4	PIDR5	RO	0x00	Peripheral ID 5 (not used)
0xFD8	PIDR6	RO	0x00	Peripheral ID 6 (not used)
0xFDC	PIDR7	RO	0x00	Peripheral ID 7 (not used)
0xFE0	PIDR0	RO	0x60	Peripheral ID 0 (Part number [7:0].)
0xFE4	PIDR1	RO	0xB8	Peripheral ID 1 ([7:4] jep106_id_3_0, [3:0] Part number[11:8])
0xFE8	PIDR2	RO	0x0B	Peripheral ID 2 ([7:4] revision, [3] jedec_used, [2:0] jep106_id_6_4)
0xFEC	PIDR3	RO	0x00	Peripheral ID 3 ([7:4] ECO revision number, [3:0] customer modification number)
0xFF0	CIDR0	RO	0x0D	Component ID 0
0xFF4	CIDR1	RO	0xF0	Component ID 1 (PrimeCell class)
0xFF8	CIDR2	RO	0x05	Component ID 2
0xFFC	CIDR3	RO	0xB1	Component ID 3

Table 81 Registers

5.1 Look Up Table (LUT) examples

The contents of the LUT can be accessed in several ways that might require different configurations of the autoincrement function of the BLK_IDX register.

- To dump the full contents of the LUT:

  1. Set the autoincrement enable bit, CTRL[8], to `0x1`.
  2. Read the BLK_MAX register. This has a value 0xN which represents the last address in the LUT.
  3. Write `0x0` to the BLK_IDX register.
  4. Read the BLK_LUT register to 0xN times to read the complete LUT.

- To rewrite the full contents of the LUT:

  1. Set autoincrement enable bit, CTRL[8], to `0x1`.
  2. Read the BLK_MAX register. This has a value 0xN which represents the last address in the LUT.
  3. Write `0x0` to the BLK_IDX register.
  4. Write the new values to the BLK_LUT register 0xN times to fill the complete LUT.

- To read-modify-write:

  1. Set autoincrement enable bit, CTRL[8], to `0x0`.
  2. Write the required address to the BLK_IDX.
  3. Read the current contents of the LUT.
  4. Write the new contents to the LUT.

Even byte accesses can be used to update only the required byte of the register without reading the full contents.

6 AHB5 TrustZone master security controller

The AHB5 TrustZone MSC is a master gasket used to connect a legacy AHB5 master to an AHB5 system and add TrustZone for ARMv8-M capability.

Each MSC provides an AHB slave interface and an AHB master interface. It also has two control inputs, one to define if an associated master interface sitting behind the MSC is Secure or Non-secure, and another to control how the MSC deals with security violations.

The signal that controls if a master is secure or not-secure is driven using the NS_MSCEXP bits in the NSMSCEXP register, allowing up to 16 MSCs to be supported in the system.

Also, the register that controls how to deal with security violation is driven by the SECRESPCFG register.

The MSC can also generate interrupts. Collectively, all MSC Interrupts are handled using the SECMSCINTSTAT, SECMSCINTCLR, and SECMSCINTEN registers.

7 Peripheral protection controller

The PPCs perform the task of gating access to the peripherals it protects, based on the security settings of each peripheral and the security attribute of the access. Each PPC can support up to 16 peripherals.

Two forms of PPC exist here:

- One that supports the AHB
- One that supports the APB.

The PPCs will therefore reside on the paths to the peripherals they control in order to gate accesses.

The security setting of each peripheral is defined using the registers within the Secure privileged control block and the Non-secure privileged control block. These blocks include AHBNSPPC*, APBNSPPC*, AHBSPPPC*, APBSPPPC*, AHBNSPPPC*, APBNSPPPC*, AHBSPPPCEXP* APBSPPPCEXP* AHBNSPPPCEXP* and APNNSPPPCEXP*.

The peripheral will gate access if the bus access has the required security and privilege access attribute required to gain access. If a security violation occurs, SECRESPCFG then defines how the access is handled. PPCs are also able to raise security violation interrupts. These are hosted and controlled using the SECPPCINTSTAT, SECPPCINTCLR and SECPPCINTEN registers.

The PPC can notify the security controller in the system that a security violation has occurred, so that secure interrupts can be raised. The control of these interrupts resides in the Secure privileged control block.