TrustZone Support to I Class Processor

A project Report

submitted by

Sunnihith Srivatsav

in partial fulfillment of the requirements

for the award of the degree of

Master of Technology

Under the guidance of

Dr. V. Kamakoti

Department of Electrical Engineering

Indian Institute of Technology Madras

May 2015

Thesis Certificate

This is to certify that the thesis titled **TrustZone Support to I Class Processor**, submitted by **Sunnihith Srivatsav**, to the Indian Institute of Technology, Madras, for the award of the degree of **Master of Technology**, is a bonafide record of the research work done by him under our supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Dr. V. Kamakoti
Project Guide
Professor
Dept. of Computer Science and Engineering
IIT-Madras, 600 036

Place : Chennai
Date :

Acknowledgement

I would like to express my deepest gratitude to my guide, ***Dr. V. Kamakoti*** for his valuable guidance, encouragement and advice. His immense motivation helped me in making firm commitment towards my project work.

My special thanks to ***Mr. G.S. Madhusudan*** for his encouragement and motivation through out the project. His valuable suggestions and constructive feedback were very helpful in moving ahead in my project work.

I would like to thank my co-guide ***Dr.Nitin Chandrachoodan*** and my faculty advisor ***Dr. Nagendra Krishnapura*** who patiently listened, evaluated, and guided us through out our course.

My special thanks to project team members Neel, Venkatesh,Suresh, Laxmeesha,Bharath,Ammannaidu for their help and support.

Abstract

The aim of the ARM TrustZone technology is to enable a device to benefit from both a feature-rich open operating environment and a robust security solution.The security of the system is achieved by partitioning all of the SoCs hardware and software resources so that they exist in one of two worlds - the Secure world for the security subsystem, and the Normal world for everything else. Hardware logic present in the TrustZone-enabled bus ensures that no Secure world resources can be accessed by the Normal world components, enabling a strong security perimeter to be built between the two. My part of the project is to add TrustZone support to I class processor.It involves designing of TrustZone Address Space Controller (TZASC) and integrating it with AXI bus and APB. TZASC is an AXI component which partitions its slave address range into a number of memory regions. Regions can be made as Secure or Non-secure. It will reject Non-secure transactions to a region that is configured as Secure and vice-versa. The entire project is implemented in Bluespec System Verilog(BSV).

Contents

List of Figures

List of Tables

Abbreviations

RISC	Reduced Instruction Set Computer
AMBA	Advanced MicroController Bus Architecture
AXI	Advanced eXtensible Interface
BSV	BlueSpec System Verilog
APB	Advanced Peripheral Bus
TZPC	TrustZone Protection Controller
TZASC	TrustZone Address Space Controller
SP	Security Permission
SI	Security Inversion

Chapter 1

Introduction

1.1 Overall Microcontroller Architecture

The processor design team of Reconfigurable and Intelligent Systems Engineering[RISE] lab in the computer science department of IIT-Madras has been actively involved in building few processors for academic purposes and other applications. The processor strictly follows the RISC-V instruction set architecture[ISA].Entire design of the processor is done using a Hardware Description Language[HDL] named Bluespec System Verilog[BSV].

I class processors

- 64-bit, 1-4 core, 5-8 stage out of order, aimed at 200-1Ghz industrial control / general purpose applications
- Devices aimed at networking applications will have dual-quad issue support
- Other features - shared L2 cache, AXI bus, threading support

Objective

- To add TrustZone Security to I class processor.
- Integrate TrustZone with AXI bus.
- Integrate TrustZone with APB.

1.2 What is security

In very abstract terms, the term security can be used to cover a number of very different underlying features of a design. However, it is essentially a property of the system which ensures that resources of value cannot be copied, damaged, or made unavailable to genuine users. Every system design will require a different set of security properties, depending on the type and value of the assets it is trying to defend against malicious attack.

Asset: An asset is a resource of value which is important to protect.This may include passwords or an intangible asset.

Attack: This is a process of trying to damage or disrupt an asset which we do not have any permission to access it. The attacker may intentionally try to run some malicious software which may cause security check code to bypass the check thereby compromising the security.The attack may also include hardware tampering and hardware monitoring.

Defend: Defending is process of designing a system that includes software or hardware methods which provide countermeasures against attacks.

1.3 Fundamental Security Properties

The fundamental security properties on which nearly every higher level property can be based are those Confidentiality and integrity.

Confidentiality: An asset is a resource of value which is important to protect.This may include passwords or an intangible asset. Defined set of attacks cannot copy or steal the data of an asset that is confidential This property is used for passwords because it should not be allowed to copy

Integrity: An asset which possess this property can be defended against any modification by defined set of attacks.This property can be used for security software based on which entire system security is dependent upon.

Authenticity: In some cases the design cannot provide integrity so in those design Authenticity is provided. If an attacker has changed some value of an asset then the defender can detect that changes before an asset is used and cause some sort of security fault. This property can be used by the security software i.e., if an attacker make some changes in the program code before loading into the secure environment without being detected,then security check can be bypassed. This property is essential for security software. If an attacker can tamper with the program code before it is loaded into a safe execution location, without being detected, then the security provided by the software can be bypassed.

1.4 Limitations of security solutions

Many security solutions are designed to defend against specific type of possible attacks that may encounter.Defending against all type of attacks is an impossible task.There will be always a attacker trying to spend good amount of time and money to crack the security using complex attacks.So before designing the security solutions one has to decide which type of assets it wants to protect.If we design security solutions for wrong type of asset then it will be very easy for the attacker to attack and crack it.The security requirement for any application should not be like impossible to crack it. It should be described in terms of time and money.

Example:attack A on asset B should take atleast Y days and Z dollars.If Y and Z happens to be larger number then we are successful in designing the security solution.Obviously attacker will move to other target.

1.5 Need For Security

Now a days Embedded devices are handling data of increasing value such as consumer banking credentials.Consumer may download some malicious software which puts these devices at high risk.

TrustZone Technology main aim is to provide security to the sensitive data from both feature rich operating system environment and a robust security solution.Also a well designed hardware architecture and secure software design ensures that data

remains safe whatever may be the operating system.

1.6 Hardware enforced security

Protecting embedded system device from malicious software has the consequences for the entire system design.The hardware and software must work together to protect an asset by using robust security countermeasures against correct type of attack.In the beginning stage of SOC design i.e., starting from processor specifications,if we can carefully integrate hardware security then protection is enabled.It would be difficult to add hardware security module in the later stages of the design process.

TrustZone technology enables high level of security by integrating protective measures into ARM processor,bus fabric and system peripheral IP.With minimum cost TrustZone is able to provide security.Diverse range of secure system architectures have been implemented by TrustZone Technology.

1.7 What are threats

Before going to TrustZone Architecture it is important to understand what is security in this context and what type of risks can attacker pose.It should also be seen in terms of cost only then we can understand hoe much to invest in the defence.The assets each field trying to protect are diverse. Examples of threats:

- Payment fraud: In case of online transaction that causes loss to the service provide.
- In Mobile sector industry IMEI number and SI card are used to provide security feature.Protection mechanism can cracked with little effort.
- Consumer electronics and embedded sector: Increasing wired and wireless connectivity, greater storage of user data, dynamic download of programmable content, and handling of higher value services, all suggest the need for a high-performance and robust security environment.

The above type of attacks can be defended with a well designed TrustZone Technology

1.8 Economic value in security issues

If an attack is successful what is the cost to the business what is the probability of the attack. What is the cost to defend it. These all should be taken care while protecting an asset. Most of the attacks are done by the professional hacker with the motivation of financial gain.

1.9 How are devices attaked

1.9.1 Hack attack

Hacker will be capable of executing a software.Example malware and viruses which are downloaded from internet

1.9.2 Shack attack

This is a low cost hardware attack.Hacker have access to the device but the equipment available with them is not sufficient to attack circuit package.

1.9.3 Lab Attack

This is highly expertise attack.Hackers will perform reverse engineering to attack the device.The attack can go all the way till the transistor level.Every device can be broken by using lab attack.

TrustZone technology is used to provide security to SoC infrastructure.

Who attack Device

Once a designer has identified the assets, and the possible attacks, it is important to identify the possible attackers. Different attackers can deploy different types of attack, and certain assets will only attract certain attackers. This analysis can help rationalize what attacks each asset needs to be protected against.

Remote attacker An attacker is a distributor of the malicious software that we see on desktop workstations: viruses and other malware.The capability to install third-party code, including the execution of browser-based content, makes it easier

to get the malicious code onto the devices to perform the attack.

Security specialist The most technically capable attackers are criminal gangs, security experts, and users attacking devices for fun. These groups are capable of executing a lab attack.

Trusted developer An often unconsidered attack is that executed, or at least assisted, by a person who could be considered trusted.

Device owner This group of attackers have a typical aim of gaining free access to services and content. In general the device owner is motivated to perform the attack, but is not technically competent.

Chapter 2

Background

2.1 System Security

Now a days embedded devices are complicated.They have multiple processor cores,Bus master and many slaves such as memory and peripheral slaves. Embedded sub-systems have to be developed and integrated in such a way that it has to work with security solutions.These sub-systems should not be designed independently with security solutions. It is not of much use if designed with security solution independently because many attackers can bypass functional checking.

Following are the security models which are already in the market:

- **External hardware security model:** Oldest security model is the addition of hardware security module outside of SoC.Example of this is the SIM card and smart card.
  - **Advantages :** These designs have been certified formally.These are used in credit and debit cards.These can provide robust security to the physical device it is used for.
  - **Disadvantages:** These are impractical to be integrated in standard SoC design.It would be compromise on power efficiency, area and performance of the device.
- **Internal Hardware security module:** Hardware module used for providing security located within the SoC provides security better than external modules.These are integrated within a module without compromise on power and

area. Example is the processor which runs parallel to the main processor to prevent unauthorised access to the sensitive resources.

  - **Advantages :** The systems that provide a dedicated general purpose processor for providing security are comparable to the TrustZone hardware solutions in terms of security.
  - **Disadvantages:** The designs that provide a secondary general processor that is dedicated to the security have minor disadvantages when compared to the TrustZone System.

    One issue is that the design needs a separate physical security processor. This processor is typically less powerful than the main applications processor, and also consumes significant silicon area. Additionally, communication between the two processors requires any data to be flushed to coherent memory, which is normally the external memory. This operation is time consuming and can use a significant amount of energy.

- **Software virtualization** Virtualisation is a software mechanism in which a highly trusted management layer known as hypervisor ,runs in the privileged mode of a general purpose processor.Hypervisor consists of MMU(memory management unit)by using this it will be able to run multiple independent software platforms .It will place each of these in a separate virtual machine controlled by hypervisor software.

  - **Advantages:** Any processor which has MMU can be used to implement software virtualization solutions.There is no for an additional hardware to implement hypervisor.
  - **Disadvantages:** Hypervisor provides isolation but is restricted to processor implementing the hypervisor.Other Bus masters such as GPU (graphic processing unit), DMA(Dynamic memory controller) must also be managed by hypervisor otherwise can be bypassed easily of protection. It would be very difficult to implement without damaging the performance of the system.GPU performance may be degraded if we do that.

2.2 TrustZone hardware security

The problems associated with the security systems discussed in the previous section exist because the design can only protect some of the assets in a restricted part of the system, or because the security solution ignores big parts of the attack problem space. In many cases the restrictive nature of the security design means that the wrong assets are protected. Cryptographic hardware module which is used to protect the keys is an valuable asset. But if an attacker keeps on trying to access decrypted data and if he is successful then there is no meaning in only providing security to the cryptographic block.

Any security solution which is able to provide protection to only one of the areas ignoring the other.These are protecting only part of an asset leaving back the big asset placing security solutions below the level which they are intended to provide.

2.2.1 System Wide security :

TrustZone has a hardware architecture that provide security to the entire design.Instead of providing security to the assets in a dedicated hardware block,TrustZone module provides security to any part of the system.

2.3 Bluespec System Verilog

Bluespec System Verilog(BSV) is a a high-level functional hardware description programming language for chip design and electronic design automation. The justification behind writing chip designs in Bluespec is that it leads to shorter, more abstract, and verifiable source code, as well as type-checked numeric code. Because of its expressive power(comparable to most advanced programming languages), full synthesizability, and quality of synthesis, Bluespec is fundamentally changing long-held assumptions about design flow.

2.3.1 Key Features of BSV

- Powerful parameterization and generate.
- High-level of abstraction.

- Fully synthesizable at all levels of abstraction.
- Advanced clock management.
- Powerful static checking

2.3.2 Overview of the BSV build process

- A designer writes a BSV program. It may optionally include Verilog, SystemVerilog, VHDL, and C components.
- The BSV program is compiled into a Verilog or Bluesim specification. This step has two distinct stages:
  - pre-elaboration - parsing and type checking
  - post-elaboration - code generation
- The compilation output is either linked into a simulation environment or processed by a synthesis tool.

  Once the Verilog or Bluesim implementation is generated, the workstation provides the following tools to help analyze your design:
  - Interface with an external waveform viewer with additional Bluespec-provided annotations, including structure and type definitions.
  - Schedule Analysis viewer providing multiple perspectives of a modules schedule.
  - Scheduling graphs displaying schedules, conflicts, and dependencies among rules and methods.

2.3.3 Bluespec SystemVerilog Constructs

Rules: Rules are used to describe how data is moved from one state to another, instead of the Verilog method of using always blocks. Rules have two components:

- Rule conditions: Boolean expressions which determine when the rule is enabled.
- Rule body: a set of actions which describe state transitions.

Modules: A module consists of three things: state, rules that operate on that state, and an interface to the outside world (surrounding hierarchy). A module definition specifies a scheme that can be instantiated multiple times.
Interfaces: Interfaces provide a means to group wires into bundles with specified uses, described by methods. An interface is a reminiscent of a struct, where each member is a method. Interfaces can also contain other interfaces.
Methods: Signals and buses are driven in and out of modules using methods. These methods are grouped together into interfaces. There are three kinds of methods:

- Value Methods: Take 0 or more arguments and return a value.
- Action Methods: Take 0 or more arguments and perform an action (side-effect) inside the module.
- ActionValue Methods: Take 0 or more arguments, perform an action, and return a result.

Application areas of Bluespec System Verilog:

- Modeling for Software development
- Modeling for Architecture Exploration
- Verification
- IP creation

2.3.4 Building a design in Bluespec System Verilog

The various steps involved in building a design in BSV is shown in Figure2.1

- The designer writes the BSV code and it may contain Verilog, VHDL and C components.
- The BSV code is compiled into a Verilog or a Bluesim specification. This step has 2 stages:
  - Pre-elaboration parsing and type checking.

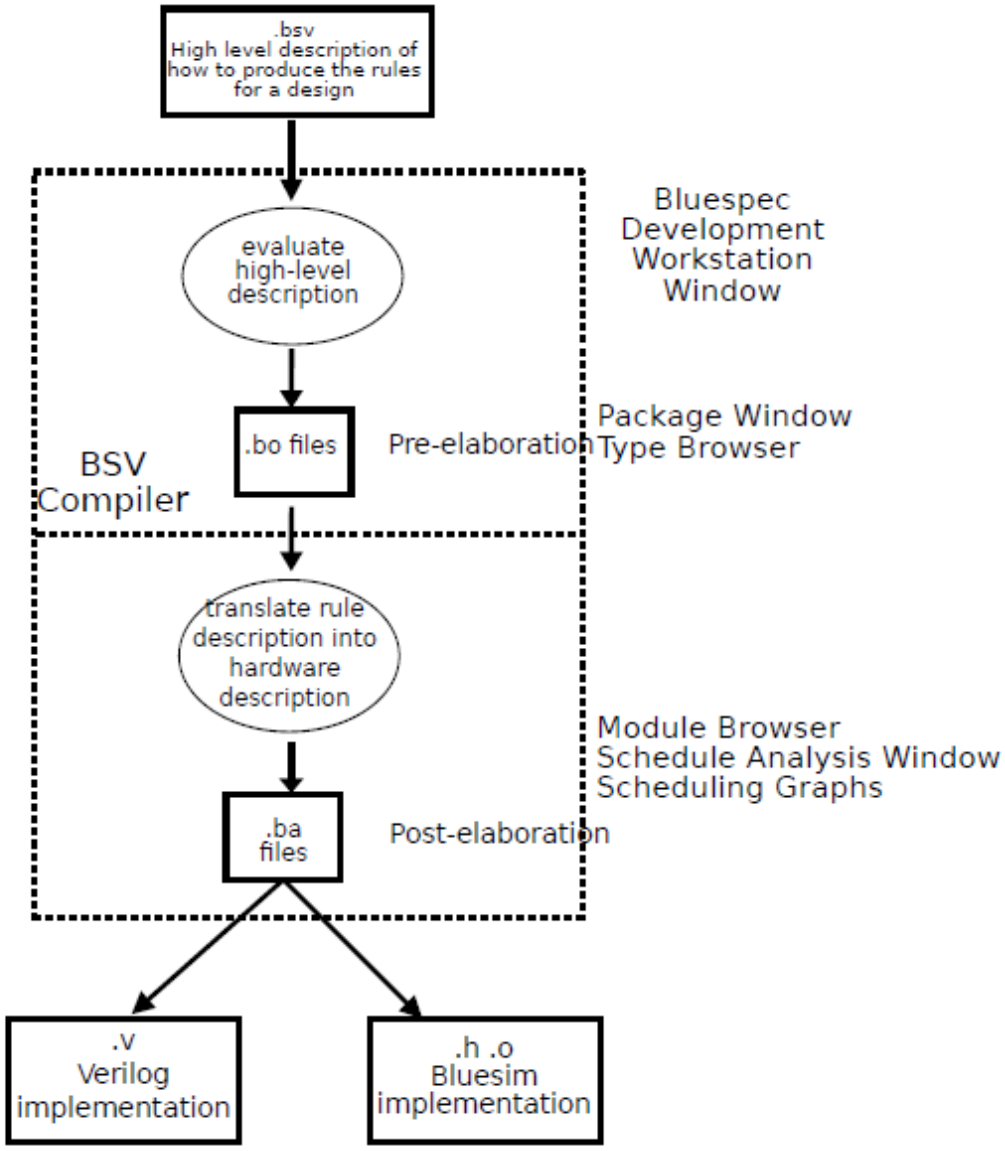

Figure 2.1: Building a design in BSV

  - Post-elaboration code generation.

- The compiled output is either linked to a simulation environment or processed by synthesis tool.

Chapter 3

TrustZone hardware architecture

3.1 System Overview

TrustZone aims to provide security that enables the device to counter many common threats.Instead of providing a one type of fixed solutions,it has options for a designer to choose from a wide range of solutions that can fulfil specific functions. The primary objective of this architecture is to provide integrity and confidentiality for any asset from specific attack.Security solutions with this can be able to provide security for any kind of attack.

The security of the system is achieved by partitioning any SoC hardware and software so that they exist in secure world and normal world.Secure world is for the security subsystems and normal world for everything else.Hardware logic must be inserted in AXI bus so that secure world resources cannot be accessed by non secure or normal world components.By ensuring this we can be sure that this type of security solutions can be used to provide security against many possible attacks.Passwords entered during any online transactions can be made secure.

There is no need for any physical core to provide security .Single core can be able to provide security by making it execute code in both secure and normal world thereby reducing chip area and power consumption.

3.2 System Architecture

There are some changes that needs to be done to implement TrustZone Type of security i.e., to separate resources into Secure and Normal World.

3.2.1 AXI Bus

Two signals are to be added to the AXI bus in addition to the signals which are present in the AXI Bus.These signals will tell whether the incoming transaction to AXI from Master is Secure or Non secure.These Signals are to be added to the existing Read and Write channels, The two important signals are:

Awprot[1] 0-secure write transaction 1-non secure write transaction.

Arprot[1] 0-secure write transaction 1-non secure read transaction.

These are just one bit signals.These are also known as non secure NS bits.When a non secure master transaction is made all bus masters set these signals to indicate that it is a non secure transaction.AXI bus and slave must decode it and permit access only if the required permissions are met which makes them impossible to access secure slaves.Vice versa in case for secure transaction.The address decode for the access will not match any Secure slave and the transaction will fail.

Interrupts are use defined.We can set an interrupt to occur if any of the non secure master tries to access secure slave or leave as such.

3.2.2 APB peripheral bus:

TrustZone architecture has ability to provide security to peripherals such as keyboard, mouse and timers.Generally peripherals are connected to APB bus(Advanced Peripheral Bus).We can write some hardware logic to provide security to these peripherals.Every peripheral can be made secure or non secure.An individual bit is provided to peripheral to make it secure or non secure.These bits are set by software interface to this hardware.When a transaction is received ,logic will check for match of security .If it doesn't match then reject transaction to propagate it till peripherals.

3.2.3 Processor architecture

Each of the physical processor core that is present in these designs provide two virtual cores,one is non secure and other one is secure.The value of NS bit tell us in which mode does the core is in.To context between these two modes,there is mode in between these modes called Monitor mode.Non secure bit is set through software.This bit tell in which mode does the virtual core is in.These two virtual processors execute code in time sliced fashion.The software that is executed within monitor mode is implementation defined.Figure 3.1 shows about the different modes and switching between them.

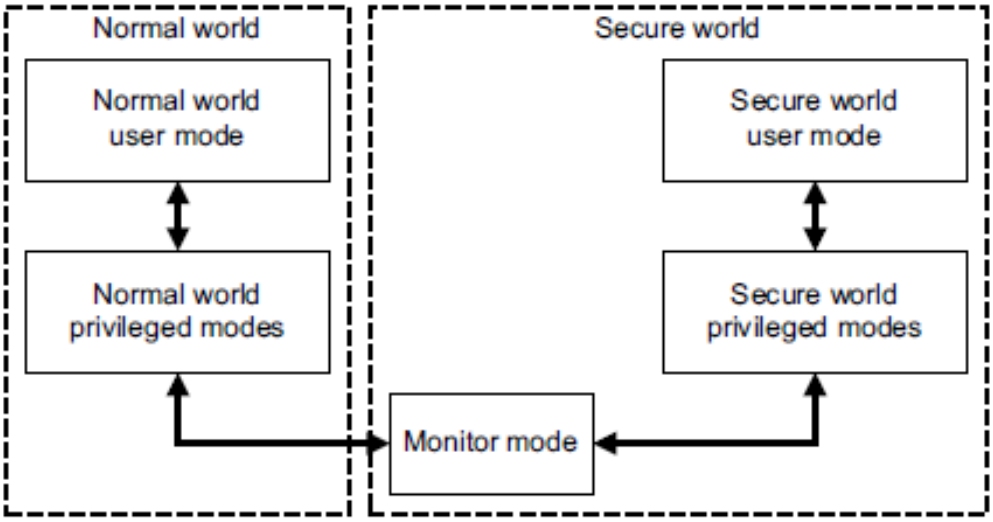

Figure 3.1: Normal world and secure world switching

3.3 TrustZone hardware modules:

3.3.1 TrustZone Protection controller(TZPC)

There is a possibility to secure APB peripherals.Each peripheral is allotted a signal which is given as input to the peripheral.The incoming AXI transaction and address decode logic will tell us which peripheral we are trying to access.Incoming signal will tell us whether peripheral is configured as secure or non secure.Non secure transaction to a secure peripheral is rejected.

3.3.2 TrustZone Address Space Controller(TZASC)

TrustZone Address Space Controller is an AXI component which is used to partition the memory into number of regions.Each region can be configured as secure and non secure.Depending upon the AXI input address,this block will allow or reject access taking into account security check.There is no latency integrating this module with AXI Bus.

The above two modules design involves almost same kind of logic.I have designed TZASC.The same module can be used as TZPC changing input signals to TZASC. TrustZone Security is a combination of hardware and software.My project deals with designing of hardware block to provide security.

Chapter 4

TrustZone Address Space Controller

The TZASC is an AMBA compliant SoC peripheral.It supports AXI protocol and it is a high performance and area optimized.TZASC can provide optimum security required for intended application.Figure 4.1 shows interfaces on the TZASC.

TZASC system contains

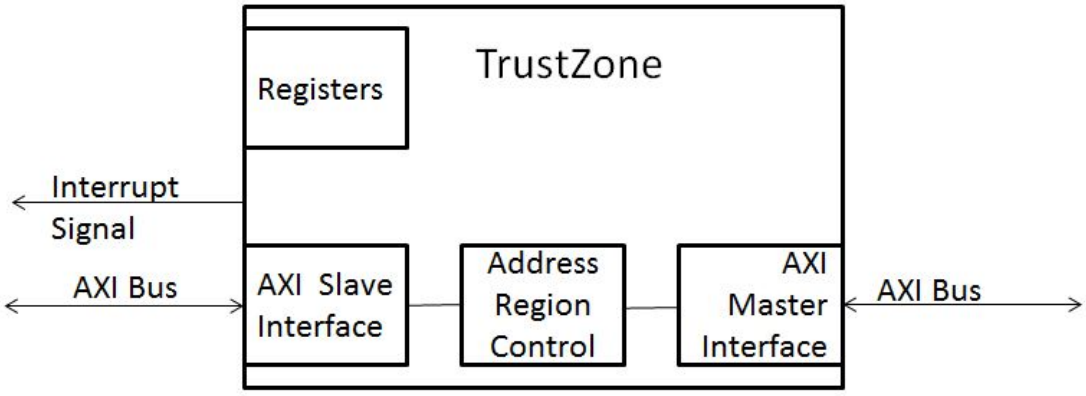

Figure 4.1: Interfaces on TZASC

- AXI bus master: a processor
- AXI interconnect

- Memory

Following features have been provided for TZASC

- It provides security for each address region .Security permissions for each can be changed depending upon our requirement
- It only allows transfers between master and slave if and only if the security permission of the transaction matches the security permission of the memory region it is trying to access
- It is provided with a enable bit for each region.We can make enable region or disable region

Functional Interfaces

- AXI Bus interface : The TZASC provides the following AXI bus interfaces
  - AXI slave bus interface
  - AXI master bus interface
- Each AXI bus interface consists of following AXI channels
  - Write Address(AW)
  - Write Data(W)
  - Read Address(AR)
  - Read Data(R)
- Miscellaneous signals : The following are the miscellaneous signals
  - Tzasc_int : This signal will be enable when any transactions fails security check for a region

TZASC is a system that performs security checks on AXI transaction to memory or off chip peripheral.It can support configurable number of regions,starting and ending address depending upon the size of the region we need,enable for each and every region,security permissions for each and every region

4.1 Regions

Region is a contiguous area of address space.The TZASC provides each region with a changeable security permissions.The security permissions value will tell whether to accept the transaction or deny the transaction.The two signals arprot[1] and awprot[1] are used to determine the type of transactions whether secure or non secure. Memory can be divided into any number of regions depending upon our requirement We can configure the following parameters for each region:

- region enable
- security permissions
- starting and ending address
- Subregion disable

Regions can be overlapped.Priority can be fixed for each region

4.2 Priority

The priority of a region is fixed and is determined by the region number.It can be configured to increase with region number or decrease with region number. Figure 4.2shows Region Priority

When a transaction is received,its address is checked for a match with all the configured regions.The order in which the regions are checked is determined by the priority level,the highest priority is the first to check for a match.The first Region that matches the transaction address match is used as matching region.The matching region security permission determine whether the transaction is permitted or not.

4.3 Subregions

The TZASC divides each region into eight equal-sized, non-overlapping subregions.Region can be divided into powers of 2 number of subregions. Figure 4.3shows Subregion example.

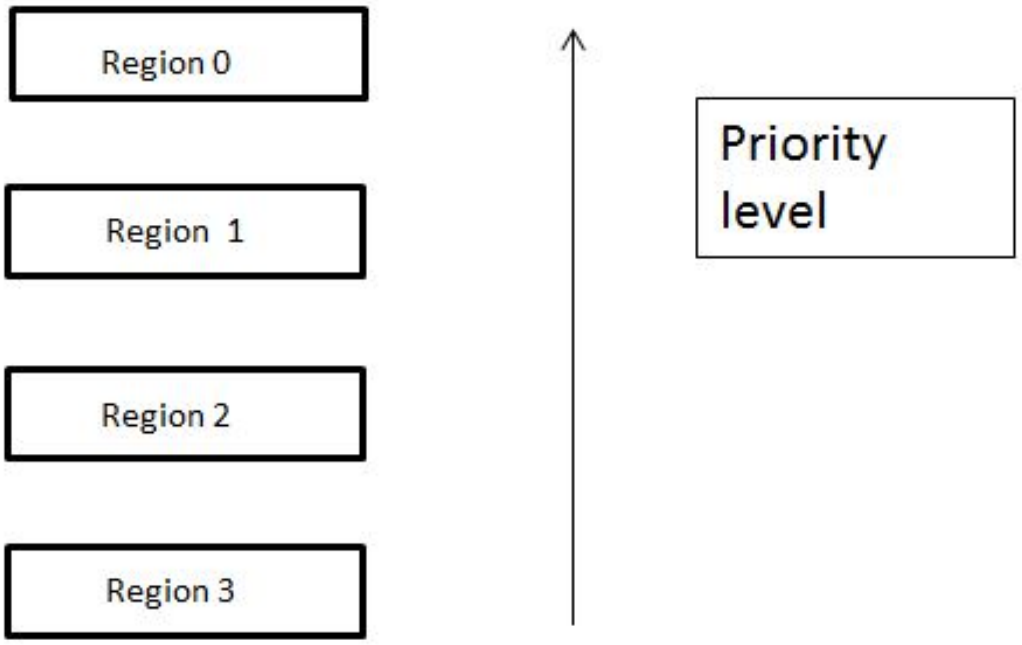

Figure 4.2: Region Priority

4.4 Subregion disable

We can disable any number of subregions in any region.If any subregion is disabled the transaction can be checked for address match in the next higher priority region or deny the transaction. Example of Subregions and region with some of subregions disabled Region subdivided into subregions.some of the subregions are disabled sp$< n >$ represents the region permissions of region n Figure 4.4 shows example of subregion disable.

4.5 Region Security Permissions

We can configure security access permissions for any region TZASC divided.A Region is assigned a security permissions field sp, in its region_attributes $< n >$ register that enables to have complete control of permissions of that region.

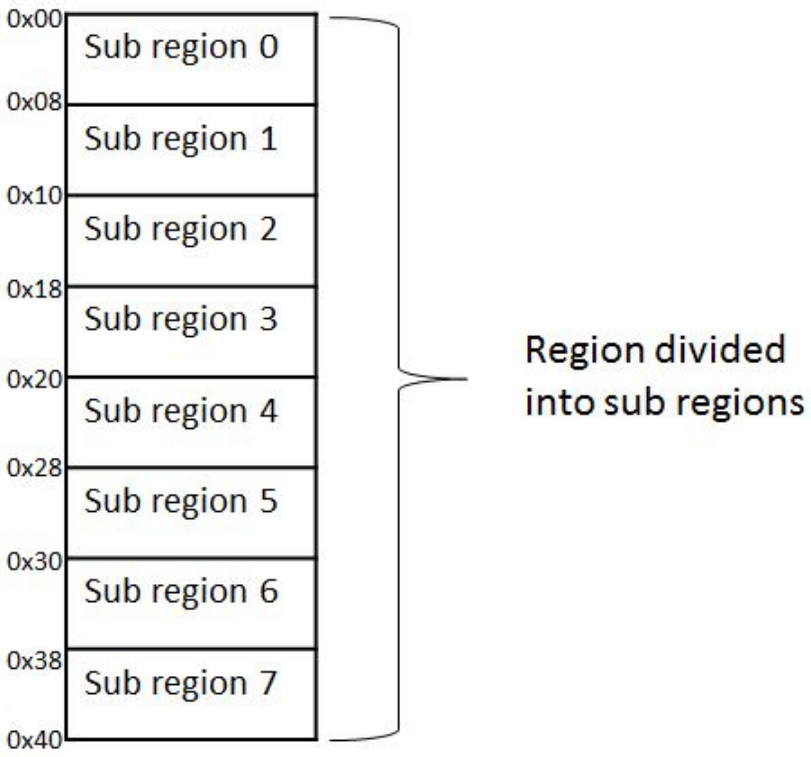

Figure 4.3: Subregions examples

4.5.1 Security Inversion

There are two modes of operation for the region security permission i. e., with or without security inversion. By default,if you program a region to support non secure accesses,then this TZASC ensures that region must also support secure accesses. For example,if you program the region permissions for region 3 to be non secure read only,the TZASC permits access to region 3 for secure reads and non secure reads.

If you require that some regions are not accessible to masters in Secure state,but are accessible in Non Secure state,then you must enable security inversion bit. Security permissions when security inversion is disabled: If security inversion is disabled,TZASC supports certain combination of security permissions.These combinations ensure that master in secure state is not denied access to a region that only accepts non secure accesses. Table 4.1 shows region security permissions when security inversion is disabled.

Security permissions when security inversion is enabled is shown in table 4.2

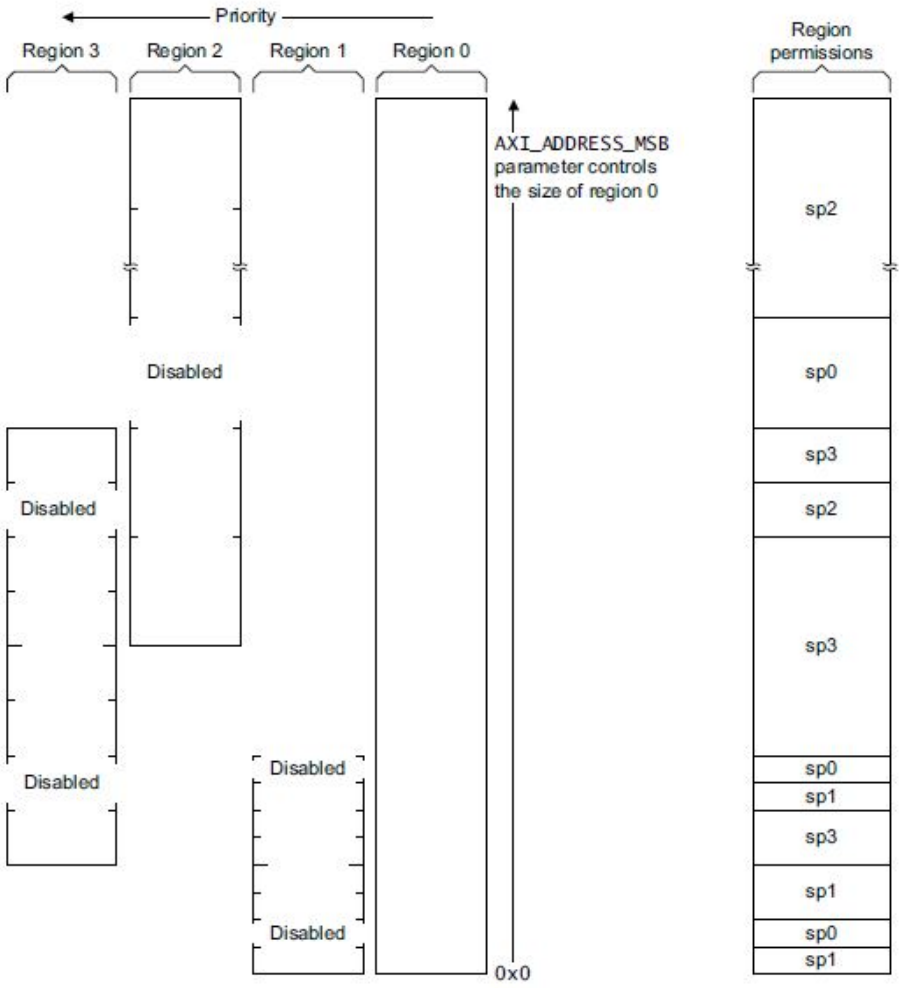

Figure 4.4: Subregion disable example

If you enable security inversion then TZASC permits you to program any combination of security permissions Table 4.2 shows region security permissions when security inversion is enabled

4.6 Denied AXI transactions

If any AXI transactions fails security check ,then that transaction has to be denied.

sp< n > field controls if the TZASC permits access for the following AXI transactions				
sp< n > field	**Non-secure Read**	**Non-secure write**	**Secure read**	**Secure write**
b0000	No	No	No	No
b0001	No	No	No	Yes
b0010	No	No	Yes	No
b0011	No	No	Yes	Yes
b0100,b0101	No	Yes	No	Yes
b0110,b0111	No	Yes	Yes	Yes
b1000,b1010	Yes	No	Yes	No
b1001,b1011	Yes	No	Yes	Yes
b11000,b1101,b1110,b1111	Yes	Yes	Yes	Yes

Table 4.1: Region permissions when security inversion is disabled

For Reads The TZASC responds to the master by setting all bits of read address to zero forcing it to access default memory location

For Writes: The TZASC prevents the transfer of data from master to slave by discarding the write data bus(making incoming data bits to zero).Write address is also made zero i.e.,allowing it access default location and writing default value.

The following sections provide information about the registers present in TZASC.It is important that these registers can only be accessed by processor in secure state only otherwise it can compromise on security of the system

sp< n > field controls if the TZASC permits access for the following AXI transactions				
sp< n > field	**Non-secure Read**	**Non-secure write**	**Secure read**	**Secure write**
b0000	No	No	No	No
b0001	No	No	No	Yes
b0010	No	No	Yes	No
b0011	No	No	Yes	Yes
b0100	No	Yes	No	No
b0101	No	Yes	No	Yes
b0110	No	Yes	Yes	No
b0111	No	Yes	Yes	Yes
b1000	Yes	No	No	No
b1001	Yes	No	No	Yes
b1010	Yes	No	Yes	No
b1011	Yes	No	Yes	Yes
b1100	Yes	Yes	No	No
b1101	Yes	Yes	No	Yes
b1110	Yes	Yes	Yes	No
b1111	Yes	Yes	Yes	Yes

Table 4.2: Region permissions when security inversion is enabled

Interrupt Status Register

Purpose:Returns the status of the interrupt. Figure 4.5 shows Interrupt status register bit assignments.

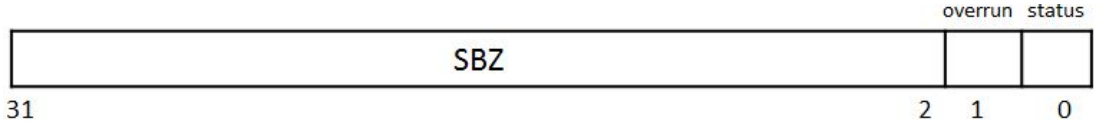

Figure 4.5: Interrupt Status Register

Bits	Names	Function
[31:2]	-	can be used in the future
[1]	Overrun	when set to 1,this bit indicates the occurrence of two or more region permission failures since the interrupt was last cleared.
[0]	Status	Returns the status of the interrupt 0=interrupt is disabled 1=interrupt is enabled

Failure Address Low Register

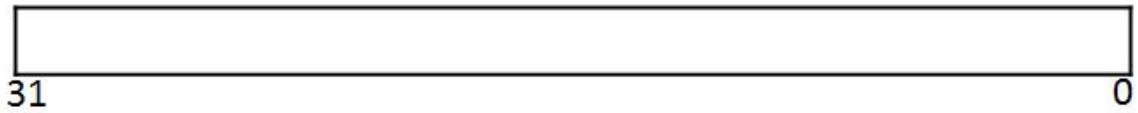

Figure 4.6: Fail Address Low Register

Purpose: Returns the address of the transaction that failed security check after the interrupt is cleared. Figure 4.6 shows the Failure Address Low Register.

Bits	Name	Fucntion
[31:0]	Add_setup_low	Returns the AXI address bits [31:0] of the first access to fail a region permission check after the interrupt was cleared.

Failure Control Register

Purpose: Returns the control status information of the first access that failed a region permission, after the interrupt was cleared. Figure 4.7 shows the bit assignments of Failure Control register.

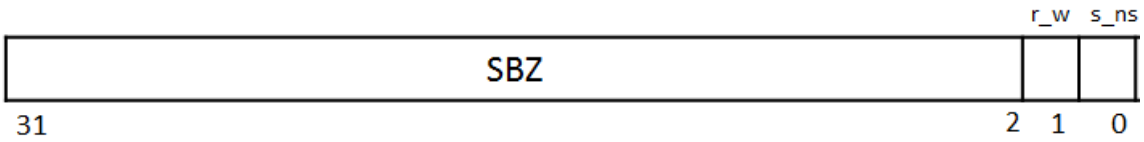

Figure 4.7: Failure Control Register

Bits	Names	Function
[31:2]	-	These bits are made zero.Can be used in future
[1]	read-write	This bit indicates whether the first access to fail a region permission check was a write or read 0= Read access failure 1= write access failure
[0]	Non-secure	After clearing the interrupt status ,this bit indicates whether the first access to fail a region permission check was non secure .0=secure access 1=non secure access

Fail Id Register

Purpose: Returns the master Id of the first access that failed a region permission

Bits	Name	Function
[31:n+1]	-	These are made Zero and can be used in future
[n:0]	id	returns the master id tag of the first access to fail a region permission check after the interrupt was cleared.

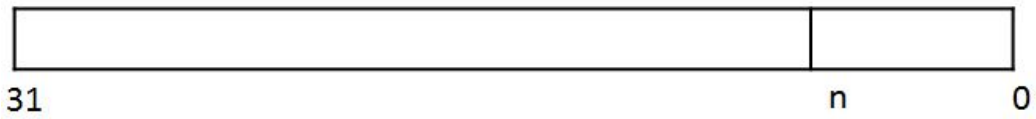

Figure 4.8: Failure Id Register

after the interrupt is first cleared. Figure 4.8 shows the Failure Id Register.

Region Setup Low $< n >$ Register

Purpose: It controls the address of lower boundary of a region. Figure 4.9 shows region setup low register

Region Setup High $< n >$ Register

Purpose: It controls the address of higher boundary of a region. Figure 4.9 shows region setup high register.

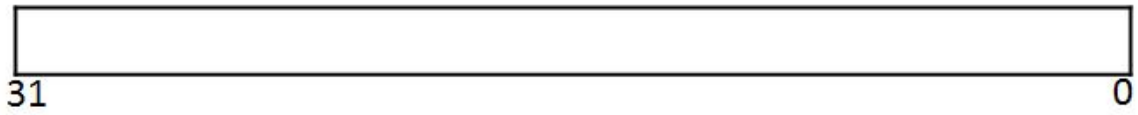

Figure 4.9: Region Setup Low Register

Bits	Name	Function
[31:0]	Address_low$< n >$	It contains the lower address boundary of the region $< n >$.This can be changed upon our requirement

Region Attribute $< n >$ Register

Purpose: Controls the permissions ,sub region disable and region enable. Figure4.10 shows the bit assignment of Region attribute register.

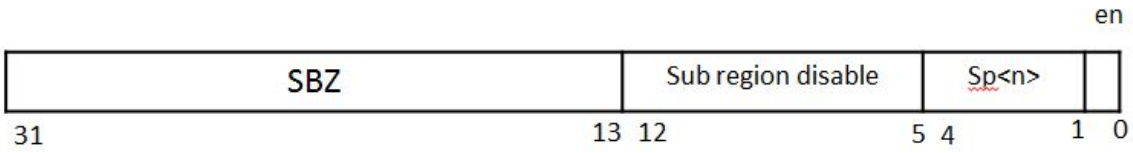

Figure 4.10: Region Attribute $< n >$ Register

Bits	Name	Function
[31:13]	-	These are made zero and can be used in the future
[12:5]	Sub region enable	Regions are split into eight equal sized sub regions and each bit in this field indicates whether the corresponding region is enabled. Bit [15] = 1 Sub region 7 is enabled Bit [14] = 1 Sub region 6 is enabled Bit [13] = 1 Sub region 5 is enabled Bit [12] = 1 Sub region 4 is enabled Bit [11] = 1 Sub region 3 is enabled. Bit [10] = 1 Sub region 2 is enabled. Bit [9] = 1 Sub region 1 is enabled. Bit [8] = 1 Sub region 0 is enabled.
[4:1]	sp$< n >$	Permission setting for a region$< n >$.If an AXI transaction occurs to region n, the value in the sp$< n >$ field controls whether the TZASC permits the transaction to proceed
[0]	enable for a region	0=region $< n >$ is disabled. 1=region $< n >$ is enabled. n=0,1,2,3.

Security Inversion Enable:

Purpose: Controls whether the TZASC enables security inversion to occur.

Bits	Name	Function
[0]	security inversion	Controls whether the TZASC permits security inversion to occur:0-security inversion is not permitted .This is default case.1-security inversion is permitted. This enables a region to be accessible to masters in non secure state but not accessible to masters in secure state.

Chapter 5

Integration of TZASC with AXI Bus:

The main components are: AXI Master: AXI master initiates transfer on the bus, and receives data. AXI Slave: AXI slave responds to the transaction initiated by the master. Fully configured wait states and slave error response. Storing ability as internal memory and supplying data on the demand. AXI Interconnect/Arbiter: AXI interconnect can take multiple masters at a time and arbitrates for the bus using configurable priority scheme.
It consist of five different channels:

- Write Address Channel as shown in table 5.1
- Read Address Channel as shown in table 5.2
- Write Data Channel as shown in table 5.3
- Read Data Channel as shown in table 5.4

There are two interfaces linked up with TZASC namely AXI Master interface and AXI slave interface. Following figure5.1shows the integration of TrustZone module with AXI bus.
TZASC performs check on the signals of slave interface.Address region control block contains logic which performs checks and permit the access to the slave only if the all required permissions are matched. As explained in the previous chapter , registers

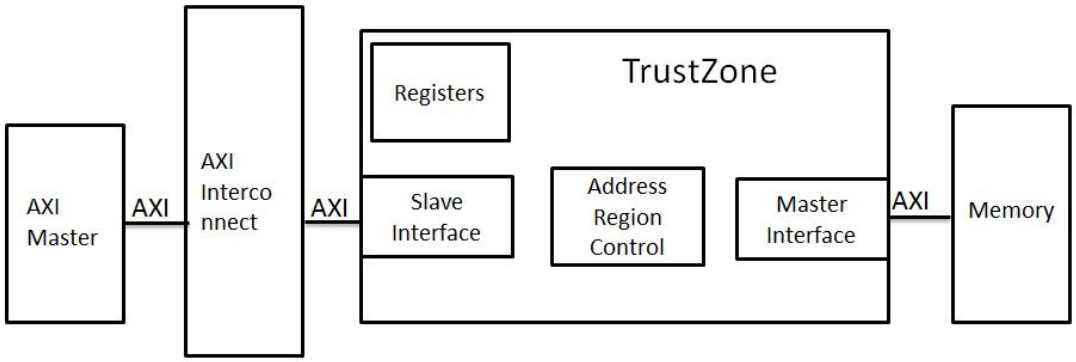

Figure 5.1: Integration of TrustZone with AXI

have control over the security permissions of each region.When an AXI transaction is received to TZASC, the hardware logic is used to check whether the transaction is permitted or not. AXI bus consists of different channels for read and write.Read and write channels are independent, so data can move in both directions between the master and slave simultaneously, and data transfer sizes can vary. The limit in AXI4 is a burst transaction of up to 256 data transfers. Each transaction is burst-based which has address and control information on the address channel that describes the nature of the data to be transferred. The data is transferred between master and slave using a write data channel to the slave or a read data channel to the master.

Master will send whether the required transaction is secure or non secure.Depending upon the security inversion bit security checks are performed. If that bit is enabled, secure access to non secure location is denied. If security inversion is disabled then secure access to a non-secure location is permitted.

Using TZASC , I have divided memory into 4 regions .Starting and ending address can be loaded from memory or can be given as input from some means. It can be configured into different number of regions .Each region has a region enable bit in region attributes register. Each region is divided into 8 sub regions. For each sub region enable bit is present in region attributes register.

Read Transaction: Read address is received on read address channel. Received address is checked for a match from 4 regions. Region priority order is region0, region1, region2, region3. Region0 has highest priority and region3 has lower priority. This order can be changed.

Write address channel

Signal	Source	Description
AWADDR	Master	Write address. The write address gives the address of the first transfer in a write burst transaction
AWLEN	Master	Burst length. The burst length gives the exact number of transfers in a burst. This information determines the number of data transfers associated with the address.
AWSIZE	Master	Burst size. This signal indicates the size of each transfer in the burst.
AWBURST	Master	Burst type. The burst type and the size information, determine how the address for each transfer within the burst is calculated
AWPROT	Master	Protection type. This signal indicates the privilege and security level of the transaction, and whether the transaction is a data access or an instruction access.
AWVALID	Master	Write address valid. This signal indicates that the channel is signaling valid write address and control information
AWID	Master	Write address ID. This signal is the identification tag for the write address group of signals
AWREADY	Slave	Write address ready. This signal indicates that the slave is ready to accept an address and associated control signals.

Table 5.1: Write Address Channel Signals

Example flow: If received address belongs to region0 and region 1,first preference is given to region0.arprot[1] signal received will tell whether this read transaction is secure or non secure.Valid bit present on address channel must be true to continue the access. Then Address is matched against sub regions. Security permission field in the region attributes register of region 0 is compared against arprot[1] signal. If these fields also match, then AXI transaction is allowed to access the memory location assuming that sub region is enabled. Suppose sub region bit is not enabled

Read address channel

Signal	Source	Description
ARADDR	Master	Read address. The read address gives the address of the first transfer in a Read burst transaction
ARLEN	Master	Burst length. The burst length gives the exact number of transfers in a burst. This information determines the number of data transfers associated with the address.
ARSIZE	Master	Burst size. This signal indicates the size of each transfer in the burst.
ARBURST	Master	Burst type. The burst type and the size information, determine how the address for each transfer within the burst is calculated
ARPROT	Master	Protection type. This signal indicates the privilege and security level of the transaction, and whether the transaction is a data access or an instruction access.
ARVALID	Master	Read address valid. This signal indicates that the channel is signaling valid read address and control information
ARID	Master	Read address ID. This signal is the identification tag for the read address group of signals
ARREADY	Slave	Read address ready. This signal indicates that the slave is ready to accept an address and associated control signals.

Table 5.2: Read Address Channel Signals

and the received address belongs to that sub region, then the transaction address match is transferred to region 1.The same process repeats. i.e., permissions checks are performed for region 1 access. If that matched sub region is enabled, transaction is allowed to access memory. If that sub region is also disabled then we say that transaction has failed permission check. Immediately address and master id tag is transferred to failure address register and failure control register respectively. Transaction is allowed on default location. In my case I made address 0 as default location. Interrupt Is generated if 2 or more transactions are denied access. Interrupt overrun

Write data channels

Signal	Source	Description
WDATA	Master	Write data.
WVALID	Master	Write valid. This signal indicates that valid write data and strobes are available
WLAST	Master	Write last. This signal indicates the last transfer in a write burst.
WSTRB	Master	Write strobes. This signal indicates which byte lanes hold valid data. There is one write strobe bit for each eight bits of the write data bus.
WID	Master	Write ID tag. This signal is the ID tag of the write data transfer.
WREADY	Slave	Write ready. This signal indicates that the slave can accept the write data

Table 5.3: Write Data Signals

Read data channels

Signal	Source	Description
RDATA	Slave	Read data.
RVALID	Slave	Read valid. This signal indicates that the channel is signaling the required read data.
RID	Slave	Read ID tag. This signal is the identification tag for the read data group of signals ge.nerated by the slave.
RLAST	Slave	Read last. This signal indicates the last transfer in a read burst.
RREADY	Master	Read ready. This signal indicates that the master can accept the read data and response information.

Table 5.4: Read Data Signals

bit is enabled in the interrupt status register. Also I have loaded all the failure transactions address.

Write Transaction: Write address is received on read address channel. Received address is checked for a match from 4 regions. Region priority order is region0,region1, region2, region3. Region0 has highest priority and region3 has lower priority. This order can be changed.

Example flow: If received address belongs to region0 and region 1,first preference is given to region0.awprot[1] signal received will tell whether this read transaction is secure or non secure.Valid bit present on address channel must be true to continue the access. Then Address is matched against sub regions. Security permission field in the region attributes register of region 0 is compared against awprot[1] signal. If these fields also match, then AXI transaction is allowed to access the memory location assuming that sub region is enabled. Suppose sub region bit is not enabled and the received address belongs to that sub region, then the transaction address match is transferred to region 1.The same process repeats. i.e., permissions checks are performed for region 1 access. If that matched sub region is enabled, transaction is allowed to access memory. If that sub region is also disabled then we say that transaction has failed permission check. Immediately address and master id tag is transferred to failure address register and failure control register respectively. Transaction is allowed on default location. In my case I made address 0 as default location. Interrupt Is generated if 2 or more transactions are denied access. Interrupt overrun bit is enabled in the interrupt status register. Also I have loaded all the failure transactions address.

Chapter 6

Integration of TrustZone with APB

TrustZone is used to provide security to general purpose peripherals such as timers, interrupt controllers, UARTs etc.,These peripherals are generally slow, these are con-

Name	**Description**
PCLK	This is the clock signal.All the signals are sampled at the rising edge of the clock
PRESETn	This is the reset signal.It is an active low signal.It is generally connected to system reset signal
PADDR	Address bus.
PWDATA	Write data bus.
PWRITE	This signal indicates the transfer direction. When LOW this signal indicates a read transfer.When HIGH this signal indicates a write transfer
PRDATA	Read data bus.
PREADY	This signal indicates if the slave is ready for the transfer or not. HIGH indicates that the slave is ready. LOW indicates that the slave is busy and wait states are inserted in the transfer
PSELx	The APB bridge unit generates the select signal to each peripheral bus slave. It indicates that the slave device is selected and that a data transfer is required
PENABLE	Enable signal.This signal indicates that the master is for transfer.

Table 6.1: List of signals

nected through a simpler APB bus protocol. TrustZone module is placed between APB bus and slaves.In my case slave is memory. Apart from all the APB signals, master(generally core) will send whether the initiated transfer is secure or non secure NS bit. Based on NS bit and read write signal, security checking is done for Secure read,secure write, non secure read and non secure write. The same type of protection is given as in the case of AXI bus.The List of signals present in this protocol are shown in table 6.1.

Example flow: Address received on is checked for match from four regions.The region that have higher priority is selected as matched region if any overlapping is present.Then address is matched for a sub block. If sub block is enabled ,then it is checked for a match with region security permissions.Access to slave is permitted only if all the permissions are granted otherwise it is denied.

Chapter 7

Synthesis

Bluespec Compiler converts the design in BSV to synthesizable Verilog code.Functionality is checked with a test bench for various possible cases.Module is also tested with test benches of AXI and APB bus protocols.

7.1 Synthesis Report

The design has been synthesized using Xilinx ISE for Virtex 5 xc5vlx110t-1-ff1136 All default settings were used. The design strategy was set to optimization for speed.

AXI read:

```
Device utilization summary:
---------------------------

Slice Logic Utilization:
 Number of Slice LUTs:                 1038  out of  69120     1%
    Number used as Logic:              1006  out of  69120     1%
    Number used as Memory:               32  out of  17920     0%
       Number used as RAM:               32

Slice Logic Distribution:
 Number of LUT Flip Flop pairs used:   1069
```

```
Timing Summary:
---------------
Minimum period: 2.305ns
Maximum Frequency: 433.839MHz
```

AXI write:

```
 Device utilization summary:
---------------------------

Slice Logic Utilization:
 Number of Slice LUTs:                 1077  out of  69120     1%
    Number used as Logic:              1045  out of  69120     1%
    Number used as Memory:               32  out of  17920     0%
       Number used as RAM:               32

Slice Logic Distribution:
 Number of LUT Flip Flop pairs used:   1109

Timing Summary:
---------------
Minimum period:2.310ns
Maximum Frequency:432.969MHz
```

Chapter 8

Conclusion and Future work

Conclusion Designing of TrustZone Address Space Controller and Integrating it with AXI bus and APB bus have been successfully implemented in BlueSpec System Verilog.TZASC provide security to memory region.TZASC module has been tested for various test cases i.e., for different combinations of Bits in security permissions block of Region Enable register and also for combinations of bits in various registers.The Test cases which are used for testing AXI bus and APB ,are also used for testing integration part. The transactions which failed security permission check are displayed in a an output.txt file to cross check the failures.

Future Work TrustZone Security provides security to each and every block in a SoC design. In the similar way of designing a TZASC, TZPC can also be designed to provide security to SoC components and peripherals such as keyboard ,mouse,LCD controller etc.
Division of memory into more regions to provide security features for various applications.Example:for address range of general peripherals such as screen control, for secure world OS kernal.

Bibliography

[1] AMBA AXI and ACE PROTOCOL specification, ARM, 2011.

[2] CoreLink TrustZone Address Space Controller TZC-380, ARM, 2010.

[3] ARM Security Technology ,Building a Secure System using TrustZone Technology, ARM, 2009.

[4] AMBA 3 APB Protocol v1.0 ,ARM,2005.

[5] PrimeCell Infrastructure AMBA 3 AXI TrustZone Memory Adapter, ARM, 2006.

[6] Bluespec, Inc, Bluespec System Verilog Reference Guide, revision: 17 ed., 2014.

[7] R. S. Nikhil and K. Czeck, BSV by Example. Bluespec, Inc, 2010.