Reliability Aware Intelligent Memory Management (RAIMM)

Student Name: Sidhartha Sankar Rout
Roll Number: MT12105

Indraprastha Institute of Information Technology Delhi, New Delhi

Under the Supervision of
Dr. Sujay Deb (IIIT-Delhi)
Mr. Deepak Baranwal (STMicroelectronics)

Submitted in partial fulfillment of the requirements for the Degree of M.Tech. in Electronics & Communication Engineering with specialization in VLSI & Embedded Systems

This research work has been done with the collaboration of ST Microelectronics Pvt Ltd., Greater Noida

Declaration

I hereby declare that the work presented in the report entitled "**Reliability Aware Intelligent Memory Management (RAIMM)**" submitted by me for the partial fulfillment of the requirements for the degree of *Master of Technology in VLSI & Embedded Systems specialization (Electronics and Communication Engineering)* at Indraprastha Institute of Information Technology, Delhi, is an authentic record of my work carried out under guidance of **Dr. Sujay Deb** in IIIT-Delhi and **Mr. Deepak Baranwal** in ST Microelectronics. Due acknowledgements have been given in the report to all material used. This work has not been submitted anywhere else for the reward of any other degree.

Sidhartha Sankar Rout **Place & Date**: .....

Certificate

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

Dr. Sujay Deb **Place & Date**: .....
IIIT-Delhi

Mr. Deepak Baranwal **Place & Date**: .....
STMicroelectronics

Contents

List of Figures

List of Tables

Abstract

The growing technology scaling and larger die size of multi-processor System-on-Chip (SoC) have increased the error rates for on-chip memories. Increased system speed for high performance, aggressive voltage scaling for power reduction and intra-die process variation have exaggerated the unreliability issues. Hence a method for memory management on SoCs to enhance their reliability is discussed, consisting of a mechanism for automatically moving the contents of a less reliable memory to a more reliable memory. The solution module designed as RAIMM (Reliability Aware Intelligent Memory Management) is an architectural framework to dynamically compute reliability of the on-chip memories and provide a better reliable solution for the application in case of any memory failure. The silicon characterization data is used in conjunction with the on-chip process/voltage/temperature sensors to correctly estimate the memory reliability status. It provides a ranking mechanism for the available memories based on the operating conditions, silicon characterization data as well as dynamic access profiling data, which can be used to provide a method to accurately predict memory failure in advance to the application. A Direct Memory Access (DMA) engine ensures the efficient working of overall application with low overhead for software in maintaining the memory configuration and contents.

Keywords: Memory Reliability, intra-die variation, memory ranking, memory characterization, System prototyping with virtual platform, dynamic memory remapping

Acknowledgments

First of all I would love to thank Almighty God for giving me such a wonderful life and showing me many hard times to make me stronger in life.

I would like to express my deep gratitude to my master thesis advisor Dr. Sujay Deb for his excellent guidance and continuous support throughout the span of the project. He was always there to improve my conceptual understanding of the subject matter and finding way out of the problems. Without his patience, motivation, critical judgment and outlook this work would never have been successful.

I convey my heartiest gratitude and gratefulness to my supervisor in STMicroelectronics, Mr. Deepak Baranwal who not only supported and guided me in technical conceptualizations from the beginning till the end of the project but also looked after me as an elder brother, a friend and an advisor. Without him I probably could not find the work place so familiar and refreshing. My sincere thanks go to Mr. Digvijay Pratap Singh for his handful of support throughout my journey in STMicroelectronics. He has always shown his presence in spite of his busy schedule to solve out the problems where I got stuck. I would like to thank Mr. Khanusiya Soyeb A for his selfless help in transferring the knowledge related to the project and Mr. Milin Raijada for his continuous help in IP verification process in STMicroelectronics.

I take this opportunity to thank Prof. R.N. Biswas for his words of inspiration, continuous support and advices. Very special thanks to IIITD and STMicroelectronics for providing me this opportunity and infrastructure support during the entire span of the research.

Special thanks to all my friends at IIITD, specially Rohit, Vijender, Vinod, Hemant, Rahul and Neeraj, who made my life lively for the last two years. On this occasion of showing gratitude I would love to say “thank you” to my parents for everything and I would be nowhere without their affection and unconditional support. Thanks to all my siblings for walking all the way with me and last but not the least a special thanks to a very special person in my life, Niru Chowdhury for encouraging me, believing on me and being with me throughout all the ups and downs.

Chapter 1: Introduction

1.1 Background

With the growing scaling of technology the recent chips have shown exponential increase in transistor densities and are now capable to contain multi-processor System-On-Chips which was not possible few years ago. However, as the geometries of the transistors reach the physical limits of operation, it becomes increasingly difficult for the hardware to achieve reliable operation. Earlier reliability issues were restricted to within different dies in the wafer (Inter-Die variations) but now the similar phenomenon is being observed within the die (Intra-Die variations). The variability in process technology, the issue of thermal hotspots and the effect of various noise sources, such as power supply fluctuations, pose major challenges for the reliable operation of current and future multi-processor SoCs. As the die-size increases, these variations become more visible and difficult to contain. Integration of multiple processors in a single SoC increases the power demand thus creating the need to drive as much possible on the lower voltages. One of the most occupying components of the die area is on-chip memories. The dual effect of intra-die process variation as well as reduction in operating voltages presents a situation for on-chip memories which limits its performance and reliability where errors can flip the stored bits, possibly resulting in a complete system failure.

1.2 Motivation

Now-a-days electronic systems have been used in many applications starting form computing devices to medical devices; from avionic applications to automotive applications. Reliability of electronic systems has always been a concern. The intensity of concern increases when the system is related to an application, where a fault or hazard may lead to an accidental situation which in turn risks the human life. In earlier days, avionics systems using electronic controls were considered as safety critical and adopted high level of costly reliability protections. But now-a-days applications involved in everyday life to a large number of population, use electronic controls to perform many tasks. For example, today's automotive industry evolved to "Drive- By-Wire" era, adopts increasing number of electronic modules to perform the jobs like brake by wire, steer by wire, shift by wire etc. If considerable level of reliability would not be provided in the electronic control systems used in safety critical applications like automotive, then the failure in system may lead to an accidental situation. As shown in figure 1.1, even a minor system fault can surge into major system failures very quickly and results in a life-threatening hazard.

Faults in electronic systems are defects that can be converted to failures. A very small defect, such as a frozen memory bit, a stuck-at fault, an uninitialized variable in software, an alpha particle hit or cosmic ray ionization can be considered as a fault.

Error may be viewed as the next level of a fault which may lead to an unexpected behavior within the system. Something like mistaken value of a state variable, entering into an infinite loop or incorrect result of a calculation might be considered as errors in system. Failure may be seen as the next level of error which may raise the situation where some part of the system does not function as expected [1].

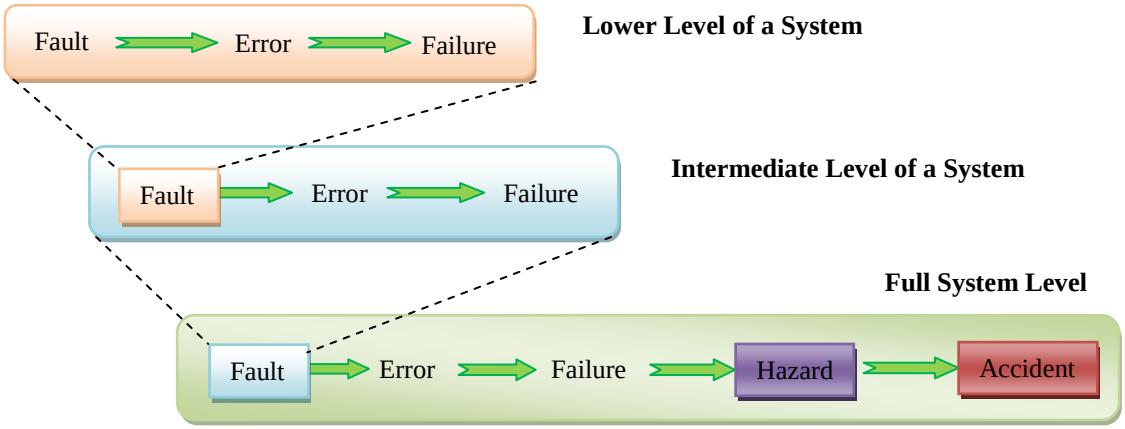


Figure 1.1: Fault-error-failure cascade can lead to life-threatening hazards [1]

As figure 1.1 shows, a failure at some low level part of a system can be seen as a fault at another upper level, which can induce errors at that level and that can stimulate failures. A failure at intermediate level of a system can be viewed as faults at full system level. If these faults are not taken care well before the time when these can avalanche to system-level failures, they can give rise to hazards that have the potential to threaten injury or loss of life [1].

The idea behind this thesis work has been emerged to provide a fault tolerant platform for system memories in the SoCs, used in safety critical automotive products. Of course this design can find its use in any embedded application where safety is the primary concern. In recent years automotive industry has adopted electronics to a large extend in its control part and has shifted to “Drive-by-wire” generation. The growth of electronic control in automotive products has replaced many mechanical control systems with its electronic counterparts and evolved with the facilities such as throttle by wire, brake by wire, steer by wire, shift by wire etc. The pervasive use of electronics in automotive has raised the importance of maintaining high reliability of the SoC used in the control circuit to achieve the functional safety.

The ISO 26262 standard defines the compliance of functional safety for all automotive electronic and electrical safety related systems. For making the SoC safety compliant; Redundancy, voltage supply monitoring, BIST and watchdog timers are some of the design techniques which are used. Redundancy is adopted as the most favourable method to provide electronic safety in automotive devices. Different forms of redundancy used in the automobile devices are like N-modular redundancy (NMR), hardware redundancy, software redundancy, information redundancy and time redundancy [2] [3] [4].

In today's System on Chip the most area occupying component is the memory device. According to the prediction of International Technology Roadmap for Semiconductors and Semiconductor Industry Association, embedded memory will continue to dominate the SoC content and will occupy more than 90% of the total die area in few years from now [5]. Hence, on-chip memory reliability draws the highest attention while the reliability of SoC is a concern. Memory reliability is indeed a significant issue, with half of system failures attributable to faults in the random access memory [6]. SRAM, the building block of on-chip memory is most vulnerable to soft error. Soft error in SRAM is a transient error which is caused by alpha particle hit emitted from chip packaging materials and hitting of atmospheric cosmic radiations. Soft error rate (SER) can be predicted as 1000 to 5000 FIT (Failure in time) per Mbit for modern memory devices [7]. To eliminate the soft errors in SRAM, many design modifications have been done at circuit level and several error detection and correction mechanisms have been adopted at system level. As a result alpha particle problems have been largely eliminated but the error due to cosmic rays is still being a subject of concern [7].

Today's SoC is encountering (1) intra-die process variability with continuous technology scaling for area reduction and increased transistor density, (2) hotspots due to rise in temperature as a result of rigorous switching with high speed operation and (3) low operating voltage to have low power applications. All these three parameters, process variability, increased temperature and lower operating voltage exaggerate the SER in SRAM hugely. Research shows that in extreme operating conditions and under random process variability, the SER of an SRAM can reach values up to 3X larger than the nominal value, or down to 2X smaller than the nominal value [8].

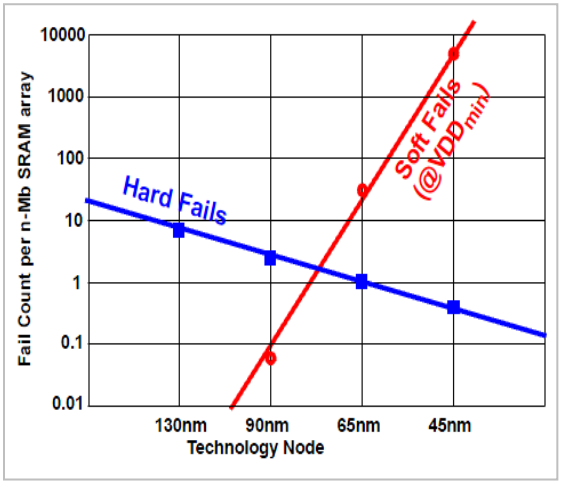


Figure 1.2: Hard and soft failure predictions vs. technology node [9]

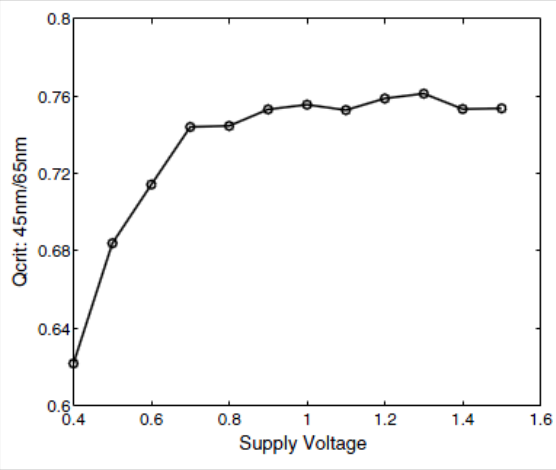


Figure 1.3: SRAM bit cell Q_{crit} (45nm/65nm) vs. supply voltage [12]

Figure 1.2 shows the growing rate of soft error in SRAM array with the shrinking technology node [9]. Though due to different design advancements the hard error is reducing in SRAM but the soft error rate is growing rapidly.

The intra-die process variation because of technology shrinking, gate oxide degradation and ageing effect has shown considerable impact on the SER in SRAM [10] [11]. It has been found that for 45nm/65nm technology SRAM cell, Q_{crit} reduces rapidly as the supply voltage scales down to 0.6 volt and below [12]. Q_{crit} is the critical charge required to upset a bit cell in SRAM, hence lesser the value of Q_{crit}, more the SRAM becomes vulnerable to soft error. The figure 1.3 clearly shows that with the reduction of supply voltage the probability of soft error increases rapidly with the decrease of Q_{crit}. Temperature is the parameter which least affect the Q_{crit} of SRAM. But still SER of SRAM may be impacted by temperature in many different ways. Temperature variation may alter the characteristics of the memory cells in terms of write speed and strength of the restoring device which in turn may affect the SER of SRAM [13].

The objective of this thesis work is to design a system level architecture in the form of a digital IP that will provide a reliable failsafe solution to the system memory by predicting the memory failure situation well before the time, using the process, voltage and temperature (PVT) variation data.

1.3 Organization of the Thesis

This thesis has been organized as follows:

Chapter 2 discusses about the related work and illustrates the proposed model. Chapter 3 provides the detailed description of the designed IP, Reliability Aware Intelligent Memory Management (RAIMM), its scope, functionality and design limitation. Chapter 4 explains the methodologies used to verify the IP by integrating it in a system level platform. Chapter 5 illustrates the simulation results proving the functionality correctness of the RAIMM IP for different test cases and analyzes its performance. Finally, Chapter 6 concludes this thesis by summarizing the contribution of the work and research done, along with the possible extension of our work that can be explored.

Chapter 2: Related Work and Our Proposal

2.1 Related Work

System reliability problem has been addressed at several fronts by the researchers and industry. This thesis work focuses on those techniques which are related to reliability issues of system memories.

The test and repair of SoCs, and more specifically of their memories, has been extensively explored. Blaum *et al.* explain many on-chip memories use single-error-correction-double-error-detection (SEC-DED) codes to provide reliability information to the systems, while studying the lifetimes of computer memories which are protected with (SEC-DED) codes in their research work [14]. Recent studies show that different program behaviour patterns can be identified, and can be used to generate various custom error correction mechanisms for different memory portions [15]. As discussed by Ohtani *et al.* in the year 2001 and Choi *et al.* in the year 2003, the recent memory architectures now integrate built-in-self-test logic and spare storage resources to achieve an optimal combination of yield and reliability for embedded memory cores [16] [17]. There has been some research assuming that the memories have error detection capabilities and a backup redundant memory is deployed to possibly use in case of error detection [18]. Various methods and metrics have been explored to estimate the failure rate with particular attention to energy efficiency, computation performance and area trade-offs.

Current memories already include extensive mechanisms to tolerate single-bit errors. The widely used error-correcting codes (ECC) such as parity-based protection and SEC-DED have only limited capability in error detection and correction with very high overhead in area, power and delay [20]. Built-in-self-test for memories and N modular redundancy (NMR) scheme for computing logic are too expensive, hence constraining the limited cost budget. All these researched techniques have their own limitations in terms of trade-off between energy efficiency, performance and area, thereby are unable to provide a total fail safe solution for maintaining system memory reliability. Therefore, with the increasing uncertainty of device operation, an effective system-level support to memory fault tolerance has gradually become the prime necessity to ensure proper functionality at a reasonable cost.

Two similar works have been done by Bathen *et al.* [19] and Wei Zhang *et al.* [20]. Bathen has introduced the concept of E-RoC (Embedded Redundant Array of Inexpensive Disk on-Chip); a distributed dynamically managed reliable memory subsystem. Wei Zhang has used ICR (In Cache Replication) for enhancing data cache reliability.

Based on these available techniques, an approach is presented for developing a system level framework that will ensure memory reliability. The system level solution has been named as RAIMM (Reliability Aware Intelligent Memory Management).

RAIMM utilizes the concept of redundant memories and replication of data from less reliable memory to the reliable memory to provide a better fault tolerant solution.

2.2 Our Proposal

This thesis work proposes an architectural framework to dynamically compute reliability of the on-chip memories and provide a more reliable solution for the safety critical applications in case of any memory failure. The model has the capability to predict the memory failure because of reliability issues well before the time and takes the corrective action to provide the application an almost fail safe situation. The proposed design is named as Reliability Aware Intelligent Memory Management (RAIMM).

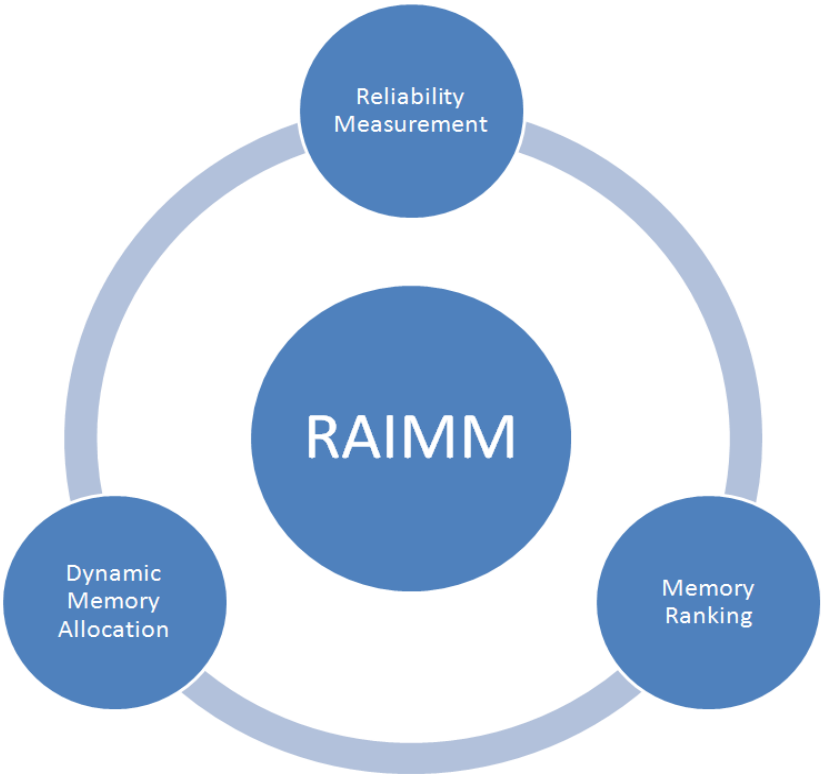


Figure 2.1: Chart reflecting the idea behind Reliability Aware Intelligent Memory Management (RAIMM)

The three major phases composing RAIMM concept as shown in the figure 2.1 have been explained in the following subsections.

2.2.1 Reliability Measurement

Operating conditions of embedded memories determine its performance efficiencies as well as soft-error rates. Memory performances are varied by process variations (intra-die), temperature variations and voltage variations. For implementing the RAIMM concept, the whole die area has been partitioned into several voltage islands. The large SRAM system memory is divided into smaller blocks and is distributed in different partitioned regions. Few redundant memory blocks of size equal to the split

system memory block are also placed in some of the regions. Reliability Monitors (PVT Sensors) are placed in each region along with the memory blocks. Process monitors provide the intra-die V_t variation data while the temperature sensors provide the information regarding local temperature variation due to high switching. Voltage sensors continuously inform the local operating voltage of the corresponding voltage island. RAIMM module collects all these PVT data and dynamically computes reliability statistics of system memory blocks using predefined reference PVT values and a comparison reliability model.

2.2.2 Memory Ranking

A central memory ranking unit accumulates individual reliability information for all memories and prepares a ranking table of the memories based on the reliability model. It also takes inputs from an on-chip profiler which records the memory accesses dynamically and thus helps providing the information regarding the usability of a particular memory area. If any critical data is present in a memory area, system can also provide explicit inputs to the ranking module to elevate its ranking. The ranking table of the memories is based on the reliability level (computed from the reliability monitor data), its usage rate (as is provided by the on-chip profiler) and the critical nature of the contained data (as is provided by application). For example, if two memory cuts are working with same reliability levels, it is possible to rank an instance higher if it contains critical data. Thus in case of alarming reliability level, the memory containing critical data would be preferred candidate for remapping than other.

2.2.3 Dynamic Memory Allocation

The memory manager module always maintains the list of usable memories and redundant memories along-with their reliability levels. As soon as the reliability level of a usable memory goes down it presents itself a candidate for remapping. Remapping of any memory is done with a reliable redundant memory and data transfer is done using a DMA controller. Memory addresses of various cuts are already programmed in the RAIMM logic and are updated dynamically to take care of the remapping effect. With the DMA mechanism the block of data is being moved without affecting the system operation. DMA is programmed automatically by the RAIMM logic.

It is also possible to raise various system level interrupts along-with different stages of the monitoring and remapping. These interrupts may provide a very informative method to manage dynamic voltage and frequency scaling (DVFS).

2.2.4 System Level Architecture

As any large RAM is implemented using smaller cuts of memories which are then placed at different locations in a SoC, it is proposed to implement few redundant memory banks in addition to the existing memory instances. With the memories being interconnected to the processors using bus communication infrastructure, these can be placed at different locations in the chip thus experiencing different operating conditions. This helps in mitigating the risks of experiencing worst case conditions for

the whole memories, and provides us a bank of memories which can operate always in a reliable operating condition. Figure 2.2 shows Processor and system memory connectivity while figure 2.3 shows the processor and memory connectivity with system level architecture of RAIMM.

Figure 2.2: Processor and system memory connection

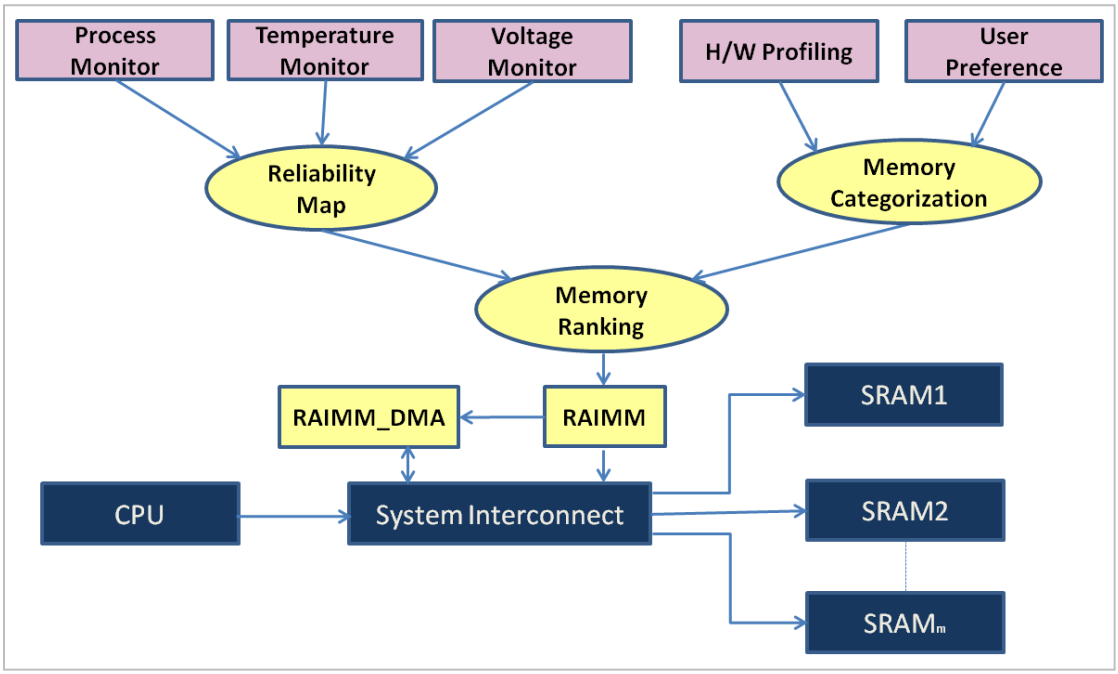


Figure 2.3: Processor and system memory connection with system level architecture of RAIMM

Chapter 3: RAIMM Description

3.1 Introduction

The RAIMM (Reliability Aware Intelligent Memory Management) is an intelligent memory manager implemented as a digital IP. This IP has been designed to provide fail safe solution to safety critical systems by maintaining the reliability of the on-chip cache memory. This method uses concepts of memory redundancy and replication of data from less reliable memory block to the reliable memory block to ensure the reliability of system memory.

Following are the important features of RAIMM:

- RAIMM is implemented as a highly configurable generic IP providing support for different reliability levels as well as number of memory modules
- It provides three, five and seven stages reliability check depending on the safety criticalness of the system. The IP has been verified for three stage reliability check: Reliable , Less Reliable and Unreliable
- Maximum sixteen memory blocks can be monitored simultaneously. The IP has been verified for six memory blocks among which four memory blocks are usable and two are redundant
- Continuous monitoring of reliability of the system memory on the basis of process, voltage, temperature (PVT) variation
- Memory profiling based on the accesses to memory
- Ranking of memories according to reliability status based on PVT condition and memory profiling
- Remapping of unreliable memory data to redundant reliable memory
- System level interrupts for unreliable memory
- Dynamically change memory addresses once data remapping done
- User has a flexibility to control interrupt generation

3.2 RAIMM Functional Flowchart

The flowchart shown in figure 3.1 demonstrates the basic steps through which the RAIMM generates system level interrupt when there arises a memory less reliable or unreliable condition and the steps for memory data transfer from the unreliable memory to the reliable memory block. This also shows the dynamic remapping of memory addresses occur at the Bus Matrix interconnect level after the memory data transfer completed by the RAIMM_DMA[#].

[#] Refer section 3.3.4 for the brief description regarding RAIMM_DMA

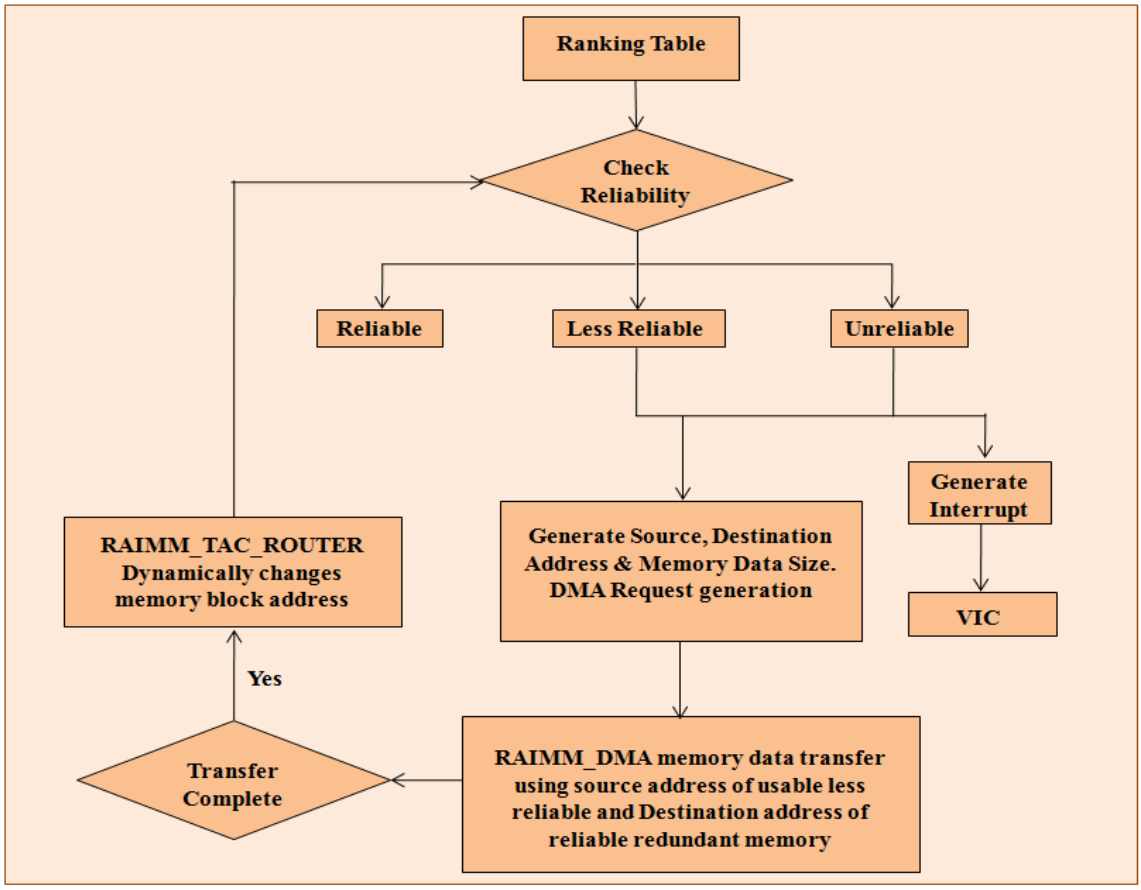


Figure 3.1: RAIMM functional flowchart

3.3 RAIMM Functional Description

The function of memory manager is to continuously update the reliability status of system memory blocks according to the PVT Variations and it provides a failsafe solution in case of memory is less reliable or unreliable. As discussed in section 2.2.1, the divided small system memory blocks along with few redundant memory blocks have been distributed in different die regions. To detect the process, voltage and temperature variations in system memory, PVT sensors are placed with the system memory blocks. These sensors continuously sense the PVT variations of the system memory and provide sensed data to memory manager. Using these PVT variation data RAIMM computes the reliability status of memory blocks. Memory profiling data has been generated depending upon the number of accesses to the same memory block. RAIMM prepares a ranking table considering the memory reliability status data and the memory profiling data. If memory manager detects any usable system memory block in less reliable or unreliable state then it provides the system level solution by generating a system level interrupt and then remapping the data from less reliable or unreliable memory to a reliable redundant memory. The functional block diagram of RAIMM IP has been shown in figure 3.2.

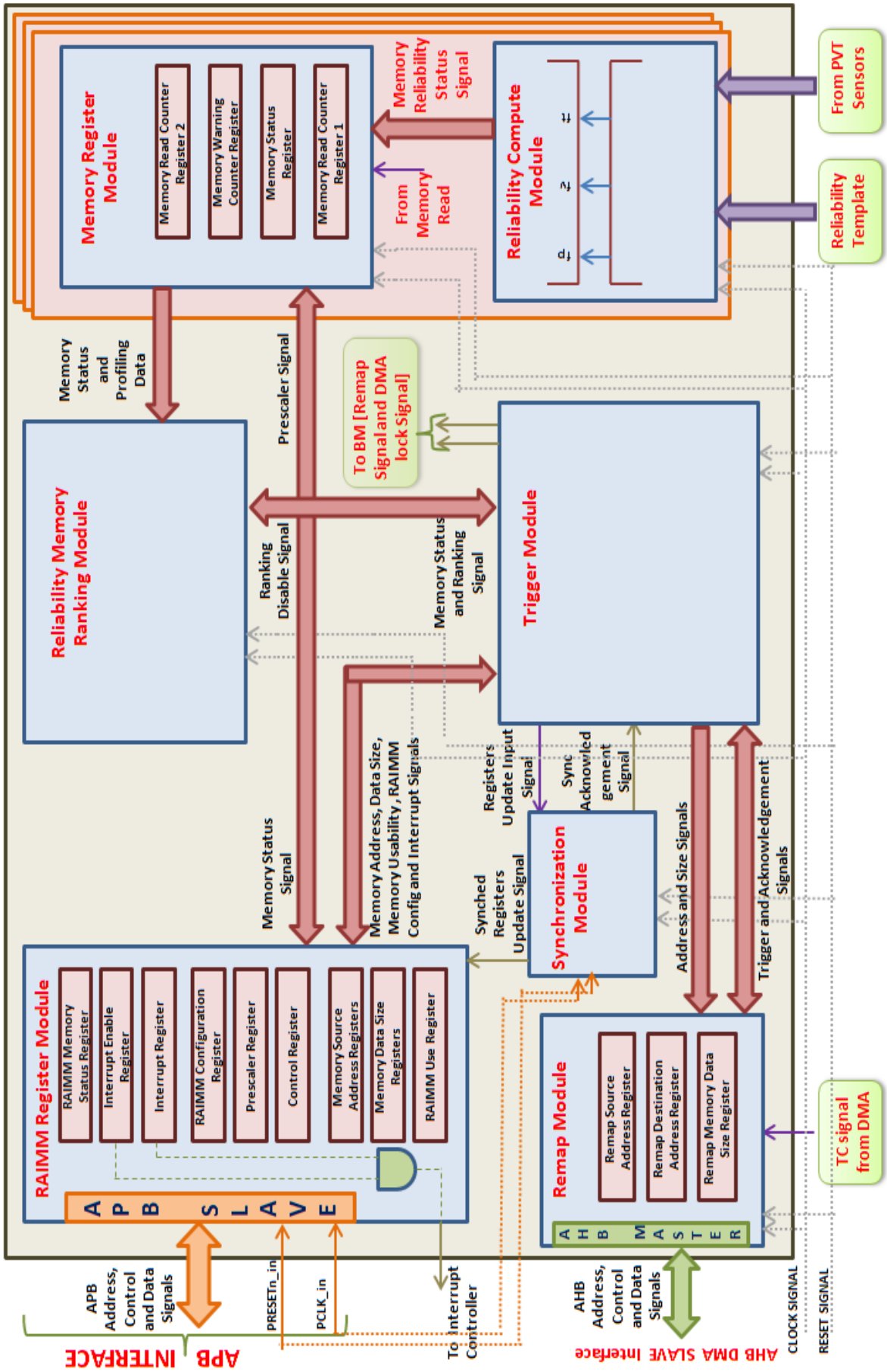


Figure 3.2: Functional Block Diagram of RAIMM

The whole memory management is performed with the help of following logic blocks:

- RAIMM Module
    - Reliability Compute Module
    - Memory Register Module
    - Memory Ranking Module
    - RAIMM Register Module
    - Trigger Module
    - Synchronization Module
    - Remap Module
- Reliability Template
- PVT Sensors
- DMA Module
- AHB Bus Matrix Module

The following subsections will illustrate the details of all the modules involved in the reliability aware memory management operation.

3.3.1 RAIMM Module

Reliability Aware Intelligent Memory Manager (RAIMM) is the heart of the system, designed as a digital IP block to supervise the reliability status of the system memory and to raise an alarm when any degradation in reliability of memory would be suspected. It provides the address and data size information regarding data transfer to the RAIMM_DMA module. It also remaps the memory blocks after the data has been transferred successfully from the less reliable or unreliable usable memory to the redundant reliable memory. RAIMM module is divided into the following seven sub modules.

3.3.1.1 Reliability Compute Module

The function of reliability compute module is to continuously check the reliability conditions of system memory block with respect to PVT variations and dynamically updates the reliability status of the memory block. Predefined PVT reference values of the memory considering all corner cases for a particular technology have been provided to the reliability compute module by the “Reliability Template”. The PVT sensors which are placed with the system memory blocks in different voltage islands, continuously (during run time) provide the PVT data to reliability compute module. Using comparison logic between sensors data and predefine reference data, reliability compute module decides the final reliability status of system memory block. There would be separate “Reliability Compute Module” for individual memory block. As here

for the verification of RAIMM; six memory blocks have been considered, so the top level RAIMM block contains six reliability compute modules.

Reliability compute module has a two stage computation method to predict the final reliability status of the system memory block as shown in the figure 3.3.

- In first stage, the module checks the reliability status of the memory blocks according to individual process , voltage and temperature variation as discussed in sub-sections (A)
- In second stage, the module use all the reliability status of the memory block depending upon individual parameters collected from the first stage and decides the final memory reliability status according to the defined reliability rules as discussed in sub-section (B)

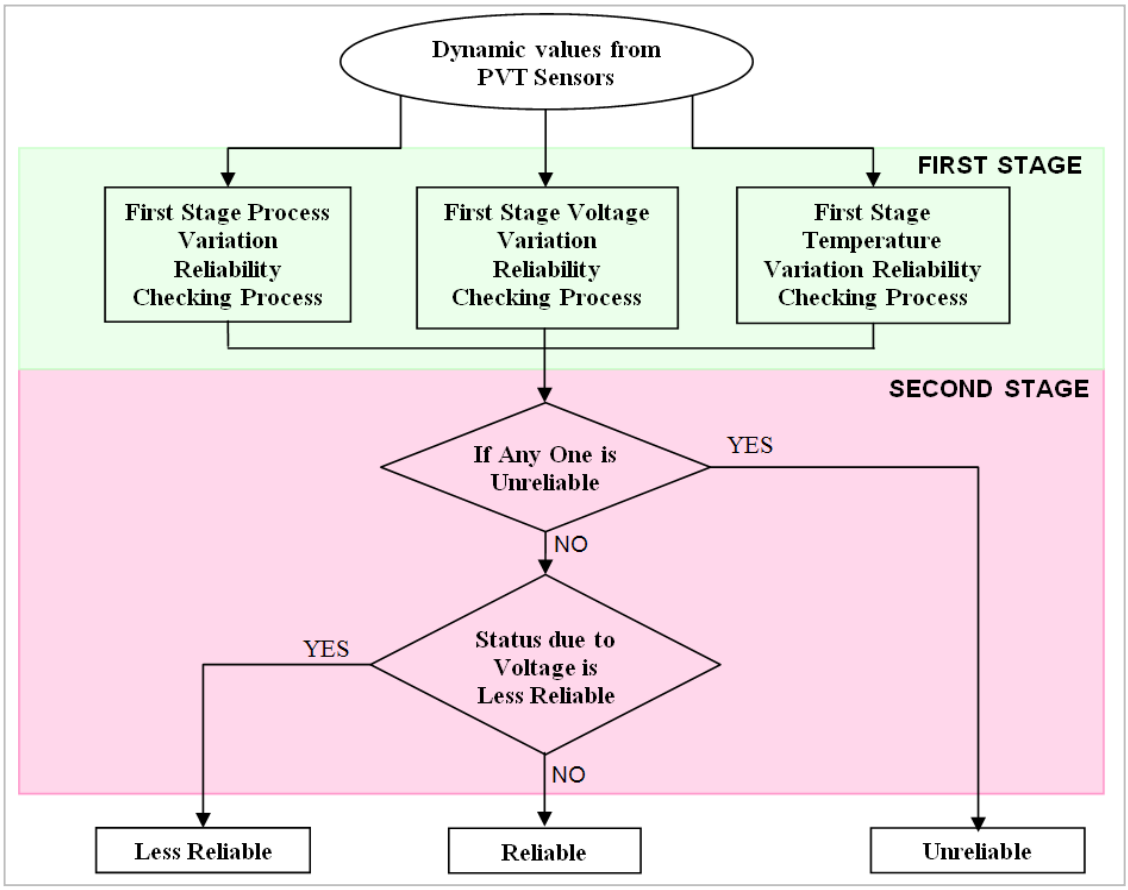


Figure 3.3: Flow Chart of the operation of Reliability Compute Module for three reliability stages

(A) First stage PVT variation reliability checking process:

This stage decides the reliability of system memory block with respect to process, voltage and temperature variation individually. Different corner cases have been considered in the Monte Carlo simulation of the memory to find out three possible outcomes in the form of color code as R - red, B - blue and G – green as explained under the sub section “Reliability Template” in section 3.3.2. Predefine reference

values, extracted from these Monte Carlo simulations are collected from the reliability template.

Reliability status of a particular system memory block depending upon the dynamically collected process, voltage and temperature values from the PVT sensors can be calculated by comparing the PVT values with the predefined reference values from reliability template. If the P or V or T value lies in the range having the color code "Green" then the memory block is "Reliable" for that particular parameter. Similarly color code "Blue" represents "Less Reliable" and "Red" represents "Unreliable".

(B) Second stage system memory final reliability status checking process:

This reliability calculation stage declares the final reliability status of the system memory block depending upon the reliability status of memory block according to individual PVT parameters collected from the first stage and based on the defined reliability rules.

Reliability Rules can be defined earlier and supplied to the RAIMM block from an external text file. The rules may be dependent on the technology and the safety criticalness of the application. This should be accessible to the user so that can be modified as per requirement without modifying the RAIMM IP.

Though the designed RAIMM IP is a generic one and suitable for three, five and seven reliability stages, the verification of the IP has been done for three reliability stages. So for the verification case, reliability compute module can assign any particular reliability status to the corresponding memory block among the following three.

- Reliable
- Less Reliable
- Unreliable

The considered Reliability Rules[#] for the verification of RAIMM is as stated below:

- If the status of memory block is "Unreliable" for any one of the parameters among P or V or T from the first stage then the final reliability status => Unreliable
- If the status of memory block is "Less Reliable" for voltage from the first stage then the final reliability status => Less Reliable
- If the status of memory block for voltage is "Reliable" and for process and/or temperature is "Less Reliable" then the final reliability status => Reliable
- If the status of memory block for all the PVT parameters are "Reliable" then the final reliability status => Reliable

[#] The Reliability Rules have been provided by the Memory Group of ST Microelectronics, Greater Noida

3.3.1.2 Memory Register Module

The memory register modules contain registers as mentioned in table 3.1 for the system memory blocks. All these registers are of 32 bits and can only be accessed by the RAIMM internal hardware and are local to the IP. This module is responsible for generating the memory profiling data depending upon the number of memory READ accesses. The module stores the memory reliability status information generated from the corresponding memory reliability compute module based on the PVT data. Memory register module transfers the memory reliability status and memory profiling information to the memory ranking module. There would be separate "Memory Register Module" for individual memory block. As here for the verification of RAIMM; six memory blocks have been considered, so the top level RAIMM block contains six memory register modules.

Sl. No.	Register Name	Description
1	Memory Status Register	Stores the memory reliability status of the corresponding memory block
2	Memory Read Count Registers	Two such registers. First one counts the number of times the corresponding memory is getting READ access. As soon as the count equals the prescaler value set by the processor in prescaler register of RAIMM register module, the second read counter register will be incremented by 1 and the first read counter register will be cleared.
3	Memory Warning Count Register	Stores the number of times the corresponding memory is getting accessed after the memory block becomes less reliable. This register is meant for showing a warning signal that memory is getting accessed even after it gets less reliable. For the current design this register has not been used.

Table 3.1: List of Registers in Memory Register Module

3.3.1.3 Memory Ranking Module

The memory ranking module ranks all the memory blocks based on the memory reliability status and memory profiling data coming from all the memory register modules as discussed in section 2.2.2. Higher the ranking of the memory block, higher the possibility of considering the memory block as the preferred candidate for remapping first. At the time of generating the ranking table of the memory blocks, the memory ranking module considers the reliability status of the memory blocks first. The memory block having the reliability status "Less Reliable" will have higher rank than a memory block having the status "Reliable". Similarly the memory block having the status "Unreliable" will climb to a higher position in the ranking ladder than the memory having the status "Less Reliable". If in case two or more memory blocks will have same reliability status, the memory profiling data would be considered for the

ranking of those memory blocks. In this case the memory block getting accessed the most, i.e., having the higher profiling would be considered as of higher usability and given the higher rank. Critical nature of data stored (as is provided by application) can also be taken as another controlling parameter for the generation of ranking of memory though the same has not been considered in the currently designed RAIMM.

3.3.1.4 RAIMM Register Module

RAIMM register module being the part of the RAIMM acts as the interface between the processor and the RAIMM module. Other than most of the global registers of the RAIMM, this module contains the APB[#] (Advanced Peripheral Bus) slave port through which the processor can configure the RAIMM IP. RAIMM register module works with APB clock unlike all other sub-modules of RAIMM which work with AHB[#] (Advanced High-performance Bus) clock. Through this module processor can enable or disable the IP, enable or disable the interrupts, set the prescaler value and the initial address, data size and usability status of the system memory blocks. This block stores the reliability status of all the memory blocks and sets or resets the corresponding interrupt bit depending upon the reliability status. This block has the capability to generate the system interrupt when there is any alarming situation because of the reliability issue of the memory blocks. The global registers accommodated by the RAIMM Register Module have been listed in table 3.2. Processor core has the ability to access these registers.

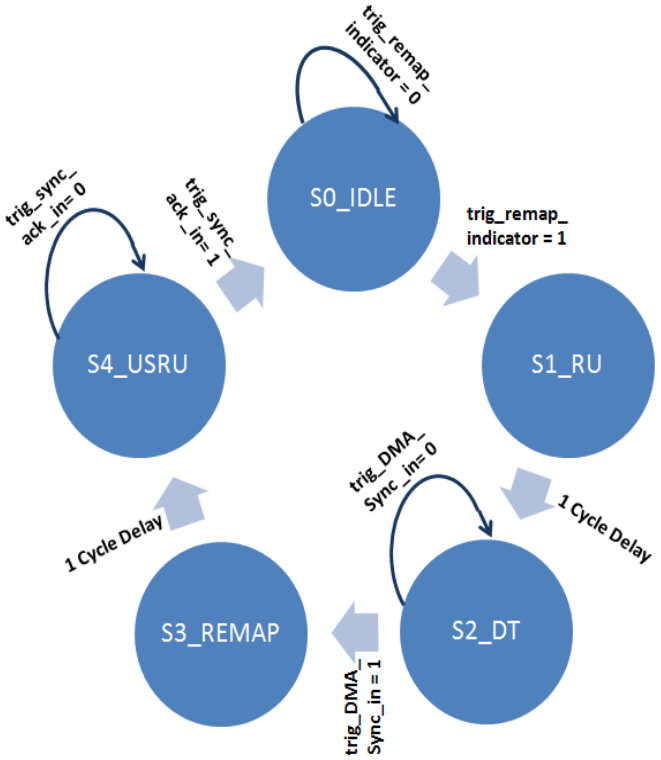
Sl. No.	Offset Address	Register Name	Processor Accessibility	Description
1	0x000	RAIMM Configuration Register (RAIMM_ConfgR)	R/W	On Page 22
2	0x004	Prescaler Register (PR)	R/W	On Page 22
3	0x008	Control Register (CR)	R/W	On Page 22
4	0x00C	Interrupt Enable Register (IER)	R/W	On Page 22
5	0x010	RAIMM Memory Status Register (RAIMM_MSR)	R	On Page 23
6	0x020	Interrupt Register (IR)	R/W	On Page 23
7	0x100	Memory Source Address Registers (M_SARs)	R/W	On Page 24
8	0x200	Memory Data Size Registers (M_DSRs)	R/W	On Page 24
9	0x300	RAIMM Use Register (RAIMM_UR)	R/W	On Page 25
For the detail descriptions of the registers refer the section 3.4 "Programmer's Model"				

Table 3.2: RAIMM Global Registers Memory Map

[#] Refer section 4.4 for the brief description regarding APB and AHB.

3.3.1.5 Trigger Module

Trigger module is the heart of the RAIMM IP. This is the module that generates the trigger signal that indicates the RAIMM_DMA module to start the data transfer from a less reliable or unreliable usable memory to a redundant reliable memory. This module continuously collects the memory ranking and memory reliability status information from the memory ranking module. If it finds a usable memory having the highest rank in the ranking table and its reliability status as less reliable or unreliable then it checks for the availability of the redundant reliable memory block. If the same is available then it generates the trigger for the start of the data transfer phase. If it does not find a redundant reliable memory block during an alarming situation, then it generates an interrupt. The trigger module works as a state machine having five states represented in figure 3.4.



S0_IDLE: Idle state

S1_RU: Source, Destination, Memsize Registers Update

S2_DT: Data Transfer from usable less reliable or unreliable memory to redundant reliable memory

S3_REMAP: Address Remap

S4_USRU: RAIMM Use & Size Registers Update

Figure 3.4: State Machine implemented in Trigger Module

During the data transfer phase trigger module generates a LOCK signal and provides it to the bus matrix so that the processor can be halted if it is accessing the memory block which is involved in the data transfer. After the DMA memory to memory data transfer has been completed, trigger module generates the REMAP signal and supplies it to the bus matrix so that the dynamic address remapping can be possible. After the generation of REMAP signal, it updates the RAIMM_UR, M_SARs and M_DSRs in RAIMM register module according to the remapping of the memory blocks.

3.3.1.6 Synchronization Module

The RAIMM register sub-module works with APB Clock and all the other sub-modules of RAIMM work with AHB clock. Trigger Module interacts with the RAIMM register module for the operation of the RAIMM block. APB clock is slower than the AHB one; here in the implemented design it is half of the AHB clock. Use of two clocks with two different frequencies demands the need of synchronization among both the clock signals for the proper operation of the device. Hence the synchronization module synchronizes the signal getting interacted between the RAIMM Register Module and the Trigger Module.

3.3.1.7 Remap Module

The Remap Module is the interfacing sub-module between the RAIMM and the RAIMM_DMA module. This is the module having a AHB master port and responsible in configuring the RAIMM_DMA for memory data transfer. The DMA is an AHB device and is configured by the remap module through the AHB interface. Remap module provides the source address, destination address and the transferable data size information to the RAIMM_DMA. Table 3.3 describes the registers in remap module. These registers are local to the RAIMM module and can be accessed only by the internal hardware of the IP.

Sl. No.	Register Name	Description
1	Remap Source Address Register	Stores the starting address of highest ranked, usable and less-reliable or unreliable memory block
2	Remap Destination Address Register	Stores the starting address of available reliable redundant memory block
3	Remap Memory Data Size Register	Store the transfer size of valid data in the memory block, whose source address is there in Remap Source Address Register

Table 3.3: List of Registers in Remap Module

3.3.2 Reliability Template

Reliability Compute Module collects the dynamic process, voltage and temperature values from the PVT sensors and compares them with the predefine PVT reference values to find out the reliability condition of the memory blocks as discussed in section 3.3.1.1. The reference PVT values are taken from the Monte-Carlo simulations, but these values are different for different technologies (i.e. if user wants to move from one technology to other then user should have to make changes in design). To eliminate this issue, RAIMM provides the flexibility of just making changes in only one text file according to requirement without making any changes in the design.

Reliability Template[#] is a text file that contains PVT reference values depending upon the number of reliability stages and for a particular technology. It provides this information to the reliability compute module during reliability checking process. In the following sub-sections (A), (B) and (C), different color codes represent the voltage,

[#] The reference process, voltage and temperature values achieved from Monte Carlo simulations have been provided by the Memory Group of ST Microelectronics, Greater Noida

process and temperature range for different reliability status as R – red for unreliability, B - blue for less reliability and G – green for reliability.

(A) Predefined voltage reference values:

Predefined voltage values have been decided for the reliability of system memory block with respect to voltage variation. Different voltage corner cases have been considered in the Monte Carlo simulation of the memory for the 40nm technology to find out three possible outcomes. Predefine reference values are extracted from this Monte Carlo simulation and it changes according to the change in technology.

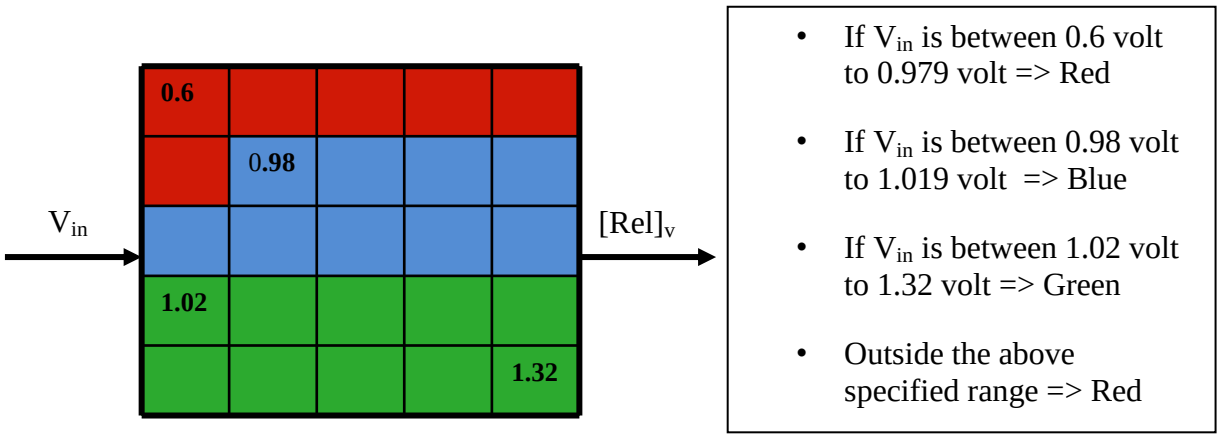


Figure 3.5: Reliability matrix according to the voltage variation

(B) Predefined process reference values:

Predefined process values for the reliability of system memory block with respect to intra die process variation have been defined. Monte Carlo 6σ – process variation simulations have been run at different corner cases for the 40nm technology memory. It is divided into two stages:

- First stage decides the reliability for separate nMOS and pMOS process variation
- Second stage decides the final reliability for process variation using the first stage outputs combinations

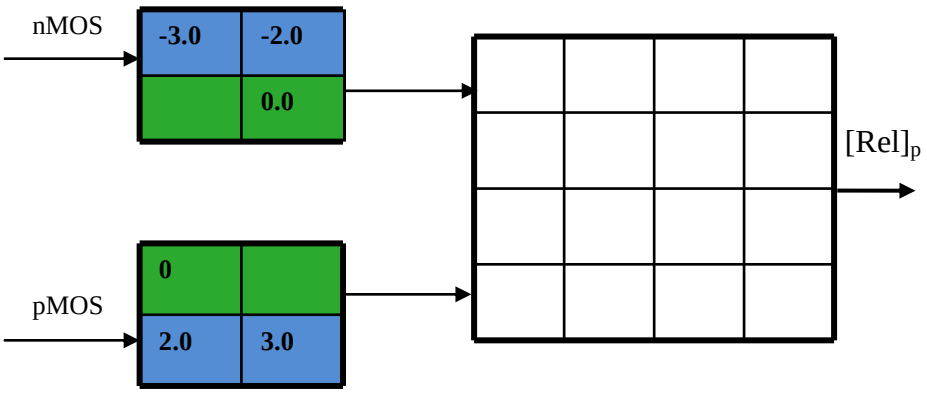


Figure 3.6: Reliability matrix according to the intra-die process variation

Rules for First Stage:

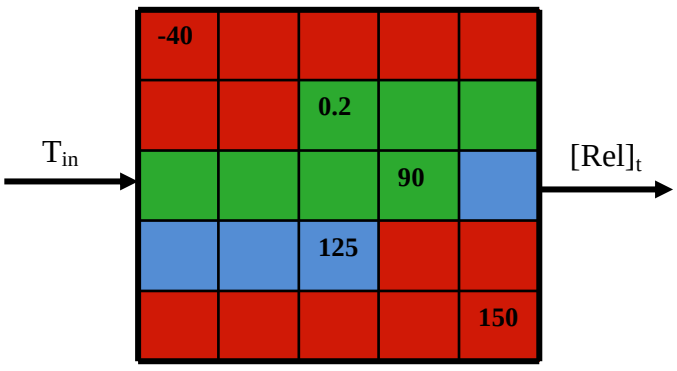
- If nMOS is between -3.0 to -2.0 => Blue
- If pMOS is between 2.0 to 3.0 => Blue
- If nMOS is between -2.0 to 0.0 => Green
- If pMOS is between 0.0 to 2.0 => Green
- If outside above range then => Red

Rules for Second Stage:

- If first stage for anyone of nMOS and pMOS is red then output => Red
- If first stage for anyone of nMOS and pMOS is blue while none of them is red then output => Blue
- If both of them are green then output => Green

(C) Predefined temperature reference values:

Predefined temperature values for the reliability of system memory block with respect to temperature variation have been decided. Monte Carlo simulations have been run for all corner cases between -40ºC to 150ºC for the 40nm technology memory.



- If T_{in} is between (-40 to 0.1)ºC or (125 to 150)ºC => Red
- If T_{in} is between (90 to 125)ºC => Blue
- If T_{in} is between (0.2 to 90)ºC => Green
- Outside the above specified range => Red

Figure 3.7: Reliability matrix according to the temperature variation

3.3.3 PVT Sensors

The whole die is divided into number of voltage islands and the split system memory blocks are placed in different partitioned die area. PVT sensors placed close to the system memory blocks in all the voltage islands, keep on sensing the memory operating conditions and process variations. These PVT sensors are responsible for providing the continuous and dynamic process, voltage and temperature values to the RAIMM module for finding out the reliability status of the corresponding memory block.

3.3.4 DMA Module (RAIMM_DMA)

RAIMM_DMA is a simple DMA controller performs only memory to memory data transfer operation in this scenario. The DMA module used in the design is a pre-verified ARM DMA Controller (PL080) IP [21]. RAIMM module interacts with the RAIMM_DMA through an AHB interface to configure and control it. RAIMM enables or disables the DMA. RAIMM remap module provides the source address, destination address, transfer size and finally configures the DMA channel control register and channel configuration register. The data width size and transfer burst size are set in the channel control register. Here in the design the data width has been fixed as 32 bits and the burst size has been set as 4. Once the RAIMM_DMA channel is configured and enabled, DMA starts data transfer operation from source address to destination address. When the memory data transfer is completed it sends a TRANSFER_COMPLETE sync signal to RAIMM module to indicate that the transfer has been completed.

3.3.5 AHB Bus Matrix Module

AHB Bus Matrix (BM) Module is another ARM provided and pre-verified IP used as the communication infrastructure in this design platform. Section 4.4 presents a brief description on AHB bus matrix. All the address decoding and arbitration schemes are implemented in this IP. Memory blocks can be accessed by the processor core through this BM module. Other than the processor, the RAIMM_DMA module uses this BM to perform the memory to memory data transfer during the memory reliability alarming situation. The BM can support maximum up-to 16 slave device connection.

After the memory data transfer through the RAIMM_DMA, it is required to remap the memory addresses so that the processor can access to the new memory block with the old address. This dynamic changing of memory address job is done through the BM at the address decoding level. The information which leads the BM to remap the address is generated by the RAIMM module.

3.4 Programmer's Model

This section describes the details of the global registers used in the RAIMM module, which can be programmed by the processor core. The following applies to the registers used in the RAIMM:

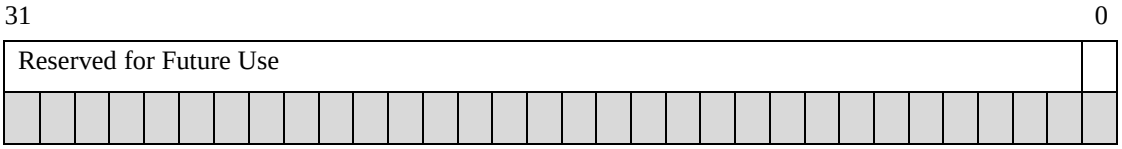
- All the registers used are of 32 bits.
- Reserved or unused bits of registers should be written as zero, and should be ignored on read unless otherwise stated in the relevant text.
- A system or power-on reset, resets all registers bits to logic 0 unless otherwise stated in the relevant text.
- A write updates the contents of a register and a read returns the contents of the register. The registers defined in this document can only be accessed using word reads and word writes, unless otherwise stated in the relevant text.

3.4.1 RAIMM Global registers description

Following sub-sections describe the details of all the global registers including the field descriptions and the processor access capability.

3.4.1.1 RAIMM Configuration Register

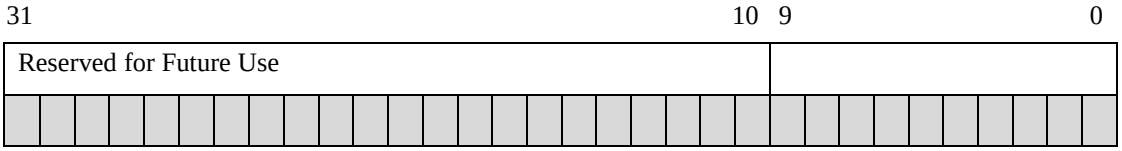
This register enables or disables the RAIMM unit. Processor has both the READ and WRITE access to this register. This register can be WRITTEN by the processor only.



- Bit [0] : 1/0 – Enables/Disables RAIMM memory manager

3.4.1.2 Prescaler Register

This register holds a value preset by the processor which controls memory read access counter overflow. Processor has both the READ and WRITE access to this register. This register can be WRITTEN by the processor only.



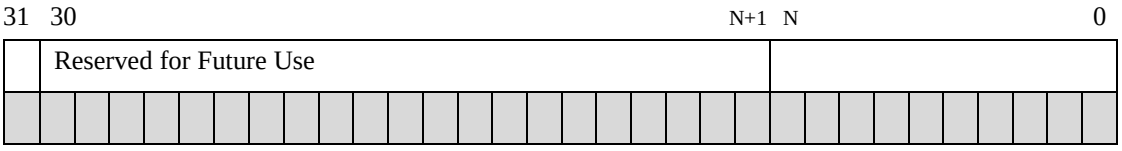
- Bit [9:0] : User Programmable, User can fix Prescaler value.

3.4.1.3 Control Register

This register is meant for configuring control bits of any interfacing IP connected to the RAIMM. This can be WRITTEN by the processor core or by the internal circuitry of the RAIMM hardware. Currently this register is not in use.

3.4.1.4 Interrupt Enable Register

This register decides whether to enable or disable the interrupt bits in interrupt register. Processor has both the READ and WRITE access to this register. This register can be WRITTEN by the processor only.



'N' is the number of Memory Blocks present.

- Bit [N:0] : any bit 1/0 – Enables/Disables corresponding interrupt bit of the interrupt register

- Bit [31] : 1/0 – Enables/Disables the interrupt generated by the system, because of the unavailability of any reliable redundant memory while any usable memory will go less reliable or unreliable

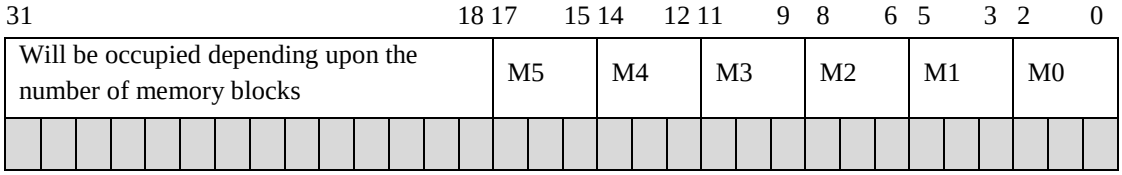
3.4.1.5 RAIMM Memory Status Register

This register stores the reliability status of each memory block. The IP has been designed for maximum 7 reliability stages so it requires maximum 3 bits to represent the reliability status of any memory block. This design can support up to maximum of 16 memory blocks (since AHB BM can accommodate maximum up to 16 slaves), so when there will be more than 10 memory blocks, we will need two Status Registers of 32 bit each within the RAIMM module and in case of less than or equal to 10 memory blocks there will be only one Status Register. This register can only be READ by the processor. This register can be WRITTEN by the internal hardware only.

Reliability status description for 3, 5 and 7 reliability state conditions are as mentioned in the following table 3.4:

Bit[2:0]	**Reliability Status Description of Memory Blocks**		
	3 states	**5 states**	**7 states**
3'b000	Reliable	Reliable	Reliable
3'b001	Less Reliable	Almost Reliable	Almost Reliable
3'b010	Unreliable	Less Reliable	Towards Reliable
3'b011		Almost Unreliable	Less Reliable
3'b100		Unreliable	Towards Unreliable
3'b101			Almost Unreliable
3'b110			Unreliable

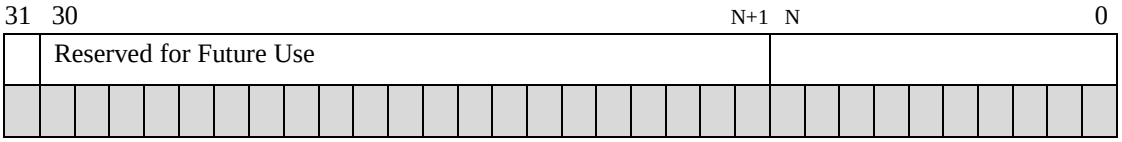
Table 3.4: Reliability Status Description of Memory Blocks



- Bit [2:0] : Reliability Status of Memory block zero
- Bit [5:3] : Reliability Status of Memory block one
- Bit [8:6] : Reliability Status of Memory block two. And so on.

3.4.1.6 Interrupt Register

This register set the interrupt bit for the corresponding memory block which becomes less reliable or beyond. This register can be WRITTEN by the internal hardware, which always keeps eye on the reliability status of the memory blocks and it immediately update the interrupt bit in the interrupt register as soon as the corresponding memory block goes less reliable or beyond. Eg. If M0 becomes less reliable then bit[0] = 1'b1. This register can be both READ and WRITE by the processor. The processor can only clear the interrupt register through WRITE operation but can't set any bit.

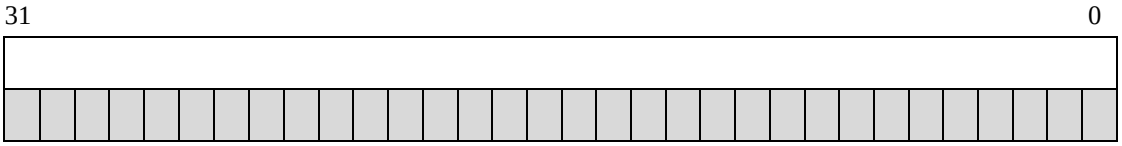


'N' is the number of Memory Blocks present.

- Bit [N:0] : any bit 1 – Means the memory block corresponding to the bit goes less reliable or beyond.
- Bit [N:0] : any bit 0 – Means the memory block corresponding to the bit is still reliable or below less reliable state.
- Bit [31] : 1 – Interrupt generated by the system, because of the unavailability of any reliable redundant memory while any usable memory will go less reliable
- Bit [31] : 0 – No interrupts as no situation as above

3.4.1.7 Memory Source Address Registers

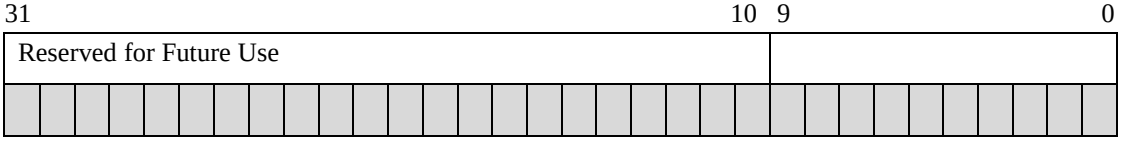
There are 'N' such registers where there are 'N' memory blocks in the system. These contain the initial address information of each memory. Processor has both the READ and WRITE access to this register. Once the initial values are written by the processor, during the operation the memory source address registers are UPDATED (WRITTEN) by the internal hardware dynamically.



- Bit [0-31]: Store the initial address information [M0-MN]
  Where MN: N^{th} memory block.

3.4.1.8 Memory Data Size Registers

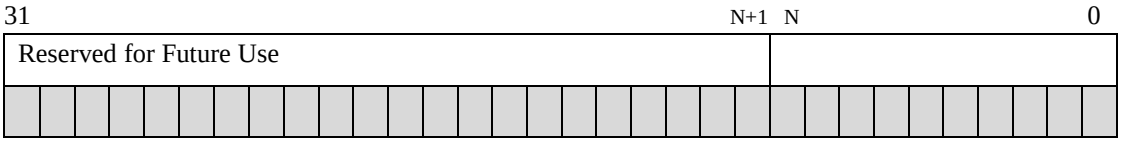
There are 'N' such registers where there are 'N' memory blocks in the system. These contain the size of the valid data present in each memory. Processor has both the READ and WRITE access to this register. Once the initial values are written by the processor, during the operation the memory size registers are UPDATED (WRITTEN) by the internal hardware dynamically.



- Bit [9:0]: To store the initial size of the valid data present in memory block [M0-MN]

3.4.1.9 RAIMM Use Register

This register decides which memory block is USABLE and which memory block is REDUNDANT. Processor has both the READ and WRITE access to this register. Once the initial values are written by the processor, during the operation the use register is UPDATED (WRITTEN) by the internal hardware dynamically.



- Bit [N:0] : any bit 1/0 – Indicates the corresponding memory block is REDUNDANT/USABLE

Chapter 4: Methodology for Implementing RAIMM

4.1 Prototyping Framework

In order to verify the designed RAIMM IP, the deployment of a virtual platform has been done. As the proof of concept had to be in simplified environment, we chose a higher abstraction level and worked with transaction level models (TLM) for interconnect and simplified and abstracted memory models. In addition, to develop the test framework, a TLM level processor model and vectored interrupt controller was incorporated in the platform. This framework is very helpful to develop initial tests in "C" and run it on the processor with all the debug capabilities. The proposed model incorporated in the virtual platform has been shown in the figure 4.1.

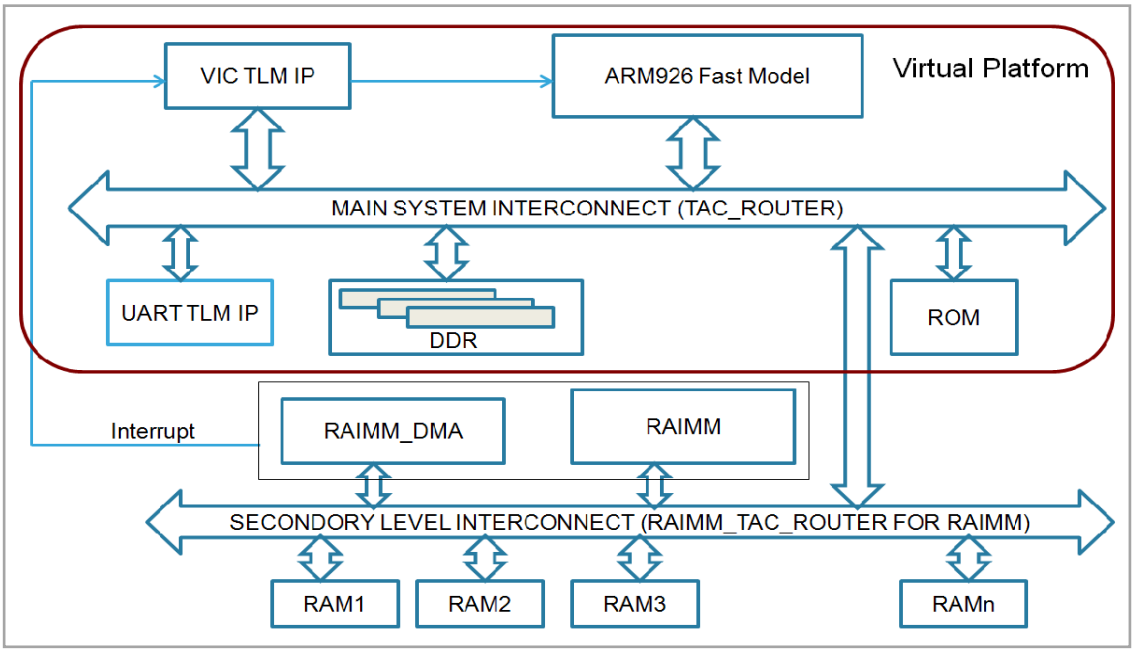


Figure 4.1: Prototyping framework - A virtual platform with RAIMM

The whole concept could have been implemented with the first level of interconnection only, as all the split memory instances and both the RAIMM as well as RAIMM_DMA would have been connected to the main system interconnect. Here a secondary level of interconnect has been introduced for ease of implementation which helped not to disturb the existing TLM level primary system interconnect which is an already verified module and interfaced with other verified modules in the system. All the memory data transfer through RAIMM_DMA and address decoding and remapping for split system memory blocks have been done at the secondary Bus Matrix level. The use of secondary interconnection also facilitates for more number of memory blocks of course at the cost of increased latency and area.

4.2 TLM Methodology

Transaction-level modeling (TLM) is an approach at a higher abstraction level for modeling digital systems. In TLM, details of communication among modules is separated from the details of the communication architecture and the implementation of functional units. In brief it separates the communication from the computation within a system [22]. Communication mechanisms such as buses are modeled as channels, and are provided to modules using SystemC interface classes. Transaction requests can be processed by calling interface functions of these channel models, which encapsulate low-level details of the information exchange. At TLM level, functionality of the data transfers is more emphasized i.e. what data is getting transferred and to which location and less emphasis on their actual implementation. This approach helps the system-level designer in many ways to experiment. This provides the platform to verify the system from the very early phase of the design that facilitates the development of the software drivers and the hardware RTL logic in parallel.

4.3 Platform and IP Integration

This proposal is basically to have an ARM based abstracted Platform. The intent of the platform is to provide a preliminary facility for the RAIMM IP verification with minimum required components which permits to run test software and permits to plug the RAIMM and RAIMM_DMA IPs.

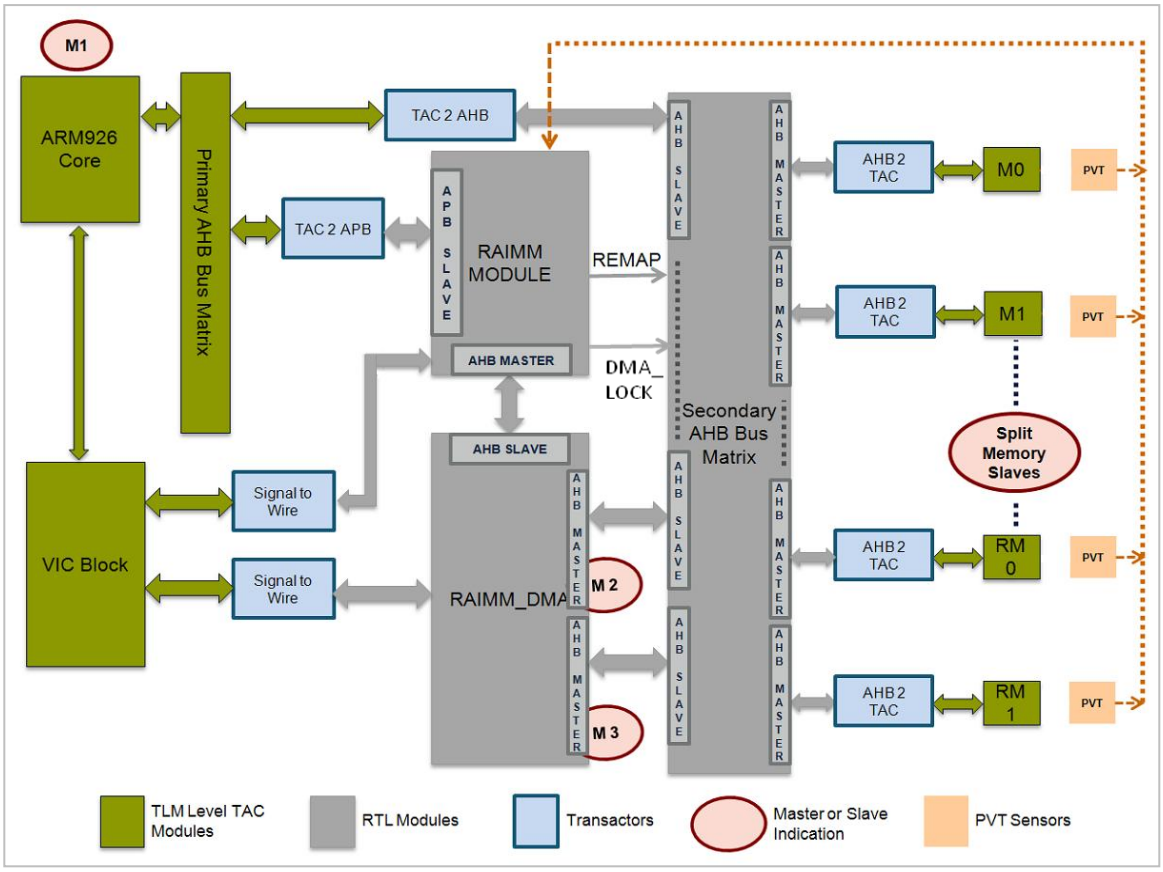


Figure 4.2: Different modules in the test frame at different abstraction levels

This System platform is a completely abstracted platform modeled in SystemC at higher abstraction level like transaction level models. The designed RAIMM IP, RAIMM_DMA and secondary AHB Bus Matrix are in RTL level as shown in figure 4.2. Figure demonstrates different modules in the system at different abstraction levels. The communication between the TLM level TAC (Transaction Accurate Channel) modules and the RTL modules can be possible by using different transactors as shown in the figure. In this scenario, the platform gives a glimpse of SoC and a complete hardware plus software prototype is available.

Such a platform could help to do early evaluation of System. This could be a suitable platform to perform hardware software partitioning and make explorations pertaining to make a better decision on what could go in software and the hardware portion of the IP. This facilitates in starting early software driver development and performs software debugging. The added advantage of platform are better simulation performance, good debugging capabilities with test benches as it permits to write C test benches.

The current platform is based on TLM TAC protocol. The available TLM IP models are ARM926, VIC, primary AHB TAC Router, memory modules. The integrated RTL modules are RAIMM, RAIMM_DMA (DMAC PL080) and the secondary AHB Bus Matrix. The platform also contains different transactors like TAC2AHB, TAC2APB, AHB2TAC and Signal to Wire to establish the communication between the TLM TAC modules and the RTL modules.

4.4 Communication Infrastructure

The AHB Bus Matrix is the interconnect backbone in the platform used. An APB interface infrastructure also has been utilized to set up the communication between the processor and the RAIMM IP.

The Advanced Peripheral Bus (APB) is optimized for reduced interface complexity and minimal power consumption. APB is one among the Advanced Microcontroller Bus Architecture (AMBA) hierarchy of buses. The AMBA APB should be used to interface to any peripherals which are low bandwidth and do not require the high performance of a pipelined bus interface [23]. Here in the design the processor configures the RAIMM IP using an APB interface.

AHB Bus Matrix is responsible for routing the transaction from one IP to other IP and IP to Memories. Implementation of AHB is simple and it is parallel in nature. AHB BM supports 1 to 16 slave ports as well as master ports. Routing data width can be chosen among 32 or 64 bits. Routing address width can range from 32 to 64 bits. This provides the facility of multiple masters accessing different slaves simultaneously. Arbitration takes place to choose the higher priority master for a slave access when request to access the same slave is raised by more than one master simultaneously. Arbitration type can be round robin, fixed or burst [24]. When a higher priority master accesses a particular slave then the bus matrix locks the slave access for the other masters. AHB

BM acts as decoder in the platform; the decoder decodes the transaction of each IP then routes according to the address. Once the decoder decodes the transaction, the router will have start address and size of the target IP. If the transaction address information doesn't match the entries of memory map, the router returns an error.

The platform has two bus matrix, one known as MAIN SYSTEM INTERCONNECT (TAC_ROUTER) which is the primary AHB Bus Matrix and other is dedicated router for the integration of RAIMM IP known as SECONDRY LEVEL INTERCONNECT (RAIMM_TAC_ROUTER) which is the secondary AHB Bus Matrix. The primary bus matrix is in its TLM TAC form and the secondary one is in RTL form. The TAC_ROUTER is responsible for routing the transactions from its bind IPs to other IPs or to RAIMM_TAC_ROUTER.

Need of RAIMM_TAC_ROUTER (Secondary AHB Bus Matrix):

As explained earlier in this chapter the use of secondary interconnect is not mandatory. But for ease of implementation and not to disturb the existing TLM level platform a secondary level of interconnect has been used in this system to integrate the RAIMM and RAIMM_DMA IPs with the platform. This also facilitates to increase the number of split memory blocks up to 16. The RAIMM_DMA uses this router to perform the memory to memory data transfer when a memory unreliability situation arises. The dynamic remapping of starting addresses of memory blocks is accomplished through this secondary router.

Chapter 5: RAIMM Verification and Results

5.1 Test Scenario

This section explains the detailed functional verification process of the designed RAIMM IP. The verification of the IP has been done for the 3 stage reliability check and for 6 system memory blocks. Out of 6 blocks 4 memory blocks are considered as usable and 2 are taken as redundant. Each memory block is of size 16 KB and of word size 32 bits. All the Usable Memory valid data size has been kept as 64 bytes (16 words) for the simulation in test case 1, 2 and 3 where as the valid data size has been kept as 200 bytes (50 words) for the test case 4. For all the four test cases the burst size of 4 has been set for the DMA data transfer.

Simple experiments imposing unreliable conditions through C test cases have been performed and correct system operation with random memory failure has been observed by proper data movement from less reliable memories to reliable redundant memories with proper remapping of addresses.

An example of system level test to create scenario

- Initially RAIMM is configured.
- Memory M0 has been initialized with data.
- Software reads the data in a loop and compares it with the written data.
- Failure (Less reliability) is introduced in M0 memory (rest all memories remained reliable)
- The RAIMM chooses any of the reliable redundant memory modules from two redundant memory modules, here M5.
- RAIMM select source address as M0 memory address and Destination Address as M5 memory address.
- RAIMM _DMA is configured with source address and destination address to remap content of M0 memory to M5 memory.
- Once data transfer is done RAIMM change the address of M0 memory with address of selected redundant memory M5.
    - M0 memory address is M5 memory address, M0 is converted as a redundant memory
    - M5 memory address is M0 memory address, M5 is considered as the usable memory
- It is verified that test is executed successfully
- Latency is estimated

5.2 General Information

A0 to A5: All the six address regions of memory blocks

M0 to M5: All the six memory blocks

Note: Always A0 to A3 address regions have been considered as the usable address regions and A4 & A5 address regions are considered as the redundant address regions. So it is important to notice that when a memory unreliability situation arises then the usable memory block gets changed not the usable address space. For example, let the memory block M0 is being the usable one gets unreliable and M5 being the redundant and reliable then the content of M0 will be transferred to M5 and the M5 will become the usable and M0 will be the redundant. Now the address space A0 will be assigned to the M5 and A5 will be assigned to the M0.

Abbreviations

U_R: Usable Reliable

U_LR: Usable Less Reliable

U_UR: Usable Unreliable

R_R: Redundant Reliable

R_LR: Redundant Less Reliable

R_UR: Redundant Unreliable

BDT – Before Data Transfer (At the time the RAIMM Memory Status Register gets modified)

ADT – After Data Transfer (At the time the RAIMM Use Register gets modified)

The latching of memory reliability status has been done at two points for every PVT error conditions generated; one when the RAIMM Memory Status Registers gets updated (4 AHB clock cycles after the PVT error condition occurrence) and another when the RAIMM Use Register gets updated after the data transfer by the RAIMM_DMA (7 AHB clock cycle after the DMA data transfer completion). The latching of address range of memory block also is done at the same two points.

5.3 Test Cases

Test Case 1: *To check whether RAIMM is successfully triggering when a memory unreliability situation comes and RAIMM_DMA is transferring the data from the usable unreliable memory to the redundant reliable memory. Proper dynamic remapping of the memories has also been checked.*

	Time (ns)	A0	A1	A2	A3	A4	A5
First PVT error condition injection at 2500 ns							
BDT	2530	M0 (U_R)	M1 (U_R)	M2 (U_R)	M3 (U_R)	M4 (R_R)	M5 (R_R)
ADT		No Less Reliable or Unreliable condition so remains same					
Second PVT error condition injection at 3000 ns							
BDT	3024	M0 (U_R)	**M1 (U_UR)**	M2 (U_R)	M3 (U_R)	M4 (R_R)	**M5 (R_R)**
ADT	3470	M0 (U_R)	**M5 (U_R)**	M2 (U_R)	M3 (U_R)	M4 (R_R)	**M1 (R_UR)**
Third PVT error condition injection at 3600 ns							
BDT	3626	**M0 (U_UR)**	M5 (U_R)	M2 (U_R)	M3 (U_LR)	**M4 (R_R)**	M1 (R_LR)
ADT	4060	**M4 (U_R)**	M5 (U_R)	M2 (U_R)	M3 (U_LR)	**M0 (R_UR)**	M1 (R_LR)
Fourth PVT error condition injection at 4200 ns							
BDT	4229	M4 (U_R)	M5 (U_R)	M2 (U_R)	**M3 (U_LR)**	M0 (R_R)	**M1 (R_R)**
ADT	4663	M4 (U_R)	M5 (U_R)	M2 (U_R)	**M1 (U_R)**	M0 (R_R)	**M3 (R_LR)**

Table 5.1: Test conditions for Test Case 1

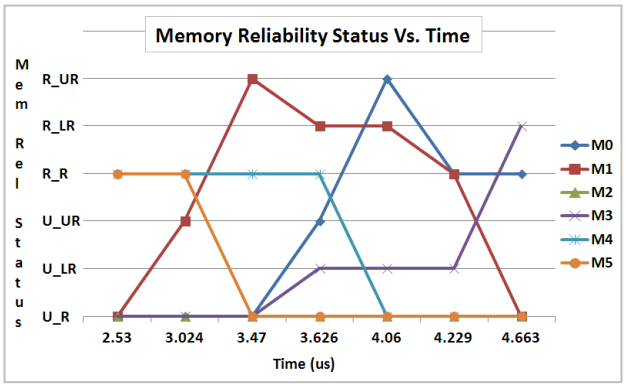


Figure 5.1: Memory Reliability Status Vs. Time (Testcase-1)

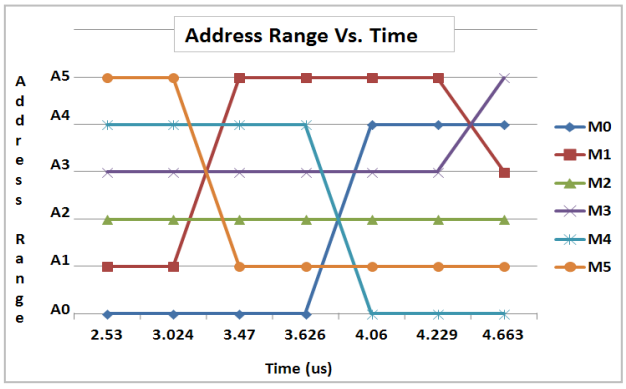


Figure 5.2: Memory Address Range Vs. Time (Testcase-1)

As shown in the test condition table 5.1, at 3000ns, usable memory M1 gets unreliable and the at the same time memory M5 is redundant reliable (figure 5.1). So the data from M1 gets transferred to M5 and the addresses of both the memories get remapped (figure 5.2). M5 became usable and M1 became redundant memory after the data transfer (figure 5.1). At 3600ns M0 gets unreliable and M3 gets less reliable while

M4 is available as the redundant reliable memory. As the ranking of M0 is higher than the M3 as M0 is unreliable, the remapping of M0 and M4 will be done. After this remapping none of the redundant memory M0 and M1 are reliable. At 4200ns M1 and M0 both become reliable being the redundant blocks. So the less reliable memory M3 gets remapped with M1. All the memory reliability status along with usable/redundant type can be noticed from the figure 5.1 and the address range assignment to memories can be observed from the figure 5.2. Signal waveforms have been shown in figure B.1 of appendix B.

Test Case 2: *To check whether RAIMM continues with the DMA data transfer operation or not while a new unreliable PVT condition arises but the DMA transfer is still going on for the previous PVT error condition. The data transfer for the later PVT error condition would continue after the completion of the first one.*

	Time (ns)	A0	A1	A2	A3	A4	A5
First PVT error condition injection at 2500 ns							
BDT	2530	M0 (U_R)	M1 (U_R)	M2 (U_R)	M3 (U_R)	M4 (R_R)	M5 (R_R)
ADT		No Less Reliable or Unreliable condition so remains same					
Second PVT error condition injection at 3000 ns							
BDT	3024	M0 (U_R)	**M1 (U_LR)**	M2 (U_R)	M3 (U_R)	M4 (R_R)	**M5 (R_R)**
ADT	**3470**	**M0 (U_UR)**	**M5 (U_R)**	M2 (U_R)	M3 (U_LR)	**M4 (R_R)**	**M1 (R_LR)**
Third PVT error condition injection at 3200 ns							
BDT	**3229**	**M0 (U_UR)**	M1 (U_LR)	M2 (U_R)	M3 (U_LR)	**M4 (R_R)**	M5 (R_R)
ADT	3952	**M4 (U_R)**	M5 (U_R)	M2 (U_R)	M3 (U_LR)	**M0 (R_UR)**	M1 (R_LR)
Fourth PVT error condition injection at 4200 ns							
BDT	4229	M4 (U_R)	M5 (U_R)	M2 (U_R)	**M3 (U_LR)**	M0 (R_R)	**M1 (R_R)**
ADT	4663	M4 (U_R)	M5 (U_R)	M2 (U_R)	**M1 (U_R)**	M0 (R_R)	**M3 (R_LR)**

Table 5.2: Test conditions for Test Case 2

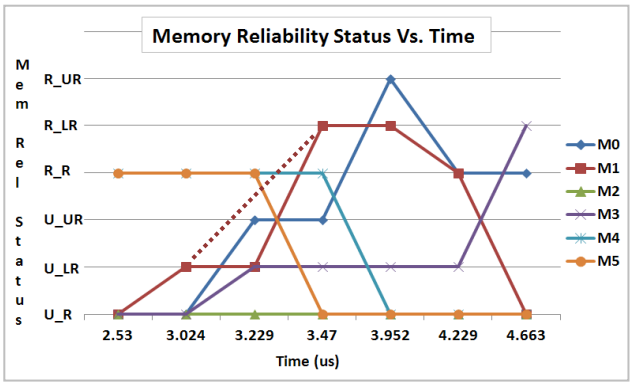


Figure 5.3: Memory Reliability Status Vs. Time (Testcase-2)

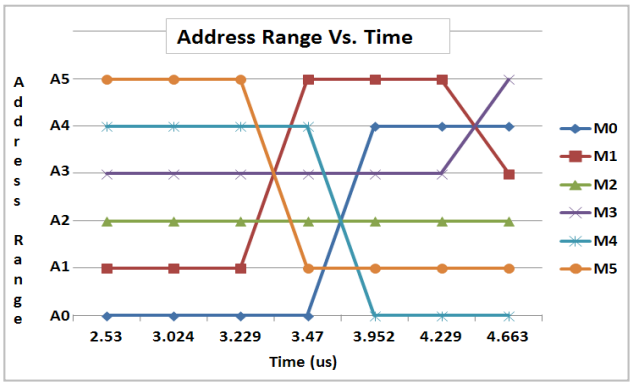


Figure 5.4: Memory Address Range Vs. Time (Testcase-2)

As shown in the test condition table 5.2, at 3000ns, usable memory M1 gets less reliable and the at the same time memory M5 is redundant reliable (figure 5.3). So the data from M1 gets transferred to M5 and the addresses of both the memories get remapped (figure 5.4). A dotted line has shown from 3024ns till 3470ns in figure 5.3. This is because the M1 memory would attain a reliability state somewhere between U_LR and R_LR during this period. But at 3229ns its reliability state has been shown as U_LR because of the third forced PVT error condition through the test case at 3200ns. M5 became usable and M1 became redundant memory after the data transfer (figure 5.3) at 3470ns. But at 3200ns before the completion of data transfer for the previous PVT error condition, another PVT unreliability condition has occurred as M0 getting unreliable with the availability of redundant reliable memory block as M4. The new remapping of memory blocks is not possible until the previous remapping has not been completed. So the new remapping of memories between M0 and M4 started at 3470ns and gets completed at 3952ns. The last PVT error condition is exactly same as the fourth one of the test case 1. Signal waveforms have been shown in figure B.2 of appendix B.

Test Case 3: *For checking whether RAIMM generates an interrupt or not while there comes an unreliable PVT condition where a usable memory gets less reliable or unreliable but none of the memory available as redundant and reliable.*

	Time (ns)	A0	A1	A2	A3	A4	A5
First PVT error condition injection at 2500 ns							
BDT	2530	M0 (U_R)	M1 (U_R)	M2 (U_R)	M3 (U_R)	M4 (R_R)	M5 (R_R)
ADT		No Less Reliable or Unreliable condition so remains same					
Second PVT error condition injection at 3000 ns							
BDT	3024	M0 (U_R)	**M1 (U_UR)**	M2 (U_R)	M3 (U_R)	M4 (R_R)	**M5 (R_R)**
ADT	3470	M0 (U_R)	**M5 (U_R)**	M2 (U_R)	M3 (U_R)	M4 (R_R)	**M1 (R_UR)**
Third PVT error condition injection at 3600 ns							
BDT	3626	M0 (U_R)	M5(U_R)	**M2 (U_UR)**	M3 (U_R)	**M4 (R_LR)**	**M1 (R_LR)**
ADT		**As none of the redundant memories are reliable so no data transfer and an interrupt signal generation**					
Fourth PVT error condition injection at 4200 ns							
BDT	4229	M0 (U_R)	M5 (U_R)	**M2 (U_UR)**	M3 (U_R)	**M4 (R_R)**	M1 (R_LR)
ADT	4663	M0 (U_R)	M5 (U_R)	**M4 (U_R)**	M3 (U_R)	**M2 (R_UR)**	M1 (R_LR)

Table 5.3: Test conditions for Test Case 3

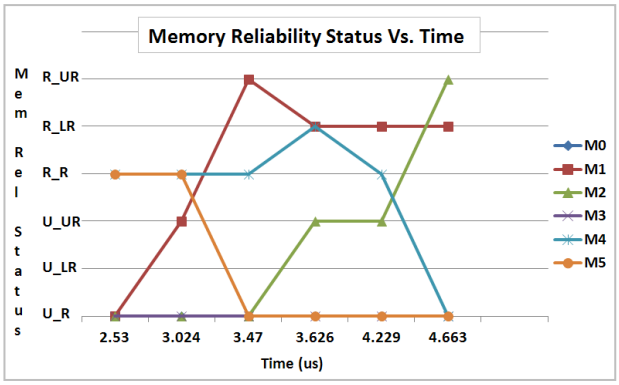


Figure 5.5: Memory Reliability Status Vs. Time (Testcase-3)

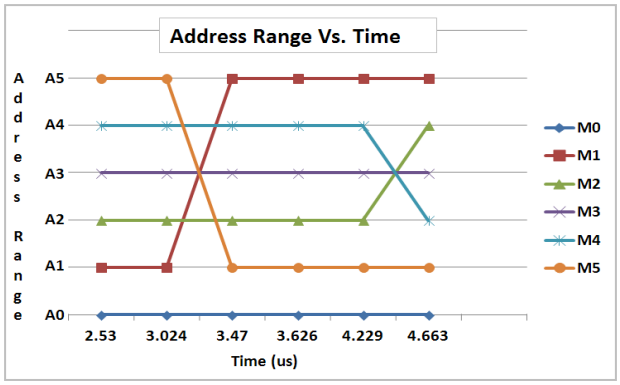


Figure 5.6: Memory Address Range Vs. Time (Testcase-3)

From test conditions table 5.3, it can be observed that the second and the fourth PVT error conditions are similar to the second and fourth ones of the test case 1. So, the remapping of memories follows the same pattern as explained for test case 1. This can also be viewed from the figure 5.5 and 5.6. There it can be seen an anomaly in the third PVT error condition at 3600ns, where there is an unreliable memory M2 but none of the two redundant memories M4 and M1 are reliable. In this type of situation there will not be any data transfer and RAIMM will generate an interrupt signal. Signal waveforms have been shown in figure B.3 of appendix B.

Test Case 4: *This is the test case to check whether the Processor is getting halted or not when the RAIMMM_DMA starts transferring data from one memory to another due to the presence of alarming condition and the processor is accessing any one of the memory which is involved in the RAIMM_DMA operation. This test case also checks whether the processor starts accessing the same location from where it left once the DMA operation would be completed.*

	Time (ns)	A0	A1	A2	A3	A4	A5
First PVT error condition injection at 2000 ns							
BDT	2024	M0 (U_R)	M1 (U_R)	M2 (U_R)	M3 (U_R)	M4 (R_R)	M5 (R_R)
ADT		No Less Reliable or Unreliable condition so remains same					
Second PVT error condition injection at 2600 ns							
BDT	2626	**M0 (U_UR)**	M1 (U_R)	M2 (U_R)	M3 (U_R)	M4 (R_R)	**M5 (R_R)**
ADT	3542	**M5 (U_R)**	M1 (U_R)	M2 (U_R)	M3 (U_R)	M4 (R_R)	**M0 (R_UR)**
Third PVT error condition injection at 3700 ns							
BDT	3723	**M5 (U_UR)**	M1(U_R)	M2 (U_R)	M3 (U_R)	**M4 (R_R)**	M0 (R_R)
ADT	4638	**M4 (U_R)**	M1(U_R)	M2 (U_R)	M3 (U_R)	**M5 (R_UR)**	M0 (R_R)
Fourth PVT error condition injection at 5200 ns							
BDT	5229	M4 (U_R)	M1(U_R)	M2 (U_R)	**M3 (U_UR)**	M5 (R_LR)	**M0 (R_R)**
ADT	6144	M4 (U_R)	M1(U_R)	M2 (U_R)	**M0 (U_R)**	M5 (R_LR)	**M3 (R_UR)**

Table 5.4: Test conditions for Test Case 4

Test case 4 shown in test condition table 5.4 is similar to test case 1. Only the valid data size of memory block taken here is 200 bytes whereas it was 64bytes in case of test case 1. A larger data size has been considered to realize whether the processor gets halted or not while the memory block which is getting accessed by the processor gets less reliable or unreliable and RAIMM raises a trigger. The address range accessed by the processor throughout the simulation is A0. Initially A0 is assigned to memory M0. At 2600ns and at 3700ns memory blocks M0 and M5 associated with A0 respectively, get unreliable. For the second PVT error condition, the HREADY signal to the processor core from the bus matrix gets low at 2728ns after the RAIMM_DMA gets configured for memory to memory data transfer and the processor READ operation gets halted. After the DMA data transfer has been completed and the memory remapping has been done for the second PVT error condition, the HREADY goes high at 3584ns. Now the processor starts accessing the memory M5 assigned to address range A0 and to the same address location from where it was stopped. Now at 3700ns for third PVT error condition M5 gets unreliable and in the similar fashion HREADY gets low at 3825ns and goes high at 4680ns. Now address range A0 has been assigned to M4 and processor starts accessing M4. At 5200ns fourth PVT error condition occurs as M3 getting unreliable and M0 being redundant reliable. M3 address range is A3 and M0 address range is A5. So during the remapping of M3 to M0 the HREADY will not go low as processor is accessing A0 and it will continue with its operation without any interruption. Signal waveforms have been shown in figure B.4 of appendix B.

5.4 Latency Calculation

Above section shows the functional correctness of the RAIMM for different test cases. In this section the performance parameter, latency introduced by the RAIMM operation has been discussed.

RAIMM IP does not follow the critical path of the processor and has been developed to operate in parallel with the processor operation. It generates the system level interrupt only when there is a memory reliability issue arises. But the generation of the interrupt does not mean that the normal operation of the processor will be halted. The processor will be halted only in the situation when it would be accessing the same memory that would become unreliable or less reliable for which the alarm situation has arrived.

The AHB clock used in the system is of 166 MHz, so the period of one AHB clock cycle is 6.024 ns. The APB clock used in the system is of 83 MHz, so the period of one APB clock cycle is 12.048 ns.

The Latency for the RAIMM operation can be expressed as follows where all the clock cycles are considered as the AHB clock cycles. The number of clock cycles has been calculated from the point of the event of a PVT error condition where as the first

two components of the latency shown in the following formula would always be parallel with the processor operation, hence ideally will not be considered for the latency calculation for the situation when the processor will get halted due to memory unreliability condition.

Latency =	(i) '7' clock cycles taken by RAIMM to generate trigger signal and updating the source, destination and size registers for DMA data transfer + (ii) '9' clock cycles for DMA configuration + (iii) 'N' clock cycles taken by the DMA module to perform the memory to memory data transfer + (iv) '12' clock cycles taken by RAIMM to generate the REMAP signal, updating the RAIMM_UR, M_SARs and M_DSRs in RAIMM register module and coming back to the IDLE state

Here N clock cycles for the DMA data transfer depends on the valid data size in the memory block to be transferred and the burst size configured in DMA.

Appendix A shows the latency calculation for the situation when the processor gets halted because of memory reliability issue with the consideration of 4KB of memory data size and the DMA burst size set at 4. The calculated latency has been found as 2330 AHB clock cycles.

Chapter 6: Conclusion and Future Work

6.1 Conclusion

A memory manager named Reliability Aware Intelligent Memory Management (RAIMM) has been developed as a digital IP to provide a failsafe solution to on chip memory failure due to PVT variations. The motivation for the design has been emerged to provide on-chip memory reliability protection to safety critical automotive products where as the product can find its use in any safety critical application. The fundamentals of memory redundancy and data replication have been used to implement the concept successfully.

The IP has been integrated into a TLM platform with the ARM926 processor model for its functionality check. The functionality of the IP has been verified by injecting PVT error conditions using the 'C' test cases. From the verification it is established that the framework is working as expected. For the quick verification 3 levels are used to predict reliability – Reliable, Less Reliable and Unreliable. It helped in dealing with simple yet effectual model in the SoC. A dynamic reliability map of memories is produced in our simulations which varied according to the injected test-cases. It has been observed that the RAIMM module is able to successfully predict memory failures and generates the alarm when a less reliable or unreliable condition arises. The memory data get transferred from the less reliable memory to the redundant reliable memory through the RAIMM_DMA with no software intervention and the dynamic remapping of memory addresses after the data transfer are achieved flawlessly. It has been observed from the latency calculation that the time required for data remapping is proportional to the size of data required to be moved. Although the RAIMM IP introduces some amount of area overhead due to the redundant memory blocks but it ensures a failsafe memory operation in case of PVT variations. The RAIMM IP is functionally ready to provide reliable solution to on-chip memory failure.

6.2 Future Work

The developed RAIMM IP is at RTL level. The synthesisation of the IP will be done along with the area, power and timing analysis. The FPGA porting of the IP can be done to realize its working on hardware level before sending it for fabrication. This IP has been verified for AHB BM interconnects. The scope of the IP can be explored for NoC architecture. At the same time, it is possible to explore the additional benefits of this model, in areas like Dynamic Voltage Frequency Scaling and Selective page table management.

Bibliography

[1] Kalinsky, David. “Architecture of safety-critical systems.” Embedded Systems Programming magazine (U.S.A.), vol. 18, no. 9, Sept. 2005.

[2] Baleani, Massimo, et al. “Fault-tolerant platforms for automotive safety-critical applications.” Proceedings of the 2003 international conference on Compilers, architecture and synthesis for embedded systems. ACM, 2003.

[3] Saini, Yash, et al. “Safety & security architecture for automotive ICs.” Sept. 2013. [Online]. Available: http://www.edn.com

[4] Negi, Deepak, et al. “Redundancy for safety-compliant automotive & other devices.” Mar. 2014. [Online]. Available: http://www.edn.com

[5] International technology roadmap for semiconductors - 2003 (ITRS-2003). [Online]. Available: http://public.itrs.net/

[6] Sarrazin, David B., and Miroslaw Malek, "Fault-Tolerant Semiconductor Memories." Computer, vol. 17, no. 8, pp. 49-56, Aug. 1984.

[7] Tezzaron Semiconductor, “Soft Errors in Electronic Memory – A White Paper.” Version 1.1, 2004.

[8] Tsiligiannis, Georgios, et al. "SRAM soft error rate evaluation under atmospheric neutron radiation and PVT variations." On-Line Testing Symposium (IOLTS), 2013 IEEE 19th International. IEEE, 2013.

[9] Pilo, Harold, et al. "An SRAM design in 65nm and 45nm technology nodes featuring read and write-assist circuits to expand operating voltage." VLSI Circuits, 2006. Digest of Technical Papers. 2006 Symposium on. IEEE, 2006.

[10] Rodriguez, R., et al. "The impact of gate-oxide breakdown on SRAM stability." Electron Device Letters, IEEE 23.9 (2002): 559-561.

[11] Cannon, Ethan H., et al. "The impact of aging effects and manufacturing variation on SRAM soft-error rate." Device and Materials Reliability, IEEE Transactions on 8.1 (2008): 145-152.

[12] Chandra, Vikas, and Robert Aitken. "Impact of voltage scaling on nanoscale SRAM reliability." Proceedings of the Conference on Design, Automation and Test in Europe. European Design and Automation Association, 2009.

[13] Bagatin, M., et al. "Temperature dependence of neutron-induced soft errors in SRAMs." Microelectronics Reliability 52.1 (2012): 289-293.

[14] M. Blaum, et al. "The reliability of single-error protected computer memories." Computers, IEEE Transactions on 37.1 (1988): 114-119.

[15] Nicolas S. Bowen et al. The effect of program behaviour on fault observability. IEEE Trans. Computing, 1996.

[16] J. Ohtani, et al. A shared built-in self-repair an embedded memories. In Proc. IEEE CICC), 2001.

[17] M. Choi, et al. Optimal spare utilization in repairable and reliable memory cores. In Proc. Int. Workshop on Mem. Tech., Design and Testing, 2003.

[18] Shivakumar, Premkishore, et al. "Exploiting microarchitectural redundancy for defect tolerance." Computer Design, 2003. Proceedings. 21st International Conference on. IEEE, 2003.

[19] Bathen, Luis Angel D., and Nikil D. Dutt. "E-RoC: Embedded RAIDs-on-Chip for low power distributed dynamically managed reliable memories." Design, Automation & Test in Europe Conference & Exhibition (DATE), 2011. IEEE, 2011.

[20] Zhang, Wei, et al. "ICR: In-Cache Replication for Enhancing Data Cache Reliability." DSN. 2003.

[21] ARM Technical Reference Manual, "PrimeCell® DMA Controller (PL080)." Revision: r1p3

[22] Ghenassia, Frank. "TLM: An Overview and Brief History". Transaction Level Modeling with SystemC. Netherlands: Springer, 2005

[23] AMBA™ Specification. Revision: 2.0

[24] ARM Technical Reference Manual, "AMBA Design Kit." Revision: r3p0

[25] Baranwal, D., SINGH, D.P., Saha, K., "Memory manager" Patent US 20140047285 A1, 13 Feb 2014

Appendix A

Worst Case Latency Calculation

Processor will be halted only in the situation when it would be accessing the same memory which has become unreliable or less reliable for which the trigger signal for DMA data transfer would be generated by the RAIMM.

Here to generate such a test condition, the PVT error condition has injected to the memory that the processor is accessing. The HREADY for the Processor goes low just when the DMA would get configured and the data transfer will start and goes high when the DMA data transfer will be completed and the trigger module will come back to its state 0. The latency of processor getting halted during this situation has been calculated.

The valid data size in the usable memory block has been set as 4KB.

Word size of the memory block = 32 bits = 4 Bytes

So total number of words in the memory block = 4KB / 4 Bytes = 1024 words

The burst size for RAIMM_DMA data transfer has been set as 4.

So total number of burst transfer = 1024 / 4 = 256

Total number of AHB cycles taken for transferring a single burst is as follows:

For bus request and grant = 4 cycles

1st word of the burst will take = 2 cycles (for non-sequential transfer)

Next three words of the burst will take = (1+1+1) cycles = 3 cycles (for sequential transfer)

So the total number of cycles taken by DMA master-1 reading from the source memory will be as follows:

For last 255 bursts: 255 x (4+2+3) cycles = 2295 cycles

For the first burst of the master-1: The transfer of the first burst will take place in only 4 cycles as the bus request and grant for the DMA would be done before the HREADY of processor would go low before the first read.

So total AHB clock cycle taken by DMA master-1 = 2295 + 4 = 2299 cycles …… (i)

Total number of cycles taken by DMA master-2 writing to the destination memory will be as follows:

DMA master-2 would write to the destination memory parallel to the read operation of the DMA master-1 with a latency of 2 burst transfer. So the extra time taken by the last two burst transfer by DMA master-2:

For last two bursts: 2 x (4+2+3) cycles = 18 cycles …… (ii)

Generation of the terminal count (TC) signal showing the transfer complete will take 1 cycle and after the generation of TC, RAIMM will take 12 AHB clock cycles to generate the REMAP signal and update the RAIMM_UR, M_SARs and M_DSRs in RAIMM register module according to the remapping of the memory blocks. Hence total of 13 AHB cycles will be consumed after the DMA data transfer. Adding to the cycles taken collected from (i) and (ii)

Total AHB clock cycles = 2299 + 18 + 13 = 2330 clock cycles

So total latency introduced = 2330 x 6.024ns = 14036ns = 14.036us

Appendix B

Waveform 1: Test Case 1 (RAIMM_Operation)

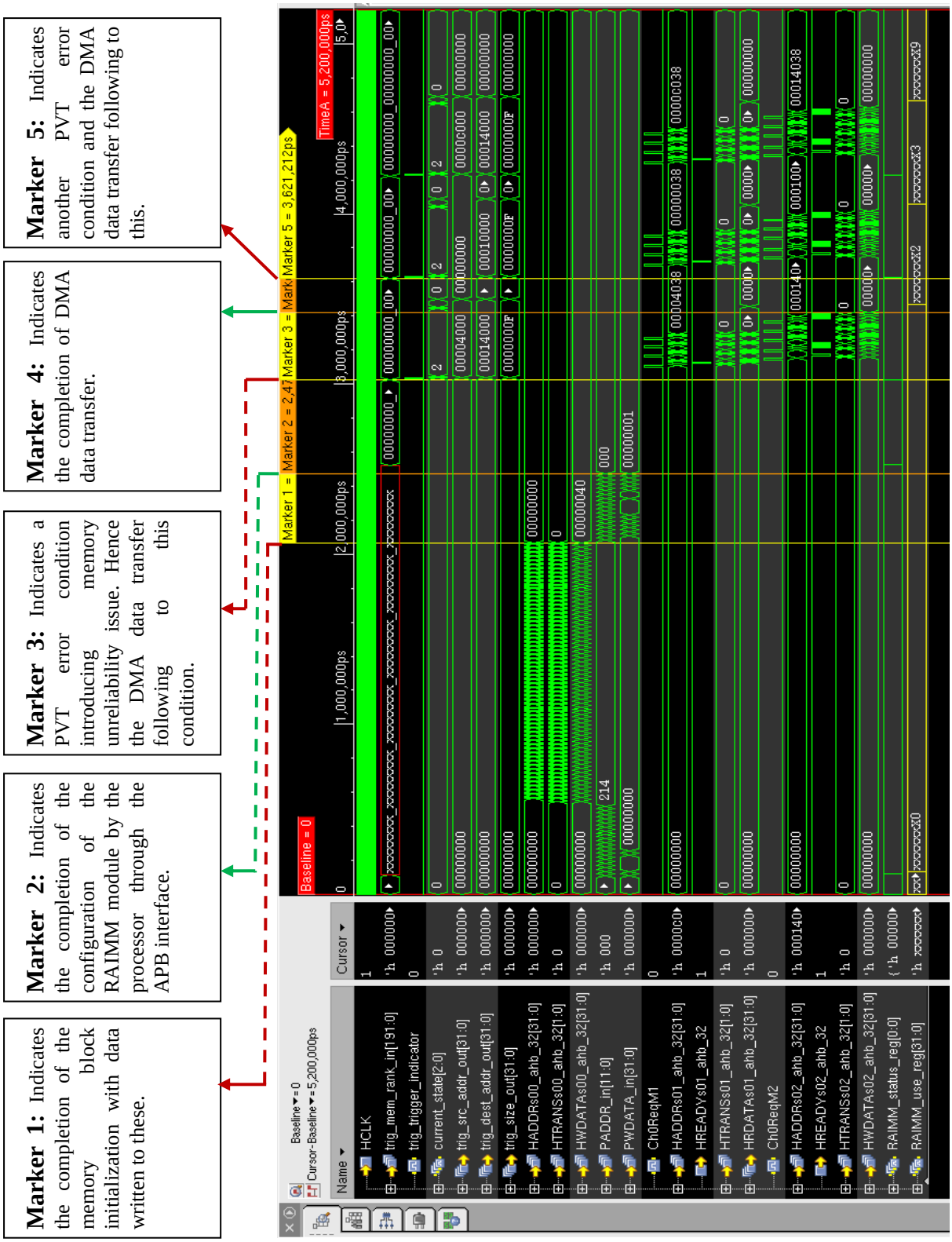


Figure B.1: Signal Waveform for Test Case 1

Waveform 2: Test Case 2 (PVT error condition before the completion of the earlier data transfer)

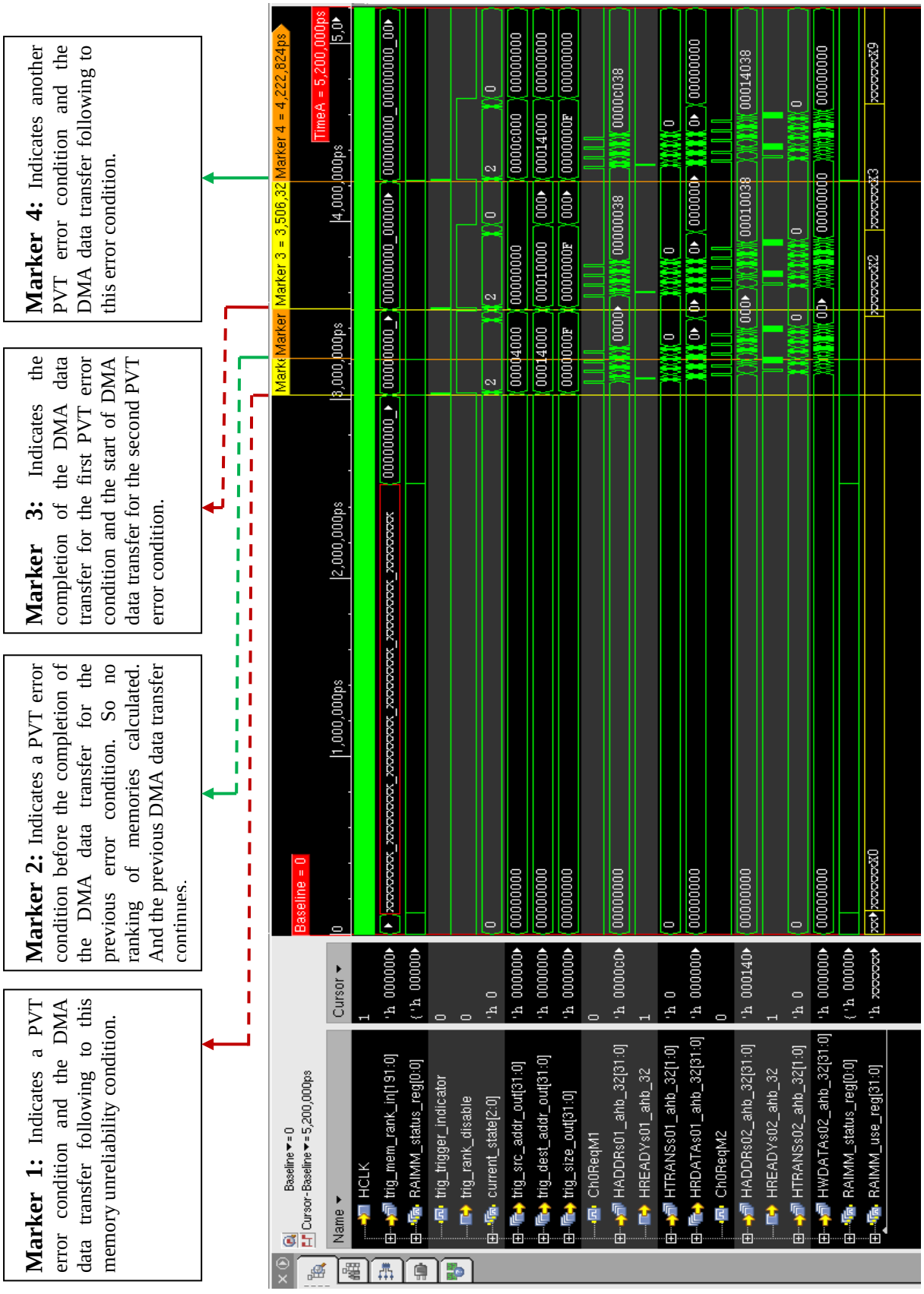


Figure B.2: Signal Waveform for Test Case 2

Waveform 3: Test Case 3 (Interrupt generation for PVT error condition but non availability of redundant reliable memory)

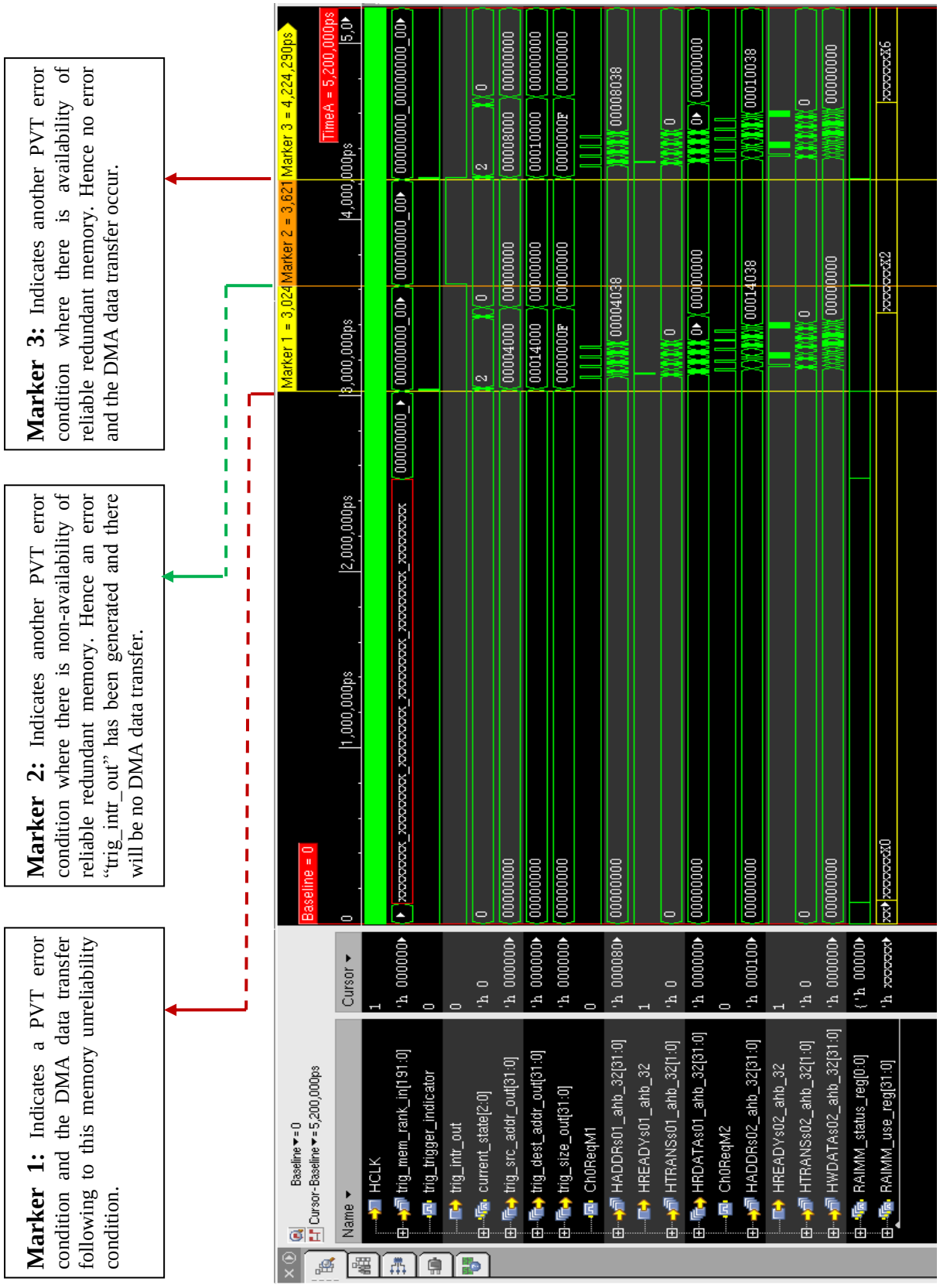


Figure B.3: Signal Waveform for Test Case 3

Waveform 4: Test Case 4 (Processor gets halted while the memory accessed by it gets unreliable)

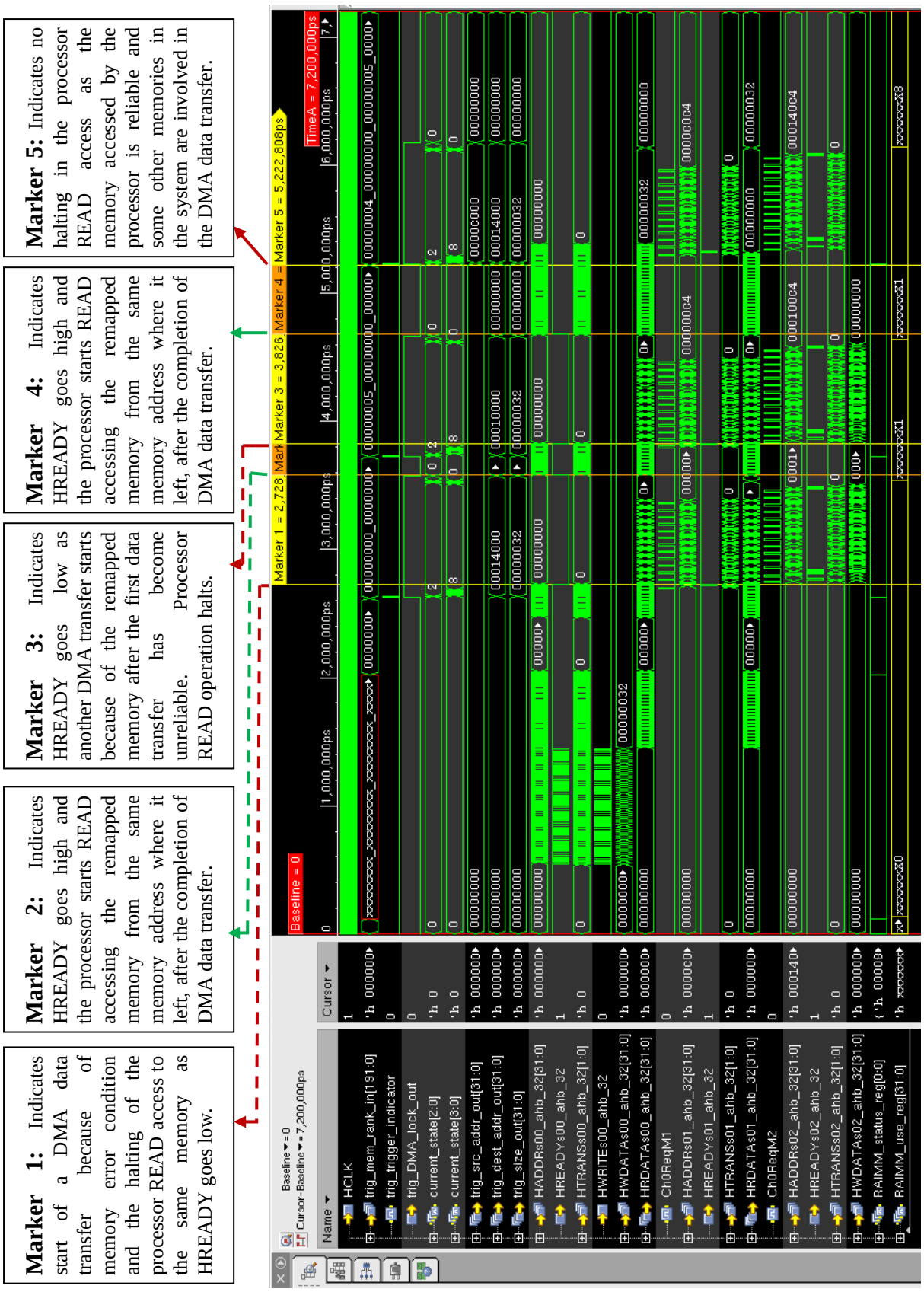


Figure B.4: Signal Waveform for Test Case 4