Data Security Architecture using Embedded Chip

***Dilip Thomas and @K.S.M. Panicker**

[*]Research Scholar, Faculty of ECE, Dr.M.G.R. University, Chennai-95
[@]Dean Research, Federal Institute of Science and Technology, Kerala.

Abstract--The technologies of computer security are more logic oriented. Designing a program with security often imposes restrictions on that program's behavior. In this work, it is proposed to add a security level in hardware such that the security will extend from the computer architectural hardware level to the software OS level. Even though the operating system controls access to the system resources and programs, the request for the access to these system resources through the Operating System (OS) is done via an integrated chip located in the Peripheral Component Interconnect (PCI) bus of the system. This will add a new dimension of security to the entire computer architecture. The level of security appropriate for a particular system depends on the value of the resources being secured. Incorporation of key operating system function into the hardware is planned so that the size of the security mechanisms still remains small, while not back tracking on the implementation of security. Use of various monitoring functions performed at regular intervals incorporated into the hardware is done so that its implementation is much faster than its equivalent software implementation.

Keywords—Embedded system, PCI, System security

1. Introduction

The main objective of any system is to share and secure information at the same time. However, sharing and protection are two contradictory goals. For protection, programs may be completely isolated from each other by executing them on separate non-networked computer, but, this precludes sharing. Similarly in the case of sharing, programs running on the same computer may be allowed to share resources extensively, but, this reduces the amount of protection possible. Thus, to add a layer of security to the existing data layers between the CPU and cache, addition of a session restoration tool that will restrict the entry of viruses and safeguard critical data is required. To maintain almost the existing data transfer ratio with the new/desired module, while retaining the memory hierarchy that is prevalent in the lower memory architecture is the focus of this work. Securing the functions is includes performing access control, logging monitoring and those which manage the real storage, virtual storage and the file system.

1.2 Security by design

The technologies of computer security are based on logic. As security is not necessarily the primary goal of most computer applications, designing a program with security in mind often imposes restrictions on that program's behaviour. There are few approaches to security in computing, sometimes a combination of approaches is valid:

1. Trust all the software to abide by a security policy, but, the software is not trustworthy.
2. Trust all the software to abide by a security policy and the software is validated as trustworthy.
3. Trust no software, but, enforce a security policy with mechanisms that are not trustworthy (again this is computer insecurity).
4. Trust no software but enforce a security policy with trustworthy hardware mechanisms.

The protection scheme is shown in Figure 1.

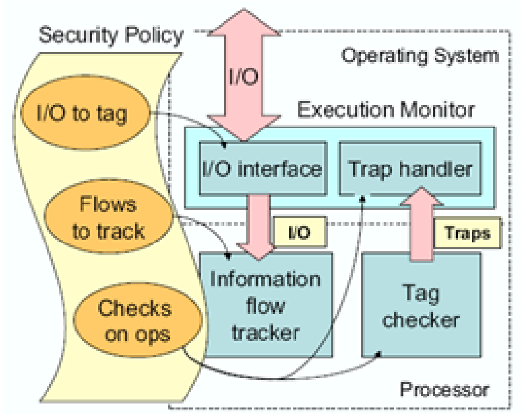


Figure 1 Overview of Protection scheme

In this work, approach is based on hardware mechanisms and avoids abstractions and offers a multiplicity of degrees of freedom, and hence, it is more practical. A secure system should require a deliberate, conscious, knowledgeable and free decision on the part of legitimate authorities in order to make it insecure. In addition, security should not be an all or nothing issue. The designers and operators of systems should assume that security breaches are inevitable. System security mechanisms controls serve the purpose to maintain the system's quality attributes, among them confidentiality, integrity, availability, accountability and assurance.

Manuscript received May 5, 2011
Manuscript revised May 20, 2011

2. Literature Survey

Ilgun, K. [1993] proposed USTAT: A Real-time Intrusion Detection System for UNIX. A number of non-expert system-based approaches to intrusion detection, while many of these have substantial promise, expert systems remain the most commonly accepted approach to the detection of attacks. T. Yamauchi et al., [1995] proposed fully self-timing data-bus architecture for 64-mb DRAMs. An area of the sense amplifiers embedded in each DRAM array the total area of the memory cells when the 128 cells are connected with each bit line. Peter Petrov and Alex Orailoglu [2001] proposed towards effective embedded processors in co-designs: customizable partitioned caches. An approach to partition the load/store instructions into different cache partitions is evaluated with a direct-mapped cache. W.T. Shiue and C. Chakrabarti [1999] presented memory exploration for low power, embedded systems. The framework of the generic enough to permit other cost functions such as power dissipation to be incorporated. W. Wolf and M. Kandemir [2003] presented memory system optimization of embedded software. A memory architecture exploration frameworks and data layout heuristics for target architectures that are primarily Scratch-Pad RAM (SPRAM) based. Asaduzzaman and I. Mahgoub [2006] proposed cache optimization for embedded systems. The computing speed of microprocessors has exponentially increased and support to computation intensive embedded systems. With such improved computing power, memory subsystem deficiency becomes the major barrier to support real-time multimedia applications on embedded systems.

S. Mohanty and V.K. Prasanna [2004] presented design of high-performance embedded system using model integrated computing. An embedded system is a special-purpose computing system embedded in an electrical or mechanical device to perform one or a few dedicated tasks repeatedly. P.R. Panda, et al., [2001] proposed data and memory optimization techniques for embedded systems. A cache memory architecture two assessments are needed: (i) an evaluation of the performance gain for a specific application and (ii) an indication of the modifications required in the software.

3. Secure Operating Systems

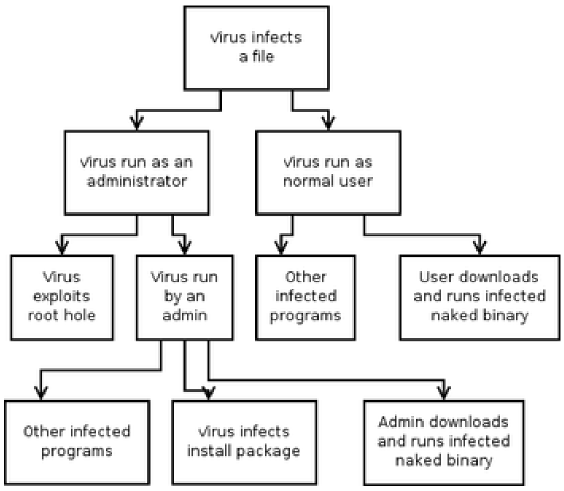
The term computer security refers to technology to implement a secure operating system. Such ultra-strong secure operating systems are based on operating system kernel technology that can guarantee that certain security policies are absolutely enforced in an operating environment. The strategy is based on a coupling of special microprocessor hardware features, often involving the memory management unit, to a special correctly implemented operating system kernel. The forms for a secure operating system, if certain critical parts are designed and implemented correctly, can ensure the absolute impossibility of penetration by hostile elements.

In the field of networking, the area of network security consists of the provisions and policies adopted by the network administrator to prevent and monitor unauthorized access, misuse, modification, or denial of the computer network and network-accessible resources. The term network security and information security are often used interchangeably. Network security is generally taken as providing protection at the boundaries of an organization by keeping out intruders. Information security, however, explicitly focuses on protecting data resources from malware attack or simple mistakes by people within an organization by use of data loss prevention (DLP) techniques. One of these techniques is to compartmentalize large networks with internal boundaries.

At the cache and Central Processing Unit (CPU) level, data is transferred directly between the two components via means of the data link layer. There is no security cover at this level. In a user authenticated security system, biometric signals are generally used for user authentication. These systems however are limited by the user himself. In some system, the system allows users to directly access critical files. Here, the unintended user would be easily in control and the operating system will not be able to monitor all the activity against an object and its modified capabilities. These issues may be taken care of by using threat monitoring are not allowed to access resource directly. Operating system routines called surveillance programs do this. Although this is a highly secure system, the bottleneck lies in its rigidity. For example if we are required to install some new applications on a Uniplexed Information System (UNIX) system it will be very difficult as a normal user has limited grants to the system.

4. Attack tree

Attack trees are conceptual diagrams of threats on computer systems and possible attacks to reach those threats. Attack trees are similar to threat trees shown in Figure 2.


Virus infects a file
Virus run as an administrator
Virus run as normal user
Virus exploits root hole
Virus run by an admin
Other infected programs
User downloads and runs infected naked binary
Other infected programs
Virus infects install package
Admin downloads and runs infected naked binary
Figure 2 Threat tree structure

Attack tree for computer viruses that assume a system, where not all users have full system access. All child nodes operate on OR conditions. Attack trees are multi-leveled diagrams consisting of one root, leaves, and children. From the bottom up, child nodes are conditions which must be satisfied to make the direct parent node true; when the root is satisfied, the attack is complete. Each node may be satisfied only by its direct child nodes. A node may be the child of another node; in such a case, it becomes logical that multiple steps must be taken to carry out an attack.

4.1 Information Assurance scheme

Attack can become largely complex, especially when dealing with specific attacks. A full attack tree may contain hundreds or thousands of different paths all leading to completion of the attack. Even so, these trees are very useful for determining what threats exist and how to deal with them. Attack can lend them to defining an information assurance strategy. It is important to consider, however, that implementing policy to execute this strategy changes the attack tree. Such that computer viruses may be protected against by refusing the system administrator access to directly modify existing programs and program folders, instead requiring a package manager be used. This adds to the attack tree the possibility of design flaws or exploits in the package manager. One could observe that the most effective way to mitigate a threat on the attack tree is to mitigate it as close to the root as possible. It is not usually possible to simply mitigate a threat without other implications to the continued operation of the system.

The protected resources surrounded by rings of the control & and rings of users. This representation is more useful for the purpose of identifying where and what kind data might be of use in detecting the exercise of one of the threat shown in Figure 3.

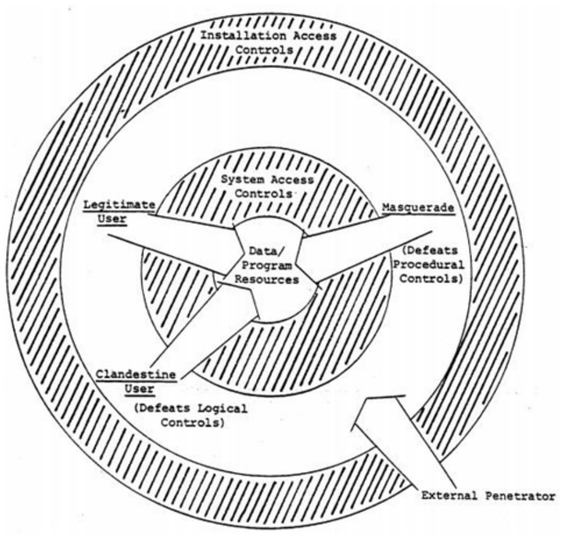


Figure 3 Process of the Data Resources

5. Hardware Security Problems, Attacks and solutions

Due to immense costs associated with fabricating chips in the latest technology node, the design, manufacturing and testing of the integrated circuits has moved across a number of countries. This makes it difficult to secure the supply chain opens up a number of possibilities for securities threats. In the simplest form an attack can be categorised as the interruption, interception, modification, or fabrication of data from a valid source to its intended destination is shown in Figure 4.

Figure 4 normal traffic flow pattern from source to destination

a) Interruption – An asset of the system is destroyed or becomes unavailable. Caused by an intruder gaining unrestricted access by means of telnet. E.g. a DoS Attack (Denial of Service). Figure 5 shows the traffic interruption between sources to destination points.

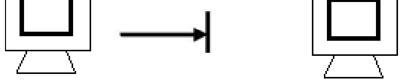


Figure 5 Traffic interruption from source to destination by an attack

b) Interception – An unauthorized party gains access to an asset.

This attack is performed by snooping on the network traffic to try to obtain data such as password, credit card number. A man in the middle attack is an example of this category and is shown in Figure 6.

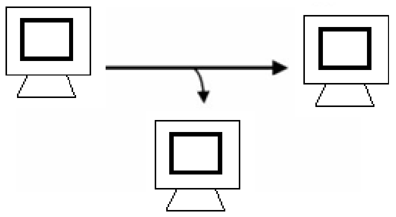


Fig 6 Interception of traffic flow from source to the destination by an unauthorized third party

c) Modification – an unauthorized party gains access to and tampers with an asset. This is shown in Figure 7.

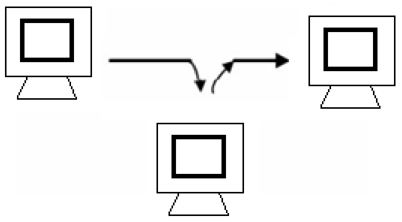


Figure 7 Modification of traffic flow source to destination by an unauthorized third party

d) Fabrication – An unauthorized party inserts counterfeit objects into the system, which is shown in Figure 8.

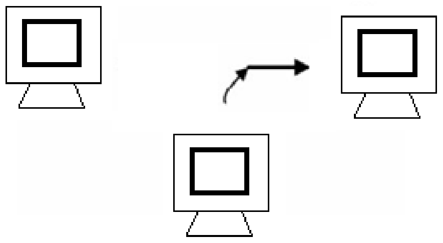


Figure 8 Fabrication of traffic from unauthorized third party to destination. Third party is posting as a trusted source of information

5.1 Hardware attacks during the Life cycle

(i) Trusted Hardware Attack

- A Trojan horse which is inserted to ICs appears to work normally but blocks the access to some resource and enables access to some restricted area.
- A KILL SWITCH once activated renders the chip inoperable.
- A BACKDOOR is form of TROJAN HORSE when activate selectively disable the encryption core.

(ii) Physical Attack

DATA REMANENCE is residual information that exists even after an attempt to erase. For e.g. Former DRAM cell contains their content for a second to minute even after the power down, even at the room temperature if removed from the mother board.

(iii) Design Tools Subversion

Programming of an FPGA relies on variety of CAD tools.

- COVERT CHANNELS are the attacks where rouge IP cores use a shared resource to transmit the information without the authorization or knowledge of the shared resource.
- SIDE CHANNEL, the IP cores have access to the internal resources of the chip, allowing them to tap into the side channel is much easier than at the chip or device level.

(iv) Design Theft

- CLONING is an unauthorised copying of the design. This attack is dangerous because once the bit stream is in hand, it is rather easy to buy another FPGA and use that bit stream to create a replica of the system from which the bit stream was stolen.
- REVERSE ENGINNERING is the process of discovering properties of design e.g. by translating the bit stream to the higher level.
- READBACK ATTACK directly obtains the bit stream from the functioning device.

6. Comparison between different Security issues

In this security mechanism uses different flows to compare Security exceptions, which is shown in figure 9.

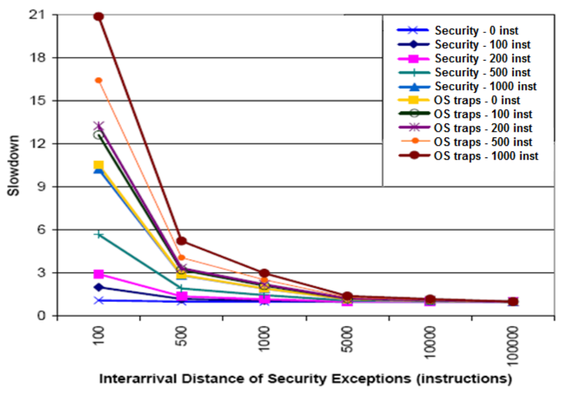


Figure 9 Comparison of Inter-arrival distance on security exceptions

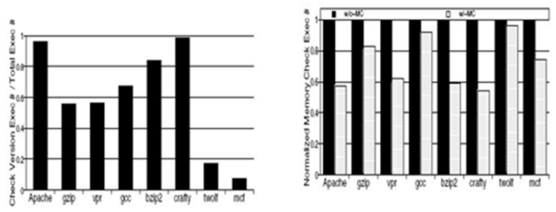


Figure 10 Check version execution

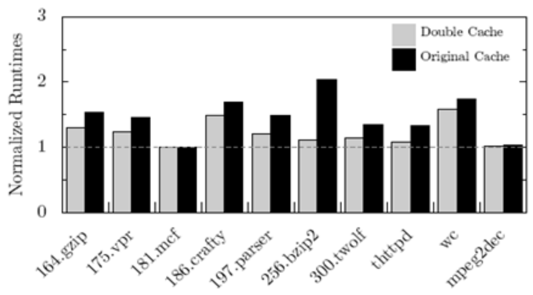


Figure 11 Performance of secured programs

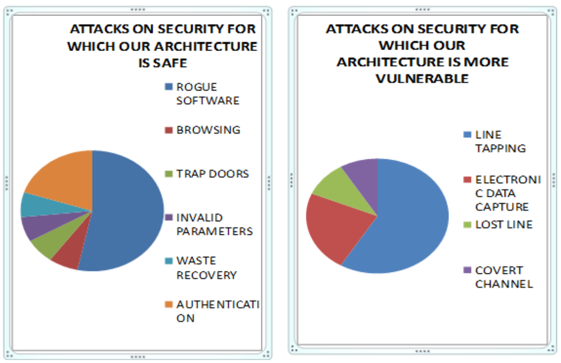


Figure 12 Safety and Vulnerability of Architecture

7. On-chip Memory Architecture of Embedded Processors

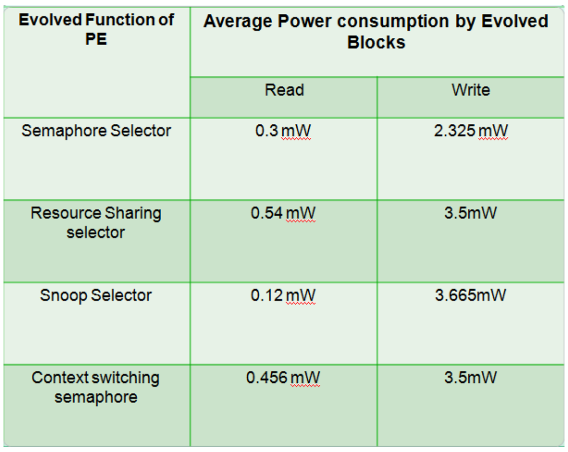
7.1 Microcontroller Memory Architecture

Microcontroller's (MCU) are designed to execute control type of applications efficiently. The applications run on microcontroller's are not very data intensive, and hence, do not require DARAM. The real-time constraints of embedded applications and the need to run the applications in a time bound manner require on-chip SPRAM. Similar to DSP memory architectures, even MCU processors have on-chip SPRAM. But unlike DSP's on-chip SPRAM, the MCU's on-chip RAM is not organized as multiple memory banks. This is because typically the MCU applications do not perform more one memory access per clock cycle. Although the MCU's on-chip RAM may be constructed with multiple physical memory modules due to practical limitations and other constraints such as:

(i) Smaller memory modules are faster and power efficient, and

(ii) It is not practical to construct one large memory module and still meet the access latency constraint.

The Power Consumed by Evolved PE is shown in Figure 13.



Evolved Function of PE	Average Power consumption by Evolved Blocks	
	Read	Write
Semaphore Selector	0.3 mW	2.325 mW
Resource Sharing selector	0.54 mW	3.5mW
Snoop Selector	0.12 mW	3.665mW
Context switching semaphore	0.456 mW	3.5mW

Figure 13 Power Consumed by Evolved PE

The memory architecture for a DSP is unique, since, the DSP has multiple on-chip buses and multiple address generation units to service higher bandwidth needs. In this work, the memory architecture optimization for DSPs because, the memory architecture of the DSP is more complex than that of microcontrollers (MCU) due to the following reasons:

(i) DSP applications are more data dominated than the control-dominated software executed on an MCU. Memory bandwidth requirements for DSP applications range from 2 to 3 memory accesses per processor clock cycle.

(ii) It is critical in DSP application to extract maximum performance from the memory subsystem in order to meet the real-time constraints of the embedded application.

7.2 Cache-SPRAM Based Hybrid On-chip Memory Architecture

Designers of real-time embedded memories have typically preferred scratch-pad memories (SPRAM) over data caches, since, the latter lead to unpredictability in access latencies as cache hits and misses result in different access times. The SPRAM memory is assigned dedicatedly to data-variables and the memory is not reused or shared among different data-variables. There are dynamic data layout approaches that aim at sharing the SPRAM by bringing the data-variables from off-chip RAM to on-chip RAM at run-time. However, these approaches are very complex and make the software very difficult to maintain and debug. Data movements in the dynamic layout approach may lead to code size and run-time overheads. Hence, architectures with only SPRAM as on-chip memory will be highly area inefficient for large applications. It is difficult to estimate the worst-case guaranteed run-time performance in a cache based system, which is a requirement for all embedded systems.

7.3 Hardware Implementation using context Switched Semaphore

In a segmentation system, two dimensional addresses representing a storage location within a segment and a capability for the segment is used. The segments are checked against the stated access rights in the capability. The users must be prevented from creating arbitrary capabilities. This can be accomplished by placing capabilities in special capability segments such that user can not access. Another approach is to add a tag bit to each primary storage location. This bit, inaccessible to the user, is set on if the location contains a capability if the bit is on. The hardware restricts the manipulation of the locations contents to appropriate system routines. The features include:

- ROBUSTNESS: To provide defense against vulnerabilities with few false positives or false negatives.
- FLEXIBILITY: It can adapt easily to cover the continuously evolving threats.
- END-TO-END: The security policy flows throughout all the seven layers of the OSI model.
- SCALABILITY: It can co-exist with the existing circuitry without any modification.

The data flow is shown in Figure 14.

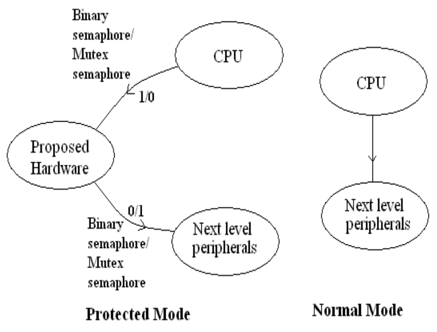


Figure 14 Data flow with hardware in loop

7.4 Snoop Cycles

Snoop cycles are used to probe the first level and second level caches when a Peripheral Component Interface (PCI) master attempts to access main memory. This is done to maintain coherency between the first and second level caches and main memory. When a PCI master first attempts to access main memory a snoop request is generated inside the PCMC. The PCMC supports up to two outstanding cycles on the CPU address bus at a time. Outstanding cycles include both CPU initiated cycles and snoop cycles. Thus, if the Pentium processor pipelines a second cycle onto the host address bus, the PCMC will not issue a snoop cycle until the first CPU cycle terminates. Thus, a snoop request is serviced with a snoop cycle only when either no cycle is outstanding on the CPU bus or one cycle is outstanding.

7.4.1 Data Transmission Reads from Main memory

Snoop cycles are performed by driving the PCI master address onto the CPU address bus and asserting EADSÝ. The Pentium processor then performs a tag lookup to determine if the addressed memory is in the first level cache. At the same time the PCMC performs an internal tag lookup to determine if the addressed memory is in the second level cache. Here, PCI master reads functions from main memory for data transmission is shown in Table 1.

Table 1 Data Transfers for PCI Master Reads from Main Memory

SNOOP RESULT (First Level Cache)	**ACTION**
Miss	Data is transferred from DRAM to PCI.
Miss	Data is transferred directly from second level cache to PCI. The line remains valid and unmodified in the second level cache.
Miss	Data is transferred directly from second level cache to PCI. Line remains valid and modified in the second level cache. The line is not written to DRAM.
Hit Unmodified Line	Data is transferred from DRAM to PCI.
Hit Unmodified Line	Data is transferred directly from second level cache to PCI. The line remains valid and unmodified in the second level cache.
Hit Unmodified Line	Data is transferred directly from second level cache to PCI. Line remains valid and modified in the second level cache. The line is not written to DRAM.
	A write-back from first level cache

Hit Modified Line	occurs. The data is sent to both PCI and the CPU-to-Memory Posted Write Buffer. The CPU-to-Memory Posted Write Buffer is then written to memory.
Hit Modified Line	A write-back from first level cache occurs. The data is posted to PCI and written into the second level cache. When the second level cache is in write-back mode, the line is marked modified and is not written to DRAM. When the second level cache is in write-through mode, the line is posted and then written to DRAM
Hit Modified Line	A write-back from first level cache occurs. The data is posted to PCI and written into the second level cache. The line is not written to DRAM. This scenario can only occur when the second level cache is in write-back mode.

7.4.2 Data Transmission Writes from Main memory

PCI master write cycles never result in a write directly into the second level cache. A snoop hit to a modified line in either the first level or second level cache results in a write-back of the line to main memory. The line is invalidated and the PCI write to main memory occurs after the write-back completes. A PCI master write snoop hit to an unmodified line in either the first level or second level cache results in the line being invalidated. For data transmission, PCI master reads functions from main memory are shown in Table 2.

Table 2 Data Transfers for PCI Master Writes from Main Memory

SNOOP RESULT (First Level Cache)	**ACTION**
Miss	The PCI master write data is transferred from PCI to DRAM.
Miss	The PCI master write data is transferred from PCI to DRAM. The line is invalidated in the second level cache.
Hit Unmodified Line	The line is invalidated in both the first level and second level caches. The PCI master write data is written to DRAM.
Hit Unmodified Line	The first level cache line is invalidated. The second level cache line is written back to main memory and invalidated. The PCI master write data is then written to DRAM.
Hit Modified Line	The first level cache line is written back to DRAM and invalidated. The PCI master write data is then written to DRAM.
Hit Modified Line	The first level cache line is written back to DRAM and invalidated. The second level cache line is invalidated. The PCI master write data is then written to DRAM.
Hit Modified Line	The first level cache line is written back to DRAM and invalidated. The second level cache line is invalidated. The PCI master write data is then written to DRAM.

8. RESULTS AND DISCUSSION

The layout of an evolved processing element that performs the function of semaphore switching for the complete data lines is shown in figure 15. The Microwind design tool package is used for study. The power consumed by this module alone is also shown in figure 16 separately to illustrate the fact that this module is the one that consumes maximum power.

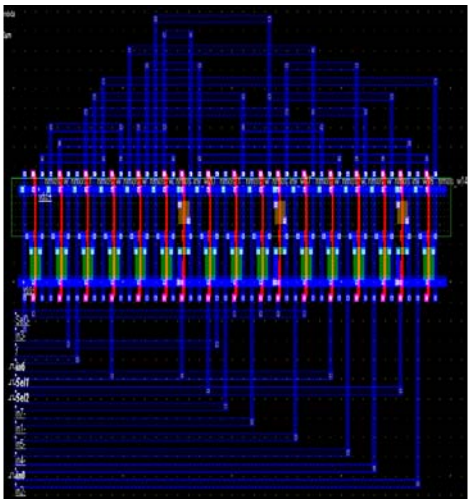


Figure 15 Layout of an evolved semaphore selection processing element (PE)

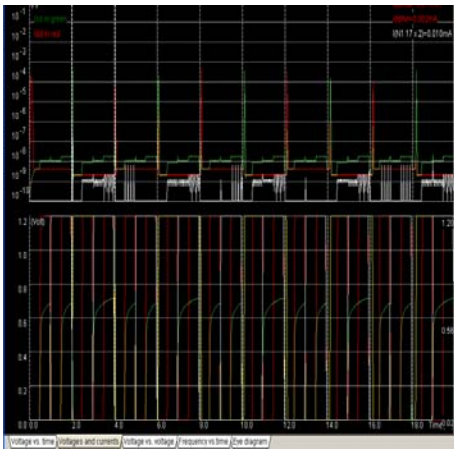


Figure 16 Power consumed by the semaphore selection PE

A single 12-T based SRAM module evolved is shown in figure 17. The complete SRAM cell used in the chip consists of several such SRAM cells and the complete schematic is shown in figure 18. The dynamic power variation of the switching gate in the SRAM cell is shown in figure 19.

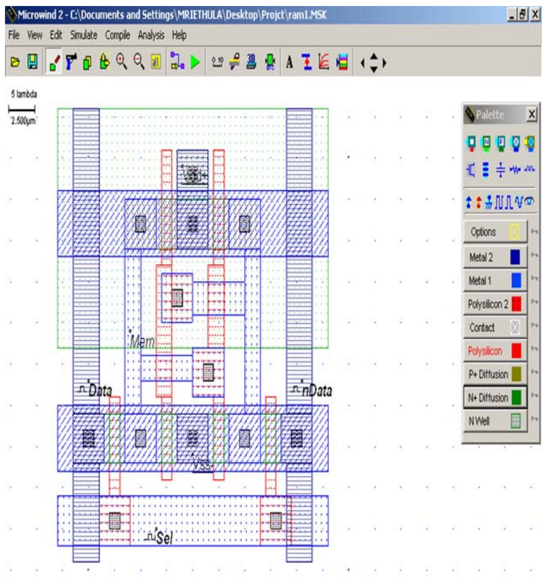

Microwind 2 - C:\Documents and Settings\MRIETHULA\Desktop\Projct\ram1.MSK
File View Edit Simulate Compile Analysis Help
Mem
Data
nData
Sel
VSS
Palette
Options
Metal 2
Metal 1
Polysilicon 2
Contact
Polysilicon
P+ Diffusion
N+ Diffusion
N Well

Figure17 SRAM cell layout

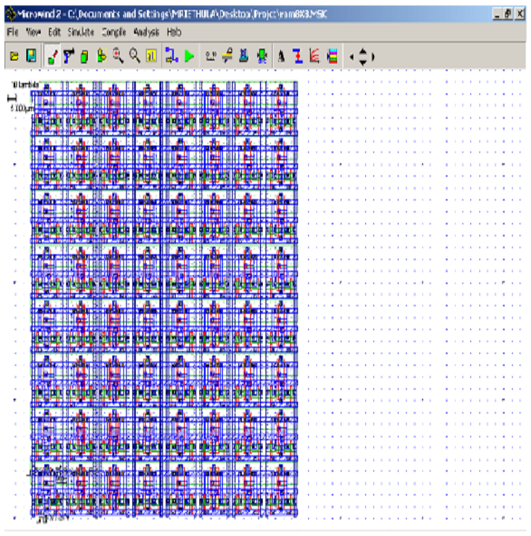


Figure 18 Complete SRAM cell layout

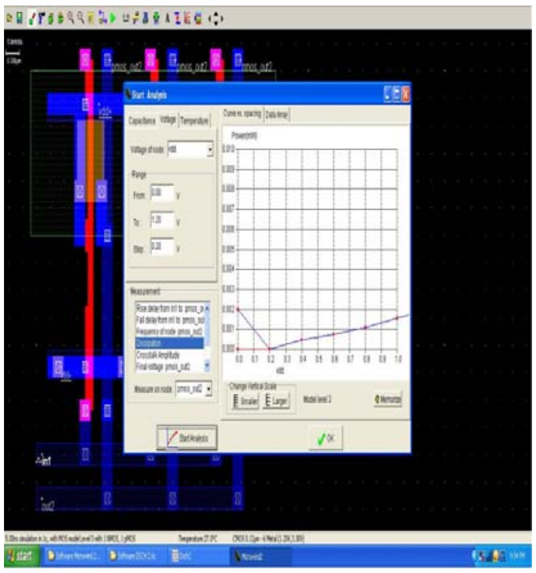


Figure 19 Power variation of switching gate used in SRAM design

9. CONCLUSION

This work provides an ideal architecture for data back-up using hardware. Security is provided independently, since, the user does not actually interact with the HDD. **It provides OS flexibility and it is possible to retain the user friendliness of any operating system.** It provides an easy and uncomplicated method of information recovery from virus infections. As a future scope, security provision can be made selective, resulting in the option of both archive and discard of the modified files that reside in different protected drives. Protection includes security threats to a network, unauthorized access to the system resources and subsequent permanent data modification, virus attack, e-mail exploitation threats to a network from both external and internal users.

REFERENCES

[1] Ilgun, K., “USTAT: A Real-time Intrusion Detection System for UNIX”, in proceedings of the IEEE Symposium on Research in Security and Privacy, pp. 16-28, 1993.

[2] T. Yamauchi et al., “Fully Self-timing Data-Bus Architecture for 64-Mb DRAMs”, IEICE TRANS. ELECTRON., VOL. E78-C, NO.7, pp. 885-865, 1995.

[3] Peter Petrov and Alex Orailoglu, “Towards effective embedded processors in codesigns: Customizable partitioned caches”, in proceedings of the 9th International Symposium on Hardware/Software Codesign, pages 79-84, Copenhagen, Denmark, 2001.

[4] W.T. Shiue and C. Chakrabarti, "Memory exploration for low power, embedded systems", in Design Automation Conference, pages 140-145, New York, 1999.

[5] W. Wolf and M. Kandemir, "Memory system optimization of embedded software", in proceedings of the IEEE, 91(1), January 2003.

[6] A. Asaduzzaman and I. Mahgoub, "Cache Optimization for Embedded Systems Running", AVC Video Decoder, AICCSA06 IEEE International Conference, UAE, pages 665-672, 2006.

[7] S. Mohanty, V.K. Prasanna, "Design of High-Performance Embedded System using Model Integrated Computing", 10th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS-2004), 2004.

[8] P.R. Panda, F. Catthoor, P.G. Kjeldsberg, "Data and Memory Optimization Techniques for Embedded Systems", ACM Transactions on Design Automation of Electronic Systems, Vol. 8, No. 2, pages 149-206, 2001.

Mr.Dilip Thomas is presently a research scholar in the faculty of ECE, Dr.M.G.R. University, Chennai-95. His active areas of research include Network security, Compiler design and Embedded systems.

Dr.K.S.M.Panicker is presently the Dean Research and planning, Federal Institute of Science and Technology, Kerala. He completed his Ph.D. from IIT Delhi and has got nearly 36 years of academic experience. He has published nearly 40 papers in international journals and conferences. His active areas of research include Power optimization techniques, Nanometer scale design, cluster computing, etc. He is a fellow of professional bodies like IETE, IEEE (USA) and IE.