Semi-Parallel Complementary Computer System

As

Optimal Protection against Malicious Software

Implications

Version 1.1

By

Marjan Lipovsek
B.Sc. of Electrical Engineering

Kamnik
Sunday, December 25th, 2005

i. Abstract

When one reads articles about various aspects of IT security they address nothing but computer software or computer user deficiencies and how they cause security breaches. In contrast this article focuses on computer hardware and software security aspects only. What are the true weak points that make a computer so vulnerable to any kind of attack? Are there options to eliminate these weak points? Yes, there is simple and straightforward method to protect a computer. Its implementation would significantly improve computer security, even though the task to implement it is huge.

Because of it nature computer security is not hardware or software issue only but both. Therefore hardware and software is to be redesigned. Because of countless software simulation options pure software protection solution is theoretically possible.

ii. Disclaimer

This article is definitely not written to discredit, offend or make damage of any kind to anybody.

Any similarity of drawn screenshots on Figures 14, 15, 16, 17, 18 and 19 to any other screenshots is unintentional and accidental. They are not to be understood in any comparative way either. They are drawn just to ease the explanation of a new idea.

iii. Outline

1 Introduction

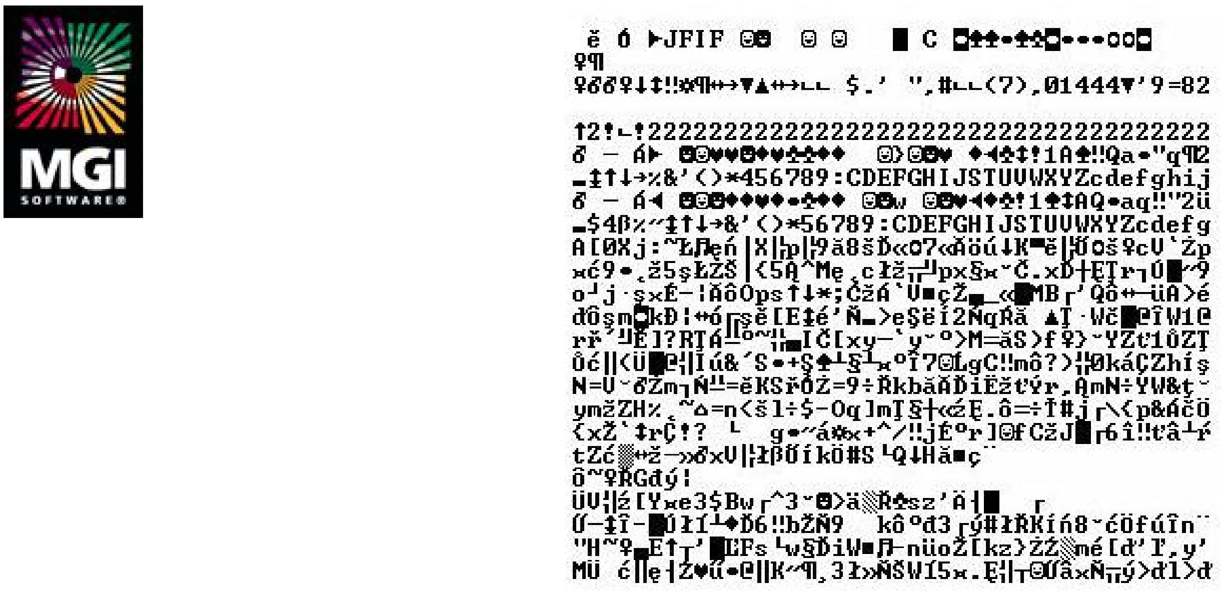
It is a sad fact that market competition so thoroughly dictates and manages implementation of new technologies and methods to improve existing technologies. Everything is ruled by the saying: "First come first served", too. It is vital and a matter of survival to be first on the market. And consequences are clear for all to see. New products are plagued by various faults because of to hasty design and lack of testing.

But to focus ones attention on computers only, there is a pure human factor that affects computer performance through hardware design on the one hand and software design on the other. Hardware designers have advantage because they work with something material what they can measure and easily express in numbers or test with relatively straightforward software step by step. In contrast to them software designers are dealing with immaterial world because, software is a world of ideas which they can't see, taste, hear, smell, touch, measure, but have to get them organized. And to have the ideas organized it is essential for everybody to understand the ideas the same way. So, how can a software developer describe an idea to others in such way that everybody else would understand it the same way? It is an impossible task and one of the fault generators. Furthermore, people involved in software developing and writing will fail in any imaginable way when they get tired because, they have to memorize too much during their work. In addition to that software packages are getting enormously huge and impossible to be adequately surveyed. Work is also often hindered with somebody else's business property or secrets. All that generates deficiencies and bugs what consequently decrease overall security.

On top of that computer protection software is installed to protect computer more or less successfully against viruses, worms, spy ware etc. Furthermore this protection software lames computer performance, too.

It is impossible to successfully improve computer security without analyzing malicious software activities in the attacked computer. What malicious software does is changing computer configuration in any imaginable way, destroying data, spying, data logging, etc. And what really does the damage is nothing but executables of any kind, not stored but running ones.

What runs a computer as it is, are numbers. And the way numbers are organized, processed and interpreted does the job. Neglecting few facts the very same numbers could be a picture, text, a piece of SW, etc. as depicted on Figure 1.



```
FF D8 FF E0 00 10 4A 46 49 46 00 01 02 00 00 01 00 01 00 00 FF DB 00 43 00 08 06 06 07 06 05 08 07 07 07 09 09 08
0C 14
0C 0B 0B 0C 19 12 13 0F 14 1D 1A 1F 1E 1D 1A 1C 1C 20 24 2E 27 20 22 2C 23 1C 1C 28 37 29 2C 30 31 34 34 34 1F 27 39 3D 38 32

18 32 21 1C 21 32 32 32 32 32 32 32 32 32 32 32 32 32 32 32 32 32 32 32 32 32 32 32 32 32 32 32 32 32 32 32 32 32 32 32 32 32
0B FF C4 00 B5 10 00 02 01 03 03 02 04 03 05 05 04 04 00 00 01 7D 01 02 03 00 04 11 05 12 21 31 41 06 13 51 61 07 22 71 14 32
16 17 18 19 1A 25 26 27 28 29 2A 34 35 36 37 38 39 3A 43 44 45 46 47 48 49 4A 53 54 55 56 57 58 59 5A 63 64 65 66 67 68 69 6A
0B FF C4 00 B5 11 00 02 01 02 04 04 03 04 07 05 04 04 00 01 02 77 00 01 02 03 11 04 05 21 31 06 12 41 51 07 61 71 13 22 32 81
16 24 34 E1 25 F1 17 18 19 1A 26 27 28 29 2A 35 36 37 38 39 3A 43 44 45 46 47 48 49 4A 53 54 55 56 57 58 59 5A 63 64 65 66 67
41 5B 30 58 6A 3A 7E 9D 0E A9 E4 B3 58 CC 70 CC 39 C7 38 E7 D2 AE 09 37 AE C6 94 A3 19 4B DF D8 CC 8A 09 E7 0C 63 56 60 BD 70
CF 86 39 07 F2 A7 35 AD 9D BD E6 B3 7B 35 A4 5E 4D A9 F2 63 88 A7 CB BC 70 78 F5 CF F3 AC 2E 78 D2 C5 A8 DD 72 BF E9 DB F1 39
6F D9 6A FA AD 9E 90 2D 7C C6 93 4F 70 73 18 19 2A 3B 8F A7 B5 60 56 FE 87 A6 DC 5F AE DB 4D 42 DA 27 51 93 1D C4 81 41 3E 82
D4 E2 AD 6D 08 6B D1 7C 1D A2 C9 AD D8 5B 45 17 82 27 D5 16 3E 65 B8 89 D6 32 E3 71 E8 C7 00 1E DD FA 57 9F DB 40 D7 57 31 40
72 FD EF BC B7 5D 3F 52 DD B5 CA F8 7E CE DC 49 AC 5B 78 79 C4 60 79 F3 F8 3E 4D CD C7 53 29 66 0C 7D F3 59 5A 9C 31 DE 5A DD
DE 86 BA 28 9A DB 40 B9 49 A3 26 EF 53 07 2B B8 06 C1 F5 C1 CF F8 D7 37 01 91 67 43 13 6D 93 3F 29 CE 30 6B A0 80 5A 68 D6 AD
4E 3D 56 F4 0B BD 6D BF E3 CA 3D D8 4B 53 FD E0 BD 3D 39 F6 FC 6B 62 C7 C6 D2 69 D3 A7 9C ED 72 2C A4 6D 4E F6 59 57 26 EE F4
79 6D A7 5A 48 25 F2 7E 7F 3D 6E 3C E7 6C F6 24 F0 4F 71 5D 6D DD F5 C5 AE AB A8 2E 93 CD F6 9B 23 6A DA 5C 7B 70 26 B5 9F 99
7B 78 A6 60 12 72 80 21 3F 20 C0 FF 00 67 07 F1 A0 0F 9E 2B 5E 2F 13 6A 90 F8 72 5D 01 66 43 A7 4A DB DA 36 8C 13 9C 83 C1 EA
74 5A 86 B1 1D A7 C4 AF 0B 78 56 CC 88 E1 8A D6 6B 99 23 53 C0 51 19 48 C7 FE 87 F9
E2 7E 0C FC 47 D0 EC 7C
9A 56 B9 AB 5B 59 CF 65 33 24 42 77 DA 5E 33 F3 02 3E 84 B0 FC 05 73 7A 27 8E B4 DB FF 00 DA
EB C4 17 D7 F0 DB E9 88 92 C1 04 D2 36 13 62 A6 D5 39 FF 00 6B 93 F8 D0 33 DA EC 23 88 FC 4B D6 E4 38 F3 86 99 66 A3 D7 6E F9
22 48 7E 0C DC 45 18 C2 27 DB 95 46 73 C0 77 15 D2 69 57 FE 0E F0 6E 81 6F A6 5B 6B 7A 7D BD 8D B0 6D 82 5B D4 27 96 2C 79 27
4D 9A 00 86 BA A9 B4 8D 03 FB 07 40 BA 4B F1 14 F7 33 88 AF D5 E6 57 92 35 CF 2E A8 B9 C2 01 EB 83 9E D5 CB EC 3E D4 6C 3E D4
```

Figure 1 A same data interpreted in different ways

Filtering through all those huge amounts of data searching for a malicious SW consumes too much computer power and resources. Therefore it is a high time to adapt a forward defense because playing the hide and search game is a game lost. After the damage has been done no protection updates can reverse it. A computer should be somehow forced into forward defense to rebuff any attempt SW, configuration or data being illegally changed or accessed. To adapt a forward computer defense it is essential to split applications or utilities that are being executed on the computer in two groups:

- Applications and utilities to serve user's needs
- Computer security issues

Even more, various software functions should be adapted. Not the tasks they are accomplishing but how a particular task is being accomplished.
At the same time a couple of questions arise immediately:

- Is special protection software really indispensable?
- Is it possible to develop a protection method that would be operating system independent at all?

What it really needs to be protected is CMOS, system memory and mass storage media. And that could be done without special protection software. The question is: how to safely run applications?

1.1 Computer safety analogy

To effectively safeguard a computer system against malicious software activities, such software should be disabled to access various memory and storage areas. At the first glance this is an impossible task because access disabling would disable or at least hinder machine performance. Even harder task to accomplish is to distinguish between particular software: genuine or hostile.
This means that simultaneously access should be allowed or disallowed to same areas so, the computer can run it tasks at all. Table 1 presents the idea how access rights should be granted.

Table 1 SW Access

Hostile Software	No Access	**BIOS**	Read/Write	**Genuine Software**
	No Access	**CMOS**	Read/Write	
	No Access	**BOOT SECTORS**	Read/Write	
	No Access	**Operating System**	Read/Write	
	No Access	**Application SW**	Read/Write	
	No Access	**Configuration Data**	Read/Write	
	No Access	**Data Files**	Read/Write	
	No Access	**Confidential Data**	Read/Write	
		****************		
		****************		

To be able to distinguish between genuine and hostile software one should just imagine there is something to be designed mimicking a high security bank. In the bank there are a bullet-proof wall with a drawer and intercom placed between a customer and a clerk. All items that supposed to be exchanged are exchanged via special drawer. The intercom provides for conversation that is needed. A clerk decides how and which items the customer will access if any at all.
The protection is to be searched for in hardware as well as in software design. Therefore it is technologically feasible to distinguish between genuine and hostile software if one combines two computers and places a semi-transitional media between them. As already mentioned two computers are mandatory: a Master Unit and a Slave Unit. Table 2 presents the idea how access rights should be granted. Furthermore Master Unit installs and configures SW for both Units, saves all data files and grants access to any data to Slave Unit according predetermined way. In contrast to that Slave Unit can only read SW files to independently boot-up and run Applications selected and supervised by the Master Unit. All Application's file manipulation is executed in Slave Unit and stored in Master Unit.
All file transfer and any other communication is done via semi-transitional media between Units and perhaps some electrical signals, control bus.

Table 2 Data Access Rights

Slave Unit System	No Access	**BIOS-master**	Read/Write	**Master Unit System**
	Read	**BIOS-slave**	Read/Write	
	No Access	**CMOS-master**	Read/Write	

	Read	**CMOS-slave**	Read/Write	
	No Access	**BOOT SECTORS-master**	Read/Write	
	Read	**BOOT SECTORS-slave**	Read/Write	
	No Access	**Operating System-master**	Read/Write	
	Read	**Operating System-slave**	Read/Write	
	No Access	**Application SW-master**	Read/Write	
	Read	**Application SW-slave**	Read/Write	
	No Access	**Configuration Data-master**	Read/Write	
	Read	**Configuration Data-slave**	Read/Write	
	No Access	**Data Files-master**	Read/Write	
	Read	**Data Files-slave**	Read/Write	
	No Access	**Confidential Data-master**	Read/Write	
	Not Existing	**Confidential Data-slave**	Read/Write	
		**** * * * * * * * * * * * * * * *		
		* * * * * * * * * * * * * * * *		

Besides two computers a semi-transitional media between them is mandatory as well. In the bank example semi-transitional media would be a drawer chest built in the bullet-proof wall. Because two computers have huge amounts of items to be exchanged something mimicking a drawer chest (Figure 2) is needed. In a computer system anything working as dual-port RAM could be implemented.

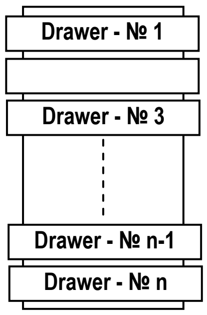


Figure 2 Drawer chest

Nevertheless in a described computer system there are four common areas where both Units come in contact on data level:

- BIOS and CMOS (to be addressed later at actual designing)
  - BIOS placed in Slave Unit EEPROM should be programmable from Master Unit only to simplify procedure
  - CMOS placed in Slave Unit EEPROM should be changed from Master Unit only to simplify procedure and prevent malicious SW to change it
- main system memory
- mass storage media
- video RAM (to be addressed later at actual designing)
  - GUI is composite of Master Unit's GUI and Slave Unit's GUI displayed as single picture on the PC monitor screen

A general idea is to introduce hardware drawers (anything that works like dual-port RAM) in computer data exchange to protect the system. Exact locations in dual-port memory are to be determined for data exchange to take place (data files, display data, etc). That would enable Master Unit to accept data at very precise locations and interpret it accordingly and consequently: activated malicious software would have only uncritical and reversible effect on computer.
And Unit to Unit communication and control should look something like Figure 3:

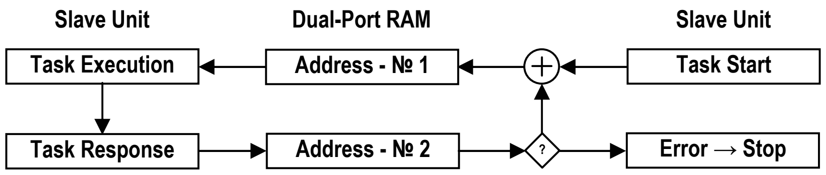


Figure 3 Execution and control loop

To achieve that kind of protection a semi-parallel complementary computer configuration is mandatory:

Semi-parallel configuration because two computers run some tasks in parallel almost independently

- Operating system

Complementary because some functions could be accomplished only by complementing each other tasks:

- SW tools and utilities

Computers share only limited and exactly defined areas of:

- System memory (which could be accessed only alternatively)
- Mass storage media (which could be accessed only alternatively)
- GUI is special composite of both systems

Computers run in a master-slave relationship. Slave computer is to run applications while Master one is to protect the system, stores data files and monitor Slave's behavior.

Configuration benefits:

- Internet surfing can't damage computer system in any way
- Opening E-mail and can't damage computer system in any way
- Perfectly safe file saving, opening, editing, closing
- Perfectly safe configurations of any kind
- Perfectly safe any stored data: system or user
- Prevents all intrusions
- Prevents spying, data logging, etc.
- Splitting tasks could potentially lead to simplified applications

Configuration protection limits:

- Running application could hang under influence of malicious software. Exiting and starting the application again or resetting a computer would cure the problem. Infected data file would be lost.
- Computer could hang under influence of malicious software. Resetting a computer would cure the problem. Infected data file would be lost.

Possible configuration deficiency:

- Such configuration could make remote computer administering a problematical case.
- File and directory sharing could be problematical too.

Implementation of described changes combined with secure IP protocols (IPSec) could have great implications in electronic banking, business, etc.

A very same configuration is applicable and effective cell phones.

2 Computer security weak points definition and their elimination options

Various types of data are being transferred between the computer and server or servers after starting a network connection. The first major security weak point is data transfer between computers utilizing direct software - software interaction. Through this interaction can a malicious software package intrude particular computer, access memory and mass storage media and self installs there. The second major security weak point is that infected files are received in and then opened, saved, edited, etc. The problem is again a possibility that malicious software directly interacts with system and application software and alters them. And in this direct access to computer system memory and mass storage media is true security weak point because malicious software can access and change CMOS, memory, master boot record, system storage areas, data areas, take over the computer, collect and send data to somebody outside, etc. Therefore this ability to illegally change the system is exactly what could one define as computer true security weak point.

Therefore, a design is to be searched for to prevent malicious software from accessing computer sensitive or confidential areas. The only way to achieve this is to implement file transfer media (Figure 4) that serves as mediator between two computers. So, a file can be transferred from the 1st computer to the 2nd one via file transfer media:

- 1st computer places a file on the file transfer media
- 2nd computer picks a file or parts of a file from the file transfer media
- Data transfer "protocol" is transferred the same way
- The transfer works vice-versa as well
- Special software drivers to execute file transfer
- Special software drivers to enable control over the file transfer

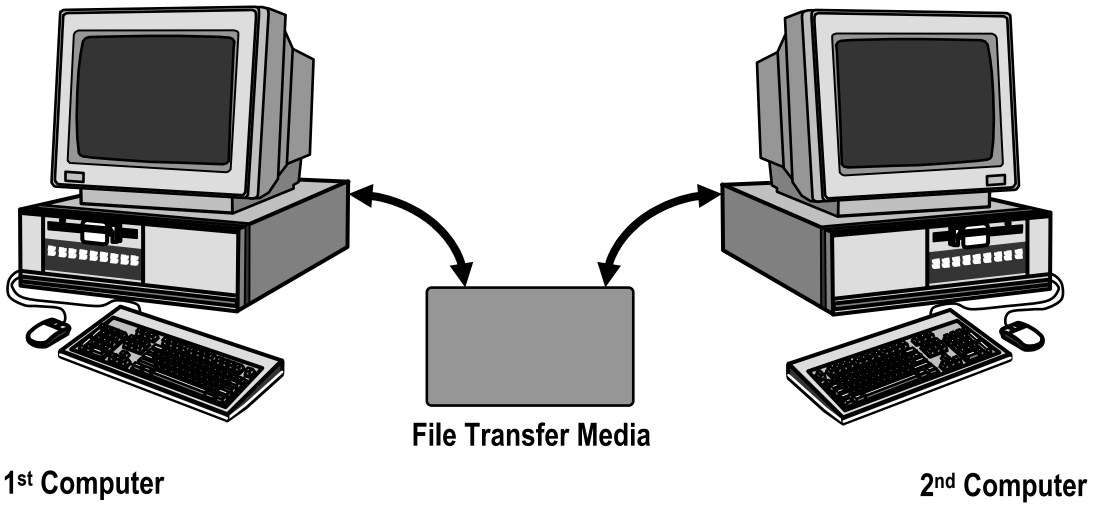


Figure 4 File Transfer Media Implementation

The first step is achieved. The second computer is indirectly connected to network and safe from the network attack. Of course a real configuration should be different, but this one helps to describe the general idea.

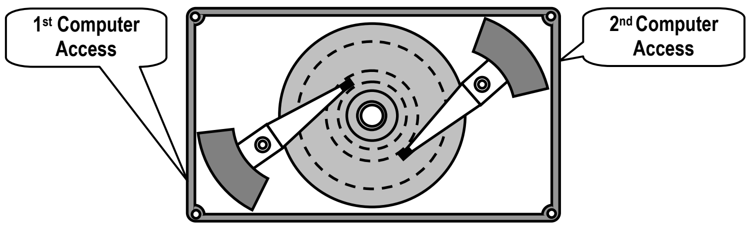


Figure 5 File Transfer Media → Idea

At this point Figure 5 depicts the idea of hard disk serving as File Transfer Media where indirect transfer is achieved by putting files in a mailbox (an exactly defined one: a directory) by one side and picking it out by another: in a sort of handshake mode. But later it will be described the true potential of double set of read-write heads combined with multiple logical disks. The 2nd computer would control access to these logical disks. The 2nd computer could have full access in contrast to 1st computer's partial one. This configuration would be ideal for the laptops because of limited space.

3 Defining a usable configuration

First of all, the configuration depicted in Figure 4 is not practical because uses to much space, it is too expensive and a single user don't need two fully functional computers at all. But configuration as it is, is not to be just waved off because it could be a good solution in future exploitation of different types of networks. From now on the 1st computer will be called Slave Unit and the 2nd one will be called Master Unit. To overcome shortcomings of Figure 4's configuration a further steps should be made:

- One unit runs applications while another stores data and controls executions
- Any data transfer is accomplished via File Transfer Media
- Slave Unit should run all applications that open, close and edit files and runs all network applications
- Slave Unit needs enough memory, enough virtual memory and enough temporary storage area
- Slave Unit system storage area should be read only
- Slave Unit shouldn't store any data files and shouldn't access any
- Master Unit should control all applications running in Slave Unit
- Master Unit needs enough memory, enough virtual memory and enough storage area because stores all data and configuration files, logins, passwords, etc.
- Master Unit should install software in Slave Unit
- File Transfer Media between Master and Slave Units should be dual-port RAM
- Special drivers to execute and control file transfer; application execution; monitor Slave Unit; etc. (handshake)

4 File Transfer Media

The purpose of this particular device is to prevent malicious software to access protected data of any kind and enable unhindered application execution and data storage. File Transfer Media is in reality a hardware firewall. Hard disk (or any other mass storage media) from Figure 5 is too slow to be really usable as File Transfer Media at least until it could compete with dual-port RAM's speed. Therefore dual-port RAM depicted on Figure 6 is to be used as File Transfer Media.

4.1 Ready available dual-port RAM

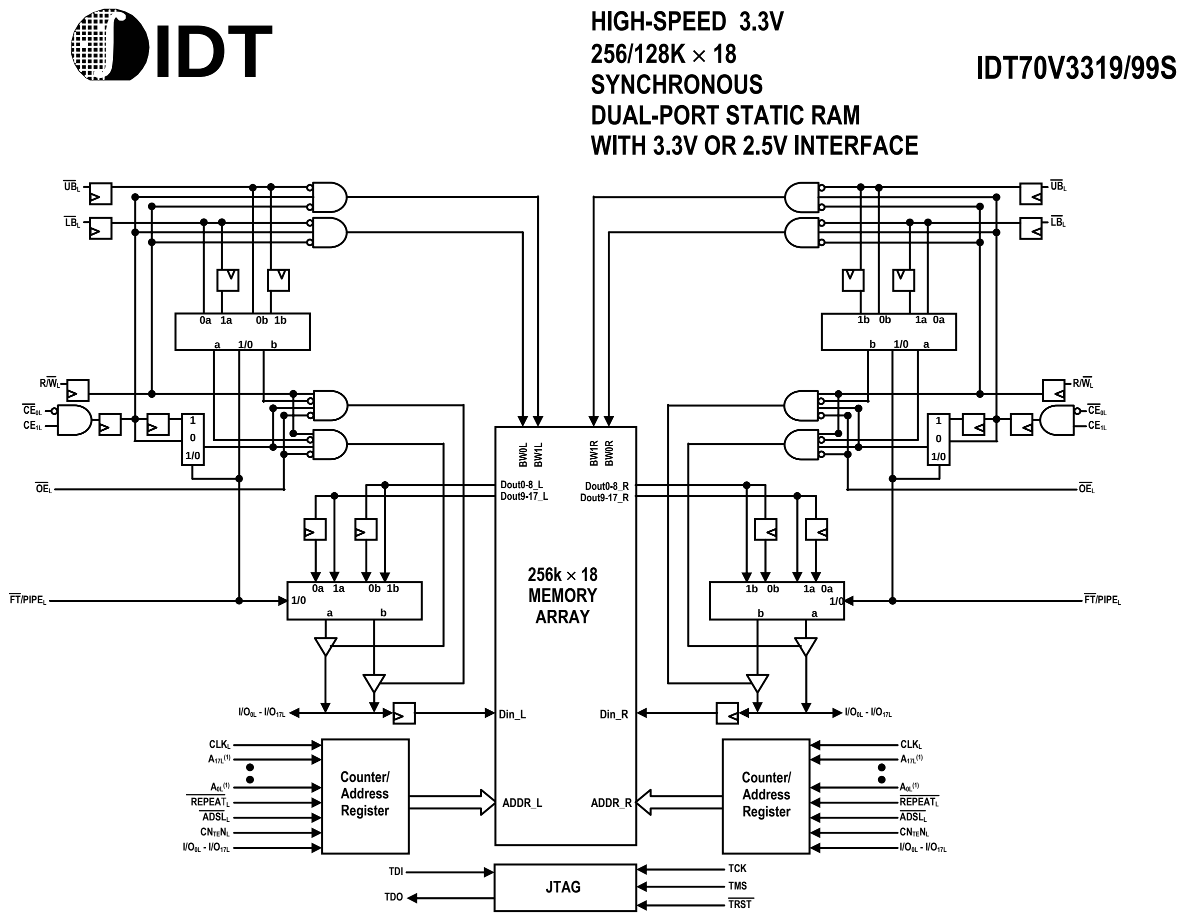


Figure 6 Dual-port RAM schematics

Features:

- True Dual-Port memory cells which allow simultaneous access of the same memory location
- High-speed data access
  - o *Commercial: 3.6 ns (166MHz)/4.2 ns (133MHz) (max.)*
  - o *Industrial: 4.2 ns (133MHz) (max.)*

- Selectable Pipelined or Flow-Through output mode
    - *Due to limited pin count PUFT option is not supported on the 128-pin TQFP package. Device is pipelined outputs only on each port.*
- Counter enable and repeat features
- Dual chip enables allow for depth expansion without additional logic
- Full synchronous operation on both ports
    - *6 ns cycle time, 166MHz operation (6 Gbps bandwidth)*
    - *Fast 3.6 ns clock to data out*
    - *1.7 ns set up to clock and 0.5 ns hold on all control, data, and address inputs @ 166MHz*
    - *Data input, address, byte enable and control registers*
    - *Self-timed write allows fast cycle time*
- Separate byte controls for multiplexed bus and bus matching compatibility
- Dual Cycle Deselect (DCD) for Pipelined Output mode
- LVTTL - compatible, single 3.3V (±150mV) power supply for core
- LVTTL - compatible, selectable 3.3V (±150mV) or 2.5V (±100mV) power supply for I/O's and control signals on each port
- Industrial temperature range (-40°C to +85°C) is available at 133MHz.
- Available in a 128-pin Thin Quad Flat pack, 208-pin fine pitch Ball Grid Array, and 256-pin Ball Grid Array
- Supports JTAG features compliant to IEEE 1149.1
    - *Due to limited pin count, JTAG is not supported on the128-pin TQFP package.*

This example proves that there is dual-port RAM available which performance could be adequate to be used as File Transfer Media and as handshake transfer media. It is just a bit slower than DDRAM already available.

4.2 Dual-port RAM implementation

Dual-port RAM should be place so that it is to be accessed by both Units as memory locations:

- When Slave Unit is designed as a snap-in card dual-port RAM is located on that card as part of Slave Unit memory system
    - Master Unit can access dual-port RAM via PCI bridge
- When Slave Unit would be a part of motherboard dual-port RAM is part of Master Unit memory system as well as part of Slave Unit memory system

4.3 Dual-port RAM located on snap-in card

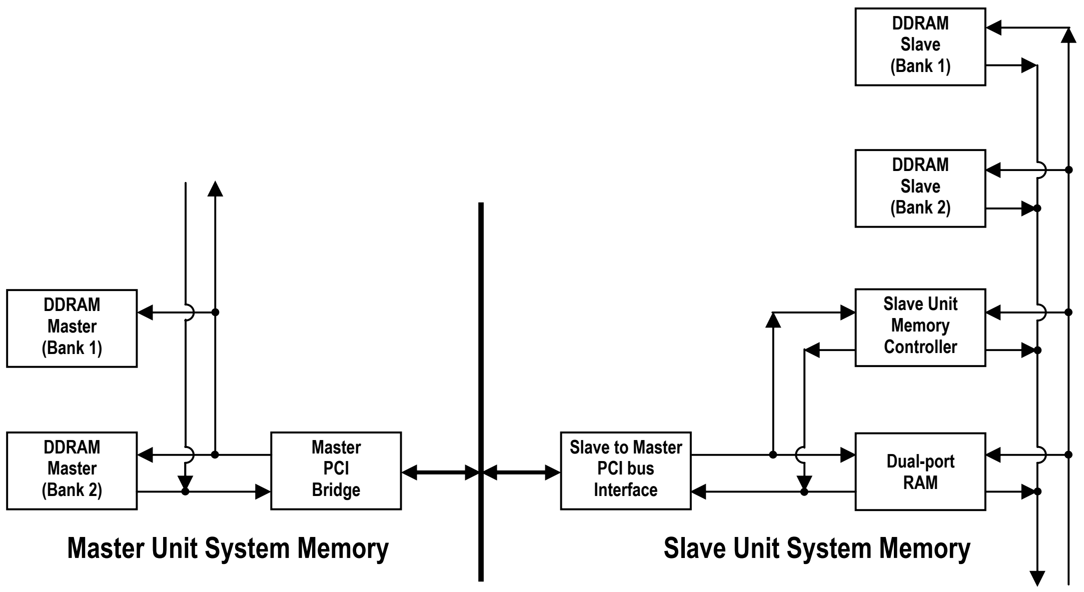


Figure 7 Dual-port RAM located on a snap-in card

Figure 7 depicts memory layout when Slave Unit is located on a snap-in card. Master Unit can access dual-port RAM via Master PCI bridge, PCI bus and Slave-to-Master PCI bus interface while Slave Unit accesses dual-port RAM directly. The only configuration drawback on the Slave Unit side is possible speed and access time difference between various types of RAM. So, appropriate design measures should be taken.

4.4 Dual-port RAM implementation when both Units are located on motherboard

In a case when both Units are located on motherboard the configuration on Figure 8 is suggested. Again the only configuration drawback is possible speed and access time difference between various types of RAM. So, appropriate design measures should be taken.

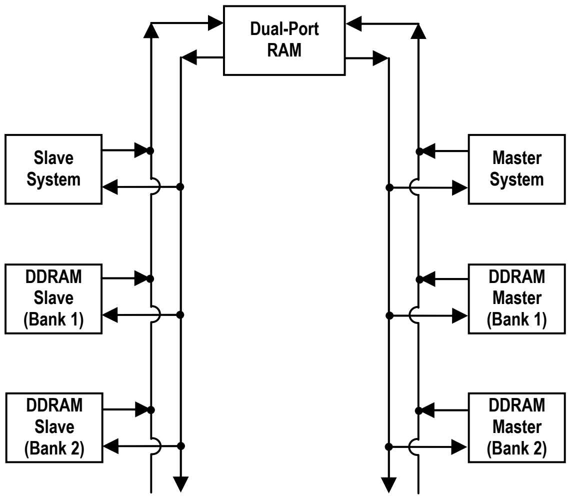


Figure 8 Master and Slave main memory configuration

4.5 Making DDRAM acting as system and dual-port RAM as well

In contrast to memory design deficiencies described previously all DDRAM design on Figure 9 has none of them, but memory sharing. A memory controller can make conventional DDRAM acting to both Units as dual-port RAM. Master Unit would program Memory Controller to allocate appropriate amounts of DDRAM to:

- Master Unit system,
- Slave Unit system and
- Dual-port RAM area.

System Request Queue would enable alternative access to memory for each Unit. After boot process is terminated allocated DDRAM is accessed as follows:

- Master Unit system memory is accessible to Master Unit only,
- Slave Unit system memory is accessible to Slave Unit only,
- Dual-port RAM can be alternatively accessed by both Units

A memory design to be implemented is yet to be assessed considering overall system performance.

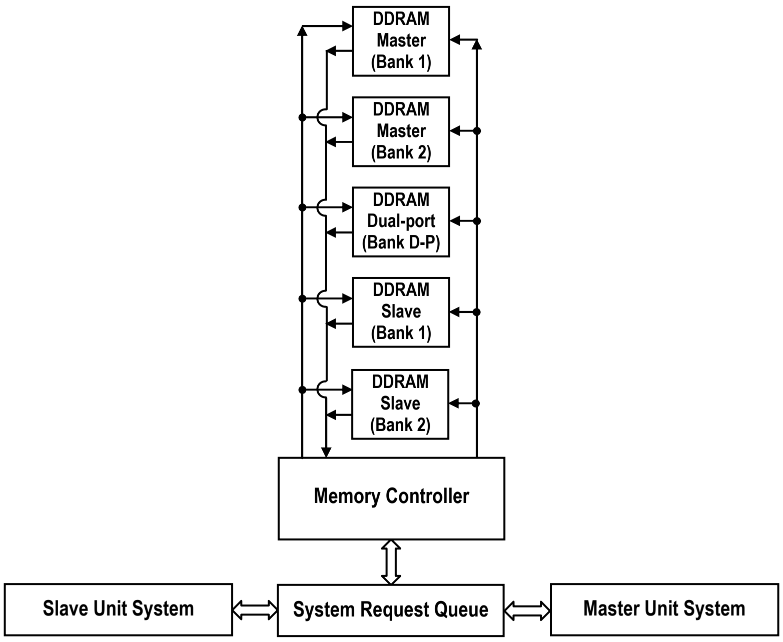


Figure 9 DDRAM acting as dual-port RAM and system memory to both Units

5 Mass storage media

Because there are two in a special way separated but fully operational computer units in one system each of them has to have its own mass storage media (hard disk).

5.1 Master Unit mass storage media

Master Unit mass storage media should be nothing but conventional mass storage media:

- To store operating system
- To store applications and utilities
- To store system data
- To store system configuration
- To store user data (files)

5.2 Slave Unit mass storage media

Slave Unit mass storage media should be a bit different than conventional mass storage media. On the one hand Slave Unit mass storage media should be read only to protect system from hostile attacks on the other hand it should be read--write so the system can run at all:

- There is read only area (partition), write option should be hardware disabled (by Master Unit):
    - To store operating system
    - To store applications and utilities
    - To store system configuration
    - To store system data
- There is read-write area (partition):
    - To act as virtual RAM
    - To store temporary data

When a file is open it is transferred via dual-port RAM from Master Unit mass storage media to Slave Unit mass storage media (read-write area) to be edited.
When a file is closed it is transferred via dual-port RAM from Slave Unit mass storage media (read-write area) to Master Unit mass storage media. Any adhering malicious software is made inactive. It's just a piece of junk adhering to data file.
When a file is saved it is transferred via dual-port RAM from Slave Unit mass storage media (read-write area) to Master Unit mass storage media. Any attached malicious software is made inactive. It's just a piece of junk adhering to data file.
Slave storage media should not store any user personal data (logins, passwords, etc) or any data considering user work.
Since system area is read only no malicious software can change Slave Unit configuration or affect its behavior after reboot. So, no infection can spread.
In event of hostile attack running application can hang or Slave Unit can hang but after reboot everything should be perfectly OK. There should be provided a means to install software on the Slave Unit and select its configuration.

5.3 Hard disk with double set of read-write heads

Two separated mass storage media performing as described in 5.1 and 5.2 is "not likely to be practical". But to accomplish demands described in chapters 5.1 and 5.2 a more radical design should be applied.
Hard disk hardware design should enable Master Unit to disable Slave Unit's read-write headset:

- To position itself over Master extended partition
- To write when positioned over Master boot partition

Therefore, design depicted on Figure 10 together with appropriate software should accomplish all demanded tasks when:

- 1st configuration: physical disk is partitioned to:
    - Master Boot partition
        - Is accessed in read-write mode by Master set of heads
        - Is inaccessible (hidden) to Slave Unit set of heads
        - Stores Master Unit's operating system
        - Stores Master Unit's applications and utilities
        - Stores Master Unit's configuration data
    - Master extended partition
        - Is Master Unit's system storage area
        - Is inaccessible (hidden) to Slave Unit set of heads
        - Is accessed in read-write mode by Master set of heads
        - Stores all confidential data (log ins, passwords, etc)
        - Stores all data files considering user work
    - Slave Boot partition

        - Is accessed in read-write mode by Master set of heads
        - Is to be accessed in read-only mode by Slave set of heads
        - Stores Slave Unit's operating system
        - Stores Slave Unit's applications and utilities
        - Stores Slave Unit's configuration data
    - Slave extended partition
        - Is accessed in read-write mode by Master set of heads
            - To potentially search for presence of illegal data or installations and delete them, because somebody could write a malicious software to clog Slave partition with garbage data, and by doing so crash Slave Unit,
            - Available to privileged users only, otherwise hidden
        - Is accessed in read-write mode by Slave set of heads
        - Is Slave Unit's temporary mass storage area
        - Is Slave Unit's system storage area

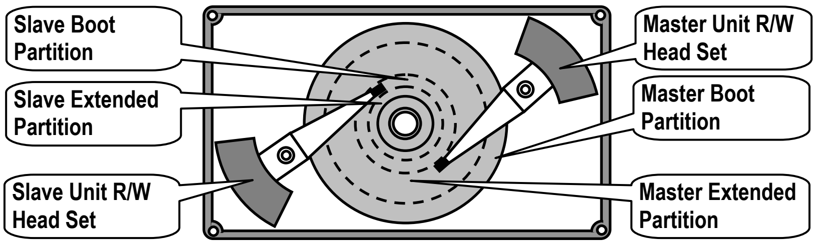


Figure 10 Double read-write head set HD → option 1

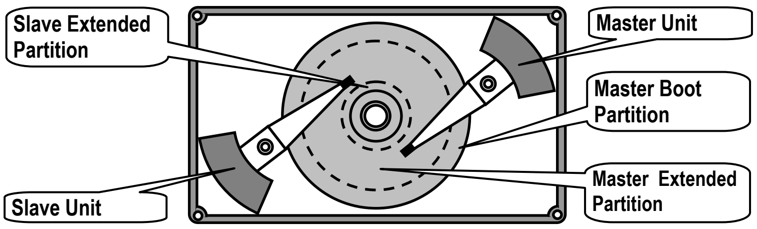


Figure 11 Double read-write head set HD → option 2

With some more endeavor more practical configuration from Figure 11 would be possible. It is obviously optimal and software installation as well as configuration friendly.

- 2nd configuration: physical disk is partitioned to:
    - Master Boot partition
        - Is accessed in read-write mode by Master set of heads
        - Is to be accessed in read-only mode by Slave set of heads
        - Stores operating system for both Units
        - Stores applications and utilities for both Units
        - Stores configuration data for both Units
    - Master extended partition
        - Is Master Unit's system storage area
        - Is to be physically inaccessible (hidden) to Slave Unit set of heads
        - Is accessed in read-write mode by Master set of heads
        - Stores all confidential data (logins, passwords, etc)
        - Stores all data files considering user work
    - Slave extended partition
        - Is accessed in read-write mode by Master set of heads
            - To potentially search for presence of illegal data or installations and delete them, because somebody could write a malicious software to clog Slave partition with garbage data, and by doing so crash Slave Unit,
            - Available to privileged users only, otherwise hidden
        - Is accessed in read-write mode by Slave set of heads
        - Is Slave Unit's temporary mass storage area
        - Is Slave Unit's system storage area

Such configuration simplifies software installation and is applicable to any kind of mass storage media. All installed software is accessible to both Units independently and simultaneously. The truth is that there should be some software adaptations implemented, too.
Options should be assessed is it feasible at all to increase data throughput by utilizing both sets of heads when such hard disk is used in conventional way.

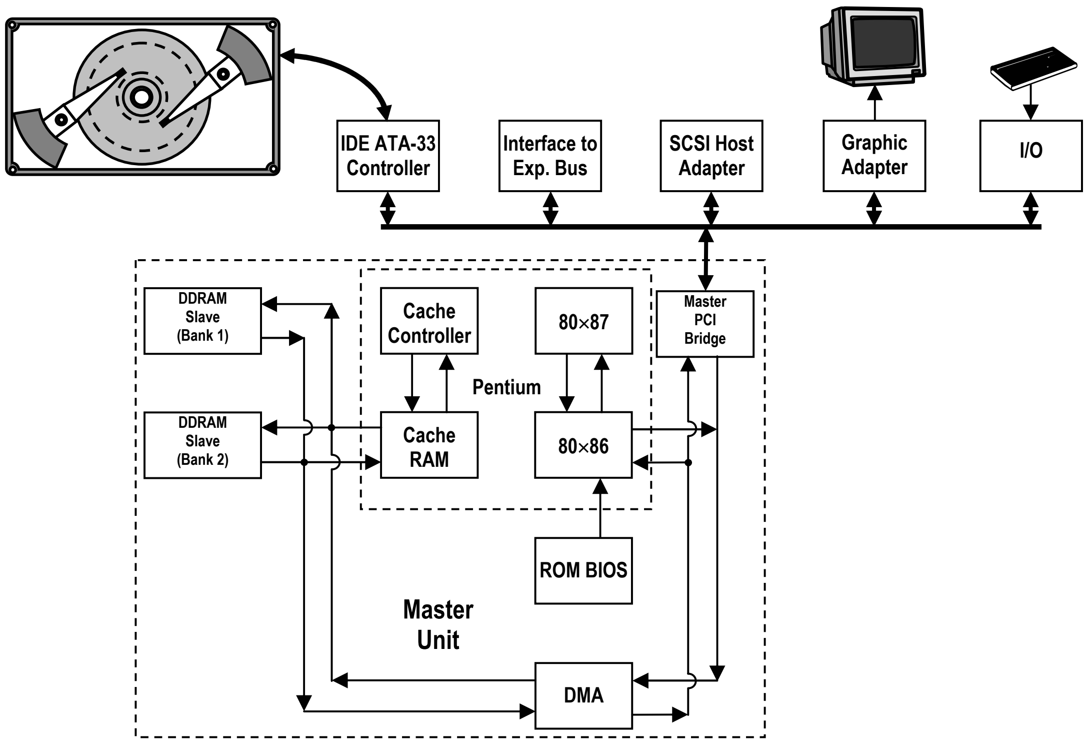


Figure 12 Master Unit → Conventional computer design

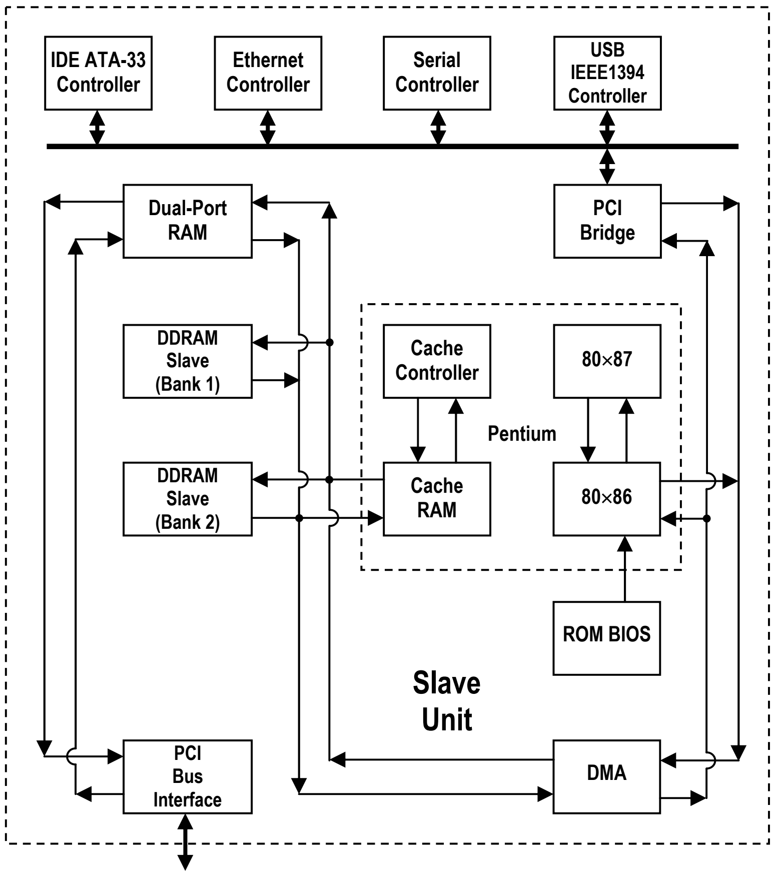


Figure 13 Slave Unit configuration → Snap-in card

6 Master Unit's Hardware Design

Master Unit purpose is to run Graphic User Interface, enable configuration of both Units, initiate and control all Slave unit functions, monitor Slave Unit overall behavior, store confidential data, store data considering user work, control file transfer between both units, control network communication, install all software running on both Units, etc. Master Unit's design should in general be similar to conventional computer one's (Figure 12).

7 Slave Unit Hardware Design

Slave Unit purpose is to run all user applications and utilities, but their execution should be absolutely controlled by Master Unit. Basically nothing could be started to run from the inside Slave Unit except tasks that would be needed for Slave Unit to be controlled and monitored by Master Unit. Files are to be transferred via dual-port RAM from Master Unit and temporary stored on Slave Unit mass storage media. Temporary stored file is to be opened and edited by application running in Slave Unit. To do that Master Unit keyboard buffer is to be mirrored dual-port RAM to Slave Unit. Data to be displayed by Slave Unit is to be transferred via to dual-port RAM to Master Unit graphic adapter video RAM. Special drivers should be implemented to make file and command transfers possible.
Because Slave unit would have to accomplish all the hard work it is expected that Slave Unit will need much rawer computer power than Master Unit.
But, except network controller and IDE ATA controller there is probably no other external device needed (Figure 13). Therefore it could be feasible to design and fabricate Slave Unit as a snap-in card. Because the purpose of Slave Unit is very specialized, it could be designed as single board computer (SBC) and snapped into Master Unit. And Slave Unit should be nothing but a very specialized Single Board Computer. An example of a small powerful SBC is depicted on Figure 14.

7.1 Slave Unit based on Single Board Computer (SBC) snap-in card

SBC example module (Figure 13) features show that snap-in card Slave Unit is technically feasible because majority of peripheral devices are not needed at all. One just needs a bit imagination to see that without majority of peripheral devices there is enough place for powerful processor, enough DDRAM, dual-port RAM, etc.
CM-i686M Highlights as a single board computer example:

- *Full-featured Embedded PC on module*
- *58 x 68 mm - smaller than a credit card!*
- *NS Geode CPU: x86 architecture, 266/300 MHz, 16 KB cache, FPU, MMX*
- *PCI, LPC, AC97 and ISA expansion busses*
- *32 - 128 Mbyte SDRAM*
- *1 - 512 Mbyte Flash Disk*
- *Graphics Controller with interface for TFT panels, NTSC/PAL TV and RGB color monitors see LCD panel support.*
- *Video Input port*
- *Host USB ports*
- *Sound codec with speaker and microphone support (optional)*
- *Standard peripherals: serial ports, LPT port, I/O ports, FDC, PS/2 keyboard and mouse, IrDA, HDD interface (optional)*
- *10/100BaseT Ethernet port (optional)*
- *Power consumption below 5W*
- *Interchangeable with other CORE modules via CAMI connectors*
- *SB-i686 - turns the CM-i686 module into a PC/104+ SBC*

The CM-i686 module *is a tiny single board computer, designed to serve as a building block in embedded applications. The CM-i686 module has all the components needed to run operating systems such as Windows, Linux or VxWorks. Ready packages for these operating systems are available from CompuLab.*
The CM-i686 is both small and inexpensive. Its small size allows integration into hand-held and mobile applications, providing a powerful computing core. Its extremely low price makes it an ideal selection for cost-sensitive applications. Based on AMD's GeodeTM architecture, the CM-i686 provides excellent price/functions/performance in comparison with any other PC-compatible embedded system.
The feature set of the CM-i686 module combines a 64-bit CPU memory architecture, Flash Disk and essential computing peripherals. For embedded applications, the CM-i686 provides a 32-bit PCI bus, 100Mbit Ethernet, serial ports, general purpose I/O lines and many other essential functions.
A latest version of the card - CM-i686-M uses standardized CAMI (CompuLab's Aggregated Module Interface) connectors which allows interchangeability with other CORE modules, enabling the flexibility required in a dynamic market where application requirements can change rapidly.

Software support for the GM-i686 includes ready-to-run packages for:

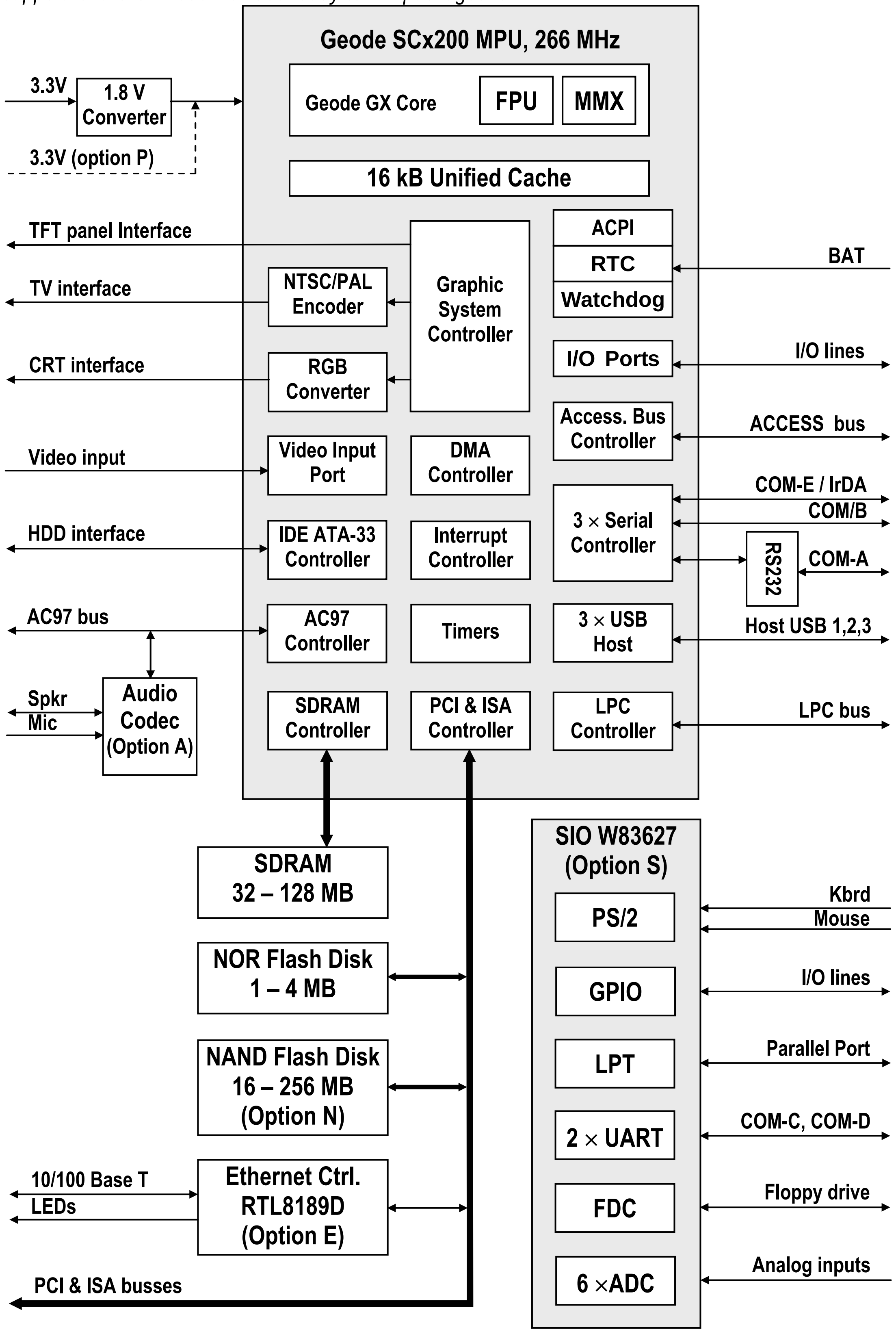


Figure 14 Single Board Computer (SBC) example

- *Linux*
- *Windows GE, NT, XP*
- *VxWorks*
- *DOS*

Covers CM-i686, SB-i686 & A TX-i686 products

CompuLab provides ready-to-run packages and support for several operating systems running on GM-i686 / SB-i686 / A TX-i686 products: Linux, VxWorks, Windows and others, including the ability to boot and run from on-board Flash Disk.

The table below specifies current and planned support coverage. This page is updated on any significant change of support availability.

8 Semi-parallel complementary computer system hardware

The idea of protection is in task division between Master and Slave Unit. Master Unit serves as I/O unit (keyboard, display, etc.), mass storage unit, control unit, software installation unit, etc. Slave Unit serves as execution and communication unit.

8.1 Computer system with Slave Unit located on a snap-in card

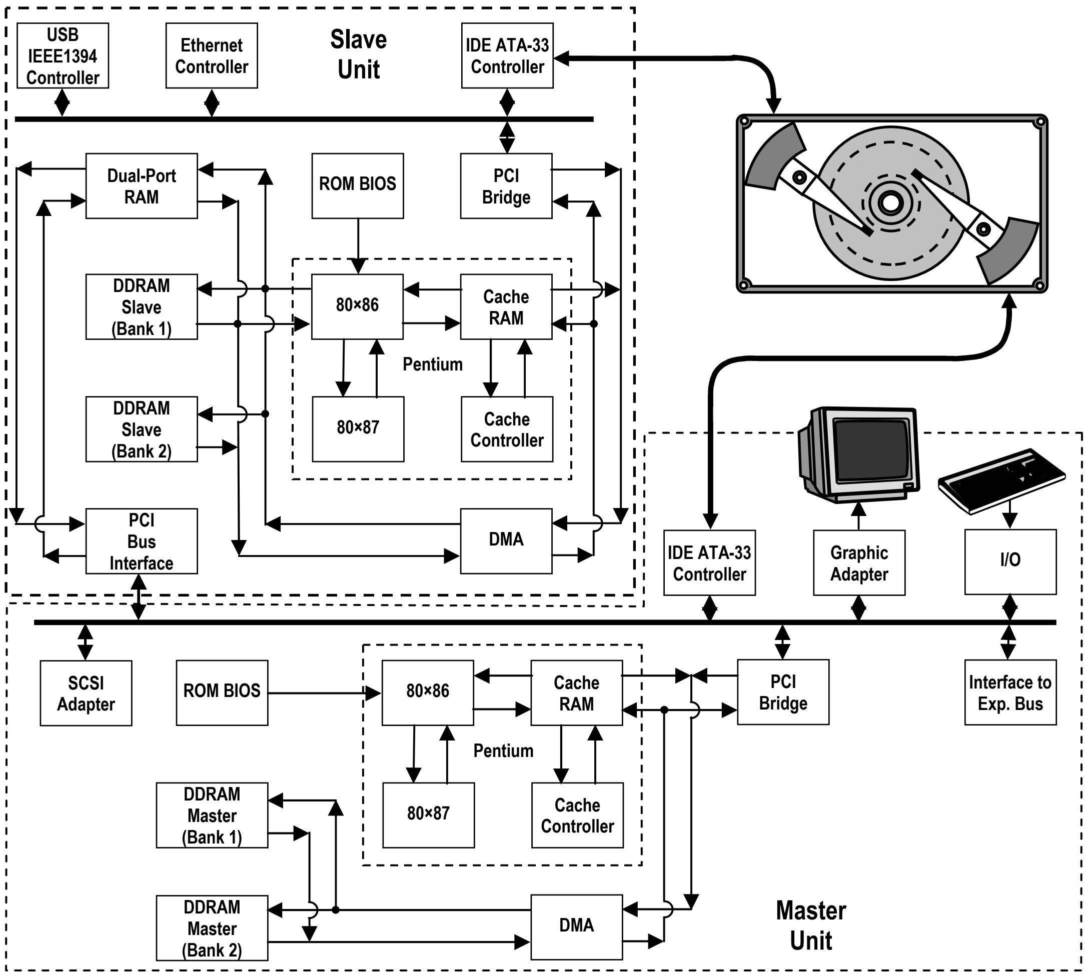


Figure 15 Computer System with Slave Unit as snap-in card

What is the basic idea behind design depicted on Figure 15?
The basic Idea is to make computer inherently resistant against malicious software attacks while maintain its configuration potential unhindered. Major design features and their meaning are:

- No application execution can be started from inside Slave Unit
- Any application execution is started, stopped, aborted, controlled and supervised from Master Unit via dual-port RAM
- All configuration files and data, etc are stored in Master boot partition (chapter 5.3)
- All confidential data and files or data and files representing user work are stored in Master extended partition (chapter 5.3)
- In general applications execute their tasks (creating, opening, closing and editing files, communications, etc) in Slave Unit
- Master and Slave unit coordinate their common tasks by exchanging directives and files through dual-port RAM
- A file that is being edited is temporary stored in Slave extended partition (chapter 5.3)
- Therefore by applying this kind of hardware no malicious software can harm computer seriously except destroying data of currently opened file, crashing running application or crashing Slave Unit.
- Anything sensitive is out of harms way (chapter 4 and 5.3).
- All communication devices are also under Master Unit hardware control
- Slave Unit CMOS should be accessible from the Master Unit only

8.2 Computer system with Master and Slave Unit located on a motherboard

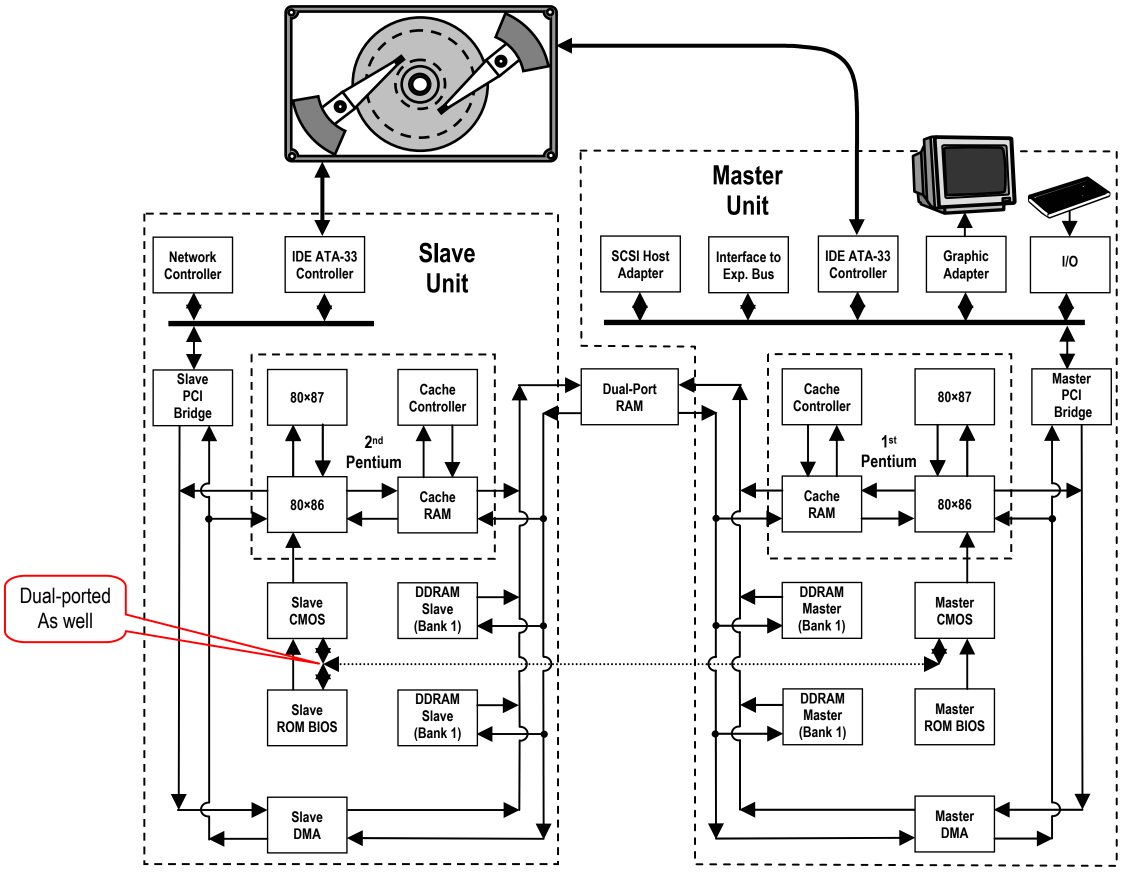


Figure 16 Master-Slave computer system motherboard → Option 1

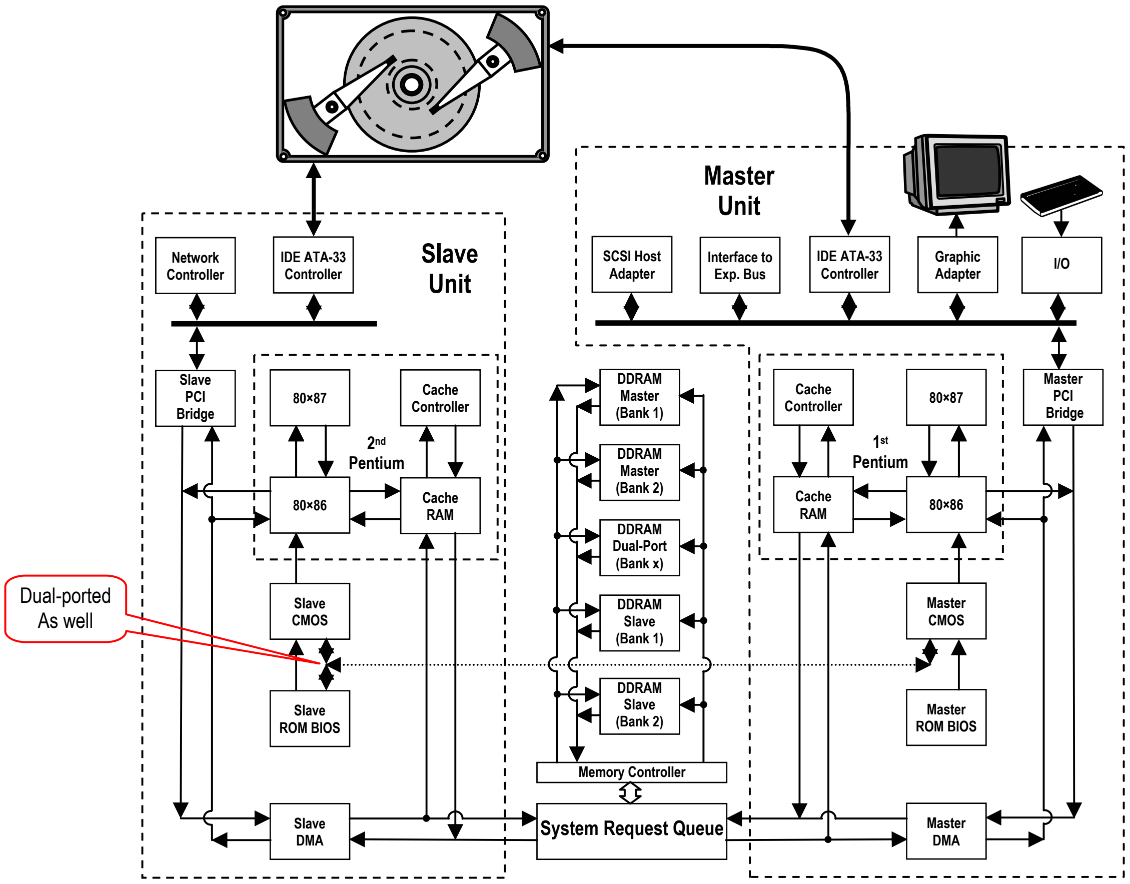


Figure 17 Master-Slave computer system motherboard → Option 2

In contrast to configuration on Figure 16 the configuration on Figure 17 is the only possible one for laptops. System memory could be modified in a way depicted on Figure 8 or Figure 9 respectively (also in chapter 4.5).
By doing so a single set of DDRAM memory banks would serve as Master and Slave system memory as well as dual-port RAM. This way the system would be extremely flexible even though a bit slower.

8.3 Suggested hardware design features summary

It is technically feasible to design hardware with built-in inherent resistance against any possible malicious software attack, because:

- Besides control lines both Units are connected only through dual-port RAM and selected areas of mass storage media
- All applications execute their tasks in Slave Unit area designed to withstand attacks by disabling direct access to configuration data, confidential data, installed software, stored files, etc.
- Basically everything running on Slave Unit is controlled, supervised and monitored by Master Unit
- All data that represents user work is stored on Master Unit mass storage media. Even when files are infected, they can't harm Master Unit because stored data is inert, even virus executables. And before they are opened (and starting viruses by doing this) they are transferred to Slave Unit.
- Malicious software can't reconfigure computer, can't install software, can't access any data stored on Master unit mass storage area, but can crash running application or Slave Unit. Resetting Slave Unit cures the problem.
- Because close monitoring Master Unit can prevent any attack from network somebody to make a zombie of user's computer. Master Unit would just reset Slave Unit.

9 Software

The best software design would be if it is to be written in the way that there was a possibility to be run unchanged by both Units simultaneously. That would simplify installation, configuration, etc. Master and Slave Unit would control different peripherals and should be aware of them all. Therefore there should be a built-in interface driver for particular Unit to interface with peripherals that exists in another Unit. And there should be special built-in software that would neglect peripherals that wouldn't be accessible to particular Unit. All this would simplify software installation and configuration because both Units could use the same installed software and its configuration all stored on Master boot partition.

9.1 BIOS

9.1.1 Master Unit BIOS

Master Unit BIOS should have built-in "awareness" of its purpose (Master) to be able to start appropriate services. Therefore there won't be any conflict during booting process. There should also be option for Slave Unit setup configuration and its transfer to Slave Unit CMOS.

9.1.2 Slave Unit BIOS

Slave Unit BIOS should have built-in "awareness" of its purpose (Slave) to be able to start appropriate services. Therefore there won't be any conflict during booting process. There should also be an option available for Slave Unit for it setup configuration to be edited by Master Unit and transferred to Slave Unit's CMOS.

9.2 Operating system

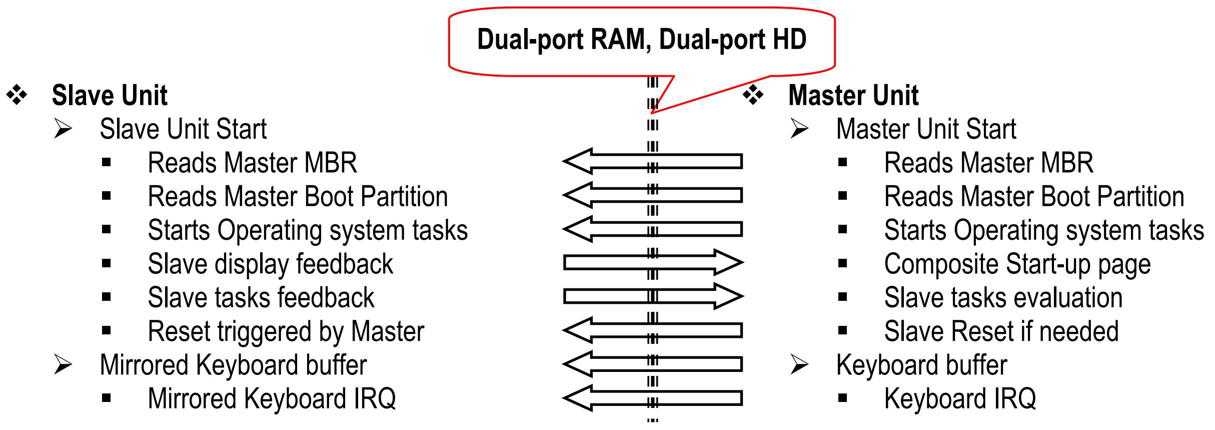


Figure 18 Operating system data exchange

The best solution would be when the same operating system package could be run both Units. And here is where the Unit's "awareness" of purpose comes into the play. For example: Slave Unit would automatically run Slave Unit's specific and complementary services and neglect the Master Unit's peripherals that are to be used by Slave Unit and, vice versa. I/O devices as a detection and interfacing example:

- I/O (keyboard, mouse and corresponding interrupt) would be detected by both Units:
    - Master Unit would conventionally detect I/O with additional buffer mirroring to a specific dual-port RAM locations
    - Slave Unit would detect I/O by probing for it at specific dual-port RAM locations
- Because Slave Unit is exposed, IRQ's and resets should be transferred to it via HW control lines from Master Unit

For this service to run a specific complementary set of drivers would be needed. Different services would need their specific complementary set of drivers.
In general all applications and utilities should be started from Master Unit only (Figure 19). There should be monitoring software to control their execution and abort any that wasn't initiated from Master Unit. Besides starting application via dual-port RAM there should be a feedback via dual-port RAM to Master Unit about actual status of that particular execution. In short: besides other tasks Master Unit should act like a user of Slave Unit and towards user it would also act as Graphic User Interface.
Most probably there will be a good decision to implement built-in Administrator Remote with limited privileges comparing with conventional built-in Administrator, because in this computer design remote administering decreases security. Therefore this issue should be carefully addressed. File and directory sharing should be less critical.
Special attention should be paid to hypertext options because one could search for files by utilizing their calls reaching into Master Unit storage area through dual-port RAM (possible solutions: disabling calls outside edited file, disabling calls outside directory in which the file is located, disabling automatic execution).

9.3 Application Software example № 1

To ease the explanation let us assume that a user is running an Internet Application (IA). Implementing already described hardware design, together with a slightly modified IA, the computer is impenetrable for any kind of attack during browsing the Internet. Figure 20 depicts IA start-up page or IA graphic user interface (GUI).
The IA start-up page is a picture on a user's display. It is composed out of many smaller parts, pictures. And it is of no importance at all where the pictures are coming from.
Imagine that there are two IA applications simultaneously running on Master and on Slave Unit. As already mentioned IA start-up page is a composed picture. And, there could be also a composed IA start-up page picture on the user display that is a part of the Slave Unit application (found Internet sites) and a part of the Master Unit application (taskbars and toolbars). Figure 21 depicts both parts of the composed start-up page picture from Figure 20.
A pointing device (a mouse) provides a coordinates for a process (or something else) to be selected by a click. That is, one positions the cursor over a certain area of the picture, usually text, and selects it by a double click. And selected text is nothing but user-friendly description of a process to be executed, together with all necessary parameters. What it really matter is the fact that a cursor position and double click start particular process execution.

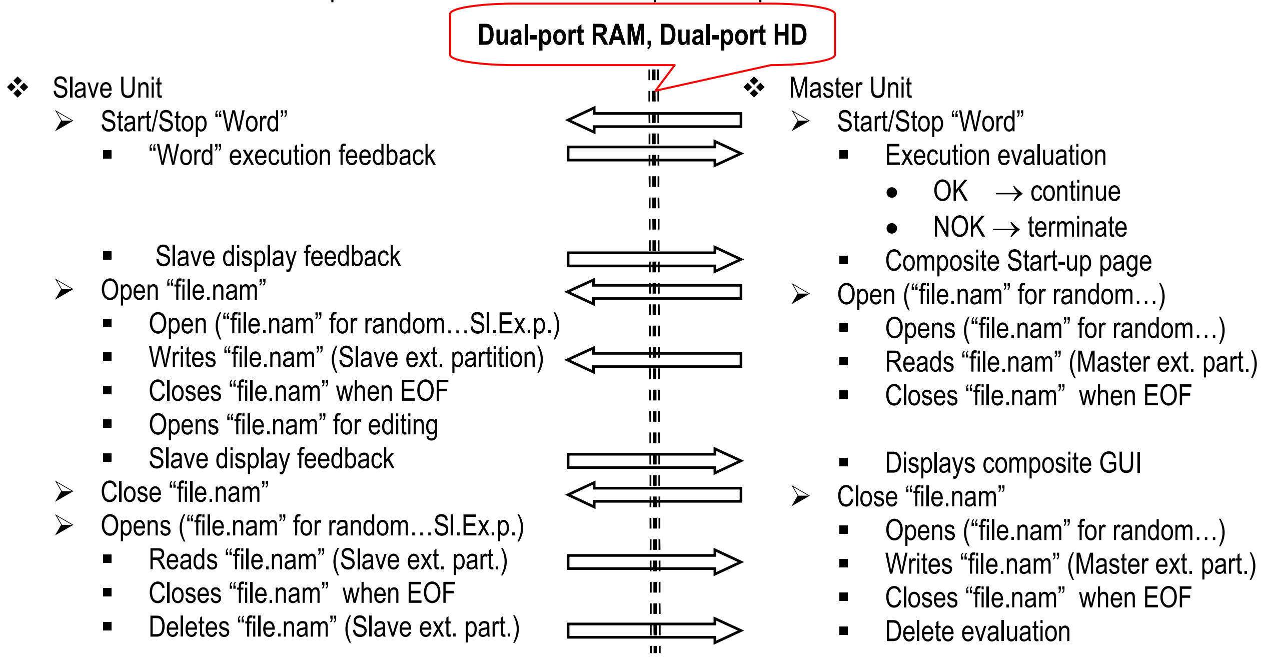
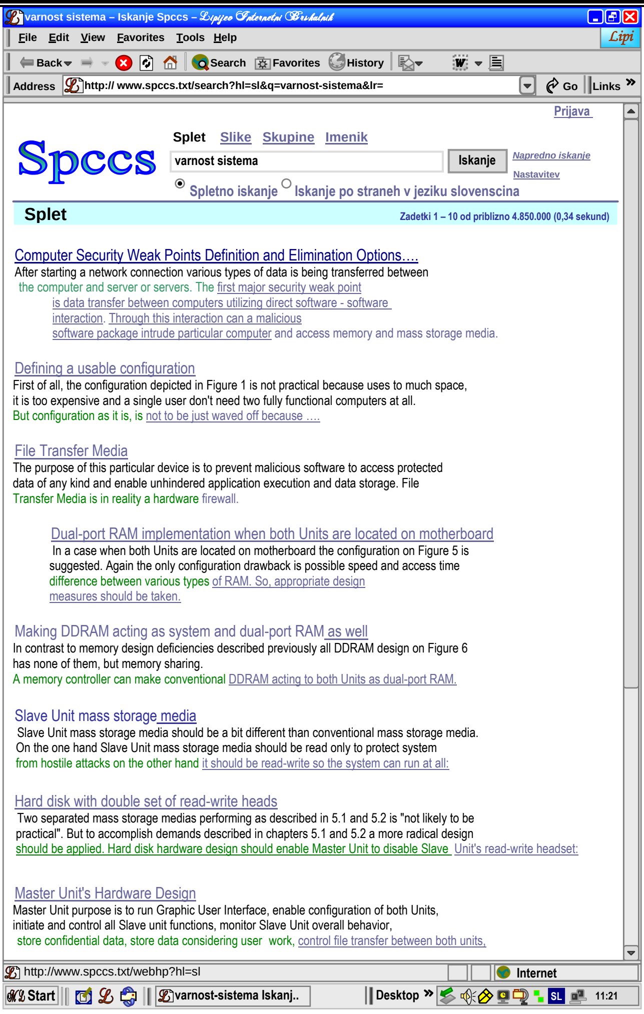

Dual-port RAM, Dual-port HD
Slave Unit
Start/Stop "Word"
"Word" execution feedback
Slave display feedback
Open "file.nam"
Open ("file.nam" for random…Sl.Ex.p.)
Writes "file.nam" (Slave ext. partition)
Closes "file.nam" when EOF
Opens "file.nam" for editing
Slave display feedback
Close "file.nam"
Opens ("file.nam" for random…Sl.Ex.p.)
Reads "file.nam" (Slave ext. part.)
Closes "file.nam" when EOF
Deletes "file.nam" (Slave ext. part.)
Master Unit
Start/Stop "Word"
Execution evaluation
OK → continue
NOK → terminate
Composite Start-up page
Open ("file.nam" for random…)
Opens ("file.nam" for random…)
Reads "file.nam" (Master ext. part.)
Closes "file.nam" when EOF
Displays composite GUI
Close "file.nam"
Opens ("file.nam" for random…)
Writes "file.nam" (Master ext. part.)
Closes "file.nam" when EOF
Delete evaluation

Figure 19 Application data exchange



http:// www.spccs.txt/search?hl=sl&q=varnost-sistema&lr=

varnost sistema

[✓ Spletno iskanje] Iskanje po straneh v jeziku slovenscina

Splet — Zadetki 1 – 10 od priblizno 4.850.000 (0,34 sekund)

Computer Security Weak Points Definition and Elimination Options….
After starting a network connection various types of data is being transferred between
the computer and server or servers. The first major security weak point
is data transfer between computers utilizing direct software - software
interaction. Through this interaction can a malicious
software package intrude particular computer and access memory and mass storage media.

Defining a usable configuration
First of all, the configuration depicted in Figure 1 is not practical because uses to much space,
it is too expensive and a single user don't need two fully functional computers at all.
But configuration as it is, is not to be just waved off because ….

File Transfer Media
The purpose of this particular device is to prevent malicious software to access protected
data of any kind and enable unhindered application execution and data storage. File
Transfer Media is in reality a hardware firewall.

Dual-port RAM implementation when both Units are located on motherboard
In a case when both Units are located on motherboard the configuration on Figure 5 is
suggested. Again the only configuration drawback is possible speed and access time
difference between various types of RAM. So, appropriate design
measures should be taken.

Making DDRAM acting as system and dual-port RAM as well
In contrast to memory design deficiencies described previously all DDRAM design on Figure 6
has none of them, but memory sharing.
A memory controller can make conventional DDRAM acting to both Units as dual-port RAM.

Slave Unit mass storage media
Slave Unit mass storage media should be a bit different than conventional mass storage media.
On the one hand Slave Unit mass storage media should be read only to protect system
from hostile attacks on the other hand it should be read-write so the system can run at all:

Hard disk with double set of read-write heads
Two separated mass storage medias performing as described in 5.1 and 5.2 is "not likely to be
practical". But to accomplish demands described in chapters 5.1 and 5.2 a more radical design
should be applied. Hard disk hardware design should enable Master Unit to disable Slave Unit's read-write headset:

Master Unit's Hardware Design
Master Unit purpose is to run Graphic User Interface, enable configuration of both Units,
initiate and control all Slave unit functions, monitor Slave Unit overall behavior,
store confidential data, store data considering user work, control file transfer between both units,

http://www.spccs.txt/webhp?hl=sl

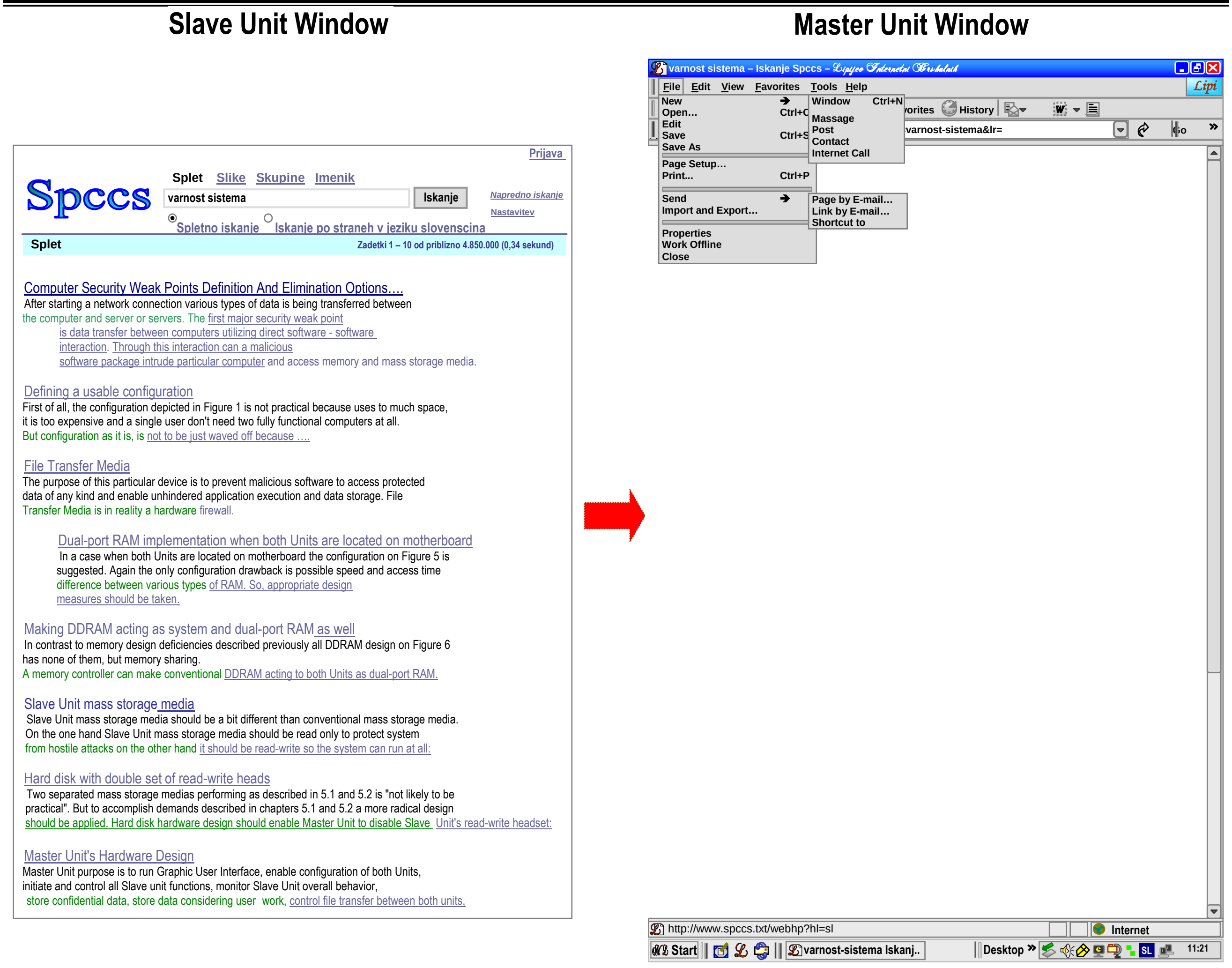
Figure 20 Internet Application start-up page

Therefore, to achieve a safe Internet browsing IA application should be modified, the one running in the Master Unit as well as the one running in the Slave Unit. Modification's end effect should be such that IA applications would run in a kind of handshaking mode:

- When IA application is started in the Master Unit, it automatically starts the IA application in Slave Unit and controls it
- Data to be displayed by Slave Unit (without taskbars and tool bars) are sent via dual-port RAM to Master Unit video RAM
- Complete IA application start-up page is then composed in Master Unit by adding taskbars and toolbars and displayed
- Therefore no malicious software can harm Master Unit because video RAM data is just a picture and nothing like executable is transferred to Master Unit

Because there is only execution control part of IA application running in Master Unit and that control is executed via dual-port RAM there is no way malicious software can affect Master Unit. There is also no way malicious software can access sensitive data stored on Master Unit mass storage media because of hardware design. The same applies also to computer configuration, CMOS, operating system, applications, etc.
But when one downloads a file, the file is just transferred via dual-port RAM from Slave Unit to Master Unit in byte-by-byte mode and stored in Master extended partition. And that is definitely no way to activate a virus in Master Unit.
In contrast to Master Unit is Slave Unit designed to be a "firing rage". Slave CMOS, boot partition with all installed software and configurations are read only. Master extended partition is hidden to Slave Unit by disabling access with programmable hardware. All these protections are configurable by Master Unit. Therefore in case of an attack:

- No configurations can be changed because they are all read only
- No data can be accessed by the attacker because Master extended partition is hidden to Slave Unit
- Computer takeover is also impossible because Master Unit monitor's Slave Unit activities and can abort any of them
- Malicious software can crash running application what can be cured with simple Slave Unit reset
- Malicious software can crash Slave Unit itself what can be cured with simple Slave Unit reset



Prijava

Spccs

Splet Slike Skupine Imenik

varnost sistema [Iskanje]

Napredno iskanje
Nastavitev

[✓ Spletno iskanje] Iskanje po straneh v jeziku slovenscina

Splet — Zadetki 1 – 10 od priblizno 4.850.000 (0,34 sekund)

Computer Security Weak Points Definition And Elimination Options….
After starting a network connection various types of data is being transferred between the computer and server or servers. The first major security weak point is data transfer between computers utilizing direct software - software interaction. Through this interaction can a malicious software package intrude particular computer and access memory and mass storage media.

Defining a usable configuration
First of all, the configuration depicted in Figure 1 is not practical because uses to much space, it is too expensive and a single user don't need two fully functional computers at all. But configuration as it is, is not to be just waved off because ….

File Transfer Media
The purpose of this particular device is to prevent malicious software to access protected data of any kind and enable unhindered application execution and data storage. File Transfer Media is in reality a hardware firewall.

Dual-port RAM implementation when both Units are located on motherboard
In a case when both Units are located on motherboard the configuration on Figure 5 is suggested. Again the only configuration drawback is possible speed and access time difference between various types of RAM. So, appropriate design measures should be taken.

Making DDRAM acting as system and dual-port RAM as well
In contrast to memory design deficiencies described previously all DDRAM design on Figure 6 has none of them, but memory sharing. A memory controller can make conventional DDRAM acting to both Units as dual-port RAM.

Slave Unit mass storage media
Slave Unit mass storage media should be a bit different than conventional mass storage media. On the one hand Slave Unit mass storage media should be read only to protect system from hostile attacks on the other hand it should be read-write so the system can run at all:

Hard disk with double set of read-write heads
Two separated mass storage medias performing as described in 5.1 and 5.2 is "not likely to be practical". But to accomplish demands described in chapters 5.1 and 5.2 a more radical design should be applied. Hard disk hardware design should enable Master Unit to disable Slave Unit's read-write headset:

Master Unit's Hardware Design
Master Unit purpose is to run Graphic User Interface, enable configuration of both Units, initiate and control all Slave unit functions, monitor Slave Unit overall behavior, store confidential data, store data considering user work, control file transfer between both units,

Figure 21 Both parts of composite Internet Application start-up page picture

- Slave Unit software running in it's memory is totally exposed and vulnerable, therefore any kind of spy-ware could be inserted in Slave Unit from the network
    - 1st solution:
        - Alternative use of Slave Unit: communication or normal work
        - A sort of "communication request" should be provided for the user to terminate work and switch to communication mode
        - Each change of use mode should be preceded by saving files and resetting Slave Unit
        - Quick reboot could be achieved by transferring operating system via dual-port RAM from Master Unit
        - This wouldn't be an adequate solution for a computer connected to network (LAN)
    - 2nd solution:
        - Implementing additional exclusively Communication Unit separated from Master Unit by the already described way (this and previous chapters) having it's own mass storage media (flash disk)
            - Semi-parallel complementary multiprocessor computer system? Intriguing, isn't it?
            - Communication Unit description is in chapter 11

9.4 Application Software example № 2

As already mentioned there should be special drivers implemented to synchronize semi-parallel complementary application execution, tasks, etc:

- User selects a menu item
- Master Unit places a command together with parameters in dual-port RAM location
- Master Unit initiates IRQ to Slave Unit to read dual-port RAM and starts command execution
- Slave Unit reads command data from dual-port RAM and starts execution
- Slave Unit is placing command execution status data in dual-port RAM during command execution

- Master Unit is reading command execution status data either in poling or IRQ driven mode
- Slave Unit is placing data to be displayed to user in dual-port RAM and generates IRQ to Master Unit to read data and pace them in Master Unit Video RAM
- Master Unit decides about terminating execution of particular command (user selects so, in case of errors)

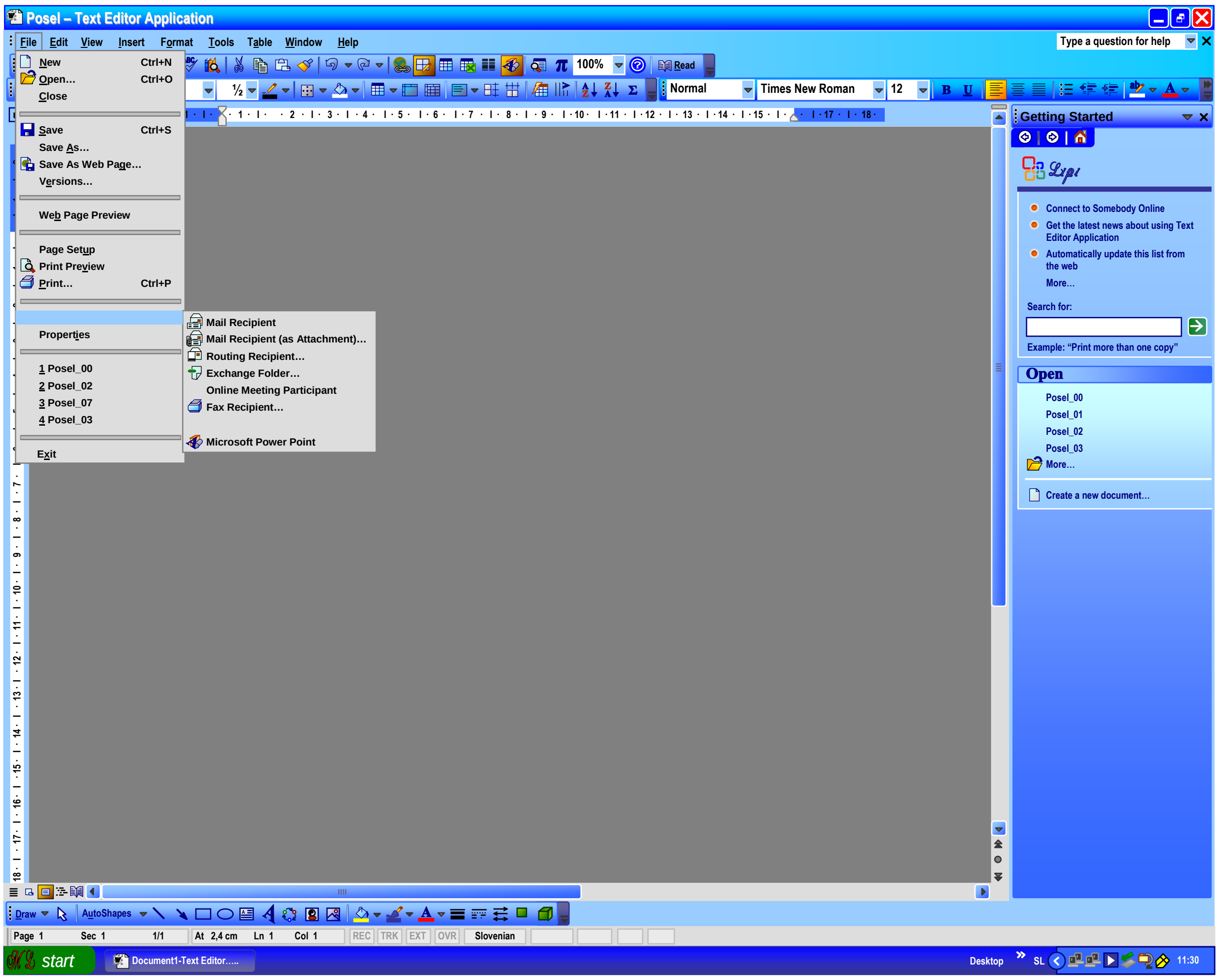


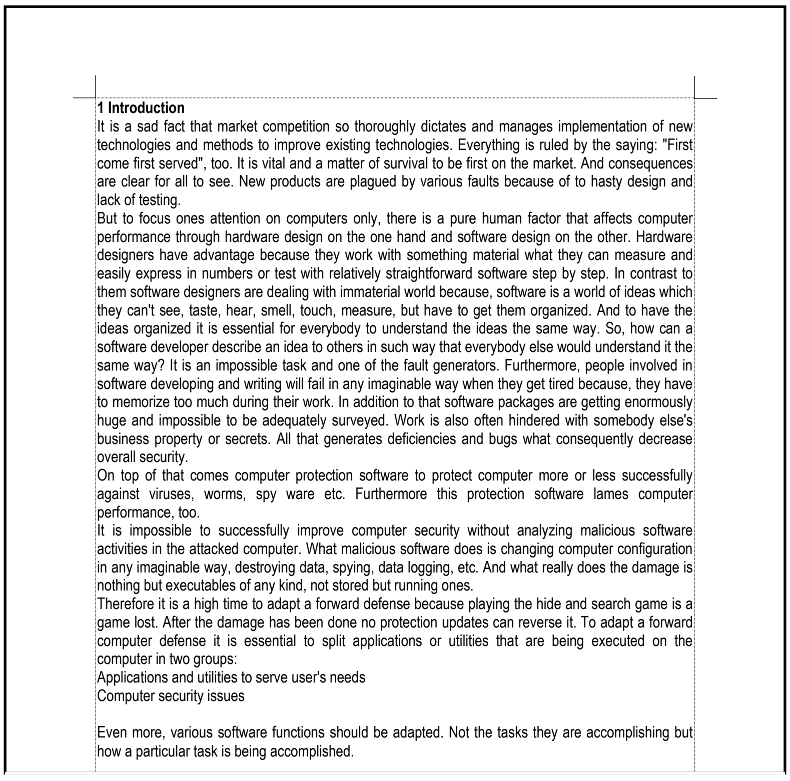
Figure 22 Text Editor Application → Master Unit part

This way Master Unit has absolute control over Slave Unit software application execution, control and monitoring.
Figures 22 and 23 depict what part of same application is executed in Master and what in Slave Unit. That helps together with the following description to prove why semi-parallel complementary computer system is so effective in computer system, application and user data protection.
Master Unit displays application start-up page with all menus, task bars, etc. To this picture is added data sent from Slave Unit via dual-port RAM: a page of a document being edited. To position a pointer on the page is to mirror keyboard buffer and IRQ to Slave Unit. The same is to be expected for how the letters are being placed in the document.

Table 3 Menu "File" items execution description

File	Figure 22
New	Initiated by Master Unit after user makes a selection from Menu File. Template is selected from templates stored in Master Unit, sent to Slave Unit via dual-port RAM and new file is created and edited in Slave Unit.
Open	Initiated by Master Unit after user makes a selection from Menu File. Existing file is selected and sent byte-by-byte via dual-port RAM from Master extended partition to Slave extended partition and saved as working copy. File's work copy is then opened and edited in Slave Unit.
Close	Initiated by Master Unit after user makes a selection from Menu File. Work copy of a file opened and edited in Slave Unit is closed. File's work copy in Slave extended partition is deleted.

Save	Initiated by Master Unit after user makes a selection from Menu File. Work copy of a file opened and edited in Slave Unit is saved. File's work copy is sent byte-by-byte via dual-port RAM from Slave extended partition to Master extended partition and saved. File's work copy in Slave extended partition is deleted.
Save As	Initiated by Master Unit after user makes a selection from Menu File. Work copy of a file opened and edited in Slave Unit is saved. File's work copy is sent byte-by-byte via dual-port RAM from Slave extended partition to Master extended partition and saved as "name.???". File's work copy in Slave extended partition is deleted.
Save as Web Page	Initiated by Master Unit after user makes a selection from Menu File. Opened and edited file in Slave Unit is closed. File's work copy is sent byte-by-byte via dual-port RAM from Slave extended partition to Master extended partition and saved as "name.httm". File's work copy in Slave extended partition is deleted.
Versions	Initiated by Master Unit after user makes a selection from Menu File. It executes and saves data in Master Unit only.
Page Setup	Initiated by Master Unit after user makes a selection from Menu File. It executes and saves data in Master Unit only.
Print Preview	Initiated by Master Unit after user makes a selection from Menu File. Data to be printed is sent from Slave Unit via dual-port RAM to Master Unit Video RAM and displayed as conventional Print Preview.
Print	Initiated by Master Unit after user makes a selection from Menu File. Printer and printer options are selected by Master Unit. File's work copy opened in Slave Unit is sent byte-by-byte via dual-port RAM from Slave Unit to Master Unit printer port to be printed.
Send To	Initiated by Master Unit after user makes a selection from Menu File. Recipient data are selected from Master extended partition by user. Opened file is sent.
Properties	Initiated by Master Unit after user makes a selection from Menu File. It executes and saves data in Master Unit only.
Exit	Initiated by Master Unit after user makes a selection from Menu File. Application is terminated in Slave and Master Unit as well.



1 Introduction

It is a sad fact that market competition so thoroughly dictates and manages implementation of new technologies and methods to improve existing technologies. Everything is ruled by the saying: "First come first served", too. It is vital and a matter of survival to be first on the market. And consequences are clear for all to see. New products are plagued by various faults because of to hasty design and lack of testing.

But to focus ones attention on computers only, there is a pure human factor that affects computer performance through hardware design on the one hand and software design on the other. Hardware designers have advantage because they work with something material what they can measure and easily express in numbers or test with relatively straightforward software step by step. In contrast to them software designers are dealing with immaterial world because, software is a world of ideas which they can't see, taste, hear, smell, touch, measure, but have to get them organized. And to have the ideas organized it is essential for everybody to understand the ideas the same way. So, how can a software developer describe an idea to others in such way that everybody else would understand it the same way? It is an impossible task and one of the fault generators. Furthermore, people involved in software developing and writing will fail in any imaginable way when they get tired because, they have to memorize too much during their work. In addition to that software packages are getting enormously huge and impossible to be adequately surveyed. Work is also often hindered with somebody else's business property or secrets. All that generates deficiencies and bugs what consequently decrease overall security.

On top of that comes computer protection software to protect computer more or less successfully against viruses, worms, spy ware etc. Furthermore this protection software lames computer performance, too.

It is impossible to successfully improve computer security without analyzing malicious software activities in the attacked computer. What malicious software does is changing computer configuration in any imaginable way, destroying data, spying, data logging, etc. And what really does the damage is nothing but executables of any kind, not stored but running ones.

Therefore it is a high time to adapt a forward defense because playing the hide and search game is a game lost. After the damage has been done no protection updates can reverse it. To adapt a forward computer defense it is essential to split applications or utilities that are being executed on the computer in two groups:

Applications and utilities to serve user's needs

Computer security issues

Even more, various software functions should be adapted. Not the tasks they are accomplishing but how a particular task is being accomplished.

Figure 23 Text Editor Application → Slave Unit part

Figure 24 depicts Menu "Edit" and it items. Menu items execution descriptions are intentionally brief. They are just to present how two computers have to interact with each other to accomplish certain task while protect computer system against hostile attacks of any kind. All this is achieved by special hardware and software design.

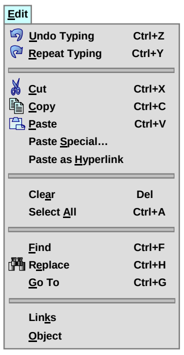


Figure 24 Text Editor Application: Slave Unit part

Table 4 Menu "Edit" items execution description

Edit	Figure 24
Undo Typing	Initiated by Master Unit after user makes a selection from Menu File. Executes undo function in a file opened in Slave Unit. Function execution is initiated, monitored and controlled via dual-port RAM.
Repeat Typing	Initiated by Master Unit after user makes a selection from Menu File. Executes repeat function in a file opened in Slave Unit. Function execution is initiated, monitored and controlled via dual-port RAM.
Cut	Initiated by Master Unit after user makes a selection from Menu File. Executes cut function in a file opened in Slave Unit. Function execution is initiated, monitored and controlled via dual-port RAM. Clipboard data is saved in Slave Unit.
Copy	Initiated by Master Unit after user makes a selection from Menu File. Executes cut function in a file opened in Slave Unit. Function execution is initiated, monitored and controlled via dual-port RAM. Clipboard data is saved in Slave Unit.
Paste	Initiated by Master Unit after user makes a selection from Menu File. Executes paste function in a file opened in Slave Unit. Function execution is initiated, monitored and controlled via dual-port RAM. Clipboard data is collected in Slave Unit.
Paste Special	Initiated by Master Unit after user makes a selection from Menu File. Executes paste special function in a file opened in Slave Unit. Function execution is initiated, monitored and controlled via dual-port RAM. Clipboard data is collected in Slave Unit.
Paste as Hyperlink	Initiated by Master Unit after user makes a selection from Menu File. Executes paste as hyperlink special function in a file opened in Slave Unit. Function execution is initiated, monitored and controlled via dual-port RAM.
Clear	Initiated by Master Unit after user makes a selection from Menu File. Executes clear function in a file opened in Slave Unit. Function execution is initiated, monitored and controlled via dual-port RAM.
Select All	Initiated by Master Unit after user makes a selection from Menu File. Executes select all special function in a file opened in Slave Unit. Function execution is initiated, monitored and controlled via dual-port RAM.
Find	Initiated by Master Unit after user makes a selection from Menu File. Executes find function in a file opened in Slave Unit. Function execution is initiated, monitored and controlled via dual-port RAM. Required data are transferred via dual-port RAM from Master Unit.
Replace	Initiated by Master Unit after user makes a selection from Menu File. Executes find function in a file opened in Slave Unit. Function execution is initiated, monitored and controlled via dual-port RAM. Required data are transferred via dual-port RAM from Master Unit.
Go To	Initiated by Master Unit after user makes a selection from Menu File. Executes find function in a file opened in Slave Unit. Function execution is initiated, monitored and controlled via dual-port RAM. Required data are transferred via dual-port RAM from Master Unit.

9.5 Other Applications Software

Multimedia software like graphics, audio and video could possibly run in Master Unit only because graphics, audio and video files don't contain any executable attachments. That would also eliminate problems with high data throughput in dual-port RAM.

Presentation software applications are expected to run in distributed mode:

- multimedia attachments in Master Unit, because graphic, video and audio files are safe
- everything else in Slave Unit

In general it is difficult to predict how particular software application would be run in advance because there is a lot of studying and testing to be accomplished first.

10 Semi-parallel computer system summary

Semi-parallel complementary computer system protects user's data by applying at least two computers (Figure 15, 16 and 17 examples). They can access each other's hardware in precisely determined way and they accomplish particular task in semi-parallel complementary way. That means:

- Master Unit
  - Has absolute control over Slave Unit
  - Can access Master boot partition, Master extended partition and Slave extended partition on computer mass storage media
  - Installs all software that run on computer
  - Stores all system and user confidential data in Master extended partition
  - Stores all data files in Master extended partition
  - Doesn't create, open or edit files
  - Stored file is transferred byte-by-byte via dual-port RAM to Slave Unit and creating a file's work copy in Slave extended partition; file's work copy is opened afterwards
  - Starts all applications and tracks it's execution status in Slave Unit
  - Data to start an application execution is transferred byte-by-byte via dual-port RAM to Slave Unit
  - Data to track an application execution is transferred byte-by-byte via dual-port RAM to Master Unit
  - Data to be displayed by Slave Unit to user is transferred byte-by-byte via dual-port RAM to Master Unit video Ram
- Slave Unit
  - Can access Master boot partition as read-only on computer mass storage media
  - Master Unit is to disable Slave Unit's hardware ability to write to this partition of mass storage media
  - Can access Slave extended partition on computer mass storage media
  - Can't detect or access Master extended partition on computer mass storage media
    - Master Unit is to disable Slave Unit's hardware ability to access this partition of mass storage media
  - Runs applications (initiated by Master unit) which create, open, edit, close files, etc
  - saves files (initiated by Master unit) by closing file's work copy and transfers it byte-by-byte via dual-port RAM to Master extended partition (path)
- Data exchange between both Units
  - Any data is exchanged via dual-port RAM
  - Master Unit commands to run applications or tasks are placed to dual-port RAM locations
    - IRQ to read commands from dual-port RAM locations and execute them can be transferred by control lines
  - Slave Unit application or task execution status data (feedback) is placed to dual-port RAM locations
    - IRQ to read execution status data from dual-port RAM locations can be transferred by control lines
    - Any unexpected data could trigger defense action by Master Unit - Slave Unit reset
  - Files are transferred from one Unit to another byte-by-byte via dual-port RAM locations (it could be handshake mode). File is effectively copied from one Unit extended partition to another one's.
  - User data file or confidential data transfer could be initiated only by Master Unit command.
- Application execution
  - A data file can be opened by an application only in Slave Unit
  - System runs applications Slave Unit indirectly started, controlled and monitored by Master Unit
    - Any abnormal activity could trigger defense action by defense action by Master Unit - Slave Unit reset
  - System creates, opens, edits, closes, saves files, etc. in Slave Unit indirectly controlled and monitored

by Master Unit Any abnormal activity could trigger defense action by defense action by Master Unit - Slave Unit reset
- System hardware allows Slave Unit read-only access to selected configuration data
- System hardware prevents Slave Unit to independently access user's data
  - No confidential data can be accessed by malicious software activity
  - Any abnormal activity could trigger defense action by defense action by Master Unit - Slave Unit reset
- System hardware prevents to change configuration in any way
  - No configuration data can be changed by malicious software activity
  - Any abnormal activity could trigger defense action by defense action by Master Unit - Slave Unit reset

This configuration contains malicious software activity in Slave Unit because all potentially dangerous tasks are executed there. It doesn't matter if a malicious attachment is added to a file edited in Slave Unit because it is inactive when stored. And no file is supposed to be ever opened in Master Unit so, all configuration, confidential and user data is perfectly safe in Master Unit's area. Malicious software can crash application running in Slave Unit and malicious software can crash Slave Unit as well but, can't change computer configuration or access any data files. In this case Slave Unit reset cures the problem.
Malicious software can potentially insert itself in running application's processes to log data about opened files and running applications and send them to predefined server during genuine network session. This could be prevented by:
- Slave Unit reset,
- And erasing all files in Slave extended partition before going online.

This configuration also prevents malicious software to spread in a particular computer as well as prevents it spreading through network. For high security systems and systems connected to network a third unit, Communication Unit should be added to computer system.

11 Communication Unit

This Unit is to be implemented in high security systems because one can always expect that well designed malicious software could take control over Slave Unit and get connected to attacker's server. But when there is no means of communication in Slave Unit such software can't do much damage because Master Unit would rebuff any attempt to get connected. By default only Master Unit can initiate network connection and communication. And before it is sent a file is first saved in Master extended partition. Therefore any malicious software activity is made inactive and can't affect Master Unit's communication processes as well as it can't send any data.
Communication Unit is essentially just another slave unit. Hardware and software design guidelines are exactly the same as Slave Unit ones. Because Communication Unit's single purpose is communication and it doesn't need to store data files. Communication Unit system software could be stored on flash disk or transferred to Communication Unit's DDRAM during booting-up session from Master Unit. Any received file is sent either to Master extension partition to be saved or to Slave Unit extension partition to be opened and displayed. It serves as a buffer against hostile attack from network because dual-port RAM prevents malicious software to change computer configuration or access confidential data.

- Communication Unit:
  - Receives addresses to connect to via dual-port RAM from Master Unit
  - Receives addresses to send an e-mail via dual-port RAM from Master Unit
  - Receives files to be sent to a recipient via dual-port RAM from Master Unit
  - Receives files from a sender and transfers them via dual-port RAM to Master Unit to be stored

- Master Unit:
  - initiates all communication activities,
  - controls and monitors processes,
  - provides addresses, log ins and passwords
  - opens received files in Slave Unit
  - saves received files in Master extension partition
  - sends files to various recipients on the network

So, in this configuration (Figure 25) two guardians protect Master Unit where computer operating system, application software, configuration data, user's data files, etc is stored. Communication Unit protects against any hostile attack attempt from the network. Slave Unit protects against any hostile attack from spawn from infected files.

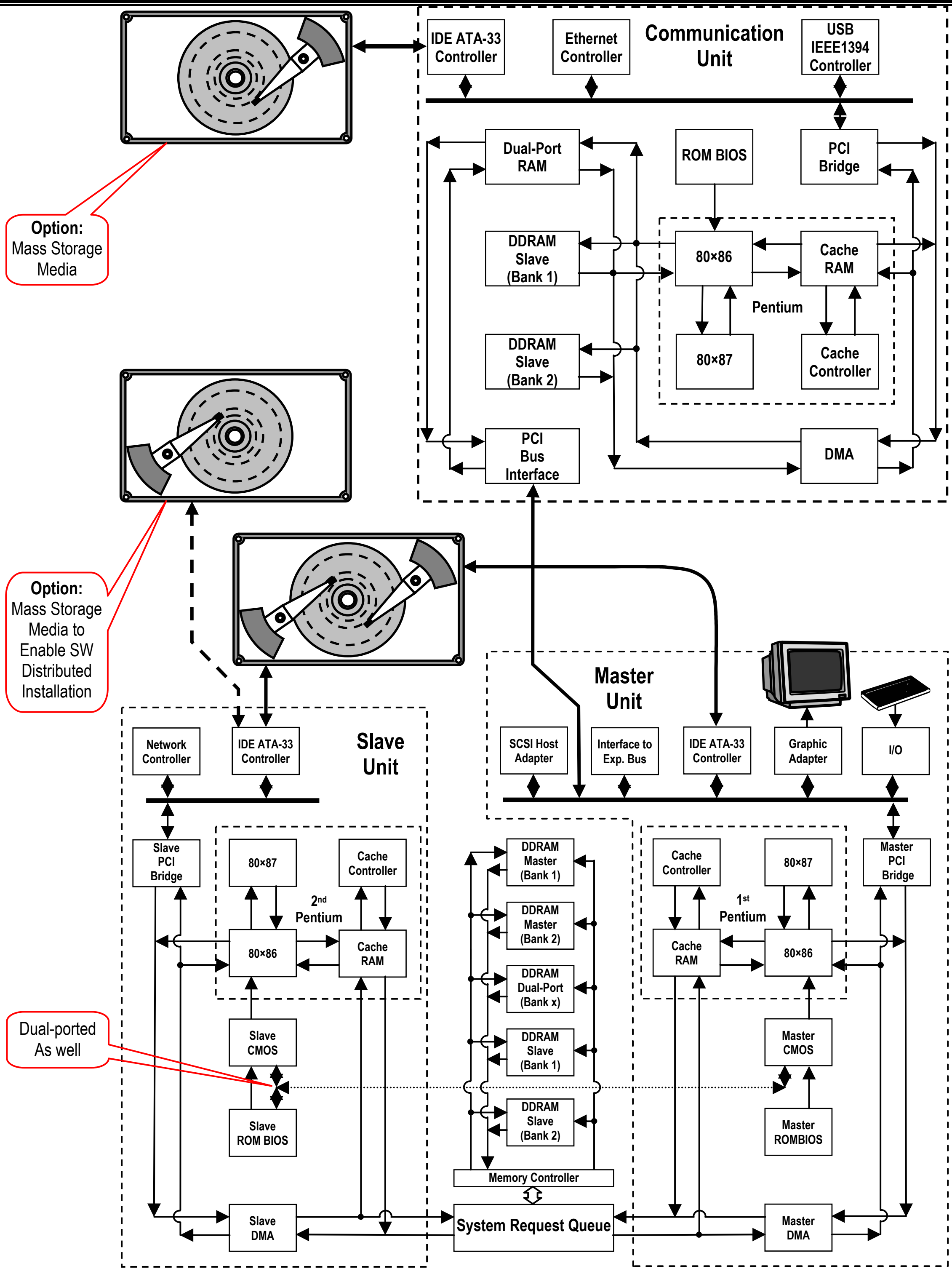


Figure 25 High Security Semi-Parallel Computer System

12 Shared Files, Shared Directories and remote Administering

After Master Unit evaluates user's name and password it could grant access to user to shared files and directories. But, a secure network protocol, like IPSsec, should be applied. Without that all endeavor to design semi-parallel complementary computer system would be in vain.

Shared file:

- Can be accessed by all users with read only rights at the same time
- Can be accessed by only one user with read-write rights and as read-only to everybody else with sharing rights
- File is sent to user and opened in user's Slave Unit

Shared directory:

- Can be accessed by all users (with sharing rights) at the same time
- The only limitation is with respect to shared files

Remote administering is a weak point because this is the entrance inside Master Unit. Therefore a special care should be put into secure network protocol, login, password, etc. Maybe it would be very usable to have two built-in accounts:

- Administrator-conventional one
- Administrator-remote one to be switched on and off by the user in coordination with network administrator. This would surely increase security because remote administering would be available only for very short periods of time.

13 All Software protection method

Semi-parallel complementary computer system previously described could be realized by software simulation running on a conventional hardware. A guideline example how to design such system could be VMware's software.

13.1 How Does VMware Workstation Work?

VMware Workstation works by creating fully isolated, secure virtual machines that encapsulate an operating system and its applications. The VMware virtualization layer maps the physical hardware resources to the virtual machine's resources, so each virtual machine has its own CPU, memory, disks, and I/O devices, and is the full equivalent of a standard x86 machine. VMware Workstation installs onto the host operating system and provides broad hardware support by inheriting device support from the host.

13.2 Multiple Virtual Machine operating system

What it is needed to be changed is to upgrade VMware from an Application to a fully functional "primary" operating system which would fulfill previously described guidelines for the hardware protection design. Conventional operating systems would run on Virtual Machines as "secondary" operating systems.

- Advantages
    - No changes in conventional hardware design are needed
    - Potentially optimal solution for cell phones
- Disadvantages:
    - Significant increase in execution time is to be expected: "primary" operating system written in assembler
    - code could reduce this drawback
- Potentially problematic when installing newer secondary operating systems with different, ever greater,
- System resource demands. For instance: Win 2000, Win XP, Longhorn, etc.
- All software weak points, bugs and security leaks are to be expected over and over again.
- Assuming current security problems it is not to be expected that Virtual Machine could compete with hardware-software method because, even suggested Virtual Machine operating system is finally just a different software package with all deficiencies (inserted by human software writers)

To summarize VMware, Intel and Microsoft already research and experiment such solutions.

14 Software installation

Software installation is a weak point because computer system is absolutely at mercy to software producer. There is no help against a "mole" inside Master Unit. So, all software is to be acquired from proven sources.
All software to be run on any computer system Unit should be able to install itself to a particular Unit via dual-port RAM from Master Unit. It could be all installed in Master boot partition only. And, SW could be transferred later during boot-up session or during starting up particular application to other units.

15 Distributed applications SW installation

Dubious software Installations should be directed to removable mass storage media as depicted on Figure 25. All configurations and registry files should be stored there. Operating system should look after them when the storage media is getting connected. This way user could have a secure computer system and SW test platform on his disposal without jeopardizing security at all. Files generated by these applications should be treated as insecure and could not be opened together with files treated as secure. Slave Unit reset and dubious SW activities termination should precede going online.

16 Network Security

Computer configuration described in previous chapters enables safe communicating with other computers because the configuration prevents uncontrollable malicious software spreading in user's computer as well as through the network. By applying secure network protocols safety increases furthermore because any spying would be very difficult. Another good practice would be to implement asymmetrical file encryption, PGP, GNU Privacy Assistant, etc as standard safety application. Before a file is sent to a recipient it would be encrypted in user's Master Unit. And the safety will be as good as it could be. That would put together totally open network communication and safe one at the same time. Of course that would work only until one installs some dubious software.

17 Internet and Electronic Business

Imagine using such practice in Internet. One can freely surf Internet at one time and does the business at some other time without jeopardizing computer overall security.
There are many tempting things found on WEB pages to be bought. It is suicidal to reveal a credit card number through the Internet. And one can never know about somebody else's honest intentions or not. But what it is really to be done is to implement:

- Already available IT technology,
- Asymmetric encryption algorithm,
- A software package to read text from Master Unit's video RAM and store it in a special file in Master extended partition and
- Old fashion bank draft to pay a bill.
    - It is important that data are collected from Master Unit's video RAM, because that is convenient way to hinder WEB page underlying scripts, potentially malicious, to affect user's intentions.

If a bank would have encryption software (GNU Privacy Assistant for instance) in their system implemented, the procedure will be as follows:

- To it's customer a bank would issue encryption software package and a credit card (smart card) with stored data
    - Customer's private data
    - Customer's bank account number
    - Credit card's number
    - Bank's particular customer public encryption key (bank's private decryption key for particular customer is to be stored together with customer's personal data in their system)
    - Customer's private decryption key (customer's public encryption key is to be stored together with customer's personal data in their system)
    - Bank's server IP address
    - Etc
- Where to store user's pass-phrase? Should it be stored in credit card chip together with all other suggested data?
    - Probably yes, because anything else would be much to complicate for a common user. Therefore all data should
    - Be protected by a PIN code and trials to guess it out should trigger erasing of all the data.

User would have to see encryption software package to be installed on user's computer. The computer should also be supplied with credit card reader.
So, when user would decide to get connected to bank inserting a credit card into a card reader will automatically reset Slave Unit and connect computer to bank's server. That means:

- User's computer would send encoded message with user's data to bank's server where it will be decoded by bank's private key for the particular customer, and
- Bank's server would send with particular customer's private key encoded message with bank's customer service internet site to user's computer
- User would select.... This way user could accomplish whatever user wishes.
- When a user would surf Internet and decided to buy particular item:
- Data about item to be bought, a seller data (name, address, bank) will be selected, picked up from Master Unit's video RAM and stored in special file in Master extended partition
- A credit card will be inserted in computer card reader
- After getting connected to bank's server user will select bank draft form
- User will fill-in recipient data
- User will open special file with previously stored data about the item to be bought and transfer that data to the
- bank draft form
- User will select send
- Software will electronically sign the form and encode it with bank's particular customer public encryption key
- Encrypted bank draft will be sent to bank's server
- Bank will sent money and purchased item's data to seller's bank

Therefore, a deal would be done without revealing any sensitive to anybody. That would render pfishing because without decoding key all sent messages are to be unusable. Furthermore, even frauds could be traced much easier. And, general public can easily use such system because it could be made simple to use nothing beyond Smart Cards and PIN numbers.

There are other implications:
- Filling in reports about income to tax-office
- Filling in insurance applications
- Similar ways to do business with other public institution could be devised
- Etc.

The bottom line is, very sensitive data could be reasonably safely sent and received via Internet as it is.

18 Cell Phone Application

Cell phone could be connected to Internet. As in any other computer system security problems are exactly the same. So, same protection methods could be implemented in cell phones.

19 Protection method summary

When one limits demands upon a degree of IT protection by neglecting infected files and focuses attention to hinder virus or hostile attack effects, a nearly perfect protection is technologically feasible. Feasible just by applying semi-parallel complementary processing consisting of Master and Slave Units to safeguard software, configuration and data files from being corrupted, changed or misused.
Nearly perfect protection means that infected files would be erased rather than infection removed. And malicious software would be able only to crash an application running in Slave Unit or crash Slave Unit itself. Considering correct design no malicious software could reach beyond that. Furthermore, resetting Slave Unit would erase malicious software and by doing so cure the problem, because no computer configuration data, operating system files, user files, confidential data, etc. could be changed from Slave Unit side. Even more, user files and confidential data are not to be accessed at all. This configuration prevents spreading infection in the computer itself as well as spreading infection over the network.
One could claim that protection is based exclusively on already developed technologies while neglecting unavoidable modifications that are to be made. Comparing Semi-parallel complementary computer system with AMD's multi core processor (Figure 26 and Figure 27) systems where a portion of system memory is set aside for data only a clear AMD's disadvantage is to be noticed:
- AMD's protection is limited to certain malicious SW only
    - o AMD's protection can't be exactly defined because the protection is limited to system memory
- AMD's protection is vulnerable to attacks because processor cores are internally directly connected on data level
    - o Therefore, a clever code can break through the protection
- In contrast to AMD's protection in this article described method enables
    - o Level of protection to be exactly defined
    - o Level of protection is exactly the same for any malicious SW even not yet written one
    - o And here is exactly the point where design applying dual-port RAM and dual-port mass storage media shows overpowering advantage because a Semi-parallel complementary computer system is vulnerable from the inside out direction only

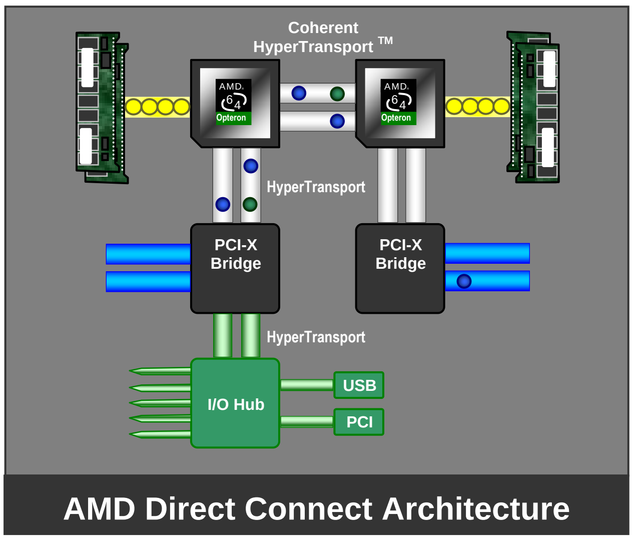


Figure 26 AMD Direct Connect Architecture → № 1

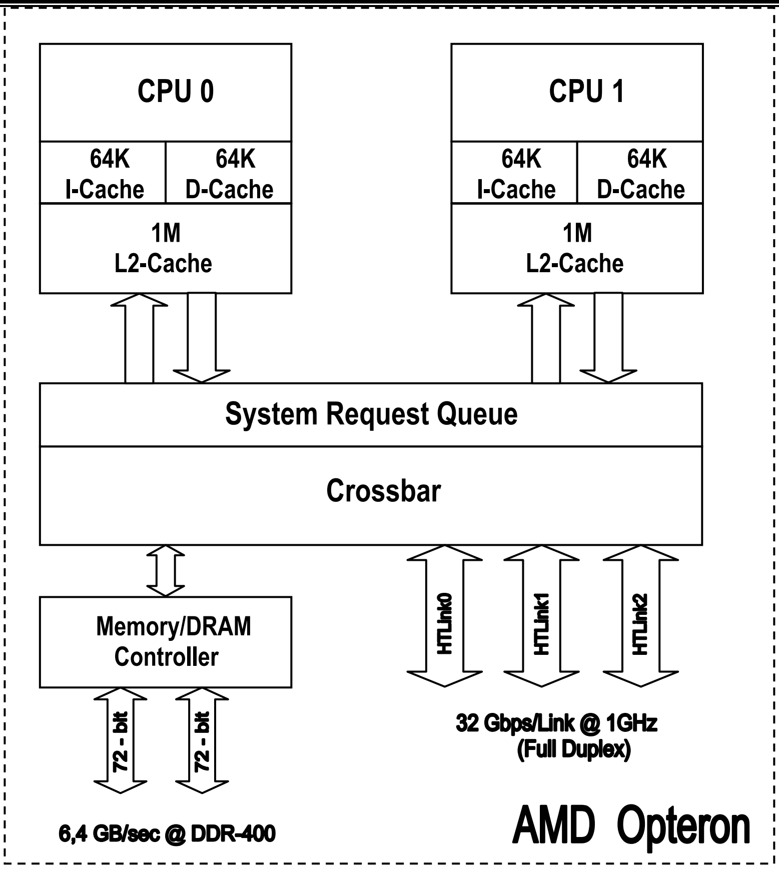


Figure 27 AMD Direct Connect Architecture → № 2

20 Conclusion

The best outcome of this study is that exactly the same technology is applicable in PC's, cell phones or in any other IT technology. Furthermore, anywhere where computers connected to networks are applied. Nevertheless, described computer configuration has the greatest possible inherent resistance against hostile attacks of today as well as of future ones and is operating system independent. It is clear fact that this article is about nothing more then a rough sketch of ideas. And it is up to IT industry to further develop initial ideas and take the advantage that they offer. The author humbly expects that SW experts can optimize suggested design performance to far outperform already available multi core processor platforms. Making computer systems safe should have great implications in electronic business because sensitive data is not accessible from the network. Therefore, various encryption algorithms with encrypting and decrypting keys could be safely stored in computer even while it is online.

About the Author

Devised and written by

Marjan Lipovšek,
B.Sc. of Electrical Engineering,
Graduated in 1984 at Faculty of Electrical Engineering, University of Ljubljana,
Ljubljana,
Slovenia,
http://www.fe.uni-lj.si/welcome-E.html

Address:

Marjan Lipovšek
Ljubljanska 1 C
1241 Kamnik
Slovenia

E-mail: marjan.lipovsek@siol.net

Reference

I tried to patent my invention. Even though I myself did a patent search I asked a particular Patent Office to do "professional" patent search for me. They received money to do the search. Results in Search Report are as follows.

Patent Document No.	Inventor
US 2003/0018892 A 1	Tello
US 2003/0070083 A 1	Nessler
US 2003/0079050 AI	Chang et al.
US 2003/0226015 A 1	Neufeld et al.
US 2004/0070920 AI	Flueli
US 2004/0123051 AI	Lloyd et al.
US 2004/0236874 AI	Largman et al.
US 2005/0091522 AI	Hearn et al.
US 2005/0172144 AI	Shao
US 2005/0204083 AI	Chu
US 2005/0216685 AI	Heden et al.
US 2005/0246469 AI	Chu
US 2005/0251867 A 1	Sastry et ill.
US 2006/0020960 AI	Relan et al.
5,283,828	Saunders et al.
5,289,540	Jones
US 6,463,537 BI	Tello
US 6,718,372 BI	Bober
US 6,749,115 B2	Gressel et al.
US 6,813,682 B2	Bress et al.

Document No.	Country & Date
2001-14239	Japan, 01/2001
2001-265727	Japan, 09/2001

Internet Documents
"Peer-to-Peer (P2P) Network Security - Four Steps To Sharing and Swapping Files Without Becoming a Victim", website: hap://netsecurity.about.com/od/newsandeditorial1/a/p2psecuritv.htm,
2 pages, printed from the Internet on April 25, 2006.

"P2P Security", website:
http://ntrg.cs.tcd.ie/undergrad/4ba2.02-03/p10.html,
1 page, printed from the Internet on April 25, 2006.

I don't agree with the search report because it failed to mention and provide any comment upon

Patent Document No.	Inventor
US2006/0031940 A1	Allen F. Rozman and Alfonso J. Cioffi

even though I have make the particular Patent Offic aware of it on March, 6th 2006 and again on April, 10th 2006. I'm absolutely sure that only patent US2006/0031940 A1 can be referenced to my invention. I believe that Search Report e-mailed to me from particular Patent Office is nothing but pure nonsense. Even more I can deduce that there is described something else than my invention in the Search Report's Summary of Invention.

For this article I have a copyright certificate from the year 2005.
Therefore I decided to send this article to as many PC Magazine editors or PC Magazine editor offices in the World as possible to make it public.

I hope readers will find it interesting at least.