R.P.L.

SGM2

SYSTEM

DOCUMENT ORDER A78138759
REVISION AA
STATUS Draft
DATE November 27,
DOCUMENT TREE
COMMESSA

ENGINEERING PRODUCT SPECIFICATION

SGM2

SYSTEM HARDWARE

PREPARED BY

G.C. Tessera

REVIEWED BY

APPROVED BY

A. Cicu

Honeywell

Honeywell Information Systems Italia

Loc. Pregnana Milanese, Italia

HONEYWELL
CONFIDENTIAL & PROPRIETARY

This document and informations contained herein are confidential to and proprietary of Honeywell Information System, and are made available only to Honeywell employees for the sole purpose of conducting Honeywell business.
This document, any copy thereof and the information contained herein shall be maintained in strictest confidence; shall not be copied in whole or in part except as authorized by the employee's manager; and shall not be disclosed or distributed (1) to persons who are not Honeywell employees, or (2) to Honeywell employees for whom such information is not necessary in connection with their assigned responsabilities. Upon request, or when the employee in possession of this document no longer has need for the document for the authorized Honeywell purpose, this document and any copies thereof shall be returned to the responsable manager.
The only exception to the terms and conditions set forth herein is applied for those companies or persons external to the Honeywell that sign a specific "Non Disclosure Agreement" document. The terms and conditions indicated in that document, constitute the basis for specific restrictions.

1.2 REFERENCE DOCUMENTS

* SGM2 PFS (Product Functional Specification) doc. N.

* SGM2 EPS for SYSTEM SOFTWARE; doc. N.

* SGM2 STATION PROCESSOR PDD; doc. N.

* SGM2 SYSTEM PDD; doc. N.

* DEMAND PAGING MMU, Nov. 15 1985; doc. N. A78138743

* VME bus
  Specification Manual edited by "VMEbus Manufacturers Group"

* INTEL MULTIBUS SPECIFICATION
  Order Number 9800683-04

* PAGED MEMORY MANAGEMENT UNIT (PMMU) MC68851
  data sheet from Motorola

* MC68020 32-BIT MICROPROCESSOR USER'S MANUAL
  ED. PRENTICE-HALL, Inc.

1.4 PRODUCT PROFILE

SGM2 is a MULTI-USERS MULTI-APPLICATIONS SMALL-BUSINESS SYSTEM designed for general BUSINESS and COMMERCIAL APPLICATION.

It is aimed at the UNIX community by endowing it with the AT&T UNIX Version V Release 2.2 operating system, which supports the virtual memory feature -demand paging-.

Obviosly, to support multiple users, multitasking is required wich allows users to execute parallel tasks. This facility often permits a more natural program structure, simplifying software development.
Yet, multitasking increases throughput, while in turn results in increased efficiency and cost effectiveness.

Today's very large scale integration and peripheral technologies enable marketing of small but powerful Multi-User System (MUS) for a very low cost.
Based on microprocessors central processor units, often coupled to Winchester disk and serial printer, they can support 4 to 16 users.
In addition, the advent of powerful standard Operating System, enables users to migrate from one hardware system to another without leaving their application programs.

The focal point for SGM2 product, is to be completely designed from STANDARD part both for hardware and for software.
We have to avoid the cost of developing proprietary chips by assembling systems using standard, off-the-shelf products.

Microprocessor chips are the functional equivalents of minicomputer processors at a fraction of the cost. And no real performance difference often exists at least at central processor unit level.
Proprietary operating systems have long been the bane of the user population.
If users wished to migrate to other systems, they could do so only at the expense of abandoning their application software.
The use of standard components allows truly portable software, obsoleting the concept of a proprietary operating system.

Bus architecture has become one of the most important factors associated with system design. A bus largely determines the types of applications that a system suits and either enhances or hinders the system's use of the latest LSI and VLSI chip. Thus, to take fullest advantage of a microprocessors based systems, it is worth to consider its bus structure very carefully.
Industry-standard bus configurations are numerous and getting more so allowing a wider choice of standard products.
It is expecially important since with SGM product line, HISI intend also to provide an opportunity for system integrators or resellers who decide to purcase peripherals and-or controllers from non HISI source.

System integrators motivation to start an integration activity using basic SGM hardware, relies upon the attractiveness of good price/performance ratio offered by the product and from

1.5. RATIONALS for MICROPROCESSOR CHOICE

The semiconductor companies, whose customers are just starting to come to grips with the potential performance capabilities of the new 16 bit machines based on such devices as Motorola 68000 and 68010, or the Intel 80186 or 286, are now designing and producing the next generation family of microprocessors.

Among the leading 32-bit microprocessor chip sets currently available or very near at the sampling stage are National Semiconductor's NS32032, Texas Instruments' TI32032 (a second source to National Semiconductor), Motorola's MC68020, AT&T WE32100, Intel's iAPX80386, and Zilog's Z80000.
Each of these basic CPU chip is at a different introduction stage as are the accompanying peripheral chips that make up a 32-bit system.

The basic building blocks of a 32-bit microprocessor system - judging from what is being offered by the top semiconductor vendors- consists of five basic chips that make up what one may label "micro-system".
These include an arithmetic logic unit (ALU) or the microprocessor kernel - MPU -, memory management unit (MMU), floating point coprocessor, interrupt controller, bus controller, and timing controller.
Minor variations of this configuration exist among vendors, but on the whole they are functionally equal.

In their design approach to 32-bit microprocessors semiconductor manufacturers tackled the architectural questions in their own ways.
The result is that system designers must familiarize themselves with the various approaches of the individual IC manufacturers to make the proper choice for their system design.

The first result of this trend is that system designer are now working with the building blocks of a "mainframe computer" in the form of a few chips. And as in mainframes, the basic building blocks must interact in a cohesive manner to ensure maximum throughput.
For these reasons, using a single vendor's blocks makes the most sense because of the sheer number of handshaking signals between the various blocks and the speed at which data must be fetched and interchanged.

The first company to implement a mainframe-type architecture similar to the VAX system, has been National Semiconductor.
Intel's and Motorola's approaches are as different and reflect a basic philosophical difference in how data is best handled between the CPU, the MMU, and the floating point unit.

Without entering in a detailed comparison among all microprocessors' chips today available, we can say that at this point in time, Motorola 68020 MPU's seems to be the best "NEAR TERM" choice due to its availability, the potential for applications and considering the previous choice done in HISI for

1.6. 68020 PERFORMANCE

The instruction timing calculation for the MC68020 is a very complex task.

It is impossible to anticipate the individual instruction timing as an absolute number of clock cycles due to the dependency of overlap on the instruction sequence and alignement, as well as the access time presented by main memory in all different sequence of operations -including I/O activity-.

We can observe that the architecture of the 68020 makes exact instruction timing calculations difficult due to the effects of:

1. ON CHIP INSTRUCTION CACHE

2. INSTRUCTION PREFETCH MECHANISM

3. INSTRUCTION EXECUTION OVERLAP

4. OPERAND MISALIGNMENT

These factors makes 68020 instruction set timing difficult to calculate on a single and unique instruction basis since instructions vary in execution time from one program context to another.

A short analysis of each of these factors influencing 68020 instruction time performance, follows.
For detailed information refer to "MC68020 32-bit microprocessor user's manual. ed. Prentice - hall, Inc."

ON CHIP INSTRUCTION CACHE

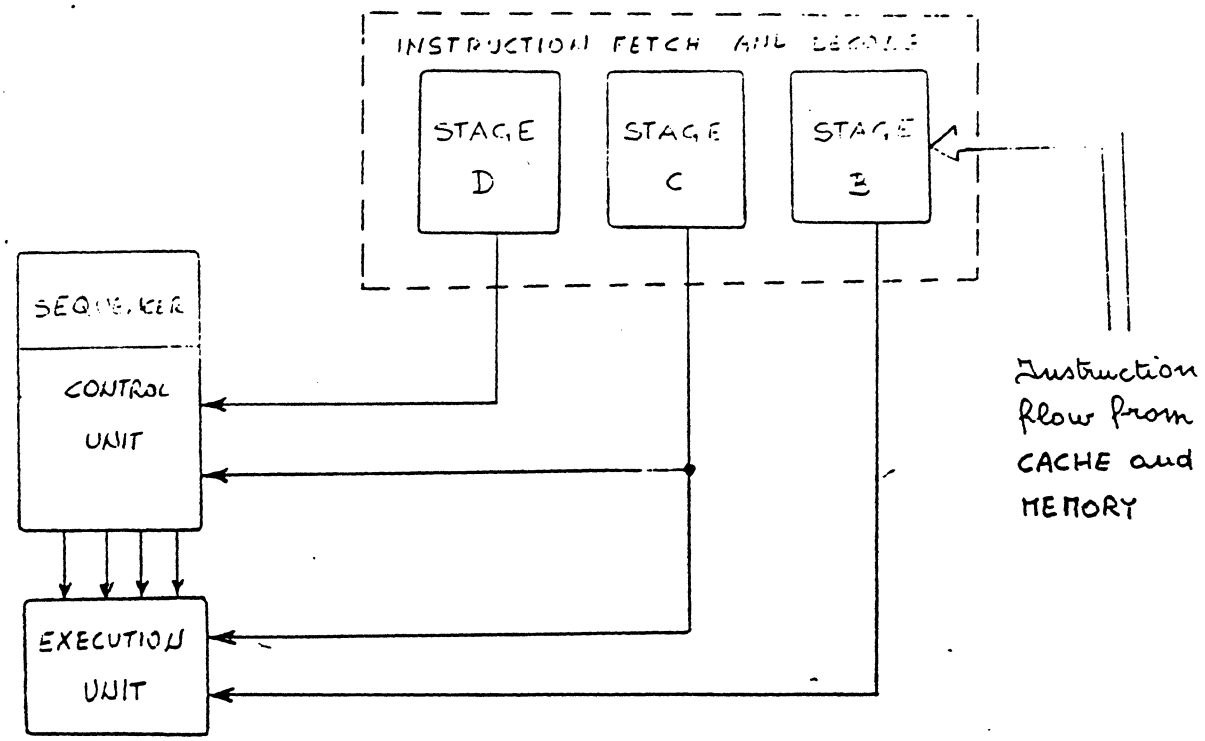
The MC68020 incorporates an on-chip cache memory as a means of improving the performance of the processor.
The cache is implemented as a CPU instruction cache -data are not cached- and is used to store the instruction stream prefetch accesses from the main memory.

Since program spend most of their execution time in a few main routines or tight loops, once captured in the high speed cache they can execute directly from the cache.
Thus the processor does'nt suffer any external memory delay.

The performance is also improved by allowing the MC68020 to make simultaneous accesses to instructions in the on-chip cache and to data in main memory.

The 68020 on-chip instruction cache is a direct-mapped cache of 64 long word entries (256 bytes); information are stored and retrived on logical -virtual- address base.
The cache itself is accessible only by the internal 68020 control unit.

To manipulate the cache entries, the software -kernel- has a


STAGE D
STAGE C
STAGE B
CONTROL UNIT
EXECUTION UNIT
Instruction flow from CACHE and MEMORY

- Figure 1.6.1. -

is maximum overlap on the instructions execution.

Starting from these assumptions, we have obtained the following AVERAGE INSTRUCTION TIMING (in number of clocks):

==========
Total n. of clocks: 6.51
==========

where for each instruction, we have:

* .343 memory access for data read operation
* 1.0 memory access for instruction fetch
* .31 memory access for write operation

CHARACTERISTICS	CONVERGENT MIGHTY FRAME	CONVERGENT MULTI FRAME
* N. of USERS	8 - 32	Up to 128
* MPU used	68020	68020
* MPU CLOCK (MHz)	12.5	16.67 (20)
* MPU BUS	PROPRIETARY	PROPRIETARY
* MMU	?	VAX - TLB
* FLOATING POINT	68881 opt.	?
* MEMORY SIZE	16 MB (2-4)	?
* MEMORY CHECK	?	EDAC
* MEM. CYCLE TIME (nsec)	?	?
* CACHE TYPE	NO Cache	Set Assoc. 2 Levels
* CACHE SIZE	Not App.	16 - 64 KB
* CACHE HIT RATE	Not Appl.	80% - 95%
* WAIT STATES per MEM. ACCESS	1	2
* WAIT STATES per CACHE ACCESS	Not App.	0
* I/O BUS TYPE	Multibus, VME	?
* S/W I/O TYPE	?	REAL
* MULTIPROCESSOR	NO	YES; Only
* MULTI MPUs S/W APPROACH	Not App.	MULTI KERNEL
* OPERATING SYST.	UNIX S V	UNIX S V R 2.2
* VIRTUAL MEMORY MB/USER	18 - 20	?
* ANNOUNCE DATE	SEPT. 85	Not Avail.
* DELIVERY DATE	3 Q 85	2Q86
* OTHERS		

2. SGM2 SYSTEM ARCHITECTURE

2.1. GENERALITY

Spanning a wide range of marketing applications as today occurs for a UNIX based system, means to provide a large amount of computer power beside complex hardware capabilities.

For instance, word processing and office automation systems, require to integrate computer and communication technologies to automate document typing, storage, transmission, retrival, and hard copy generation.
That means to assure fast character transfer between bulk storage and CPU, as well as high speed communication between the CPU and the terminals, without impacting the significant processor power required for shared resource word processing.

Different system capabilities are required for traditional stand alone system even if used with interective application programs.

To match these different and for many aspects opposite requirements with a product characterized by a shop cost goal quite tight, it is necessary to design an hardware system whose architecture relies on the following major points:

- assure the speed of CPU -i.e. Interior Decor Processor- as high as possible (high instruction execution rate).
The I/O operations must produce minimum or no impact on IDP instruction execution activity. Orders exchange between IDP and I/O controllers must be as fast as possible.

- I/O operations (mainly for disk devices and local terminals) executed under the control of autonomous dedicated controllers or processors.
In other words instead of relying on a unique input/output channel shared by all I/O devices for data transfer operations and very simple device controller specific for each device, it has been chosen to design independent controllers or processors performing both tasks.

For I/O controllers or processors, major objective is to achieve low shop cost with no sacrifice in performance.
The solution rely on the capability to devide the workload of the functions to be performed in different groups identifying an aggregation for those devices that have some level of commonalities, like rigid disks and floppy, and provide a unique controller or processor for these homogeneous peripheral units.
This allows a cost optimization design since it is possible to share the intelligence of the controller/processors among multiple I/O peripheral devices.

To assure a good level of flexibility for connection of I/O devices to the SGM2 system, a bus structure has been adopted.
In SGM2 there is a "SYSTEM BUS" that is the main path used by controllers-processors to communicate each other and with the

main memory block.

AS the needs of the customer grow and the use of the computer increase, the system must be expandable to provide enhanced performance and new application capabilities.

SGM2 system has been designed to offer an "expandable architecture" which will accomodate added processor (CPU and I/O), added memory, peripherals and communications equipment in a wide variety of configurations.

The architecture is adequate to support 32 bit wide data transfer and accepts also off-the-shelf standard controller/processor boards (at First Customer Shipment, VME based and later also MULTIBUS I).

The Interior Decor Processor -IDP- and related Sub-Systems including I/O controllers/processors and I/O peripheral devices, are field upgradable whithout requiring substitution of already existing peripheral equipment.

Hardware and Software shall be upward compatible from the smallest configuration to the largest.

2.2. HARDWARE OVERVIEW

SGM2 System is a high performance, 32-bit computer, based on Motorola 68020 microprocessor and UNIX System V Release 2.2 demand paging virtual memory.

It will be characterized by these two basic configurations:

* MONO PROCESSOR version (or MONO IDP) using a unique IDP -one 68020 MPU, one MMU, one cache, and one Main Memory sub-system-

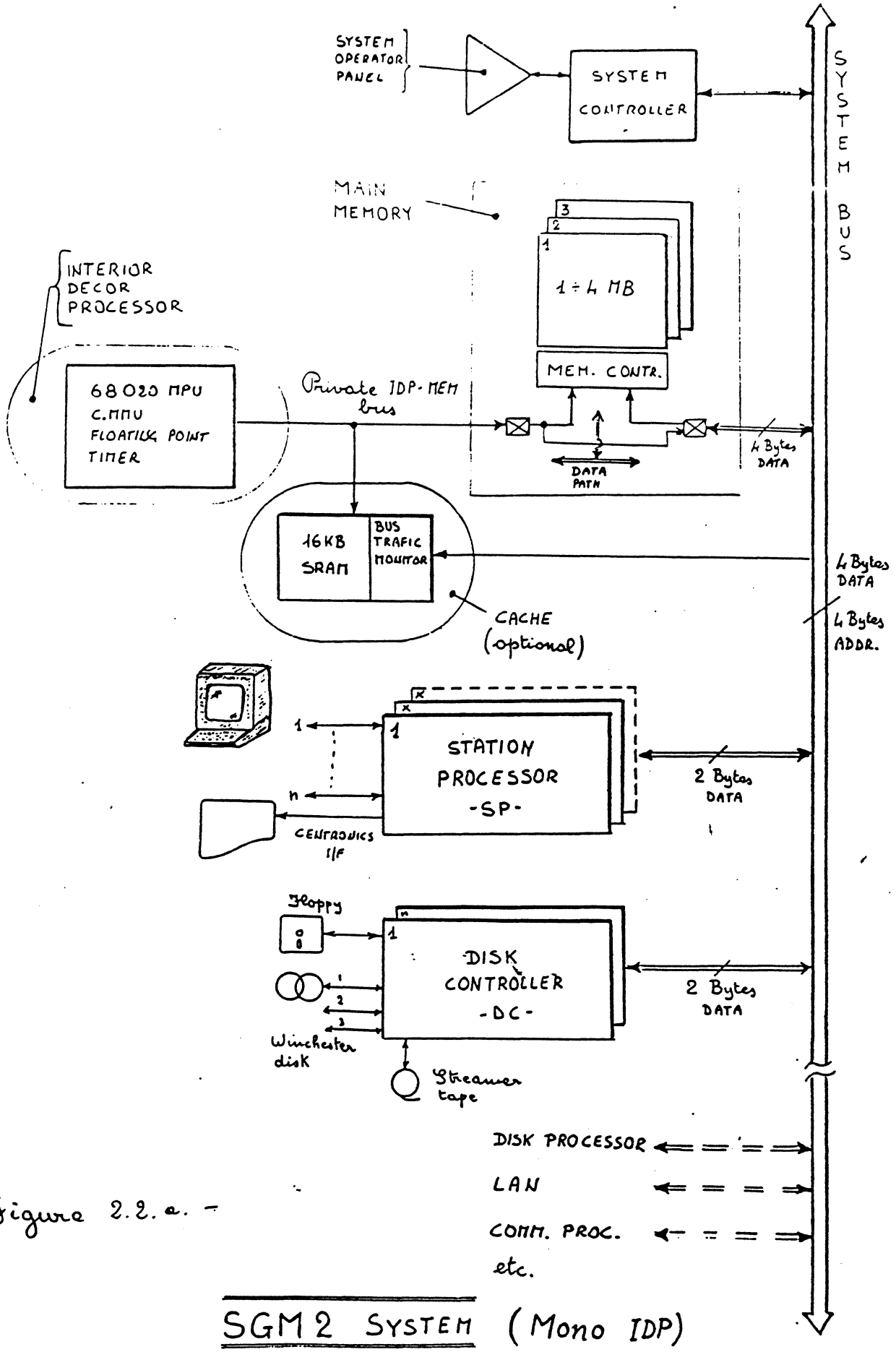
* DUAL PROCESSOR version (or DUAL IDP) having two IDPs sub-systems each one with proper cache and Main Memory Sub-system.

From the point of view of an application program, a computer system can provide one or more "PROCESSING ENGINES" to execute application code.
A computer that provides exactly one such "ENGINE" is considered as a MONO-PROCESSOR system.
A computer providing two or more such "ENGINE" is a MULTIPROCESSOR.

Note that this application program viewpoint makes this classification independent of other, "SPECIAL PURPOSE PROCESSOR" which do not directly execute application programs, namely smart disk controller or processor, terminal processor, LAN processor, etc.

In SGM2 system, I/O Controllers/Processors can be added to the basic system according to the specific user's needs.
Easily expandible, the SGM2 can grow from a dedicated dual/three

- MAJOR BLOCK DIAGRAM -



- Figure 2.2.a. -

SGM2 SYSTEM (Mono IDP)

2.2.1. SYSTEM CONTROLLER

System Controller -SC- PWA, provides support for:

* Clock- calendar battery powered.

* System bus control and monitor, namely: VME bus arbiter, and bus time-out control.

* "Interrupter" hardware support for IDP to IDP communication. This feature is used in SGM2 dual-processor configuration, to notify interrupts between IDPs processors.
"Interrupter" function is implemented using Motorola 68153 Bus Interrupter Module chip (see circuit data sheet for details).

* "EPROM Primary Bootload"

* System "Power Supply" control and monitor, for:

  -- Power on procedure

  -- Power off procedure (software controlled)

  -- Memory battery backup supervisor.
     "AC FAIL" interrupt generation;it appears to IDP at level 7

* System Operator Panel -SOP- control.

  SOP contains:
  -- two exadecimal disply used mainly by Diagnostic purpose.
  -- AC ON; lamp.
  -- DC ON; lamp.
  -- MEMORY POWERED via BATTERY BACK-UP; lamp.
  -- STAND-BY; lamp. It indicates that the system is under controlled power-off sequence.
  -- HARDWARE SYSTEM FAULT; lamp. Software driven; used to indicate that the system has a blocking hardware failure.
  -- SOFTWARE FAULT; lamp. Driven by the software and indicating "UNIX panic" status.
  -- DC POWER ON; pushbutton. Causes system power on and initializatiion.
  -- RESET; pushbutton. Causes system initialization when the system is already powered up.
  -- POWER OFF; pushbutton. Causes a software controlled power off sequence.

System Controller is always present in a unique copy physically placed in the first SGM2 cabinet.

2.2.3. INTERIOR DECOR PROCESSOR

Based on Motorola 68020 MPU running at 16.67 MHz clock rate, which corresponds to 60 nsec. clock period.

* "0" Wait State when operating with the CACHE
* "2" Wait States when operating with Main Memory (4 Bytes acces)

To access controllers/processors' resources connected to system bus, the 68020 takes 4 - 8 wait states according to the delay experienced to grant VME bus.
This figure doesn't include any additional delay caused by accessed resources (access time resource specific).

A floating poit co-processor -68881- is optionally available.

256 KBytes EPROM (128 KWords) is used for diagnostic purpose and initialization.

Custom Memory Management Unit -CMMU- hardware support, is provided.
This CMMU is derived from SUN MMU providing adequate enhancement to support demand paging virtual memory.

On IDP PWA, are implemented also the following functions:

-- system interval timer (68230 chip) -24 bit parallelism-.
   50 Hz frequency, 20 milli sec. period, software programmable, interrupt to IDP level 7.
-- profile timer (asynchronous, 100 Hz frequency, interrupt level 7) used to dynamically "profile" the usage of system resources produced by the program
-- status register
-- command register.
   Used to: control the cache, and 68020 interrupt mask.
-- arbiter to control memory access
-- VME bus interface
-- memory extension interface
-- cache interface

Interrupt handler hardware allows to map internal to the board and external originated interrupts, to be communicated to 68020.

IDP is implemented on a single PWB.

2.2.5. CACHE

SGM2 system can be equipped with an OPTIONAL CACHE aimed to increase the internal calculation speed of 68020 reaching the top allowed by this MPU and, at the sametime, decreas main memory loading produced by internal calculation.

Performance objective of the present cache, is to offer an hit rate at least equal to 80%.

Follows a summary description of this sub-system:

* Caching function, performed for both data and instruction (user and supervisor)

* Size: 16 KBytes (SRAM chip)

* Sector Associative (2 KBytes x 8 Sectors)

* Write through strategy

* Updating algorithm: FIFO

* Cache to Memory path: 4 Bytes

* Parity checked at Byte level on data RAM.

* 68020 instruction rerun on Missing occurrence
"EARLY HIT" technique has been implemented

* Three "Looh-aside" registers are used to speed up cache and memory accesses (1 for program space, 2 for data space)

* "VME bus traffic monitor" to invalidate cache entries when I/O operations are in progress.

These lines can come also with RS422 interface, asynch., with max speed up to 56400 bps.

* 2 lines 9 pins connector; RS232/422, asynch.; usable for local connection only; speed for RS232 up to 19.2 kbps, RS422 up to 56400 bps.

* 2 lines 15 pins connector; RS422 ; used for local connections only; synch. with the speed up to 100 kbps, asynch. with speed up to 56400 bps.

LPO is implemented on 1 PWA.

2.2.8. PHYSICAL STRUCTURE and CONFIGURATIONS

SGM2 full configuration, consists of two floor-stand cabinets.

First cabinet: for MONO-IDP system with the possibility to connect in a balanced configuration, up to 32 users (terminals).

First plus Second cabinet, for DUAL-IDP system with the possibility to support up to 64 users (terminals).
The second cabinet expansion, can be used also just as I/O expansion of SGM2 first cabinet configuration; so it is not a must to have in the second cabinet the 2nd IDP sub-system.

Anyway when 2nd IDP is added, it always recall for the second cabined with its own memory and cache.

A third cabinet will be available as an expansion option for those customers that would like to connect to SGM2 system a MULTIBUS I standard bus.
MULTIBUS expansion will be obtained through an "adapter board" that provides conversion from SGM2 system bus (VME) to MULTIBUS I.

SGM2 system bus is VME Standard Bus.
So standard VME board can be directly plugged on the SGM2 system bus.
Since SGM2 native PWBs, have larger dimension compared to VME ones, (SGM2 native PWB is 10.4" x 14.4") mechanical adaptation is required in order to plug VME boards into SGM2 slot.

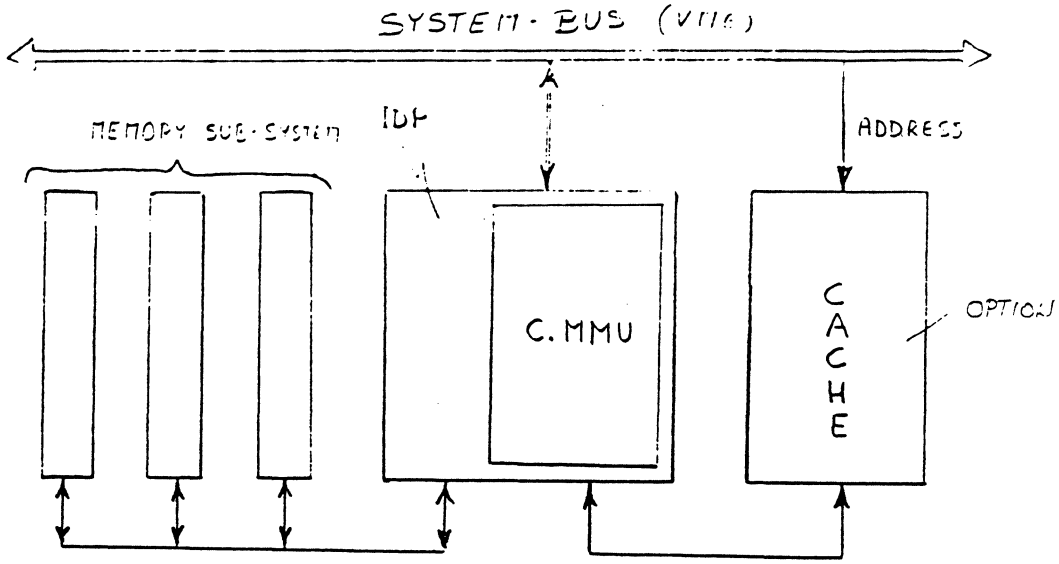
Figure 2.28.a. depicts the physical structure of SGM2 card cage.

As can be seen by the figure, some slots position are exclusively dedicaded to hold specific sub-systems; other are "anonymous positions" and can host any kind of PWA either native or VME standard.
For each cabinet we have 7 slots available to accept any kind of boards (DC, SP, LP, or VME boards).

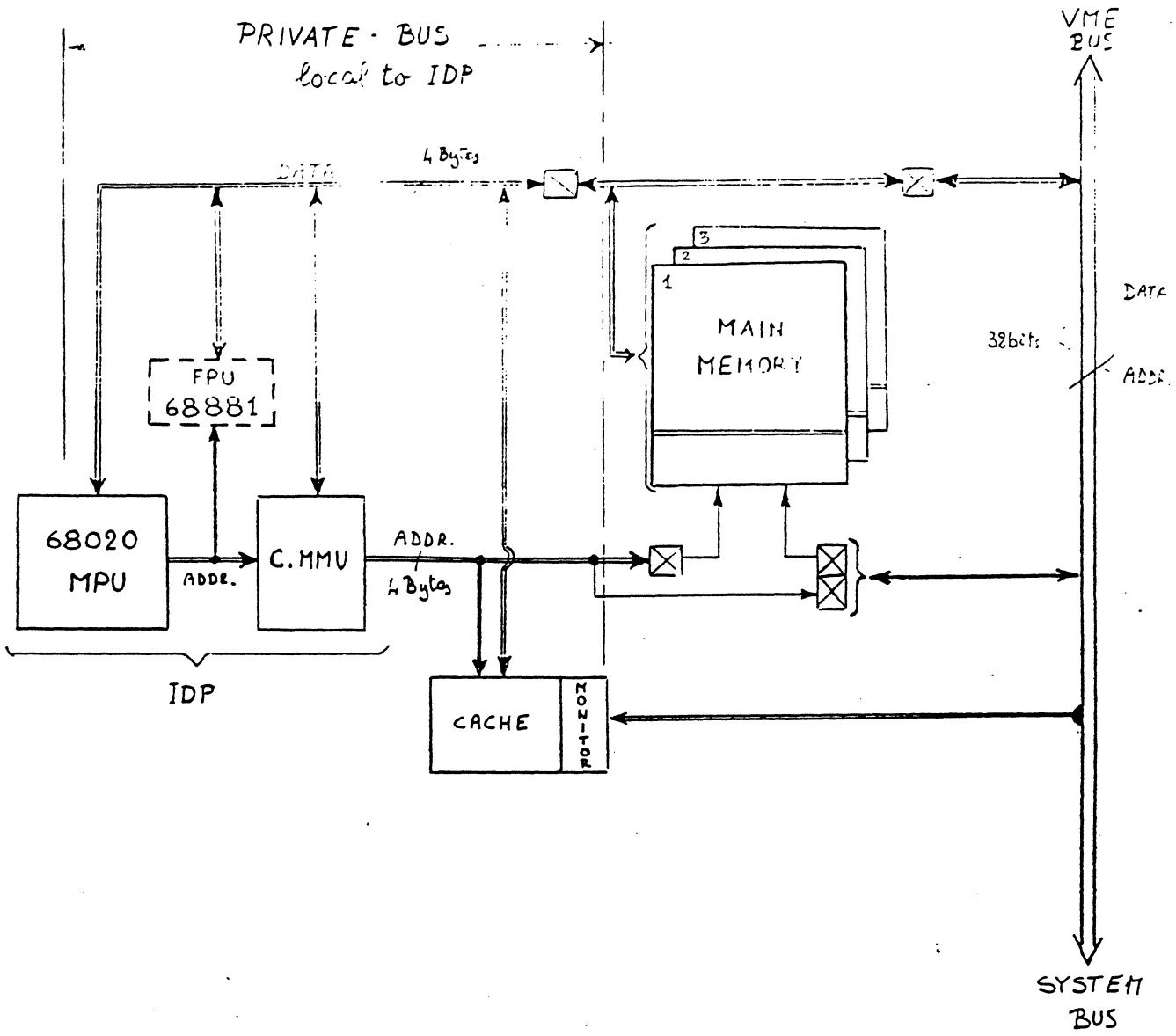
Each slot, through a predefined cabling on the back-plane, has assigned a "code" that can be read by the controller/procesor installed.
Using this feature, the software can determine the system configuration in place.

As far as configuration is concerned, SGM2 can accept any mix of native controller/processor.
Specifically, disk controller and disk processor can coexist in the same configuration.

Figure 2.2.8.b. depicts the physical connections that allow the IDP to dialogue with Main Memory and with the Cache.
As can be seen from this figure, C.MMU board is physically installed on IDP's PWA as a daughter board, so it doesn't appear on the block diagram of figure 2.2.8.a.

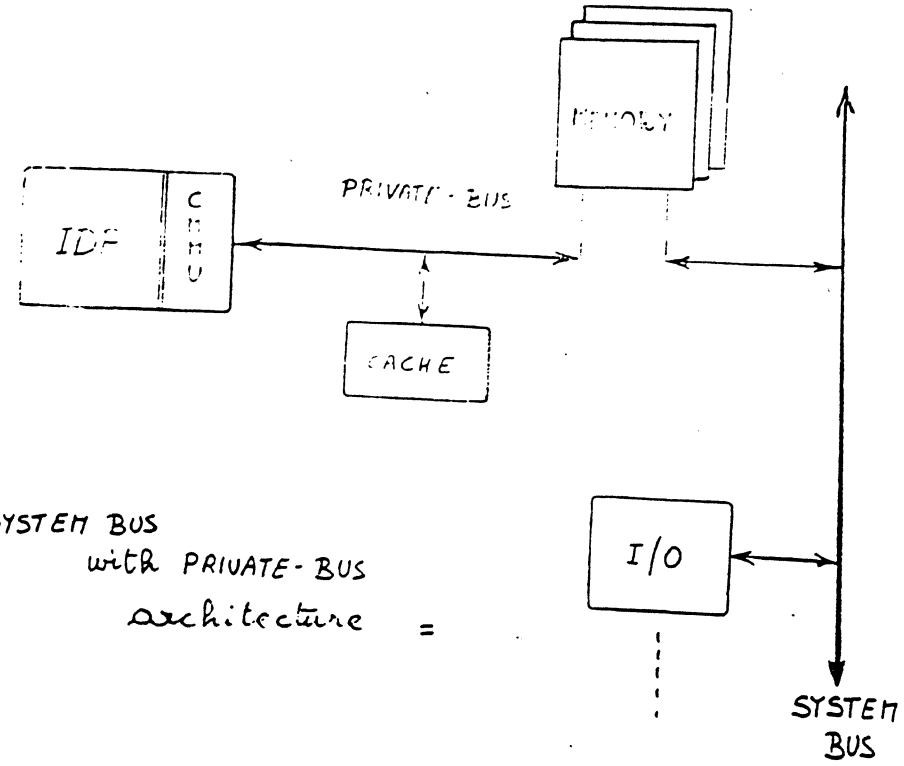

SYSTEM · BUS (VME)
MEMORY SUB-SYSTEM
IDT
ADDRESS
C. MMU
CACHE
OPTION

- Figure 2.2.8.b. -

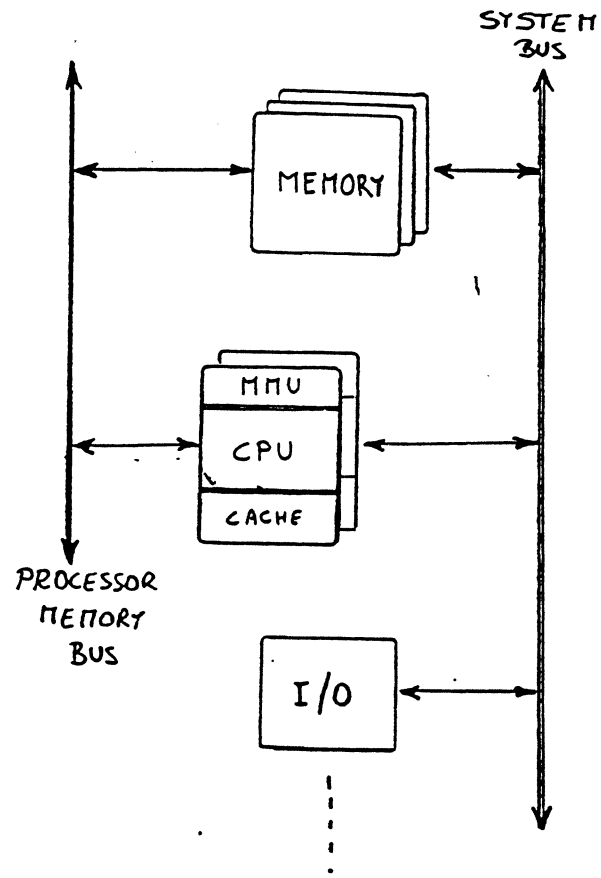


SGM2 - "SYSTEM BUSES" STRUCTURE

- Figure 2.3.1.a -



= MONO-SYSTEM BUS
with PRIVATE-BUS
architecture =



= FULL DUAL-SYSTEM BUS
architecture =

Figure 2.3.1.b. -

So, even one good reason to have Multibus II, that is full compatibility with previous products developed with Multibus I interface, is no longer totally applicable.

In SGM2, VME bus is fully supported, that is all the features provided by the standard bus are available.
Specifically SGM2's backpanel will support VME, VMX, and VML "buses" connections; therefore those controller/processor that will use this channels, will be fully supported.

According to the VME standard, SGM2's controllers/processors developed by HISI have the following characteristics:

	1	2	3	4	5	6
SYSTEM CONTROLLER				X	X	X
IDP	X	X	X			
MAIN MEMORY				X		
STATION PROCESSOR				X	X	
DISK CONTROLER	X	X		X	X	
MULTIBUS INTERFACE	X	X	X	X	X	

WHERE:

1 - MASTER
2 - REQUESTER
3 - INTERRUPT HANDLER
4 - SLAVE
5 - INTERRUPTER
6 - BUS CONTROLLER

For a detailed description of the above indicated functions, refer to "VME System Architecture Manual".

2.3.3. SYSTEM MEMORY ARCHITECTURE

The organization and management of system main memory has always been one of the most critical factors in system design. Advancing integration has recently removed the memory capacity constraints on most microprocessor based systems by allowing memories of 4 MBytes or more on a single board (PWB).

System memory provides a shared memory space that can be accessed by IDP and I/O processors connected to the System bus. Sharing system memory among all processors is a very cost-effective solution.
Since system memory typically can be expanded simply by installing additional memory boards, this modular flexibility allows the system to grow in terms of configuration, according to the user needs.

The tradeoffs with using shared system memory instead of local memory in multi-processors systems (dual IDP), include the fact that access to system memory may be slower than access to local memory because processors must contend with each other for access to the system bus.

For SGM2 system in Dual-IDP configuration, the overhead payed for having shared system memory, is in any case very low due to the specific architectural solution adopted for multiprocessor (see specific paragraph on Dual-processor).

In SGM2 the system memory has been designed according to the dual-port technique.
Dual port memory is a single memory sub-system that combines many of the advantages of both "Local" (to the MPU) and System - global- memory.

Generally speaking as a Local memory we mean a memory that provides a processor with private memory space that is not accessible to other processors, except those that share the processor's local bus.

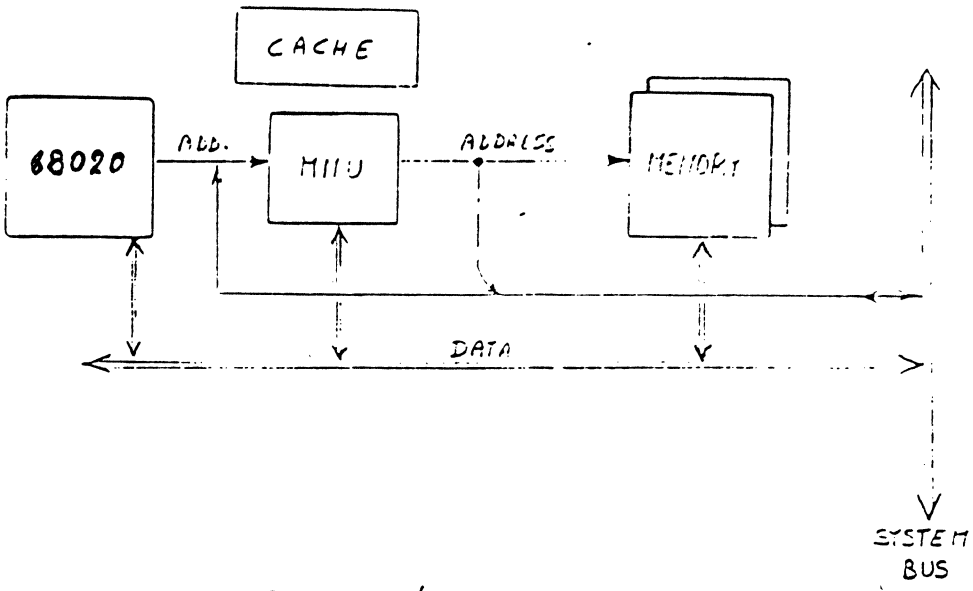
System memory provides a shared memory space that can be accessed by all processors connected to the system.

Shared system memory is often a cost-effective alternative to using local memory when individual processors may occasionally require e relatively large physical memory space.

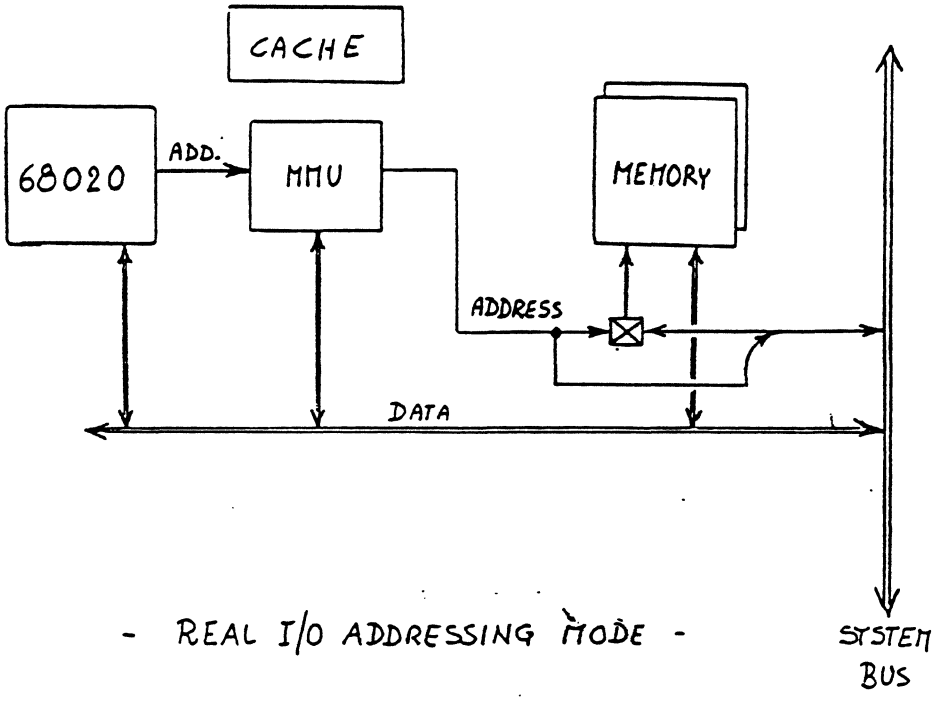
Dual-port memory appears both as local memory to the processors on a single local bus, and as system memory to other processors in the system.

Dual- port memory permits the local processor to have high-speed access to the dual-port memory without tying up the system bus. In this way, when the system is equipped with the cache, the main memory can be used in overlapped way by controllers/processors connected to the system bus.

Offsetting these advantages is the fact that dual-port memory typically is more complex than a single-port memory sub-system.



- VIRTUAL I/O ADDRESSING MODE -



- REAL I/O ADDRESSING MODE -

- Figure 2.3.4.a. -

(See also "Memory utilization" chapter)
This leads to have a theoretical memory reference every 3.9 clock cycles, (60 nsec.) x (3.9 cycles) = 234 nsec.

With a "I/O real addressing" mode architecture and a cache sub-system, main memory accesses can be reduced to 35% of all references.

This conclusion is derived from the following formula:

(percentage of bus accesses = read miss rate x percentage of read accesses + percentage of write accesses)

= (0.2 x 0.81) + 0.19 = 0.162 + 0.19 = 35%

Thus, with a cache, the system can sustain an higher throughput since the main memory occupation for internal calculation, equals 35%.

2.3.5. MAIN MEMORY UTILIZATION

Many features of the 68020 MPU contribute to an high utilization of the system memory.

These features include:

* instruction prefetch mechanism

* high number of memory references per "Average instruction".

Depending on the particular program that is being executed and the speed of the main memory sub-system, the intrinsic feature of 68020 can produce a very high memory utilization.

For typical types of software, as an average, we have for every instruction execution (that is 6.51 number of clocks), 1.65 memory references.

With a 16.67 MHz clock frequency, 68020 engage the memory every 237 nsec.

If the number of wait states equals zero, having a memory cycle time of 180 nsec, we obtain a memory occupation of:

(180 x 1.65) / (6.51 x 60) = 76%

For SGM2, this corresponds to the "cache" case with 100% hit rate.

With 2 wait states, and hence having a memory cycle time of 300 nsec, we have:

(300 x 1.65) / (6.51 x 60) = > 100%

that is, the basic instruction execution time that theoretically equals to:

2.3.6. INPUT/OUTPUT ARCHITECTURE

Hardware system architectures can be characterized by the functional capabilities and interconnection structures of major system components.

In the case of I/O system architecture, it is usefull to distinguish between:

a. basic hardware capabilities and inter-connection structures required simply to perform data transfer operations between main memory and peripheral devices.

b. extended hardware facilities used to support, in appropriate way specific I/O operations, e.g. disk, communications, etc.

The basic I/O hardware facilities must provide for:

1. coordinated initiation and termination of I/O operations in conjunction with CPU operations -68020-

2. control of device operations as required by mechanical part of I/O devices.

3. multiplexed use of interconnection path between I/O controller, devices and main memory.

4. access rights control of main memory areas -addresses- used by I/O operations.

In many computer system basic I/O hardware architectures, these functions are distributed accross memory controller, I/O controller or processor, I/O channel, and device controller components.

Extended I/O hardware facilities typically involved the use of processors and memories, in conjunction with basic I/O hardware, that are dedicated to executing same portion of specific I/O activity.
Within this concept of extended I/O facility falls also technical solutions identified by terms such as Front End Processor, File Caching processor,Backend processor,Communication processor, etc.

As far as SGM2 hardware I/O architecture is concerned, it is worth to note that the system performance can increase significantly if the burden of handling I/O chores (both serial and parallel and other related functions) is offloaded to dedicated controllers or intelligent processors.

Unix works in a highly multiplexed environment; it place a heavy load on its I/O devices expecially on disks.

As a result of the heavy load placed on the disk file system for program, data, and temporary storage, a UNIX based system is typically paced by its system disk.

2.3.7. WORKSTATION PROCESSOR AND COMMUNICATION

As per PFS requirements, SGM2 Communication Processor must be able to connect serial lines with a single line max speed and throughput of 19.2 kbits/sec for local/remote connections.
Each line must be configured to support, in any mix, the following:

* asynch./synch. modes
* bit/byte oriented protocol
* 7 + 1 and 8 bits character format
* RS232 and RS422 electrical interface

As far as SGM2 to host connection is concerned, PFS requires to have high speed channel (remote link) with a data rate higher than 72 kbits/sec. since products' example exists that today offer line speed to 100 - 150 kbits/sec.

To serve local link connection -CPU to CPU- several hundreds of kbits/sec., is required -up to 500 kbits/sec.-

<Ref. SGM2 PFS draft version; Nov., 11, 1985>

In order to properly define this "Processor" the following points must be considered.

* 8 lines with max speed and throughput per line of 19.2 kbits/sec. (TWA), means a processor with a real throughput of 153.6 kbits/sec if we consider a continuous stream of characters transmitted over the serial line.
In other words this case correspond to a theoretical situation in which burst of endless characters "cross" the processor without rest.
More real environment, applicable either to terminals controlled by operator or remote communication lines, operates on the basis of burst packets of characters paced by pauses in between packets. This fact leads to design a processor which real throughput is not simple the "line speed times the number of lines connected", but this figure is calculated on a statistical base.

* Considering to design a processor with a "real throughput" of 153.6 kbits/sec., with 10-bits character, asynchronous procedure, TTY protocol, it is required to handle one characters in 65 microsec. -character processing-. This if we have 8 lines each one with equal speed of 19.2 kbps.
Even worst are the things when one line has a speed very high, e.g. 76,8 kbps., compared to the other lines. In this cases, attention must be paid to verify that within the interrupt period of fastest line it is possible to service at least two characters.

If the "processor" is implemented with a micro whose average instruction (or microinstruction) time is 1 microsec., we have just 65 assembler's instructions available to completely handle one character (in addition to byte processing we have also to consider contest switching, interrupt servicing, etc.).

* All the above required features combined together in a unique processors, leads to a product very cumbersom either from the point of view of physical implementation -number of IC components required- and of course from the point of view of the shop cost.

If the ICs "real estate" occupation exceeds one single PWB, we are faced with a problem of maximum number of serial I/O supportable by the entire SGM2 system (this number for dual-IDP configuration is 64); that is to have just 8 PWB slots reserved for 64 terminals connection.

Since the lines' connectors are physically placed on the edge of processor's PWA, we are limited by the number of connectors (and pins per connector) supported by PWA.

Taking into consideration all the above points, we have approached the communication processors' definition, according to the following rationales:

* design different products that responds to these basic requirements:

  ** high-performance low-cost processor optimized for local terminal connections; 8 lines; Asynch. TTY protocol only (byte oriented)
  This processor is named Station Processor SPO.
  See for details "SGM2 PDD"

  ** high-performance processor for local/remote connections; 64 lines; Synch./Asynch.; byte-bit protocol oriented.
  This processor is named Line Processor LPO.
  See for details "SGM2 PDD".

  ** controller with 8 lines for local terminal connection; asynch.; TTY protocol.
  This controller is a very low-cost product and uses the power of IDP -68020- to directly control the serial I/O circuits. Neither a microprocessor nor memories are provided on this controller.
  The name of this controller is: Workstation Controller WSO and up to now it is just provided as a "debugging tool" for SGM2' workstation software driver.

While SPO has as a primary objective of low shop cost with competitive performance, LPO is aimed to reach a high level of performance and a complete support of various line protocol disciplines.

SPO and LPO are microprocessor based processor; they both use 68000 MPU running at 12,5 MHz clock rate that corresponds to 80 nsec. clock period.
With such clock rate, the fastest instruction takes 4 clock cycles that corresponds to 320 nsec.
As an average, we can consider that with "0 wait states", the micro processor runs average instruction in 800 - 1000 nsec.

The SIO used on both products, is Motorola 68564 (see Motorola data sheet, for details).
Considering this component and the physical interface to VDU, the

2.4. CACHE

2.4.1. GENERALITIES

The high-speed data processing that is possible to gain using Motorola 68020 MPU running at presently maximum clock rate of 16.67 MHz, can be lost if it is not adequately supported with fast main memory.

For SGM2 System, "Cache memory feature" attempts to overcame this potential problem since being main memory -implemented with DRAM chips modularly expandable up to 12 MBytes- we have to pay 2 wait satates for every 68020 memory access.

A cache memory is a fast buffer located between the "68020/MMU" and the "main system memory".
Keeping frequently accessed data in the cache produces at least two benefits.

First, it reduces the average access time for the processor's memory requests, increasing the processor's throughput.

Second, it thereby reduces the processor's utilization of the available memory bandwith, allowing other devices on the system bus to use the memory without interfering with the processor.

Cache "efficency" is based on "property of locality" probability principles, which experience has shown to have the following characteristichs:

* first, over short time period, most MPU memory accesses are made to adjacent, small groups of locations; therefore, even a small cache, storing carefully selected data, will have data the MPU needs most of the time.

* second, data stored in the cache and recently used will likely be reused shortly thereafter.

* finally, data adjacent to data that have been recently used will most likely be used next.

For SGM2, "cache feature" has to be considered as an optional board to be offered as an way to increase 68020 MPU internal calculation speed.
Following this approach, it is possible to design caches with different complexities, in such a way to have complementary choices that allow to reach different level of system performances.

The basic problem that a designer is faced with is to design a cache with proper carachteristics that leads to maximize the ratio between performance/shop cost.
One difficulty in providing definitive statements about aspects of cache operation, is that the effectiveness of a cache memory depends on the workload applied to computer system.

parameters can be varied at will and experiments can be repeated and reproduced precisely.

The principal advantage of measurement over simulation is that it requires 1 to 0.1 percent as much running time and is thus very valuable in establishing a genuine, workload based, actual level of performance (for validation).
Actual workloads also include supervisor code, interrupts, context switches, and other aspects of workload behavior which are hard to imitate with traces.

2.4.3. OVERVIEW OF SGM2 CACHE DESIGN

Starting with the goal to minimize the time to access information that are in cache, i.e. minimize the cache access time, and considering that 68020 MPU requires the data after 115 nsec of address issuing -logical addresses-, we have verified that it is possible to design a cache that respects these constraints.

Designing a cache with 115 nsec of access time, leads 68020 MPU working with "0 wait states", that is exactly the objective to operate with the maximum speed allowed by the MPU working at 16.67 MHz.

As before explaned the effect of the cache is to increase the processor's throughput by eliminating "wait states" of 68020 and to reduce the processor's utilization of the available memory bandwith, thus allowing other bus masters to operate without reducing the main processor -i.e. 68020- throughput.

To run without "wait states", a 16.67 MHz 68020 requires a transfer acknowledgment from the memory within 115 nsec after the logical address issuing.

The MMU -Motorola PMMU in this case- takes 60 nsec to map the logical address into physical address.

A cache able to access its "directory" and comparing the "Tag" in the remaining 55 nsec would be a complicated and costly design.

Of course we consider here a "Real Addressing Cache".
In computer systems with virtual memory -the case for SGM2 system-, the cache may potentially be accessed either with a real address (Real Address Cache) or a virtual address (Virtual Address Cache).

If real addresses are to be used, the virtual addresses generated by the 68020 must first be translated; this operation is generally done by MMU.

Direct virtual address access is faster, since no translation is needed, but causes some problems.

In a virtual address cache, all the content must be invalidated every context switching, (i.e. at process swapping or

Other placement algorithm may be defined.

LINE SIZE

"The line" is the unit of transfer -usually in bytes- between the main memory and the cache.

The data width of main memory should usually be at least as wide as the line size, since it is desirable to transmit an entire line in one main memory cycle time.

To define this parameter it is also worth considering the relationship between prefetching and line size

REPLACEMENT ALGORITHM

When information is requested by the 68020 -from main memory- and the cache is full, some information in the cache must be selected for replacement.

Various replacement algorithms are possible, such as FIFO, LRU and RANDOM.
As can see the set of feasible solution is still large, but many of them can be rejected for cost and complexity reasons.

MAIN MEMORY UPDATE ALGORITHM

When the 68020 executes instructions that modify the content of the current address space, those changes that are immediately applied to the cache content when there is cache hit, sooner or later must be reflected in main memory.

There are two general approaches to updating main memory: stores can be immediately transmitted to main memory -WRITE THROUGH- or stores can initially only modify the cache and can later be reflected in main memory -WRITE BACK-.

USER/SUPERVISOR, DATA/INSTRUCTION CACHE

Cache design strategy can contemplate the possibility to split the cache into many parts each one dedicated to store specific information: user, supervisor, data and instruction.

The frequent switching between user and supervisor state in most systems results in high miss ratio because the cache is often reloaded (process switch).
One way to address this is to incorporate two cache memories, and allow the supervisor to use one cache and the usert programs to use the other.

The same approach can be applied to data and instructions.

CACHE SIZE

It is obvious that the larger the cache, the higher the probability of finding the needed information in it.

Of course detailed performance prediction activity must be pursued in order to reach a good level of confidence mainly related to specific workloads to measure SGM2 against.

2.4.4. SGM2 CACHE CHARACTERISTICS -TENTATIVE-

* Real addressing cache

* Cache-memory mapping algorithm: SET ASSOCIATIVE (4 sets levels-)
Actual SGM2 implementation is based upon SECTOR ASSOCIATIVE approach. This mapping algorithm has been chosen because the better access time achievable.
Assuming as a constatnt fact, that the cache has to provide a "0" wait state access to 68020, it is worth to note that the mapping algorithm to adopt must be selected in order to reach the highest hit rete possible; in any case no lower than 80%.

* Size: 16 KBytes (each level is 1K by 4 Bytes)

* Line size (unit of transfer from memory): 4 bytes (two consecutive memory fast read -nibble or page mode-, under evaluation)

* Unique cache for data and instructions

* Cache fetch algorithm: on demand (no prefetching)

* Replacement algorithm: FIFO (or RANDOM)

* Main memory update algorithm: write through

* Access time: such to allow 68020 running at 16.67 MHz to spend "0 Wait states" when cache acceses occur.
No penalities on access time to memory payed, when cache misses occur.

* Directory and data banks, parity checked.

* Input/output activity: hardware monitor that provides to invalidate cache entries corresponding to address space affected by I/O operation.

2.4.5. DATA from COMPETITION's SYSTEMS

SYSTEMS	-	KNOWN CACHE CHARACTERISTICS
* ALTOS 3068 (68020 MPU)	-	8KB unique cache for data and instructions - Set associative: two set -level- each of 4 KB - Cycle time: "0 wait states" when 68020 operates with cache. 68020 clock frequency: project design up to 18 MHz.
* MULTI FRAME	-	16KB unique cache for data and instructions

2.5. MULTIPROCESSOR SYSTEM (DUAL-IDP)

2.5.1. INTRODUCTION

Multiprocessing is not a new idea in computer architecture, rather it is an idea whose time has come.
Neverthelless, designers of UNIX system during the inception of the product in 1969, could not have anticipated the various environments in which their OS eventually would be applied.

Thus, the UNIX system until recently had no provision for supporting a multiprocessor environment or transaction processing mission.

Nowdays, the 32-bit microprocessor toghether with its co-processor extension capability promise to support a design technology that will enable the computer architect to configure high performance system made of low-cost components.

This goal is certainly appealing and its potential as a solution to the performance problem has found a favorable consensus among computer engineers, however, such architectures may encountered some problems.

Multiprocessor systems can offer some unique capabilities and advantages compared to more conventional designs.
GRACEFUL GROWTH and GRACEFUL DEGRADATION are key capabilities that multiprocessor systems can support.

With properly designed hardware and software, it is possible to increase the processing capacity of the system merely by plugging in additional processors.
Conversely, failure of a particular processor can be made to result in merely reducing the system's throughput rather than in a crash.

Graceful growth has great appeal to end-users. With conventional systems, users who anticipate growth in demand for system capacity have basically only two option: they can either buy excess capacity initially or undergo a painful upgrade from one computer model to another later.
Either alternative is far less desirable than the ability to expand the system gracefully.

AT&T began attacking the UNIX multiprocessor issue itself when it introduced the 3B20A and 3B20D dual-processor systems to the commercial market.

To support true multiprocessing situations, however, AT&T Bell Labs' staffers recently described a set of modifications to the UNIX system that allow the multiple processors to execute kernel functions concurrently.

The basic UNIX system assumes a single-processor environment and hence is constructed under the assumption that only hardware interrupts can pre-empt kernel execution; when an interrupt

2.5.2. DUAL-IDP HARDWARE ARCHITECTURE

From the hardware point of view, the architecture defined for SGM2 dual-processor (or dual-IDP) can be considered as TIGHTLY-COUPLED with MEMORY HIERARCHY.

Dual-IDP configuration is obtained starting from a SGM2 mono configuration and adding, by connecting through the system-bus, a Main Memory sub-system, an IDP sub-system, and the cache.
In this way we have an additional complete autonomous "Interior Decor processing element with local Memory" dedicated to the execution of parallel UNIX tasks (see figure 2.5.2.a.).

Each IDP has its own local memory used to store:

* kernel code (UNIX), stack, U-block
* User code, stack, U-block

A portion of the Main Memory connected physically to the first IDP, is also used by both IDPs, as a "Global Shared Memory".
This Global Memory is used to store:

* kernel globals data, and
* shared segments

Each CPU (i.e. 68020, CMMU, CACHE and up to 12 MBytes of Memory), runs the own UNIX kernel plus a set of assigned processes.

Shared memory is only used to hold system wide data such as process tables and disk buffers.
User programs execute entirely within local memory and may not share writable memory segments.
As previously said, a copy of UNIX operating system resides in local memory and each processor is capable of concurrently executing it.

Assignment of "processes" to the "processors -IDP-" are done in a static way namely at "EXEC" time.

Due to the architecture implemented, this solution can be defined as "MULTI EXECUTIVE with COMMON GLOBAL MEMORY", whereas SHARED-MEMORY multiprocessor system provides multiple processing engines with a unique system main memory shared among all processors, that contains both user programs and a single copy of operating system.

No prevision exists to have more than two IDPs connected to the SGM2 system; in other words SGM2 system has been designed for dual-processor configuration only.

For more detailed informations related to the software aspects of Dual-IDP architecture, refer to "SGM2 software EPS".

Through the system-bus, each local-memory is completely accessible by each IDP's; the local-memory have different physical address when accessed from VME-bus (24 MBytes physical linear address space).

With this architecture, while the access of 1st IDP to the global memory costs 2 waits states, the 2nd IDP has to pay 6 to 9 wait states when accessing the data stored in the global memory. This is due to the fact that there is a system bus to travel across and potential contention with I/O devices can arise.

The IDPs synchronization on shared resources, i.e. global memory data, is achieved using "Test and Set -TAS-" instruction, provided by 68020.

even though the bus is free of traffic during memory access time.

In the "Split-bus-protocol", one processor can issue a request to memory while the data requested by another processor is being read.

A memory sub-system and cache, using the "Split-bus-protocol" must be smarter than simple memory and simple cache system.

2. Using of "interleaving" technique for main memory.
This approach allows to reduce the access time to fetch data from main memory.

3. Increase the speed of data transfer between main memory and processors by time multiplexing two or more double words on data path.

4. Increase the cache size in order to obtain an higher "hit rate" and then reduce references to main memory.

All these solutions, while allow to reach the objective to assure a good level of operability, as a counter-part they lead to a system shop cost significant higher particularly for mono-processor configuration.

With SGM2 architecture, the mono-processor solution is absolutly unpenalized in term of performances when compared against a system endowed with the features above described while giving a system shop cost significantly lower.

As far as multi-processor configuration is concerned (dual-IDP in the SGM2's case), the following considerations are offered:

* SGM2 multi-processor configuration is limited only to DUAL-IDP.

* SGM2 Dual-IDP configuration is not symmetric, since the second IDP's access to the shared memory area has an access time penalty of 6 to 9 wait states against 2 of first IDP (first IDP shared memory is physically coincident with local memory).

As a consequence, while on symmetric system, adding a processor increase throughput capability but will not affect an individual process's execution time, on asymmetric systems, it can affect the process's execution time.
According to the assignement of "Process to Processor" actually determined during the program run-time, users should be aware of the possibility of a small difference in program execution time.

On SGM2, this difference would likely only be seen on programs which may heavy use of system calls.

Let us now try to quantify the time penalty and the raw computational power that can be obtained connecting to SGM2 system a second IDP.

The following assumptions has been made:

due to the fact that the memory is busy being servicing 1st IDP (this last configured with cache option).

If we consider that the 1st IDP gives a raw computational power of 1.9 - 2 MIPS, that corresponds to an AIET = 526 - 500 nsec., we obtain:

Number of WAIT STATES to access Global Mem.	ΔT (nsec.)	ΔT/500 (%)	AIET (MIPS) of 2nd IDP
4	37	7.4	1.86
6	56	11.4	1.80
9	84	16.8	1.71

We can conclude that SGM2 Dual-IDP architecture, even in the worst case situation, can efficiently respond to the performance requirement expected to reach with a multiprocesor system.

In fact, in terms of raw computational power in the worst case situation, adding a 2nd IDP we can increase of at least 1.71 MIPS the power of the system (that is 85.5% more power than with a single IDP).

(see "Standard workload definition" chapter).

A picture of the model is shown in figure 2.5.4.a.
We suppose that a population of N active users interacts with the system through terminals.
A transaction is divided into two phases. The first phase refers to the time the user spends to prepare and to imput his message; the second, to the time spent waiting for and receiving service from the system.
These intervals are called "think plus input character time" and "response time", respectively.

We assume that the dashed box of figure 2.5.4.a. always contains a certain number of programs circulating between CPU and I/O (disk only) processors and competing for the use of these resources. This implies a heavily loaded system, so that there will be as many programs in main memory as allowed by the operating system.
No programs swapping is foressen.

During its lifetime, a transaction requires an average number of I/O operations -as defined by the workload profile-.
In other words, a program relinquish the CPU a given number of times in order to acquire (sometimes to write) data from (into) auxiliary storage.

Processing a transaction requires to wait for CPU and I/O processor availability in a "input message queue".

If the attention is focoused on the closed loop inside the dashed box of figure 2.5.4.a., it can be seen that the cycle queueing system therein corresponds to the "repairman model".
We can therefore compute the CPU utilization "U" using the formula indicated in figure 2.5.4.a.

Since we are going to study the system behaviour as a function of the number of terminals -users- connected to the system, we have to consider also the effect of Q1.
To this end, we simplify the model by observing that, because of the internal congestion, the system can be thought of as a single server (in which the server is the dashed box) with mean service time

$$C` = C / U$$

where: C = CPU service time
U = CPU utilization as computed by repairman model when applied to dashed box.

operating on input message queue Q1.

Now is possible to compute the "utilization" factor for the entire system, applying the formula of "repairman model". for the external loop.

Khintchine-Polloczek equation shows that as the utilization of a given resource approaches 1, i.e. 100%, requests` mean queue length and mean wait time grow very rapidly.
In other words, if the economic advantage of high utilization is pressed too far by encouragging greater service volume, the price of the near-unity utilization is a very long mean wait time.

The conclusion is that in order to avoid service times that increase in exponential way the server utilization must stay in between 50% to 70%.
On this premise, using the analytic model above described, it is possible to derive the number of active users supported by a SGM2 mono-IDP configuration.

Referring to diagram of figure 2.5.4.b. we can see that, according to the heavy or light transaction workload, from the point of view of calculation power, with a mono-IDP, SGM2 can support respectively 18 or 32 active users.

Since the workload profile used to load SGM2 requires also I/O disk activity, we have to include in this study also the analysis of disk sub-system.

As before mentioned, for SGM2 in mono-IDP configuration, we intend to offer the disk-controller.
Let us assume to have as a disk device, Wren II units connected to the system.
The "average access time figure" for this kind of device, can be assumed to be about 50 milli second.
This figure derives from the following assumptions:

* 35.0	milli sec.	<average seek time>	+
* 8.8	milli sec.	<average disk latency>	+
* 2.0	milli sec.	<1kB of char. transfer>	+
45.8	milli sec.		=

that makes a roud figure of 50 milli sec. and corresponds to a theoric figure of 20 I-O OPERATIONS per SECOND per DEVICE with a controller/device 100% loaded.

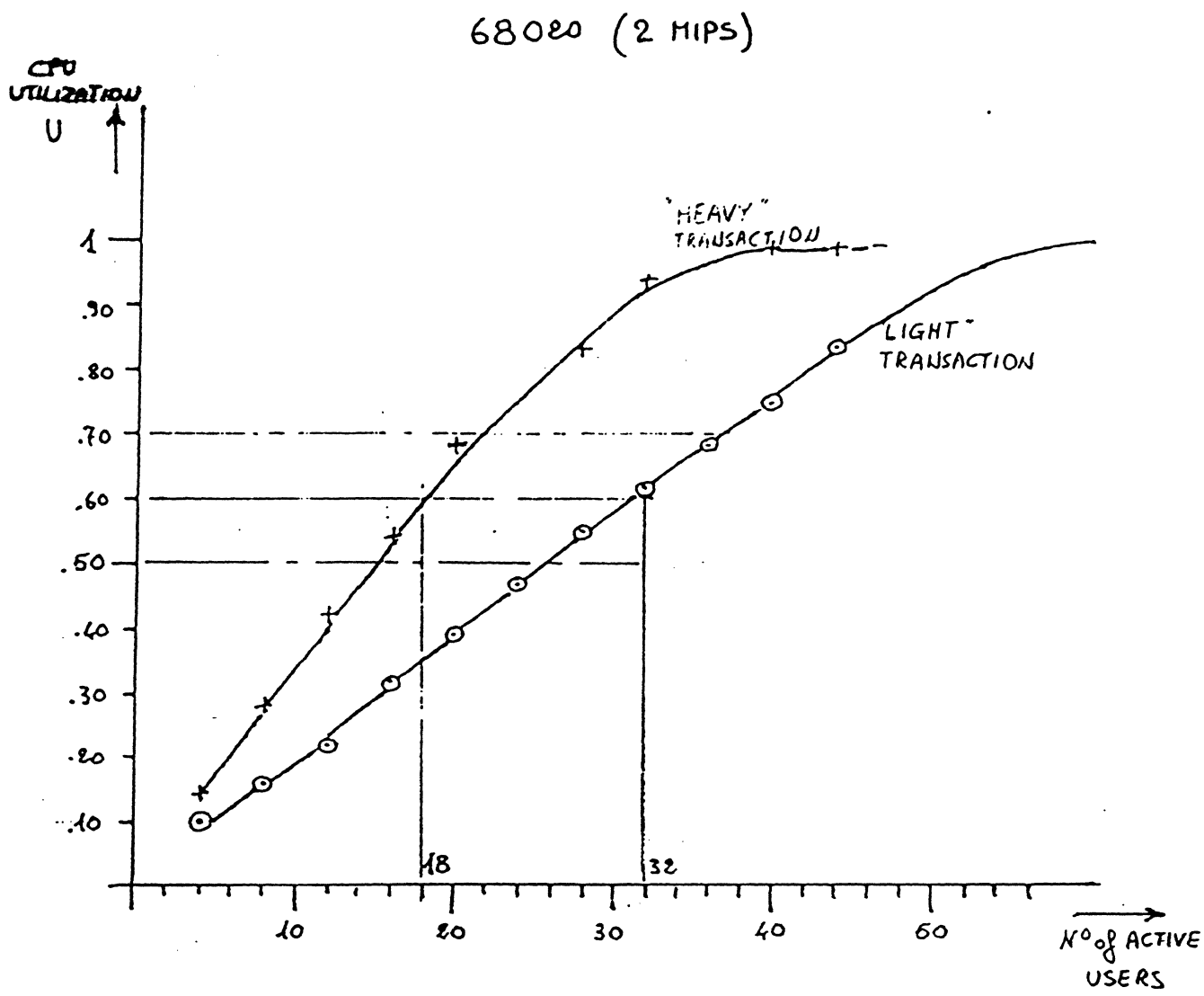
Also for this case, if we would not experience an unacceptable increase of the service time, the server -disk controller plus device in this case- must be loaded no more than 60%; in terms of a range we can say in between 40% to 60%.

Considering 50% of loading and applying a Khintchine-Polloczek formula, we can derive the service time for disk I/O operation.
According to the diagram depicted in figure 2.5.4.c. we can assume as a service time for one disk I/O operation, 100 milli sec.
This figure has been used in repairman model and was indicated as "i" (service time).

As far as the number of I/O operations supported by disk controller is concerned, since it is not possible to assume a uniform distribution of the file among the dispacks when more than one disk device per system is present, we can assume the following:

1 disk drive	10 I-O/sec	
2 disk drive:	10+8 = 18 I-O/sec	(corrective factor .8)
3 disk drive:	10+8+6 = 24 I-O/sec	(corrective factor .6)

*the corrective factore has been used to account for a non


68020 (2 MIPS)
CPU UTILIZATION
U
1
.90
.80
.70
.60
.50
.40
.30
.20
.10
"HEAVY" TRANSACTION
"LIGHT" TRANSACTION
18
32
10
20
30
40
60
Nº of ACTIVE USERS

- Figure 2.5.4.c.-

3. MANUFACTURING AUTOMATION PROTOCOL (MAP)

3.1. INTRODUCTION

World wide competition is exerting increasing pressure on the automotive industry in its own basic business - providing transportation products.

To meet these challenges, General Motors Corp. has stepped up the pace of a number of programs aimed at continuous improvement in the cost and quality of manufactured products.

In this process, large numbers of robots, programmable controllers and other programmable devices have been introduced into the manufacturing process, and statistical process control tools are being used to monitor "as built" quality.

Accompanying this growth in the number of programmable devices has been a growth in GM's requirements in other areas.
In addition to an increased need for computer support for programming, documentation and backup, the need for sophisticated communication has been autstanding.

The need for communication is considerable. The communication system must be able to accomodate voice, video and high speed data.
The communications network therefore becomes an interesting force for factory operations.

On these premises, GM has defined the MAP protocol with the goal that it become an open national standard with widespread use.
In other words, GM's objective for MAP are to define factory networking standards that support application - to - application communication, to identify application functions to be supported by the standards, and to recommend protocols that meet the functional requirements.

The GM initiative has been inspired by the need to build multi-vendor manufacturing systems, without incompatibility problems that would otherwise create "island of automation" on the factory floor.

The standards are aimed at defining an accepted communication method for a multivendor environment.
Recently Ford has also planned to join the multitude backing MAP.

As a result of this, hundreds of companies, large and small, have already agreed to meet GM's specifications for its Manufacturig Automation Protocol.
Those that do not agree, will not see their products used in GM's factory automation systems.

Specified as IEEE 802.4 token bus broadband (amplitude modulation).

MAP currently specifies broadband coaxial cable for the physical media in factory floor local-area network.

GM believes broadband is the best choice for the reasons explained in the follow.

First, broadband allows multiple networks to exist simultaneously.
In present configurations at GM and elsewhere that still permitt the use of proprietary networks, a broadband system can support as many as 100 such networks (depending of the bandwidth of each).

Second, broadband will support both the high speed data requirements of many factory LANs and the voice and video requirements needed for security, teleconferencing, and training.

Third, broadband is part of the IEEE 802.4 (token bus) standard of communications and is under study for 802.2.
It is also the foundation for the IEEE metropolitan area studies group.

Fourth, and possibly the most important, GM has a sizable investment in the broadband coaxial cable systems installed in existing plants many years ago.

Fifth, in a real time environment such as manufacturing which must be designed knowing the worst case delays for message delivery, it is unacceptable to have the degradation that occurs on networks based upon other protocol, e.g. Ethernet, when there is an increase on the traffic.

Additional and detailed information about MAP can be obtained from:

T.H. WATSON
SYSTEMS ENGINEER
CREATIVE CONTROL SYSTEMS
4115 CLUBVIEW Dr.
FORT WAYNE, IN 46804

Requests for copies of a publication detailing MAP specification, VERSION 2.1, should be submitted to:

MANUFACTURING ENGINEERING and DEVELOPMENT
GENERAL MOTOR TECHNICAL CENTER
MANUFACTURING BUILDING, A/MD - 39
30300 MOUND Rd.
WARREN, MI 48 090 - 9040 attn.: MAP CHAIRMAN

It is suggested to investigate as soon as possible on these alternatives in order to be ready to support on SGM2, the MAP offering.

the benchmark programs for each machine and the premise of benchmarking computers using exactly the same programs, become a myth.

4.2. STANDARD WORKLOAD DEFINITION

The process of applying a known workload to a computer under controlled conditions, is known as BENCHMARKING.

"Benchmarking" is a term that has been used in computer system evaluation to describe a number of different functions.
In its broadest sense, the term has been used to describe the entire process of preparing and executing a mix of representative programs on a target computer system (or on a test vehicle) in order to validate (or predict) system performance.

In a more limited sense, the term benchmarking has been used to describe any activity which involves using test workloads or "benchmarks" to study computer system performance.

This process includes a quantitative description of the user imposed real workload, and the construction and validation of the REPRESENTATIVE TEST WORKLOAD.

Benchmark tests imply a comparison, which may be among multiple computers or may be against the "same" computer under alternative hardware or software configurations.

The term "representativeness" has been traditionally employed when referring to workloads used in performance evaluation studies.
If we agree that, independent of their type, these workloads are in fact workload models, we should use the term "accuracy" instead.

When is a workload model "representative" (or "an accurate reppresentation") of a given workload? Probably the most popular answer to this question is: when the model consumes the same physical resources at the same rates as the workload it tries to represent.
The closer the test workload is to the real workload, the better the selection decision.

Thus , it is evident that great care must be taken in the generation of a test workload, and that the degree of accuracy achieved must be explicity evaluated.

Since the methodology to follow for the correct workload definition is outside the purpose of this document, it is suggested to refer at specific technical literature.
In particular it is suggested to deserve particular care about the computer evaluation environment in which test workloads are typically used and correct methods for sound test workload generation in each of these environment (including standard "Data Base" usage).

Since at this point in time (September 1985) the activity for the definition of "HISI STANDARD WORKLOAD" for SGM family of computers, i.e. based upon the use of UNIX operating system, is yet under definition, for SGM2 performance prediction activity we have started to use published data coming from HISI external source.

Since SGM2 system is based upon the use of Motorola microprocessor i.e. 68020 MPU, we have used data published by Mr. Doug MacGregor, a designer for Motorola's MOS Microprocessor Design Group in Austin (Te), during February 1985.

In this publication, D. MacGregor writes:

"..... one of the most popular applications of the 68000 family has been as a general processor in UNIX-based supermicrocomputers and engineering workstation.
Although such systems can differ in many ways -by supporting a single user or several, having more than one processor, or running radically different programs- there is a common factor: they all typically execute high level structured code.

That common thread makes it possible to predict how the 68020 will perform on an equivalent system.....

The task typically performed by high level code can be described in terms of the individual activities that take place (see Figure 4.2.a.).

To evaluate the 68020's strength in performing each of these activities, about 20 programs were assembled or compiled into processor instructions. Then, each category of instructions was traced for its dynamic frequency of occurence and the types of addressing modes used.

For completeness, the programs covered applications, benchmarks, utilities, and development software and together constituted over 500 million instructions.....

For tracking the occurrence of different addressing modes, the effective source and destination addresses of instructions that move operands were traced.

Studying the length of branch displacement (which are positive or negative and measured in bytes) is a good way to measure the cache's effectiveness in a program.
A statistical study of branch displacements for workstation software indicates that most branch destinations are within 32 bytes of the branch instruction.
Because the average length of an instruction is 1.66 words, branch destinations are usually within 10 instructions of the branch occurrence.

This distance of 10 instructions places the branch instructions well within the on 68020 chip cache's capacity of 64 long words.
In other words, these commonly executed instructions make good use of the on chip cache.....".

machine instructions

* Disk block size = 1024 bytes

4.3. MACHINE INSTRUCTION FREQUENCIES (INSTRUCTION MIX)

Analysis of an instruction set as large and varied as the one specified for 32 bit microprocessor, e.g. Motorola 68020, is important for aiding processor design evaluation.

Many modern computer architectures supply instructions geared to specific applications as well as general purpose software. For example, instructions can be included for scientific computation, transaction processing, commercial data processing, or operating system software.

It must be the goal of a computer designer to obtain statistical data related to the "INSTRUCTION SET" available on a given MPU, in order to be able to properly evaluate the effectiveness of a system from the architectural point of view.

These statistical data must be collected using "STANDARD WORKLOAD" that is the representative set of jobs that as an average should be run on the system.

Statistical data to be collected with this activirty, should be, among the other, the following:

* OP CODE frequency and time distribution

* memory references per instruction (specified for read, write, read-modify-write operations)

* ratio between master-slave instruction frequency

* instructions length

* branch op code analysis; execution distances for branch

* op code pairs (pairwise instruction frequencies)

* address calculation

* MPU's internal registers usage

* operands length

* character and decimal instructions

* control transfer (contest switching)

4.4. MILLION of INSTRUCTIONS PER SECOND (MIPS)

1. 68020 ~~working~~ with 16,67 MHz clock rate (60 nsec clock period)

2. 68020 on chip instruction cache hit rate = 15%

3. Average execution time figure for each 68020 instruction, has been used (best case timing, not considered)

	R-W Memory with 0 Wait States	Cache with 80% H.R.	R-W Memory with 2 Wait States
AIET 68020	448-434 nsec	515-500 nsec	760-714 nsec
MIPS	2.2 - 2.3	1.9 - 2	1.3 - 1.4

Notes

1. For cache access, 0 Wait States is required;
   Write through algorithm has been adopted.

2. 300 nsec. for main memory cycle time has been assumed. This implies that 2 wait states are required when read/write operations with main memory are performed.
   It is assumed that this cycle time is applicable for whichever kind of sequences of memory operations are performed.

4.5. MEMORY MANAGEMENT UNIT (MMU)

A good multiuser system must provide a sophisticated computing environment and support large virtual and physical memories. The Operating System is responsable for insuring the proper execution of user task in the system environment and memory management is basic to this responsability.

The hardware support to Operating System for memory menagement, is achieved through the use of specialized "logical network" referred as MMU.

Some microprocessors' families had no MMU on chip (in other words, the MMU unit attached physically to the processor board), a design carried forward by instance in Motorola and National MPUs.

These processor families require an off-chip MMU in order to

execution time overhead on the user program.

This measurement session can be executed on "SGM2 Test Vehicle", i.e. LISE System, as soon as the UNIX port (now in progress, -Sept. 1985-) will be available.
Since for this test vehicle, will be also available a PWA with Motorola MMB, the comparison of the performances offered by Motorola PMMU (i.e. 68851) vs. Custom MMU, must be evaluated.

During the evaluation of custom MMU, it is suggested also to provide specific tests able to measure the effect on the global performances at the system level, having different memory "page size" and different number of context simultaneously storable on the MMU bloch.

plugged on a connector and residing on System Controller PWB.
The program stored on PEB is SGM" system configuration dependent and allows to have a great flexibility on initialization sequence.

* the execution of the PEB's program, causes the starting of active initialization phase.
From a general point of view, the software stored on the PEB will provide the INITIALIZING DEVICE selection.
This is obtained by an automatic scanning of magnetic devices, i.e. floppy, 1st. disk, 2nd. disk, etc. until the media with "SOFTWARE BOOTLOAD" is encauntered.
At this time the SOFTWARE BOOTLOAD is loaded into main memory and executed.
The execution of this program will provide for the loading of the software local resident to controllers/processor and finally for UNIX loading and launching.

to the TEST VEHICLE a custom MMU (for example a SUN like MMU) in order to obtain some "performance indicator".

Since SGM2 "TEST VEHICLE" constitutes the most viable way to obtain as soon as possible data useful to direct SGM2 design choices, it is strongly raccomanded to produce one more system to be assigned for these specific activities (in charge of care to System Engineering Group).

The system configuration of "TEST VEHICLE" that could satisfy performance measurement requirement, is the following:

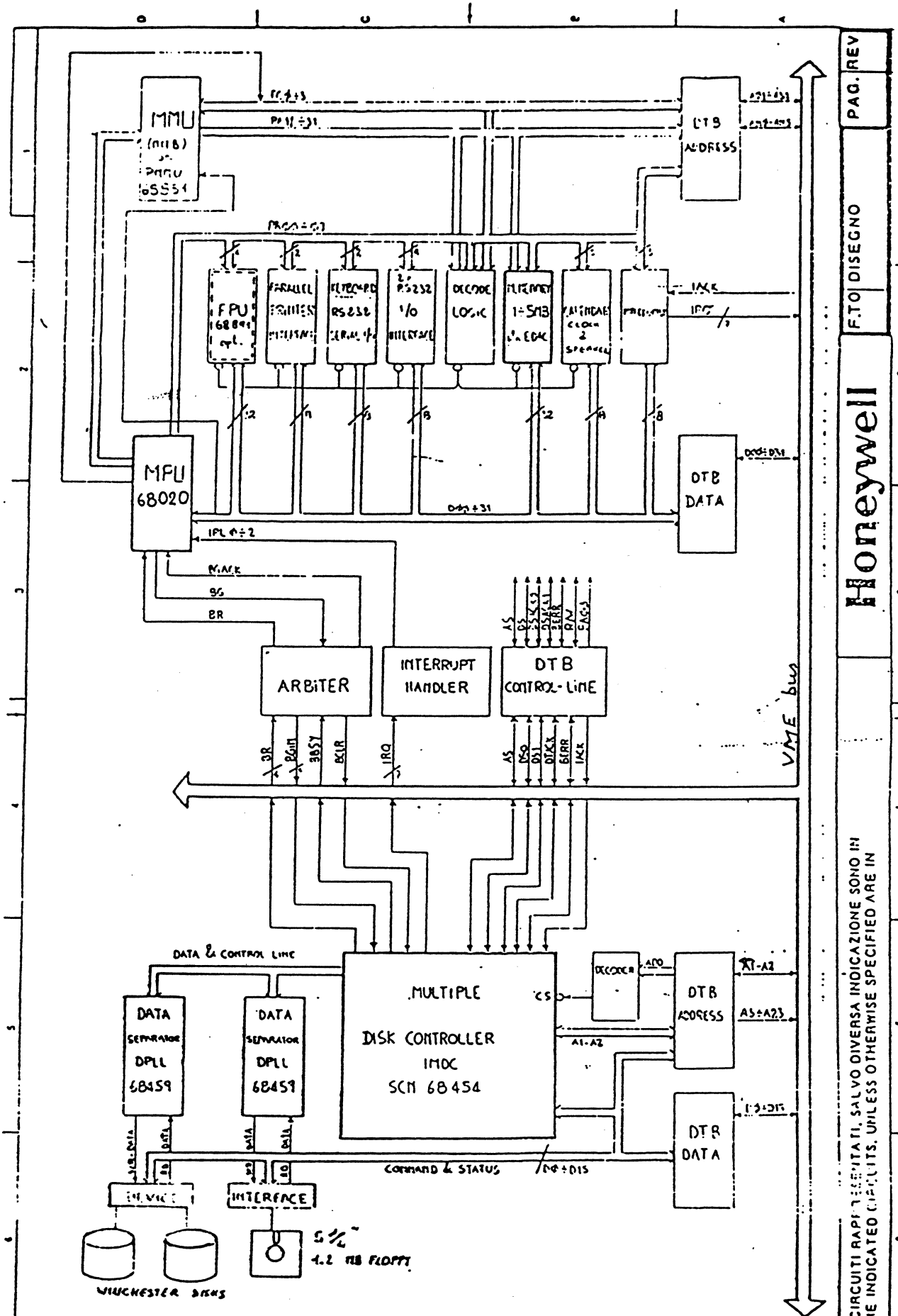
* basic mainframe with at least 2 MByte of memory

* 1 floppy disk (5 1/4"), AX Superteam compatible.

* 2 winchester disks -Wren I or II-

* one terminal to be used as console

The availability of "VME adapter board" has to be considered a nice to have expansion, that would allow to connect to "TEST VEHICLE", standard controllers.



"LISE" major block diagram PRELIMINARY Dec. 1984

Honeywell

F.TO DISEGNO

PAG.

REV

"CONFIDENTIAL & PROPRIETARY"

I CIRCUITI RAPPRESENTATI, SALVO DIVERSA INDICAZIONE SONO IN
THE INDICATED CIRCUITS, UNLESS OTHERWISE SPECIFIED ARE IN