INTERNATIONAL ATOMIC ENERGY AGENCY
UNITED NATIONS EDUCATIONAL SCIENTIFIC AND CULTURAL ORGANIZATION
INTERNATIONAL CENTRE FOR THEORETICAL PHYSICS
P.O.B. 586, 34100 TRIESTE, ITALY, CABLE CENTRATOM TRIESTE

H4.SMR. 405/10

SECOND WORKSHOP ON TELEMATICS

6 - 24 November 1989

Digital Signal Processing

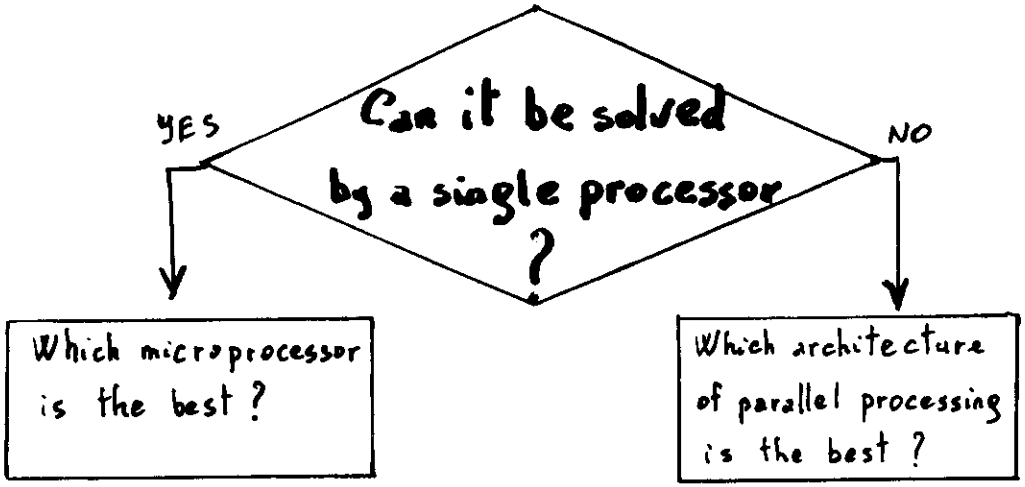
D. CROSETTO
CERN, SPS Division, Geneva, Switzeralnd

DIGITAL SIGNAL PROCESSING (DSP)

SUMMARY

- Parallel Processing
- μPs Comparison
- DSP evolution
- DSP Application Area
- General Purpose DSPs
- Special DSPs
- DSP Applications Examples

PROBLEM DEFINITION



- MICROCONTROLLER
- CISC
- RISC
- CRISP
- DSP
- TRANSPUTER
- BIT SLICE
- DATA FLOW
- SYSTOLIC

- CONCEPTS
- PARALLEL COMPUTER STRUCTURE
- METHODS OF CLASSIFICATION
- INTERCONNECTION SYSTEM

CONCEPTS:

Parallel processing is concurrent execution of multiple functions.

Concurrency implies concepts of:

- Parallelism : parallel events may occur in multiple resources
- Simultaneity: events in different pieces of hardware at the same time
- Pipelining: overlap of independent portion of multiple instructions

COMPUTER STRUCTURES

Parallel computer systems may be divided into three architectural configurations.

Pipeline computers

Perform overlapped computations to exploit temporal parallelism

Array processors

Uses multiple, synchronized aritmetic logic unit to achive spatial parallelism

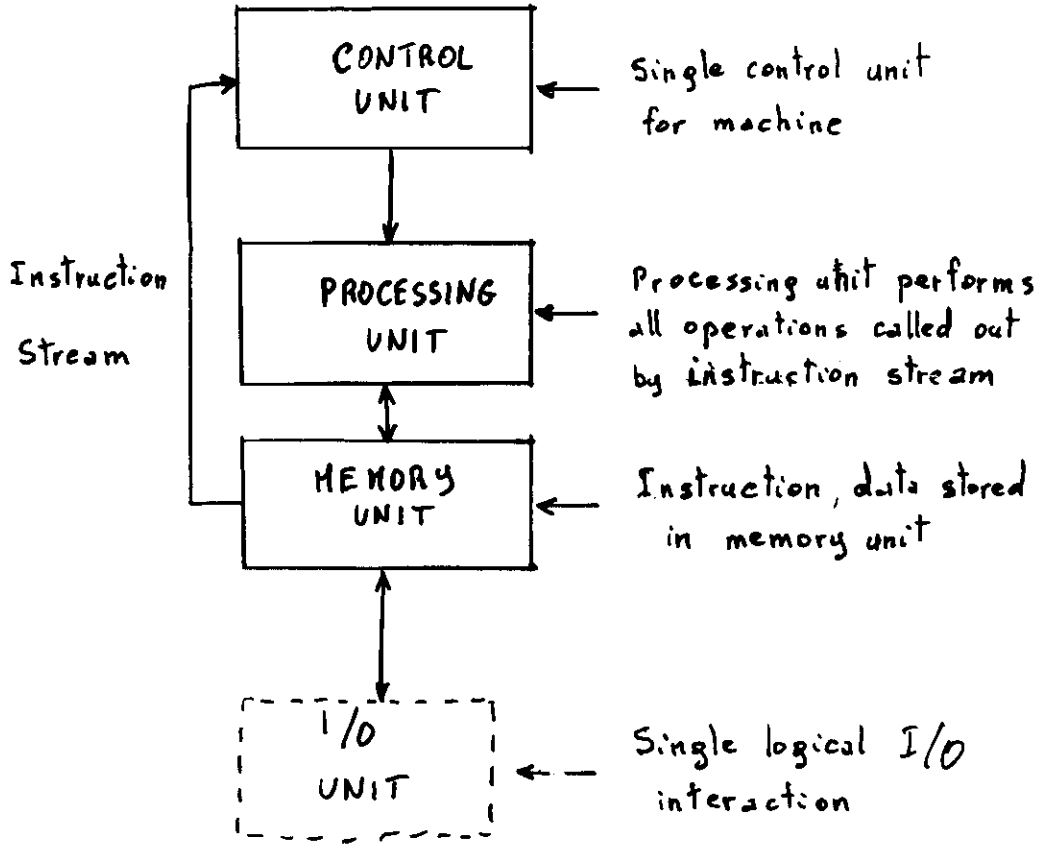
Multiprocessor system

Achive asynchronous parallelism through a set

METHODS OF CLASSIFICATION

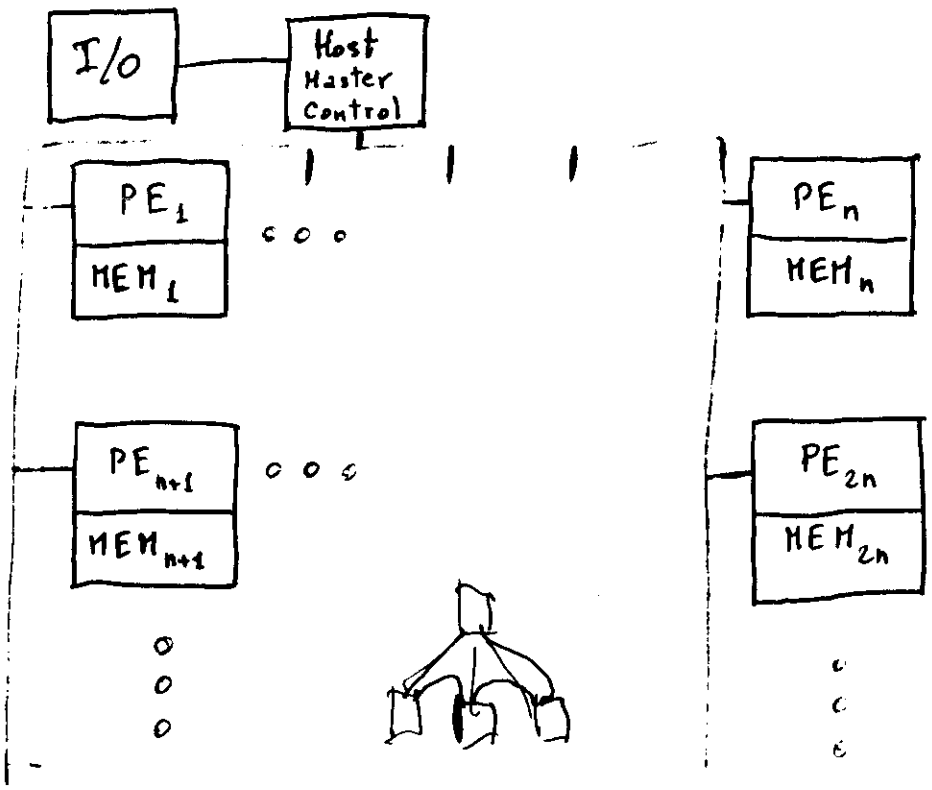
- Performance metrics
- Technological contributions
- Architectural methods
- Generation of computers (1-5)
- Architectural descriptions
  - SISD: Single Instruction Stream, Single Data Stream
  - SIMD: Single Instruction Stream, Multiple Data Stream
  - MIMD: Multiple Instruction Stream, Multiple Data Stream
  - MISD: Multiple Instruction Stream, Single Data Stream
- Interconnection techniques

BASIC "SISD" ARCHITECTURE



- One stream of instructions, one stream of data
- Most "normal" machines including pipelined implementations
- Single control unit

BASIC "SIMD" ARCHITECTURE



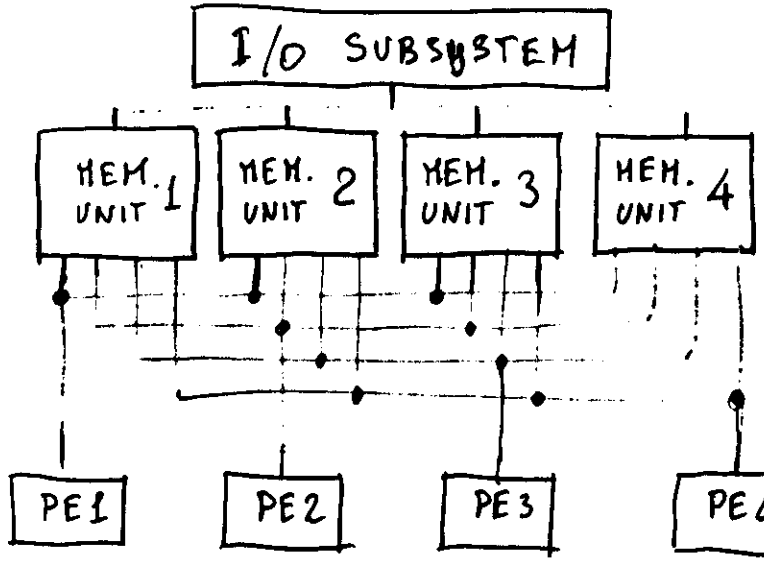
PE = Processing Element

All "PE" Execute Instructions on data in own memory

- Early attempt at parallel processing - ARRAY PROCESSOR
- One instruction stream for all operations
- Multiple data streams one for each PE
- Serial portion of instruction stream executed only by master

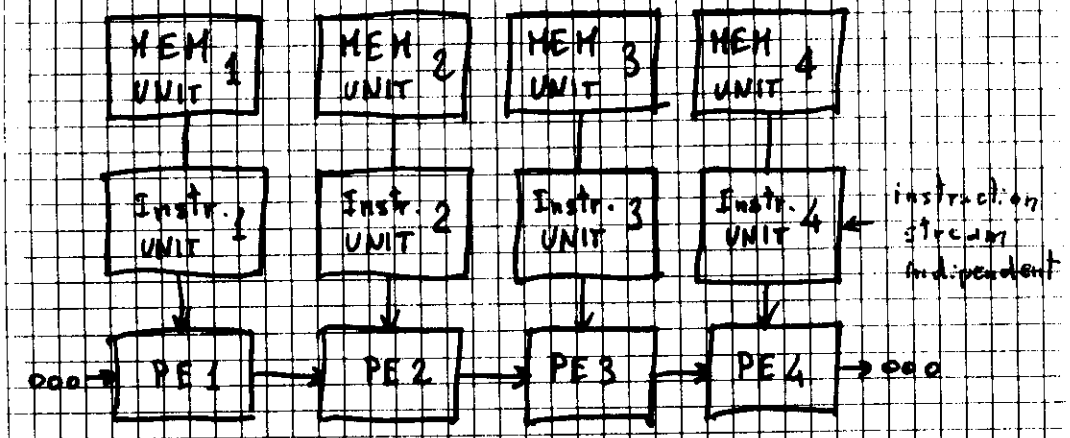
BASIC "MIMD" ARCHITECTURE

(Multiport)



- Instruction partitioned into independent streams
- Each instruction stream has own data stream
- Model fit multiprocessor, multicomputer organizations
- Memory can be shared or independent
- Tightly coupled (special purpose) and loosely coupled (general purpose machines)
- Interconnection methods
  - Multiport memory
  - Regular conn. system
  - Routing schemes
  - Cross bar switch

HYPOTHETICAL "MISD" ARCHITECTURE



Each PE operates on data and passes it on

- Multiple sets of instructions to execute
- Single stream of data
- No real processors of this type known

INTERCONNECTION SYSTEMS

Switching systems: multiple simultaneous transfers

- BUS system
- Crossbar switch
- Omega network (Multiport)

Shared resources: OS driven

- Shared memory

Regular, structured interconnection: message passing sys.

- Hypercube structures
- Tree connected machines
- 2-D grid

Assuming to have a problem to solve that requires more computing power and I/O then any μP or μC existing on the market.

Which Parallel Processing Architecture do we choose ?

Is there the possibility to design an architecture flexible enough that can solve efficiently all types of applications ! (I/O, matrix calculation, Filters, Real-time, vector proc., many data - little process, a lot of process on few data, programmable interconnection configurable as three, matrix, etc. tigtly or loosely coupled...)

If such a super computer that will solve all types of problems will ever be build the ratio cost/performance will certainly be very high and for some specific applications the efficiency and the facility to use it would not be the best

- Start from the Problem Definition
- Define the best Architecture
- Select the tools (~~Opera~~ BUS, Operating System, Languages, ~~Debug~~)
- Select the best and (more suitable to the application) most advanced technology (μP, ASIC, VLSI, μC, etc)

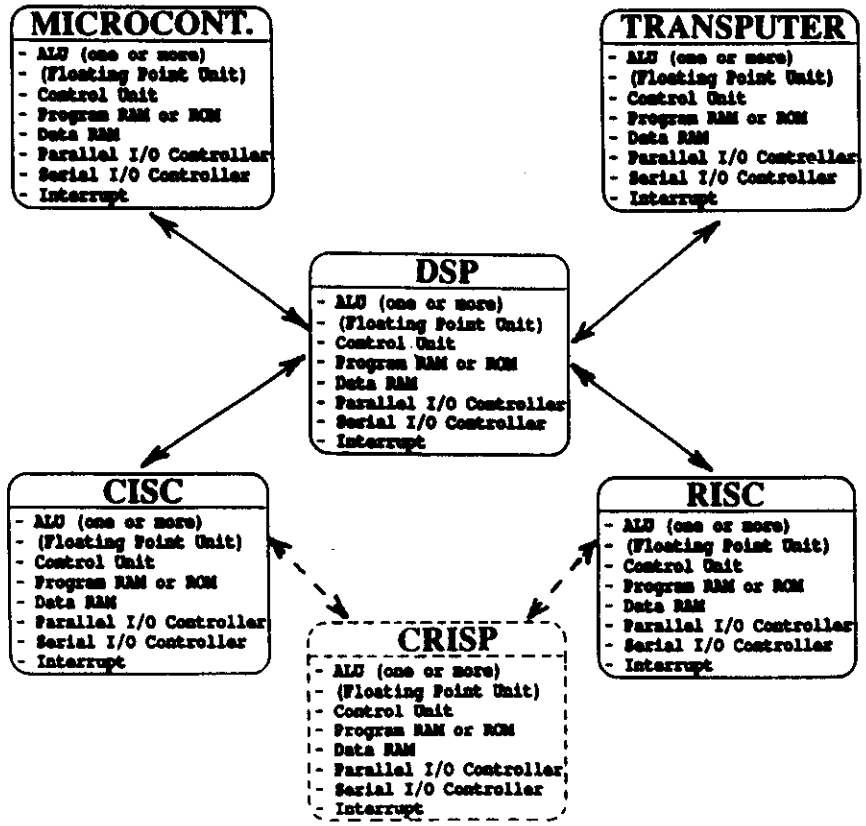
After performance comparison with other components it will become clearer why the DSP is more suitable for several types of applications.

The basic elements of a real-time processor are:

- ALU (one or more for Addr. and Data)
- (Floating Point Unit)
- Control Unit
- Program RAM or ROM
- Data RAM
- Parallel I/O Controller
- Serial I/O Controller
- Interrupt

There are several ways to realize a concurrent system of many basic element described above, each one having a different throughput, privileging in one case one aspect respect to another

Differences on performance respect to a basic system that make use of DSP



MICROCONTROLLERS < > DSP

- **A "microcontroller" contains all the necessary components of a complete system on one piece of silicon (E.g. Intel 8051, Motorola MC6804, MC6805, MC68HC11, etc.)**

- Less performance than a DSP

- 4, 8, 16-bit

- instruction set more like CISC processor (using more then one cycle per instruction)

- some extra programmable peripherals on chip, like A/D converters are not available on DSP.

- Is not designed to build concurrent systems but for economical applications in embedded systems where is necessary only to have the capability of one of the most common 8-bit or 16-bit microprocessor instruction sets.

Applications:

- industrial control
- device controller (printers, plotters, etc.)
- in an array of front end processors in a High Energy Physics Experiment for slow calculations

- DSP is replacing to this component in the most sophisticated applications where speed is an important factor.

TRANSPUTER < > DSP

- A Transputer contains in a single chip:
  - an integer processor
  - a Floating Point Unit
  - 4 Kbyte of memory
  - 4 high speed serial links (20 Mbit/sec)

- Transputer is designed as a programmable component to implement a system with much higher degree of concurrency then is currently common.

- The Transputer, together with the formal rules of Occam, provides the design methodology for this family of concurrent systems.

- Special instructions divide the processor time between the concurrent processes and perform interprocess communication

- In addition the transputer is designed so that its standard behavior corresponds to the formal model of a process. As a consequence it is possible to program systems containing multiple interconnected transputers in which each transputer implements a set of processes.

- Since a program is defined as a set of processes, it can be mapped onto such a system in a variety of ways, for example to minimize cost or to optimize throughput, or to maximize the responsiveness to specific events.

- The architecture should give the possibility to span the range of application from microcontrollers to supercomputers

ADVANTAGES AND DISADVANTAGES:

- It is easier to build concurrent systems because of the good coordination between hardware and software (Occam)

- Easier to transport software on different concurrent systems with different number of transputers.

- Good concurrency, and good flexibility, but the throughput respect to another architecture that makes use of DSP for a more specialized application has to be verified.

- The time required for a multiplication is 500 nsec average for T800 (~ 2 usec for a T414). Most DSP's do it in one cycle (75 to 200 nsec), the same is true for the division and for the floating point operations.

- The performance of a Transputer begins to drop noticeably as soon as the on-chip memory is too small to hold all the frequently accessed data. And its premise that the world is process-shaped rather than procedure-shaped may well be true, but the majority of available software doesn't reflect that belief.

- on the contrary the DSP don't have special signals or instructions foreseen to implement a concurrent system with message passing.

RISC < > DSP

- Deeper investigation is merited by the RISC architecture because it is the most innovative and is based on concepts that are attractive for applications in: workstations, superminis and also as embedded controllers.

- From the initial simple concepts of a register-intensive cpu design from Seymour Cray in 1960 to the modern notion of RISC architectures emerged from John Cocke's project at IBM in 1970.

- Cocke's team goal was to design the best CPU architecture for an optimizing compiler

- the machine should be register-to-register with only load and store accessing the memory.

- the architecture eliminated microcode and microsequencers in favor of simple, hardwired, pipelined, one-instruction-per cycle CPU design.

- RISC technology created an almost insatiable demand for memory speed. The answer to the problem come with high performance memory hierarchy, including general purpose registers and cache memories.

- the instruction set is regular and simple with few addressing modes: indexed and PC-relative.

There are then some RISC variations from these common theme.

- IBM 1975 with 801 minicomputer

- BERKELEY 1980 with RISC I and RISC II

- STANFORD 1981 with MIPS (Microprocessor Without Interlocked Pipeline Stages)

- IBM and Stanford pushed the state of art in Compiler Technology to maximize the use of registers.

  - the key idea is to expose in the instruction set all the processor activity that could effect performance. This philosophy, coupled with the concept of a streamlined instruction set, allows a shift of functions from hardware to software.

  - Hennessy's team at Stanford recognized that with a clever compiler, interlocking the pipeline wasn't necessary. The compiler simply had to make sure that the instruction directly after the LOAD didn't use the new data.

- The BERKELEY team did not include compiler experts, so a hardware solution was implemented to keep operand in registers.

  - to optimize the task switching time they have defined many sets or windows of registers (global and local) so that registers would not have to be saved on every procedure call. The disadvantage of register windows is that they use more chip area.

- As Compiler Technology improves and sufficiently fast processors become available, there should be decreasing necessity to program in assembly language.

Each vendor has had to improve some characteristics to translate the University design into workstations products.

- SUN Microsystems adopted some ideas of Patterson's work at Berkeley and designed a system that should be portable between implementation technologies (CMOS, ECL, etc.) calling it SPARC (Scalable Processor Architecture).

  Customers are looking at SPARC as an instruction-set definition, with many implementations available.

- MIPS Computer System turned Stanford team's effort into a product. They retained the Stanford design's delayed load and branches, and focused on a single high-speed implementation rather than scalability.

- The MIPS designers felt strongly that the key to performance was the ability of the compiler to manage CPU pipeline during floating point as well as integer operations. The floating point unit must understand the state of the integer unit's pipeline at all times (R3000, R3010)

Other RISC's vendors:

-ACORN VL86C010
tech. VLSI Technology - Sanyo
- INTEL: 80960 (for embedded applications)
- AMD: 29000 Family
- HARRIS: TRX2000 (high integrated FORTH executing microcontroller

Univ:	BERKLEY	STANFORD
Vendor:	SUN	MIPS Comp. Sys.
Type:	SPARC	MIPS (Microp. Without Int. Pip.)
-		

Second Sources

- FUJITSU 1.3um 25MHz
- Bipolar Integrated Technology
- Cypress .8um 33MHz
- LSI Logic .9um 25 MHz
- Performance Semicond. .8um 25MHz
- Device Technology

Competitors recognize as long-term microprocessors innovators:

MOTOROLA, INTEL, AMD

- The Fairchild team did not attempt to reimplement either research chip.

The Fairchild Clipper now available from Intergraph Advanced Processor Division, was the first microprocessor design to recognize the growing memory bandwidth.

Their solution was to separate the relentless demand of instruction fetches Load/Store activity by providing separate instruction and data buses.

The Clipper supports the dual busses with a pair of integrated caches and memory management chips.

- The MC88000 from Motorola appears to be a blend of the purity of MIPS' CPU concepts and the innovations of Clipper's bus architecture.

It follows the dogma of simple, one-cycle, fixed-length instructions and load/store architecture.

Like Clipper, the MC88000 is a dual bus, three-chip layout (HCMOS). All the execution units work over the same two source buses and a single destination bus.

The most important characteristics of the MC88000 may be the system's ability to incorporate new, specialized execution units. That starts to make room for some really interesting special purpose units, like vector processors or graphic engines.

- MIPS, Intergraph and Motorola all have taken different architectural approaches to a common goal:

A FAST UNIX WORKSTATION

- Chip vendors, with characteristic optimism, are already discussing coprocessors for signal processing, message handling and graphics.

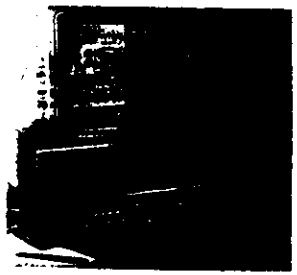
This time the breakthrough could come in multiprocessing, and again it could be led by software.

Just as RISC Technology will let Compilers make CPU pipeline more efficient, perhaps a new technology will let compilers make a cluster of CPU more efficient.

FIRST 88000-BASED UNIX-WORKSTATION APPEARS

The first Unix workstation based on the Motorola 88000 RISC processor has arrived from Opus Systems. The Personal Mainframe Series 8000 is a 17-MIPS machine featuring a dual-processor architecture that pairs the RISC chip with an 80386 I/O subsystem.

The machine brings mainframe technology to the workstation environment, thanks to its I/O subsystem designed by Everex Systems. The subsystem is a dedicated I/O processor capable of simultaneously running Unix and MS-DOS. Opus Systems' 88000-based CPU board handles all system functions.



An important benefit of using the 88000 processor is full binary compatibility with other 88000 Unix systems and products. Motorola's Binary Compatibility Standard provides a low-level specification for interaction with the Unix kernel for all applications running 88000-based Unix. Many vendors, including Opus Systems, have agreed to support this standard. As a result, applications running on the Opus machine will be assured of running on any BCS-compliant 88000-based Unix system, and vice-versa.

The company's port of Unix includes all commands, utilities, and other programs that are part of the standard AT & T release, as well as the portable C compiler and an ANSI-standard Fortran 77.

The system, which is the first product of a strategic alliance between Opus Systems and Everex Systems, will be marketed separately by the two companies. Prices for Opus' Personal Mainframe Series 8000 machine start at $9995, and shipments begin during the second quarter. Everex Systems has yet to announce its pricing structure for its machine.

Opus Systems, 20863 Stevens Creek, Building 400, Cupertino, CA 95014; (408) 446-2110. CIRCLE 323

Everex Systems Inc., 48431 Milmont Dr., Fremont, CA 94538; (415) 498-1111. CIRCLE 324

CISC < > DSP

- CISC (Complex Instruction Set Computer) architecture use a large amount of hardware complexity to provide high degree of instruction set capability.

  - They are characterized by a large instruction set with some very complex instructions.

  - The length and execution time of instruction is different from one another. Instructions can manipulate bit, byte, word and long word

  - The dynamic bus interface allows for simple, highly efficient access to devices of different data bus width.

  - The latest components of this technology support, directly via BUS Monitoring, Multimaster and Multiprocessor applications.

Advantages and Disadvantages

- Some instructions needs more then 50 cycles to be executed. However it does have control lines to support a multiprocessing environment and is connectable to different bus width devices. On the other hand, DSP executes most instructions in one cycle.

- Benchmark analysis of CISC, RISC, DSP and Transputer instruction sets measured in MIPS cannot be used to make a meaningful comparison of performance, particularly between machines with different architectures.

  Each MIPS unit should be multiplied by a normalizing factor that reflects overall system performance.

  The real comparison between one component to the other in a particular task, is the time required to execute that particular task.

- RISC and CISC may become more alike in the future. RISC is a technology, a philosophy of design, not a product.

  Some design techniques that have been applied to RISC machine can be applied to CISC architecture to improve performance.

- An example is the National Semiconductor 32532 general purpose processor which incorporates many RISC features. It has:

  - on-chip data and instruction caches
  - direct-mapped caches for stack access
  - pipelining and branch-prediction logic
  - it uses microcode (not used in RISC) for only the most complex instructions and hardwired logic elsewhere.

- Processors like the 32532 with Intel 80486 and Motorola 68040 (expected next year), incorporate more RISC-like features to push the number of cycles for most of the instructions below 2.

  These new features will probably characterize the new type of processor as:

CRISP

(Complexity-Reduced Instruction Set Processor)

to think how one component will fit better than another.

- An investigation must certainly include the component instruction set, speed and internal architecture that is best suitable for the application algorithms or for the needs of the more general project.

- More decisive in determining the overall throughput in a project, can be the harmonious interaction and intercommunication of the various components rather than the speed of any one single component.

- Thus the real effort in designing a specific application or project should be based on defining

"THE MULTIPROCESSING ARCHITECTURE"

for the best performance solution to the application or project.

Looking forward to a long-term solution

MULTIPROCESSING
LED BY SOFTWARE

Just as RISC technology will lead Compilers make CPU pipeline more efficient, perhaps a new technology will let Compilers make a cluster of CPU more efficient

Technology (DSP, RISC and CRISP) are improving very rapidely, faster than software.

Should we then look at the

SOFTWARE

that is leading the

HARDWARE

- With the advent of Fast A/D converters and new processors oriented towards signal processing (DSP), arose the tendency to treat analog signals in digital form, thus using discrete algorithms instead of analog functions. The advantage of the digital circuitry over the analog components is: high density, precision, programmability, stability and testability.

- Digital Signal Processor are special purpose microprocessors optimized for the processing of digitalized analog signals, which are discrete in both amplitude and time

- In 1982 there appear on the market a new externally programmable DSP family of processors, the TMS320XX of Texas Instruments.

- 1st DSP 1978 AMI S2811

- 1979 INTEL 2920/21 (Telecommunication)

- 1980 NEC 7720

- 1982 Texas 32010 First with the option of having the program memory on RAM. Ideal for low volume applications.

- The typical specifications that was distinguished a DSP with respect to other microprocessors, has changed since 1982. As a consequence, also the field of applications is increasing.

- **At the beginning most DSP's had in common the characteristics of:**

  - Harward architecture (separation between Program and Data memories)

  - very small Program and Data memory area

  - small instruction set and most of them executed in one cycle (for this reasons similar to RISC)

  - special instructions for treatment digitalized analog signals (such as: parallel multiply, barrel shifting, auxiliary registers for single cycle manipulation of data tables, etc.)

- From these characteristics, the user could see the best application of DSP in the area where a short but very efficient algorithm should be used in order to replace a function that was previously done with analog components; such as filters, fast Fourier transforms.

- Assembly language was sufficient because the code needed be optimized and not be longer than a few pages.

- In recent years instead we see that the characteristics of the DSP's are improving very rapidly. No one features of the past was dropped (hardware multiplier, special instructions, etc.) but in addition we see that address capability has increased very much and some powerful one cycle instructions has been added (floating point).

- The DSP96002 from Motorola is capable of addressing 12 Gigabyte of memory: 4 gigabyte for Program memory, and 4 gigabyte each for the two Data memories banks.

- This DSP has also a high degree of parallelism and pipelining. One can write in a single line of assembly code the following operations that will be executed in one cycle

```
FMPY D9,D7,D1 FADDSUB.S D5,D2  D4.S,X:(R5)  Y:(R1),D7.S
```

- Some of the characteristics that make the DSP particularly suitable to treat discrete signals are found in its instruction set.

  - Several presently available DSP's can: perform a simple operation y = ax + b in one cycle (75 nsec) while at the same time performing some operations on addresses by updating pointers.

  - can have hardware "DO LOOP" instructions.

  - can have compare magnitude instructions

- Assembler language may be convenient to optimize a fast algorithm, but is a limitation for large programs. The principle firms: AT & T, Motorola, Philips and Texas Instruments are already providing "C" compilers for their DSP's.

- The leading firms in designing and manufacturing DSP's are:

  Texas Instruments,
  NEC,
  Motorola,
  Philips,
  Zoran,
  Analog Devices,
  AT & T,
  Honeywell Inc.,
  Thomson,
  Inmos,
  Fujitsu.
  Intel
  National Semiconductor

- The software development support is given by the firms themselves and also by:

  - TEKTRONIX that offers the Signal Processor Workstation (SPW) that runs on VAX or Apollo Computer Domain.

  - DATACUBE offers Euclid Tools and DSP-1000

  - DSP Development introduced DADiSP which is a menu driven software for displaying and analyzing digital waveforms.

  - STEP Engineering offers Step-4 SDT running on IBM PC AT

- SPECTRUM offers DSP~LINK letting you match analog or digital interface boards with DSP chip of your choice from Texas, Motorola and Analog Devices.

[redacted] SOFTWARE PACKAGES TO DESIGN DIGITAL FILTERS:

- Hyperception HYPERSIGNAL

- Atlanta Signal Processors, Inc.

Sometimes though these software packages are not suitable for solving high performance Real-Time applications. In that case it is necessary to use a particular type of DSPs more specific solving filtering problems, or in other cases, communication.

DSP
worldwide third party support

FRANCE

- EIA
- OROS
- SYMINEX
- TEXAS INSTRUMENTS
- XCOM

GERMANY

- DSPACE
- ELECTRONIC TOOLS
- DISTEC
- FUCHS MESSETECHNIK
- KONTRON ELECTRONICS

ITALY

- PRACTICA SRL

UNITED KINGDOM

- BEDFORD RESEARCH
- COMPUTER SOLUTIONS LTD
- ENSIGMA
- JOYCE-LOEBL

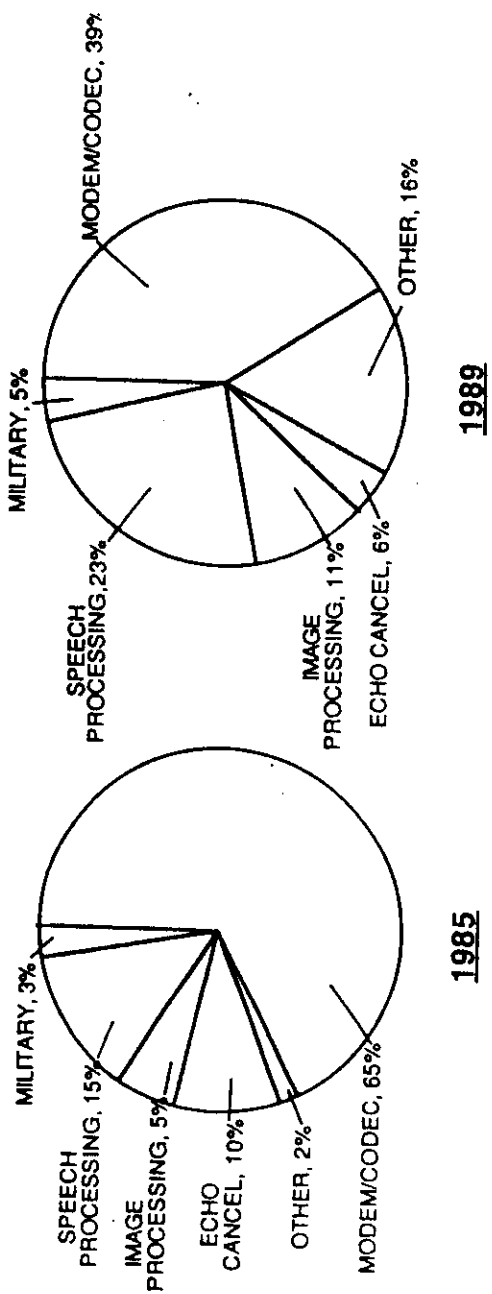
- LOUGHBOROUGH SOUND IMAGES
- RACAL MICRO-ELECTRONICS SYS
- THORN EMI
- ULTRA DIG. SYST.

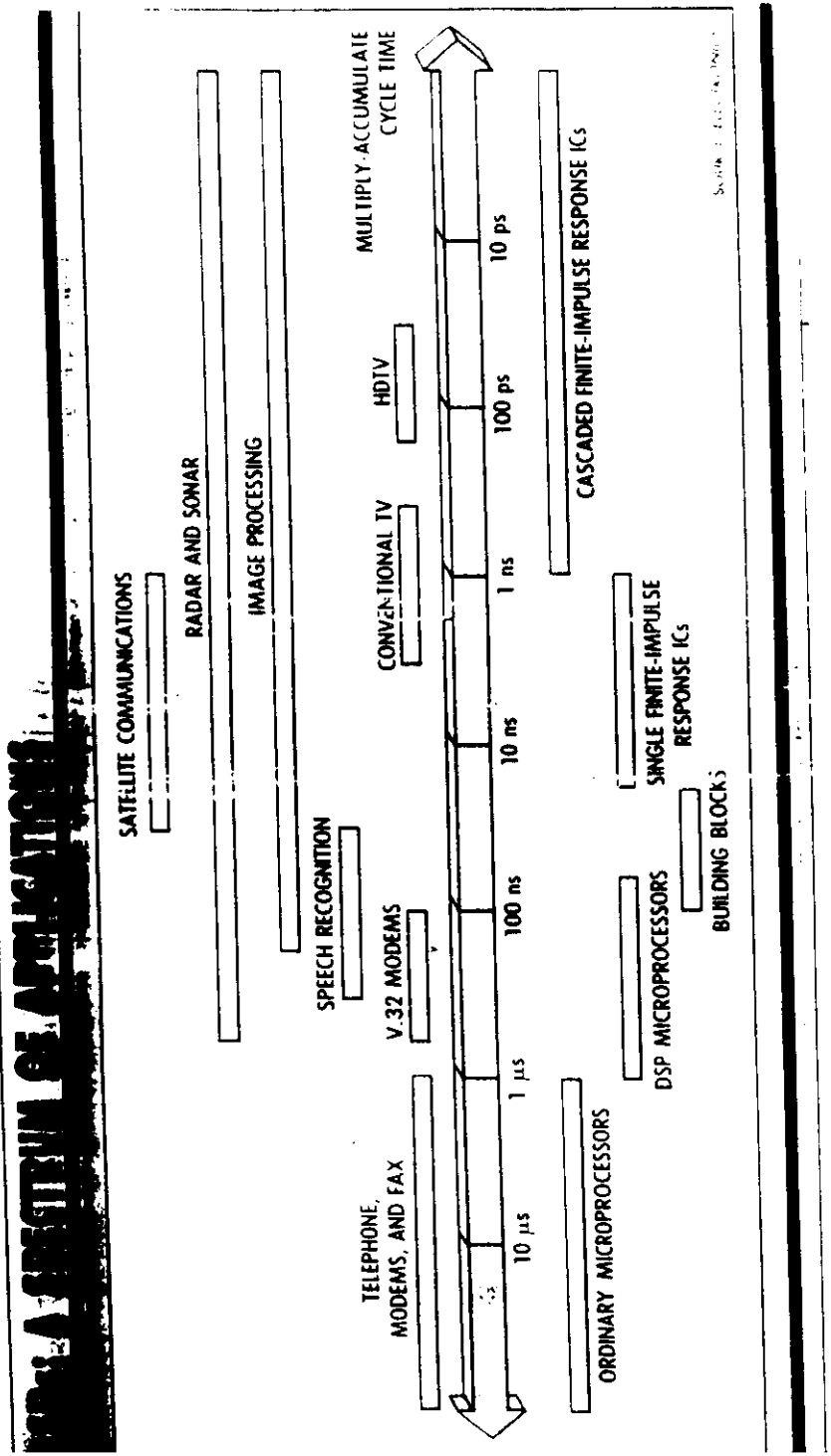
USA

- ADVANCED DIGITAL SYS
- AI WARE INC.
- ALLEN ASHLEY
- APPLIED BUSINESS
- ARIEL CORP
- ATHENA GROUP, THE
- ATLANTA SIG. PROCESSORS
- AVOCET SYSTEMS
- BURR BROWN
- CALCOMP
- CASUAL SYSTEM INC.
- Comm. Autom. and Control
- COMPUTALKER
- CYBERNETIC MICRO SYS
- 3D SYSTEMS INC.
- DAISY SYSTEM CORP
- DALANCO SPRY
- DIGITAL AUDIO CORP.
- DIGITAL SIGNAL PROC SOF.
- DIGITAL SOUND CORP
- DSP APPLICATIONS
- DSP TECHNOLOGY CORP
- EIGHTEEN EIGHT LAB.
- ELECTRO RENT CORP
- EMONA ENTERPRISES LTD
- FACS INC
- FORTH INC
- GAS LIGHT SOTWARE
- HEWLETT-PACKARD
- HYPERCEPTION
- JOHN WILEY & SONS
- KAY ELEMETRICS CORP
- MEMOCOM DEVELOP. TOOLS
- METME CORP
- MICRO K SYSTEMS
- MICROCRAFT CORP
- MICROWORKSHOP
- MOMENTUM DATA SYSTEMS
- NAVTROL COMPANY INC
- PC ELECTRONICS
- PH ASSOCIATES
- PRENCTICE-HALL INC
- RAPID SYSTEMS INC
- SARIN
- SENTRY
- SENTRY TEST SYSTEMS
- SIGNAL TECHNOLOGY INC
- SIGNIX CORP
- SIGNUM SYSTEMS
- SKY COMPUTERS
- SONITECH
- SPECTRON
- SPECTRUM SIGNAL PROC
- SPECTRAL INNOVATION
- SYMMETRIC RESEARCH
- TEKTRONICX INC
- TELEPHOTO COMMUNICATION
- TELERIC
- TELEVIC
- TIAC
- WADIA
- VALID LOGIC SYSTEMS
- VOTAN
- WHITMAN ENGINEERING

DSP APPLICATION AREAS

(BY UNIT VOLUME)





Low-cost and high-speed favors the use of DSP' in these applications.

- telecommunication (high speed modems)
- image processing and pattern recognition
- speech recognition, musical synthesizer
- direction finding in radar,
- target tracking (closed loop systems)
- ultrasound medical imaging
- automobiles: antiskidid braking systems, adaptive suspension, engine control and instrumentation
- disk drives, tape drives
- printers, plotters and consumer products
- digital filters
- digital HIFI, digital AM/FM radio
- workstations (?)
- robotics
- spectrum analysis

D.S.P. Performance Comparison

Firm	type	Technology	Cycle (nsec)	Architecture	Size-Instruction bit	Program Memory	Data Memory	N. Bus	ALU	Others
ANALOG DEVICES	ADSP 2100	CMOS	125	Hardware	24	16/32K	16 K	1	16-bit Fix	cash memory on chip
AT&T	DSP 32	NMOS	250		32	2K	56K Ext 4K Int	2	32-bit Flo 16-bit Int	Serial I/O Parall I/O $175 UNIX DSP32 SLib.
FUJITSU	8764	CMOS	100		24	1K	1Kx16 Ext 256x16 Int	1	16-bit Fix	Edit-Ass-Debug CP/M-86 Hand. Emul.
INMOS	IMS A100	CMOS	100		16	Input 2.5 to 10 million samples/sec Processing rate 80-320 million instr./sec				Av. Oct. 86 $500
INTEL	2920		400		24					A/D 4 inputs D/A 2 outputs
MOTOROLA	DSP 56000	HCMOS	97.5	3 exec. in. Data ALU Addr. ALU Prog Contr.	24	64K Ext 2K Int.	15K ROM 15K RAM 128K Ext	4 Data Bus 3 Add Bus	24-bit Fix	24 GP I/O 16 bit port DMA RS232C Codec Sync IBI
NATIONAL	LM 32900	CMOS	100	Hardware	28	64K	2K (16K-64K)	7	32-bit Fix	IBM VRX NATIONAL NS32-VR
NEC	μPD 77230	CMOS	150	Hardware	32	2K	1K RAM 1K ROM	2	55-bit Float	VAX (Unix-VMS) Evakit 77230
PHILIPS	DSP	CMOS	125	Hardware	40	64K	2x Ram (128x16) 512x16 Rom	2	16-Fix	
TEXAS	TMS320M10 TMS320M20 TMS320C25	NMOS NMOS CMOS	200 (160) 100	Hardware Hardware	16-bit 16-bit	4K 64K 4K Int	144x16 64K 544 Int	1 1	32bitFix 32bitFix	8-I/O $30 con $240

Others from: Advanced Micro Devices, American Microsystem Inc, Thomson, Pertec, IBM

2.1.1 Harvard Architecture

The TMS32010 utilizes a modified Harvard architecture in which program memory and data memory lie in two separate spaces. This permits a full overlap of instruction fetch and execution.

Program memory can lie both on-chip (in the form of the 1536 X 16-word ROM) and off-chip. The maximum amount of program memory that can be directly addressed is 4K X 16-bit words.

Instructions in off-chip program memory are executed at full speed. Fast memories with access times of under 100 ns are required.

Data memory is the 144 X 16-bit on-chip data RAM. Instruction operands are fetched from this RAM; no instruction operands can be directly fetched from off-chip. However, data can be read into the data RAM from a peripheral by using the IN instruction or read from program memory by using the TBLR (table read) instruction. The OUT instruction will write a word from the data RAM to a peripheral, while a TBLW instruction will write a data RAM word to program memory (presumably, off-chip).

Figure 2-2 outlines the overlap of the instruction prefetch and execution. On the falling edge of CLKOUT, the program counter (PC) is loaded with the instruction (load PC2) to be prefetched while the current instruction (execute 1) is decoded and is started to be executed. The next instruction is then fetched (fetch 2) while the current instruction continues to execute (execute 1). Even as another prefetch occurs (fetch 3), both the current instruction (execute 2) and the previous instruction are both still executing. This is possible because of a highly pipelined internal operation.

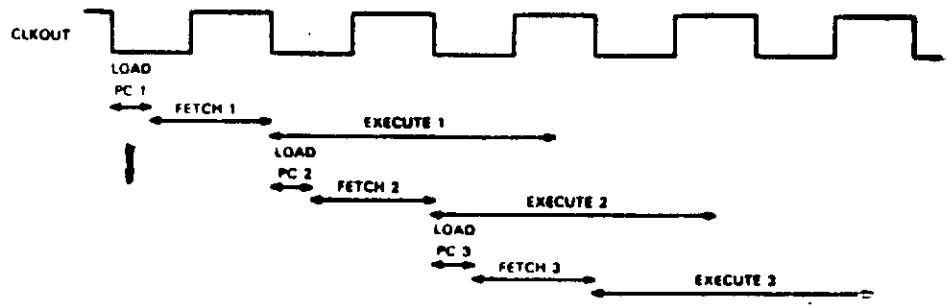


FIGURE 2-2 – HARVARD ARCHITECTURE

KEY FEATURES

- 16-bit instruction/data word
- 32-bit ALU/accumulator
- 16 x 16-bit multiply in 200 ns
- 0 to 15-bit barrel shifter
- Eight input and eight output channels
- 160 ns or 200-ns instruction cycle
- 288-byte on-chip data RAM
- ROMless version – TMS32010
- 2.7-micron NMOS technology or CMOS
- Single 5-V supply
- 40-pin DIP
- 16-bit bidirectional data bus with 40-megabits-per-second transfer rate
- Interrupt with full context save

DSP SELECTION

Selection of a general purpose DSP to be used in the network system for general application.

Among existing DSPs, I will take the following into consideration for their performance and their diffusion (consequently hardware and software support):

- **AT & T DSP32C**
- **Motorola DSP96001**
- **TEXAS INSTRUMENT TMS320Cxx**

Features

- Full standard C language compiler for the *WE* DSP32 Digital Signal Processor
- Complete development environment, with symbolic debugging and assembly-language interface
- Standard *UNIX*® System library support, including libm and a subset of libc
- Includes *WE* DSP32-AL Application Software Library of signal processing functions

Compiler Support

The DSP32 CC package provides a complete development environment for writing, debugging, and testing programs. In addition to the C compiler, an enchanced version of the *WE* DSP32-SL Support Software Library, a symbolic debugger, and other utilities are included

AT&T

WE® DSP32C Development System

Clockwise from top: DSP32C software development card, dynamic RAM expansion card, multiple in-circuit emulation interface, in-circuit emulation card (and cable), and PCbus interface card

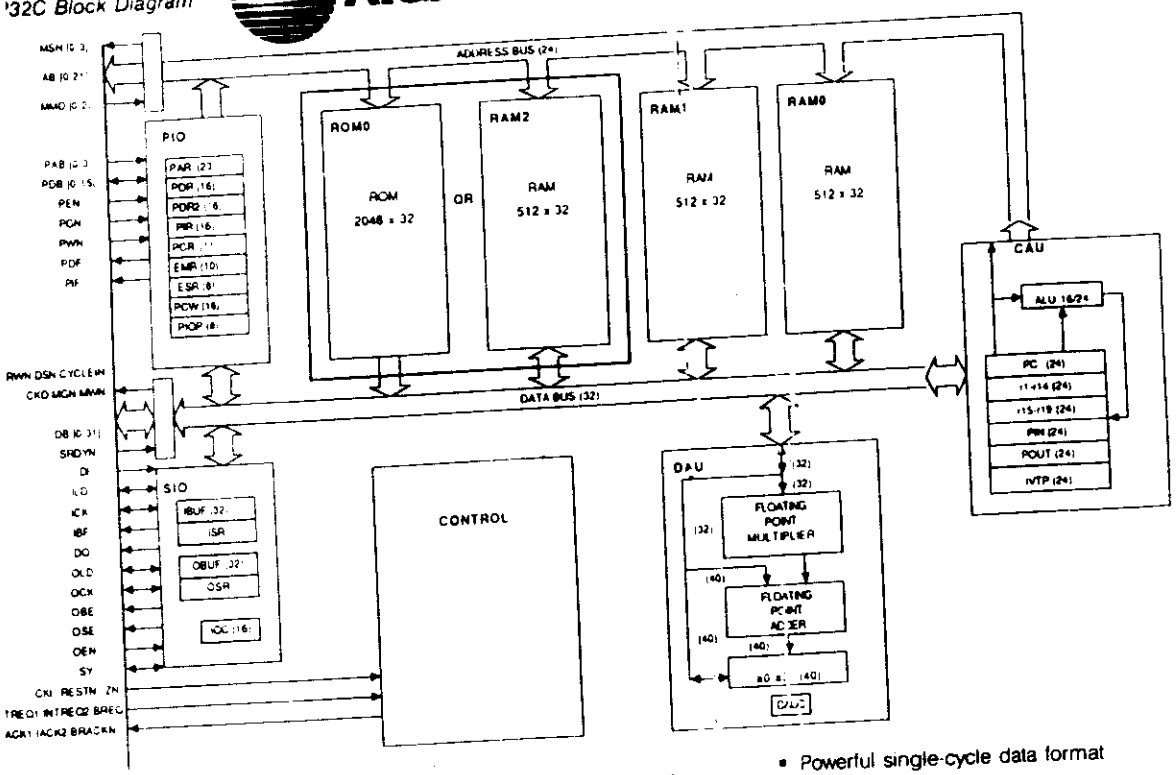
Features

- Software development card for IBM PC/XT/AT® (and compatible) personal computers, featuring:
  - Full-speed operation of DSP32C (50 MHz)
  - 16 Kwords of static RAM (0 wait states)
  - Optional 48 Kwords of additional static RAM (1 wait state)
  - Optional expansion card with 1 Mword of dynamic RAM
  - Serial I/O through an AT&T T7520 High-Precision Codec
- In-circuit emulation card with high-speed PC interface
- Multiple in-circuit emulation interface allows up to four emulation cards to be controlled through one PCbus interface card
- Software development and in-circuit emulation cards are controlled by the DSP32C software simulator

Description

The DSP32C Development System is actually a family of five products. This modular design allows you to purchase only the components you need to assemble the exact development environment your application requires. The two major uses of the system are real-time software development and in-circuit

'32C Block Diagram



LEGEND

a0—a3	Accumulators 0—3
ALU	Arithmetic Logic Unit
CAU	Control Arithmetic Unit
DAU	Data Arithmetic Unit
DAUC	DAU Control Register
EMR	Error Mask Register
ESR	Error Source Register
FPA	Floating Point Adder
FPM	Floating Point Multiplier
IBUF	Input Buffer
IOC	Input/Output Control Register
ISR	Input Shift Register
IVTP	Interrupt Vector Table Pointer
OBUF	Output Buffer
OSR	Output Shift Register
PAR	PIO Address Register
PC	Program Counter
PCR	PIO Control Register
PCW	Processor Control Word
PDR	PIO Data Register
PDR2	PIO Data Register 2
PIN	Parallel Input Register
PIO	Parallel I/O Unit
PIOP	Parallel I/O Ports
PIR	PIO Interrupt Register
POUT	Parallel Output Register
r1—r19	Registers 1—19
RAM	Read/Write Memory
ROM	Read Only Memory
SIO	Serial I/O Unit

Features

- 50 MHz (80 ns) or 40 MHz (100 ns) operation
- All instructions are single cycle
- 32-bit floating-point arithmetic
- Full C compiler
- Data addressable as 8-, 16-, or 32-bit words
- Bit reversal for FFTs
- Powerful single-cycle data format conversions:
  - IEEE 754 floating point
  - 24-bit and 16-bit integer
  - 8-bit linear byte
  - 8-bit μ-law and A-law
- Serial and parallel ports with DMA

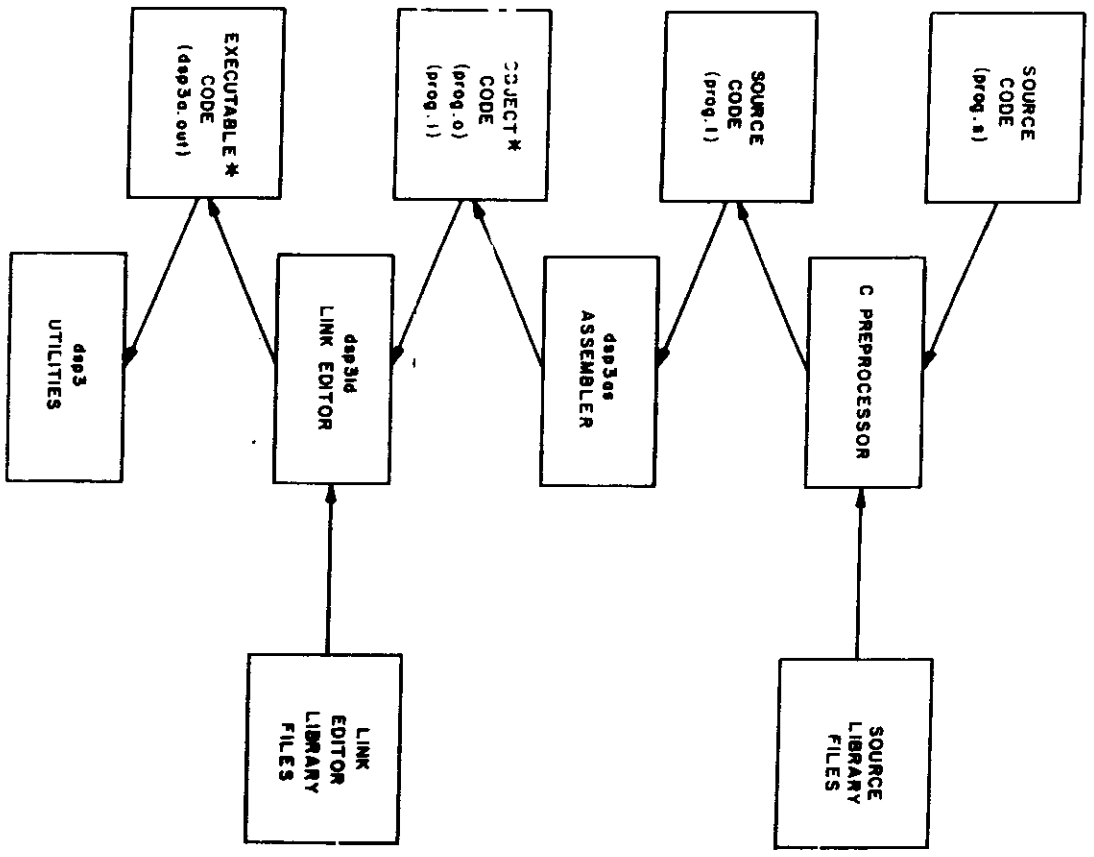
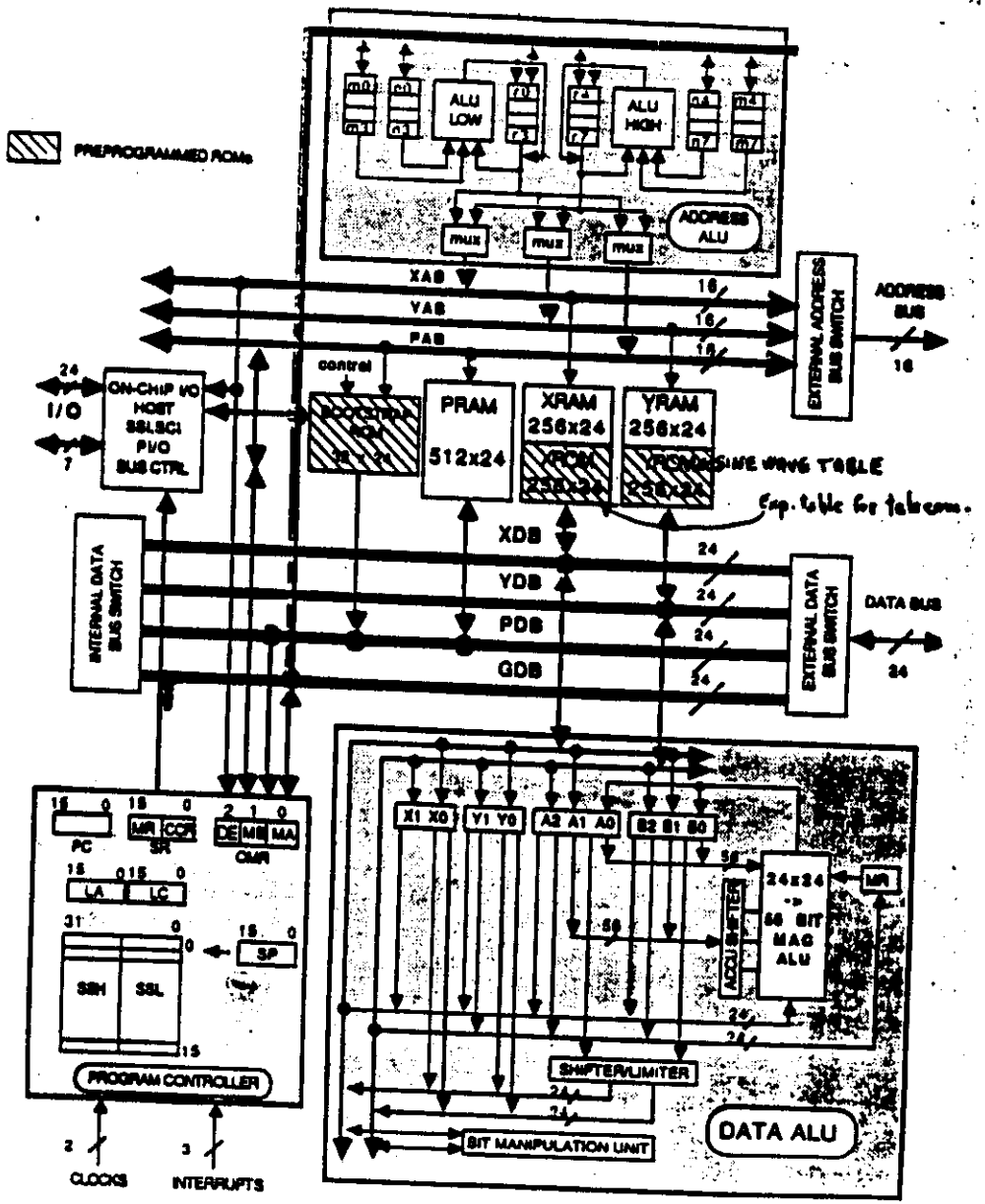


Figure 1. Software Generation Process

dsp3sim	Simulator	**dsp3init** Development System Initializer
dsp3as	Assembler	**dsp3size** Object File Code Size Reporter
dsp3ld	Link Editor	**dsp3strp** Symbol Table and Line Stripper
dsp3make	Make Utility	**dsp3nm** Symbol Table Lister
dsp3ar	Archive Utility	**dsp3dump** Object File Dumper

DSP56001 GENERAL BLOCK DIAGRAM



ADD x0,A B,x1 Y0,y: $3F80

ADD x0,A x1,x:(R1)+ y:(R4)-,B

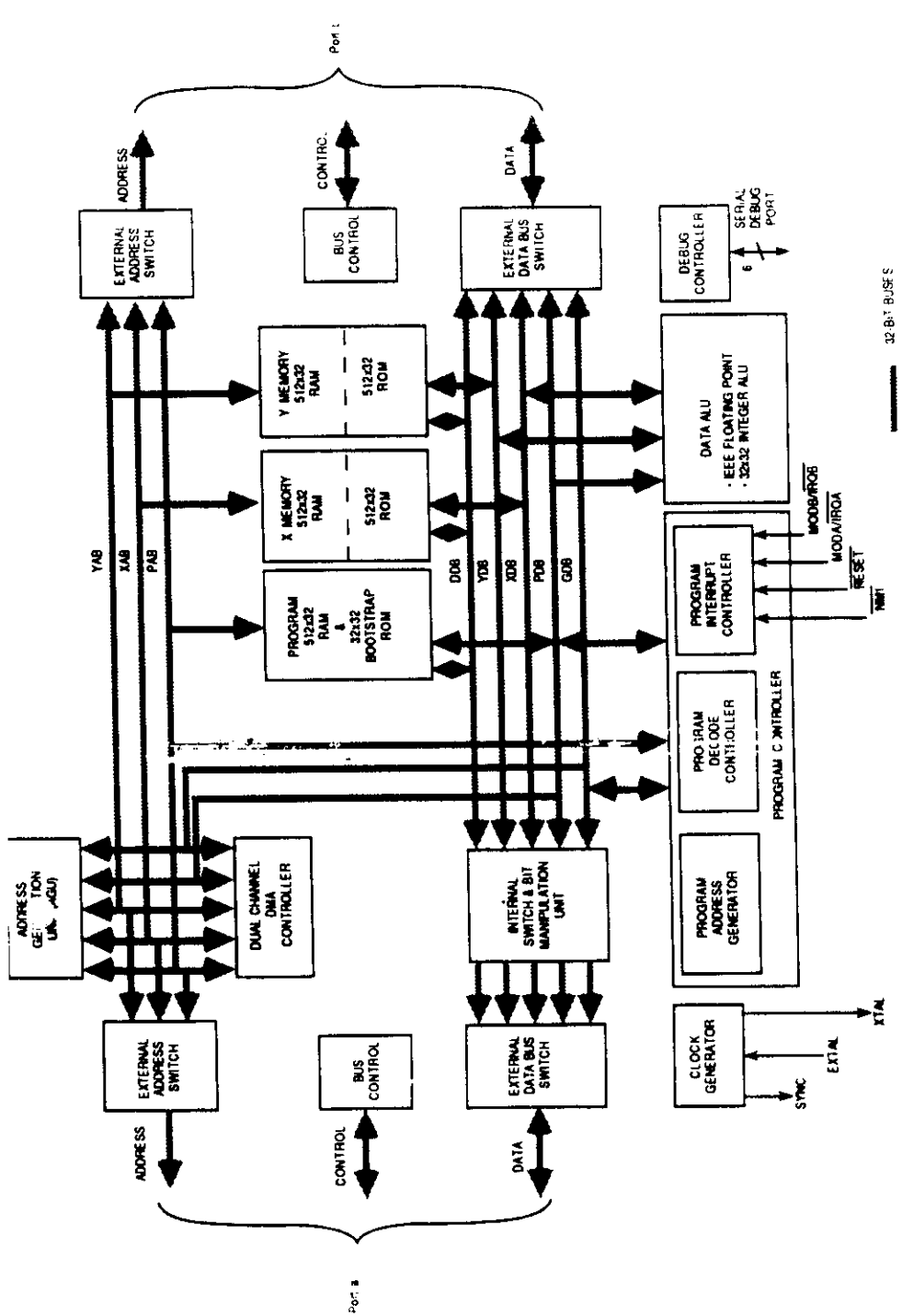
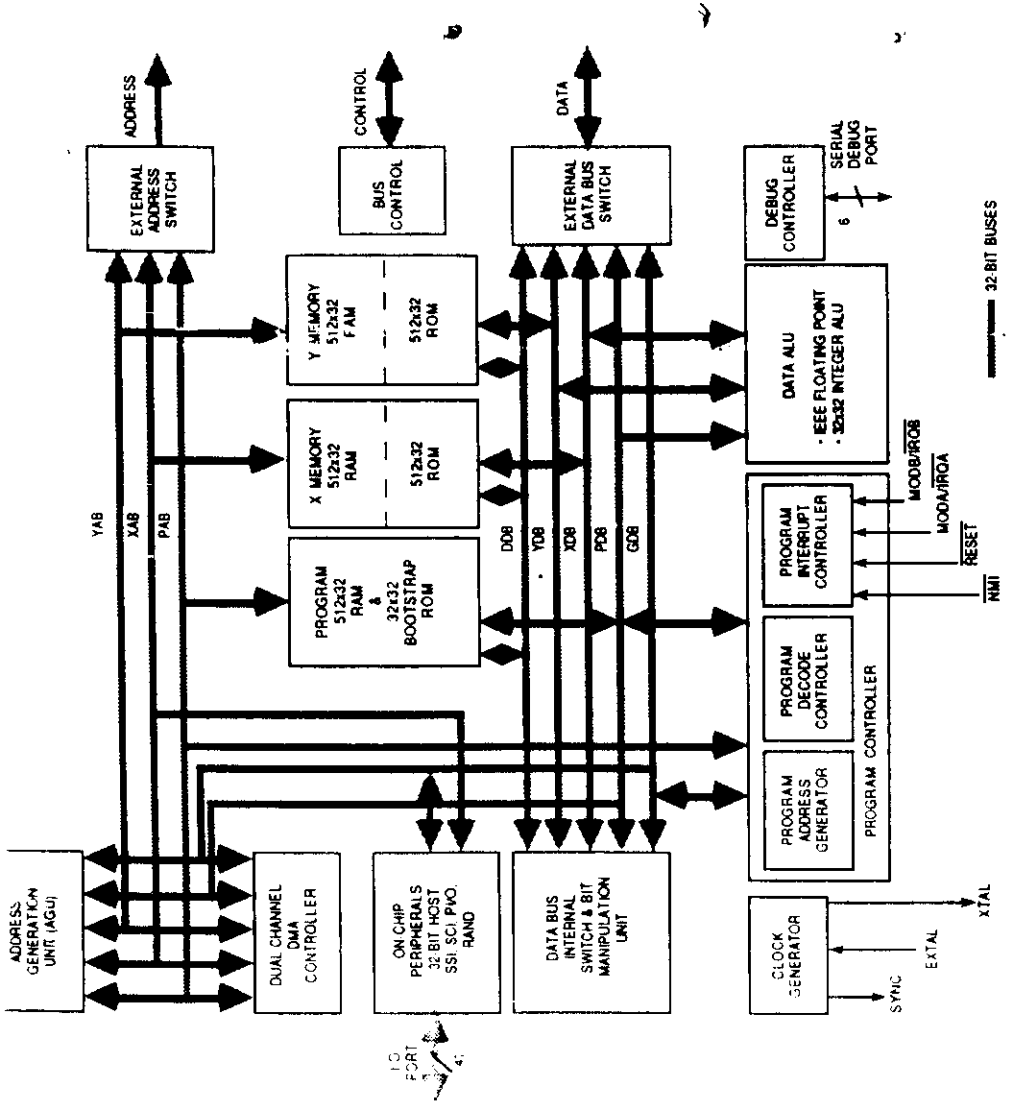


Figure 2. DSP56001 Block Diagram



MOTOROLA DSP96000

- SIMULATOR PROGRAM
-MACRO CROSS ASSEMBLER
- LINKER/LIBRARIAN

```
MOTOROLA DSP96002 SIMULATOR:  VERSION 1.0 3-27-89
>ad TEST
>eading file:TEST.lod
>isassemble
>:$00000000 01b003f4 00000005 - do #<$1f4,$6
>:$00000002 e98H0b2a          - fmpy d6,d8,d2 fadd.s d3,d0 x:(r0),d4.s d2.s,y:(
-5)+
>:$00000003 eae66c2f          - fmpy d7,d8,d3 feddsub.s d4,d0 x:(r1)+,d6.s d5.s
,y:(r4)+
>:$00000004 00000000          - nop
>:$00000005 00000000          - nop
>:$00000006 00000000          - nop
>>

asm    break    change    copy    disassemble    display    <space>=more


   m7= $ffffffff   n6= $00000000   r6= $00000000
     = $ffffffff   n5= $00000000   r5= $00000006
   m4= $ffffffff   n4= $00000000   r4= $00000000
   m3= $ffffffff   n3= $00000000   r3= $00000000
   m2= $ffffffff   n2= $00000000   r2= $00000000
   m1= $ffffffff   n1= $00000000   r1= $00000000
   m0= $ffffffff   n0= $00000000   r0= $00000000
d9.h= $00000000 d9.m= $00000000 d9.l= $00000000 d9=0.0000000000000000e+00
d8.h= $40000380 d8.m= $00000800 d8.l= $00000000 d8=1.1210387714598540e-44
d7.h= $40000380 d7.m= $00000700 d7.l= $00000000 d7=9.8090892502737190e-45
d6.h= $40000380 d6.m= $00000600 d6.l= $00000000 d6=8.4077907859489020e-45
d5.h= $40000380 d5.m= $00000500 d5.l= $00000000 d5=7.0064923216240850e-45
d4.h= $40000380 d4.m= $00000500 d4.l= $00000000 d4=7.0064923216240850e-45
d3.h= $40000380 d3.m= $00000400 d3.l= $00000000 d3=5.6051938572992680e-45
d2.h= $40000380 d2.m= $00000200 d2.l= $00000000 d2=2.8025969286496340e-45
d1.h= $40000380 d1.m= $00000200 d1.l= $00000000 d1=2.8025969286496340e-45
d0.h= $40000380 d0.m= $00000200 d0.l= $00000000 d0=2.8025969286496340e-45
  pc= $00000000   sr= $70000000   la= $00000000   lc= $00000000
 ssh= $00000000  ssl= $00000000   sp= $00000000  omr= $00000001
 cyc= 00000000000 ictr= 00000000000 cnt1= 00000000000 cnt2= 00000000000
>:$00000000 01b003f4 00000005 - do #<$1f4,$6
>>

asm    break    change    copy    disassemble    display    <space>=more
```

Software Summary

DSP56KCCx
DSP56000/1 Family C Language Compiler

DSP56KCCx is a full Kernighan and Ritchie C implementation supporting development of DSP56000 Family applications.

Features include support of:

- Structures/Unions
- Floating Point
- Pointer Variables
- In-Line Assembly Language Code Compatibility
- Full Function Pre-processor supports:
  - — Macro Definition/Expansion
  - — File Inclusion
  - — Conditional Compilation
- Low Compiler Overhead (approximately 20%)
- Full Error Detection and Reporting

Ordering information:

Host Platform	Operating System	Order Number
IBM®-PC	DOS 2.X, 3.X	DSP56KCCA
Macintosh® II	MAC OS 4.1	DSP56KCCB
SUN-3®	UNIX® BSD 4.2	DSP56KCCC
VAX®	VMS 4.X	DSP56KCCD
VAX	UNIX BSD 4.2	DSP56KCCE

Each package consists of:

- Software
  - — C Compiler (CC56000)
  - — Macro Cross Assembler Program (ASM56000)
  - — Linker/Librarian (LNK56000/LIB56000)
- Documentation
  - — DSP56KCC Compiler User's Manual
  - — C Pre-Processor User's Manual
  - — Macro Cross Assembler Reference Manual
  - — DSP56000/1 Data Sheets
  - — DSP56000 User's Manual

IBM is a trademark of International Business Machines.
Macintosh is a trademark of Apple Computer, Inc.
SUN-3 is a trademark of Sun Microsystems, Inc.
UNIX is a registered trademark of AT&T.
VAX is a trademark of Digital Equipment Corporation.

MOTOROLA
SEMICONDUCTOR
TECHNICAL DATA

BR522/D

DSP320to56001

Software Summary

DSP320to56001
Translator Software

The DSP320to56001 translator software will convert any 32010 code into code for Motorola's powerful new digital signal processor chip, the DSP56001. The primary features of DSP320to56001 are:

- Translation of any 32010 applications software into DSP56001 source code
- Two modes of operation:
  - — Translates to 56001 source code for potential optimization and assembly with the DSP56000SASMA or DSP56000CLASA software
  - — Translates and runs 32010 Code "as is" directly and immediately on the DSP56000ADS, Motorola's DSP560001 Applications Development System
- Runs on IBM®-PC under MS®-DOS or PC-DOS
- C source code of DSP320to56001 program is provided on diskette
  - — User may modify for 32020 and 320C25 translation
  - — Third party vendors may contact Motorola for licensing details
- Registration card provided so users can obtain future optimized versions of DSP320to56001 software, hand-coded macro routines, etc.

MOTOROLA DSP DEVELOPMENT SOFTWARE
32010 TO 56000/1 CODE CONVERSION

HARDWARE REQUIREMENTS

The conversion programs are delivered on one double-sided, double-density 5 1 4 inch floppy disk and may be run from either a floppy disk or a hard disk. They require only enough disk space to hold the output of the converted source file.

The minimum hardware requirements for the conversion programs are:
IBM-PC, XT, AT, or compatible with 256K bytes of RAM and one 5 1.4 inch floppy disk drive.
PC-DOS/MS-DOS v2.0 or later.

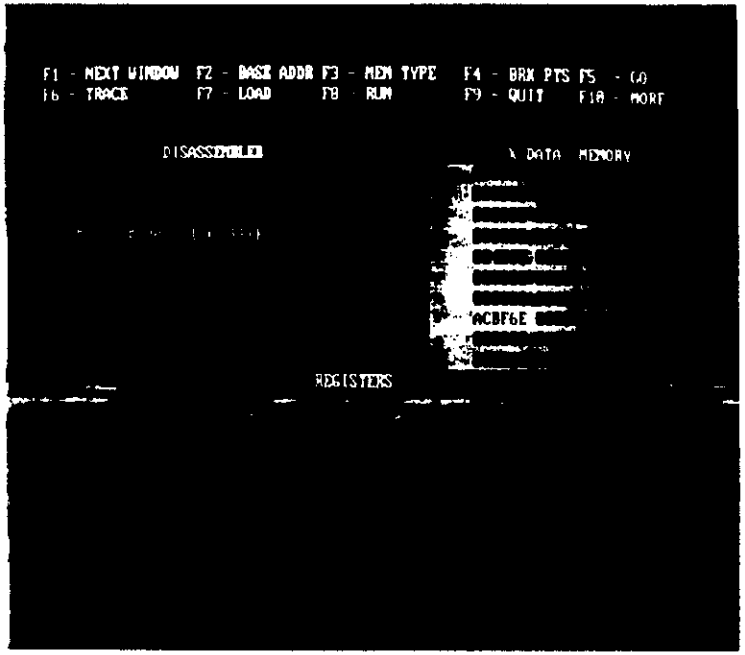
The DSP56000 Application Development System (DSP56000ADS) is recommended as a development tool for designing real-time DSP56000/1 signal processing systems.

IBM is a registered trademark of International Business Machines Corporation
MS-DOS is a trademark of Microsoft, Inc.

Loughborough
Sound Images Limited

MOTOROLA DSP56001

Development and Applications



- *PC plug-in with 20 MHz Processor*
- *144 KBytes Fast SRAM Supplied User-expandable to 576 KBytes*
- *Window-based Debug Monitor*
- *Motorola Support Software included - Assembler, Linker and Simulator C Compiler also available*
- *Twin Channel 16 bit A/D and D/A*
- *Serial and Parallel I/O Expansion*

TMS320 Applications

TMS320C3X

320C30 32-Bit CPU
Virgule glissante
Super DSP

Work stations

Réglages complexes

TMS320C2X

32020 16-Bit CPU
320C25 100 ns Cycle
320E25 128 K Mots adressage
320C26C 1,5 K Mots de RAM

Industrie *Véhicules à moteur* *Domaine militaire*

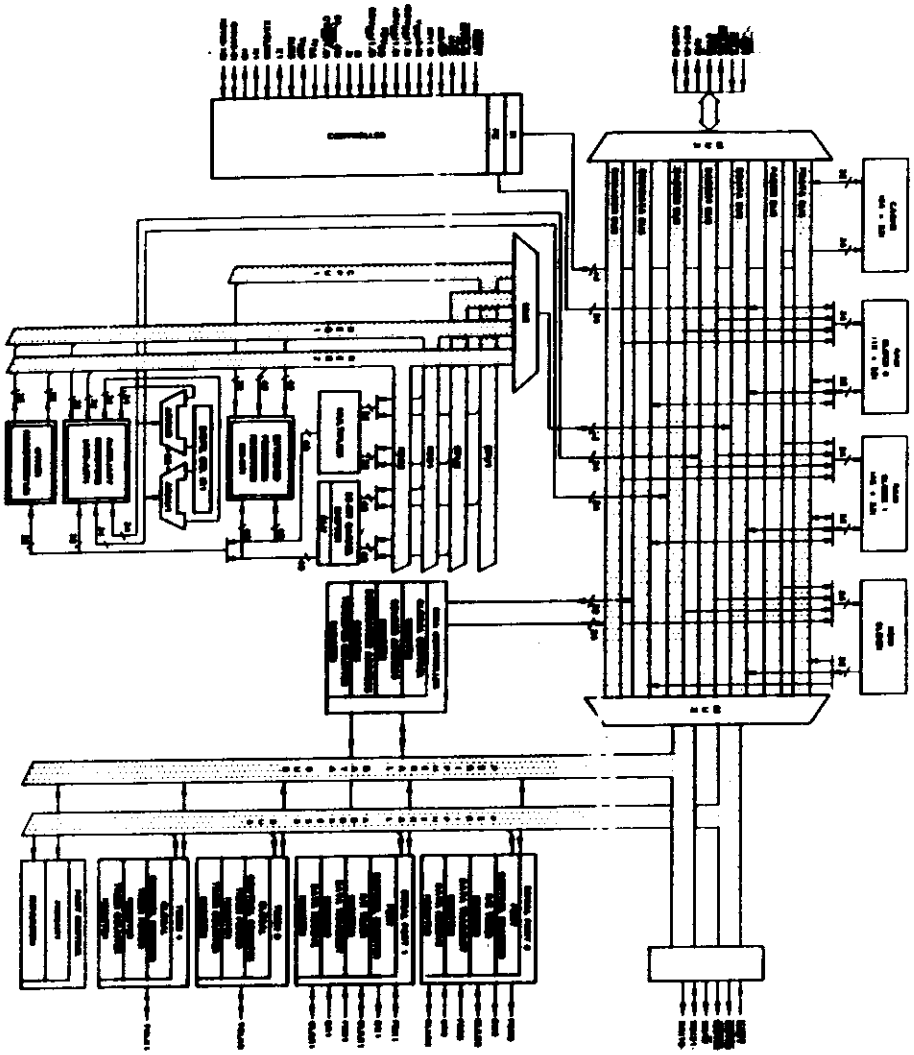
Télécommunication

TMS320C1X

32010 Low Cost
320C10 16-Bit CPU
320C15/E15 ROM/EPROM
302C17/E17 160 ns Cycle
32014/E14 Fonctions contrôleurs

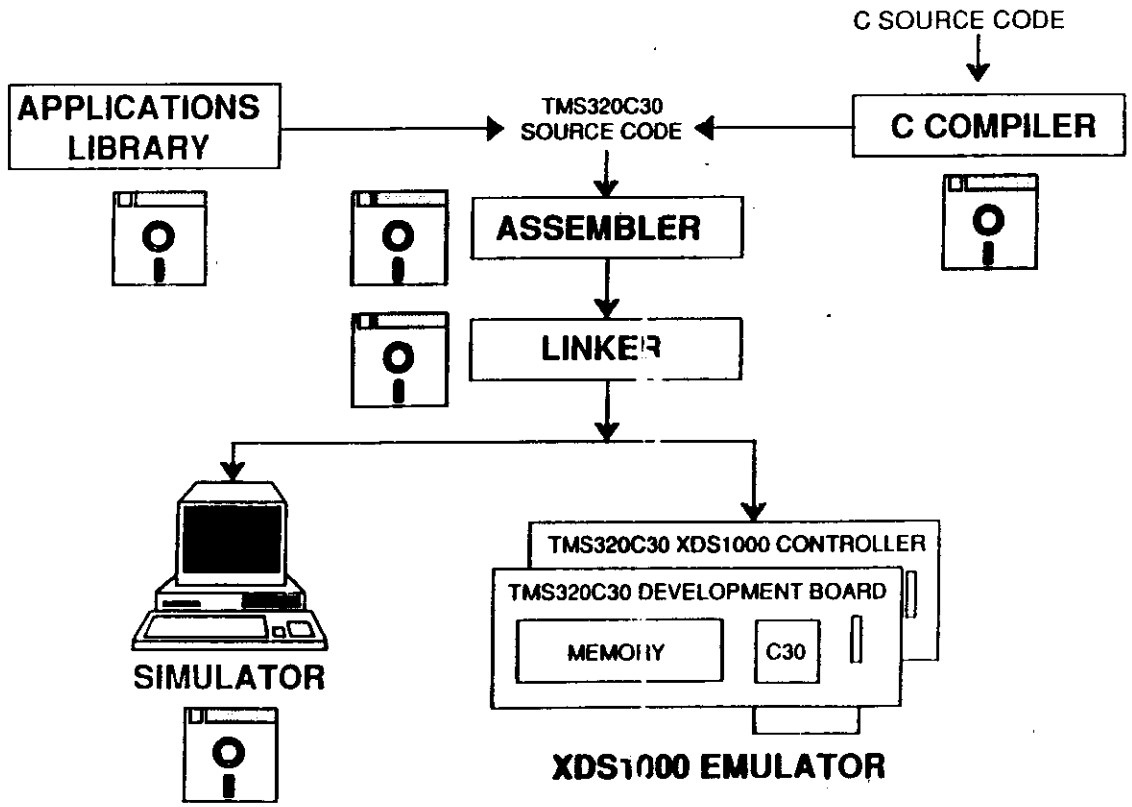
Modems *Bien de consommation*

Traitement de parole *Téléphonie* *Ordinateurs* *Saisie des valeurs de mesure*



TMS 320-DSP FAMILY OF CHOICE

TMS320C30 DEVELOPMENT TOOLS



C32

TMS320 BENCHMARKS

BENCHMARK	TMS320C1x CYCLES	@ 160nS	TMS320C2x CYCLES	@ 80 nS	TMS320C3x CYCLES	@ 60 nS
FIR FILTER						
20 TAP	49	127.5 KHZ	29	431.03 KHZ	25	667 KHZ
64 TAP	133	47.0 KHZ	73	171.73 KHZ	73	228 KHZ
67 TAP	139	45.0 KHZ	76	164.47 KHZ	76	219 KHZ
IIR FILTER						
4 X BIQUAD	44	142.0 KHZ	36	347.22 KHZ	23	725 KHZ
5 X BIQUAD	56	111.6 KHZ	43	284.09 KHZ	27	617 KHZ
TRANSPOSE BIQUAD	69	90.6 KHZ	54	231.48 KHZ	37	450 KHZ
DOT PRODUCT	6	0.96 µS	6	.48 µS	4	.240 µS
MATRIX MULTIPLY						
2 X 2 TIMES 2 X 2	24	3.84 µS	21	1.7 µS	12	.720 µS
3 X 3 TIMES 3 X 1	24	3.84 µS	22	1.8 µS	13	.780 µS
MEMORY TO MEMORY FFT						
64 POINT RADIX 2	3687	590 µS	3088	247 µS	2603	156 µS
256 MOINT RADIX 2	41478	6.64 mS	17602	1.408 mS	12857	.771 mS
1024 POINT RADIX 2	331237	53.0 mS	109755	8.784 mS	61511	3.67 mS
PORT TO MEMORY FFT						
64 POINT RADIX 2	2954	473 µS	1621	130 µS	1370	82 µS
256 POINT RADIX 2	41418	6.64 mS	8520	.682 mS	6734	.404 mS
1024 POINT RADIX 2	331237	53.0 mS	56286	4.503 mS	32354	1.94 mS

A39

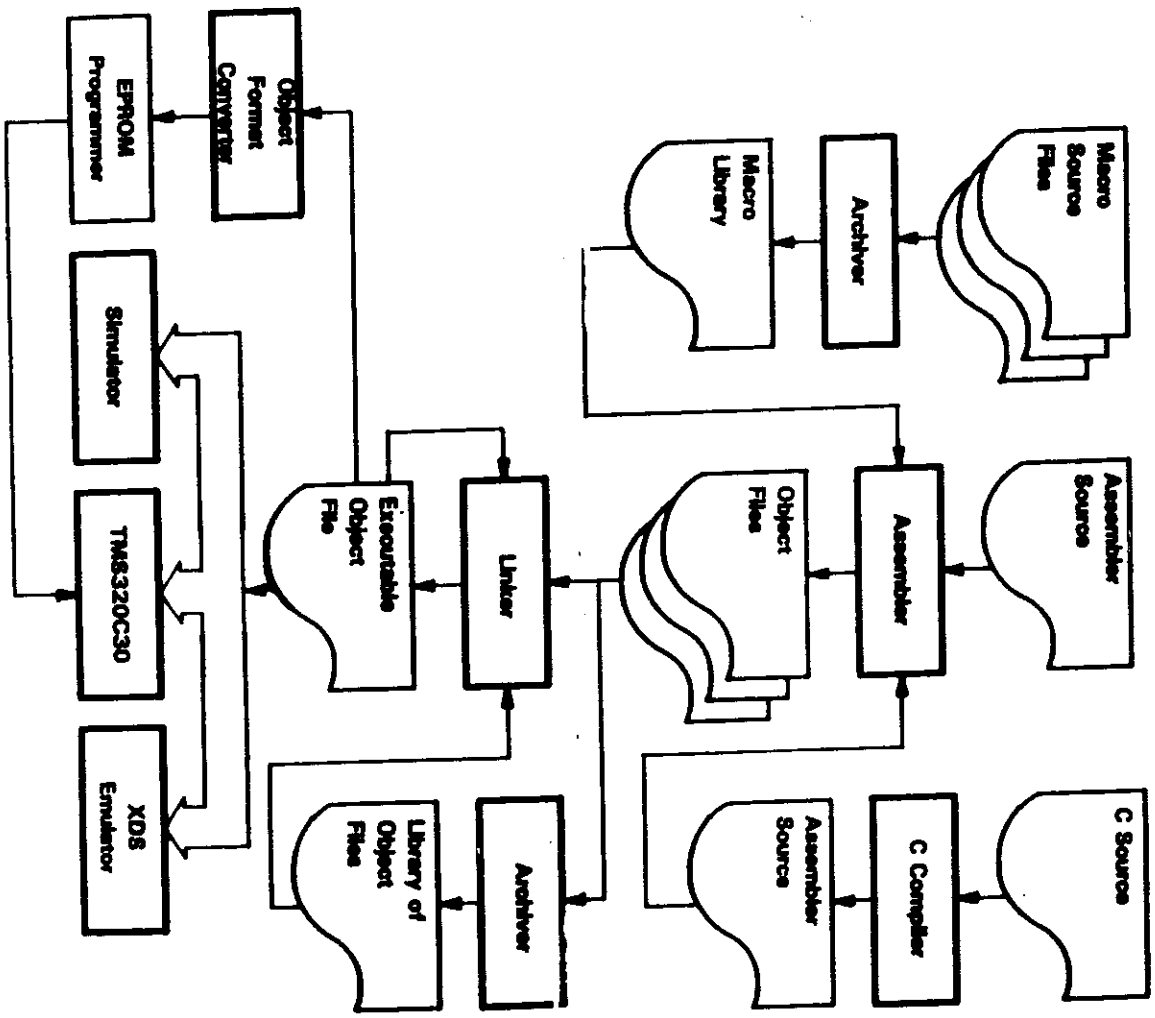


Figure B-1. TMS320C30 Development Environment

DIFFERENT TYPES OF DSP

Among all the DSP that are today on the market, which one should we select ?

Do they have all the same characteristics ?

Which criteria do we use to select them ?

- performance ?
- world-wide diffusion ? (consequently hard
- ware and software support)
- cost ?

Though different types of DSP exist: some more suitable for executing filtering functions and some with more general purpose characteristics, at least two types must be chosen in order to satisfy maximum performance and also to have the best compatibility and standardization among the system.

inmos

IMS A100
Cascadable Signal Processor

Preliminary

FEATURES

- Full 16 bit, 32 stage, transversal filter
- Fully cascadable with no speed degradation or reduction in dynamic range.
- Coefficients selectable as 4, 8, 12, or 16 bits wide
- Data throughput to 10 MHz
- High speed microprocessor compatible interface.
- Data input and output through dedicated ports or via the microprocessor interface
- Fully static high speed CMOS implementation
- TTL compatible
- Single +5V ±10 % power supply
- Power dissipation < 1.5 Watts
- Standard 84-pin ceramic PGA

APPLICATIONS

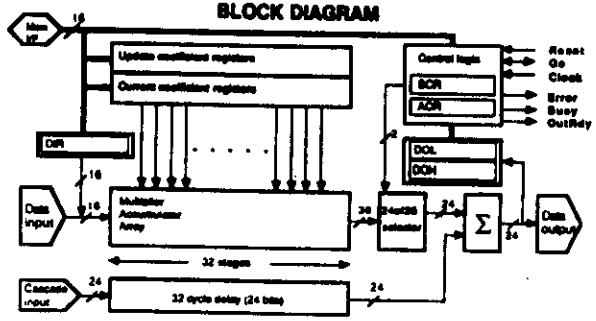
- Digital FIR filtering
- High speed adaptive filtering
- Correlation and Convolution
- Discrete Fourier Transform
- Speech processing using Linear Predictive Coding
- Image processing
- Waveform synthesis
- Adaptive and fixed equalizers and echo cancellers
- Spread spectrum communication
- Beamforming and beamscanning in sonar and radar
- Pulse compression
- High speed fixed point matrix multiplication

DESCRIPTION

The IMS A100 is a high speed, high accuracy 32 stage digital transversal filter. It's flexible architecture allows it to be used as a "building block" in a wide range of Digital Signal Processing (DSP) applications. The part is capable of performing high speed DFTs, convolution, and correlation, as well as many filtering functions.

The input data word length is 16 bits, and coefficients are programmable to be 4, 8, 12, or 16 bits wide; two's complement numerical formats are used for both data and coefficients. The coefficients can be updated asynchronously to the system clock during normal operation, allowing the chip to be used in a variety of adaptive systems. The IMS A100 can also be cascaded to construct longer transversal filters with no additional logic or degradation in speed, whilst preserving a high degree of accuracy. The device is controlled through a standard memory interface, allowing use with any general purpose microprocessor. Data communications can be either through the memory interface, or through dedicated data ports.

BLOCK DIAGRAM

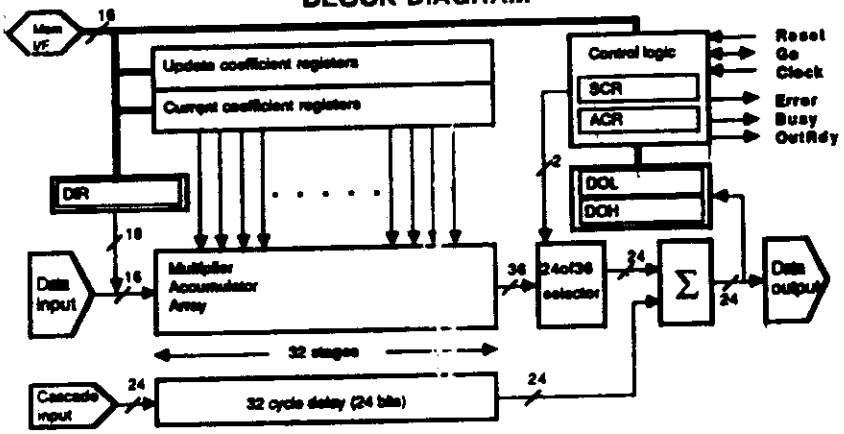


DESCRIPTION

The IMS A100 is a high speed, high accuracy 32 stage digital transversal filter. It's flexible architecture allows it to be used as a "building block" in a wide range of Digital Signal Processing (DSP) applications. The part is capable of performing high speed DFTs, convolution, and correlation, as well as many filtering functions.

The input data word length is 16 bits, and coefficients are programmable to be 4, 8, 12, or 16 bits wide; two's complement numerical formats are used for both data and coefficients. The coefficients can be updated asynchronously to the system clock during normal operation, allowing the chip to be used in a variety of adaptive systems. The IMS A100 can also be cascaded to construct longer transversal filters with no additional logic or degradation in speed, whilst preserving a high degree of accuracy. The device is controlled through a standard memory interface, allowing use with any general purpose microprocessor. Data communications can be either through the memory interface, or through dedicated data ports.

BLOCK DIAGRAM

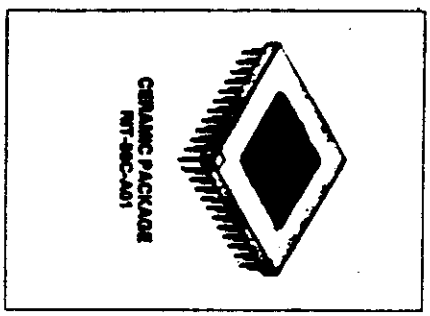


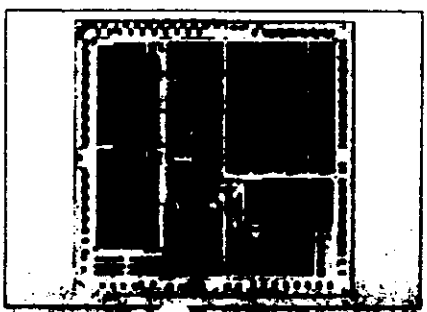
APRIL 1987

FUJITSU (available)

General Purpose Digital Signal Processor

MB 8764





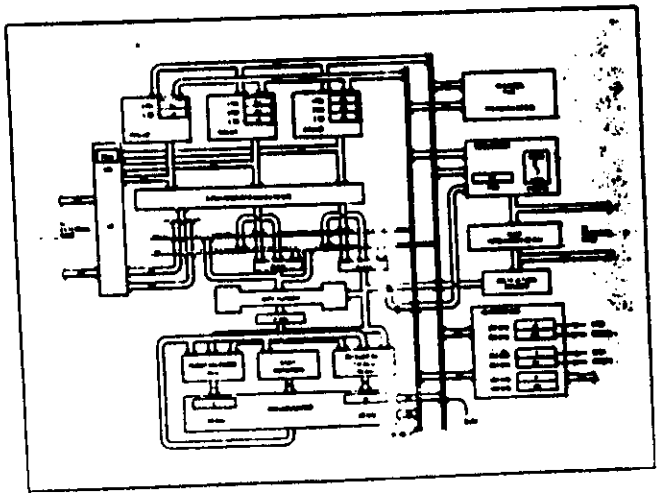
FUJITSU (on paper)

3a Next Generation DSP

F³ - DSP

F³-DSP (Fastest Fujitsu Floating-point DSP)

Features	
Cycle time	55 ns
Mult/Acc time	55 ns
Multiplier (Input 2x32 bit = result 48 bit)	FIX/FLOAT
ALU (Input 32 bit/48 bit = result 48 bit)	FIX/FLOAT
M-ROM internal prog. mem.	4 K words x 32
E-ROM external prog. mem.	64 K words x 32
RAM on-Chip 3-port	512 words x 32
RAM external DATA-RAM	1 MEGA words x 32
Register file	16 words x 32
Ext RAM I/O controller	
2 internal DATA BUSES	
Ext. RAM DATA BUS and common I/O BUS are separated	
I/O controller for common I/O operations	
3 independent Addressing Units for DATA-RAM	
2 Serial Input and 2 Serial Output lines	
3-Stage PIPELINE operation	
4 subroutine levels	
16 interrupts	



ANALOG DEVICES

DSP Microprocessors

ADSP-2100

FEATURES

Separate Program and Data Buses, Extended Off-Chip
Single-Cycle Direct Access to 16K x 16 of Data Memory
Single-Cycle Direct Access to 16K x 24 (Expandable to 32K x 24) of Program Memory
Dual Purpose Program Memory for Both Instruction and Data Storage
Three Independent Computational Units: ALU, Multiplier/Accumulator and Barrel Shifter
Two Independent Data Address Generators
Powerful Program Sequencer
Internal Instruction Cache
Provisions for Multiprecision Computation and Saturation Logic
Single-Cycle Instruction Execution
Multifunction Instructions
Four External Interrupts
125ns Cycle Time
660mW Maximum Power Dissipation with CMOS Technology
160-Pin Grid Array

APPLICATIONS

Optimized for DSP Algorithms Including
- Digital Filtering
- Fast Fourier Transforms

Applications Include
- Image Processing
- Radar, Sonar
- Speech Processing
- Telecommunications

GENERAL DESCRIPTION

The ADSP-2100 is a single-chip microprocessor optimized for digital signal processing (DSP) and other high-speed numeric processing applications. It integrates computational units, data address generators and a program sequencer in a single device.

The ADSP-2100 makes efficient use of external memories for program and data storage, freeing silicon area for increased processor performance. The resulting architecture combines the functions and performance of a bit-slice/building block system with the ease of design and development of a general-purpose microprocessor. The ADSP-2100 (K Grade) operates at 8.192MHz. Every instruction executes in a single 125ns cycle. Fabricated in a high-speed 1.5 micron double-layer metal CMOS process, the ADSP-2100 dissipates less than 600mW.

The ADSP-2100's flexible architecture and comprehensive instruction set support a high degree of operational parallelism. In one cycle the ADSP-2100 can:

- generate the next program address
- fetch the next instruction
- perform one or two data moves
- update one or two data address pointers
- perform a computational operation.

DEVELOPMENT SYSTEM

The ADSP-2100 is supported by a complete set of tools for software and hardware system development. The Cross-Software System provides a System Builder for defining the architecture of systems under development, an Assembler, a Linker and a Simulator. The Simulator provides an interactive instruction-level simulation. A PROM Splitter generates PROM burner compatible files. An Emulator is available for hardware debugging of ADSP-2100 systems.

ADDITIONAL INFORMATION

For additional information on the architecture and instruction set of the processor, refer to the *ADSP-2100 User's Manual*. For more information about the Development System, refer to the *ADSP-2100 Cross-Software Manual* and the *ADSP-2100 Emulator Manual*. For examples of a variety of ADSP-2100 applications routines, refer to the *ADSP-2100 Applications Handbook, Volume 1*. Manuals are available from your local Analog Devices sales office. See ordering information.

NEC
NEC Electronics Inc.

μPD7723(
ADVANCEI
SIGNAL PROCESSOI

Description

The μPD77230 Advanced Signal Processor (ASP) is the high-end member of a new third-generation family of 32-bit digital signal processors. This CMOS chip implements 32-bit full floating-point arithmetic, and is intended for digital signal processing and other applications requiring high speed and high precision.

All instructions execute in one instruction cycle. The μPD77230 executes a 32-bit by 32-bit floating point multiply with 55-bit product, sum of products, data move, and multiple data pointer manipulations—all in one 150-ns instruction cycle.

Features

- ☐ Fast instruction cycle: 150 ns using 13.3-MHz clock
- ☐ All instructions execute in one cycle
- ☐ 32- x 32-bit floating point arithmetic
- ☐ Large on-chip memory (32-bit words)
  - — 1K data RAM (two 512-word blocks)
  - — 1K data coefficient ROM
  - — 2K instruction ROM
- ☐ 8K- x 32-bit external memory; 4K may be instruction memory
- ☐ 1.5-μm CMOS technology
- ☐ 32-bit internal bus
- ☐ -bit ALU bus
- ☐ Dedicated internal buses for RAM, multiplier, and ALU
- ☐ Eight accumulators/working registers (55 bits)
- ☐ 47-bit bidirectional barrel shifter
- ☐ Two independent data RAM pointers
- ☐ Modulo 2^n incrementing for circular RAM buffers
- ☐ Base and index addressing of internal RAM
- ☐ Data ROM capable of 2^n incrementing
- ☐ Loop counter for repetitive processing
- ☐ Eight-level stack accessible to internal bus
- ☐ Two interrupts: maskable and nonmaskable (NMI)
- ☐ Serial I/O (5 MHz)
- ☐ Master/slave mode operation
- ☐ Three-stage instruction pipeline
- ☐ Single +5-volt power supply
- ☐ Approximately 1.2 watts

Ordering Information

Part Number	Package Type
μPD77230R	68-pin PGA

Applications

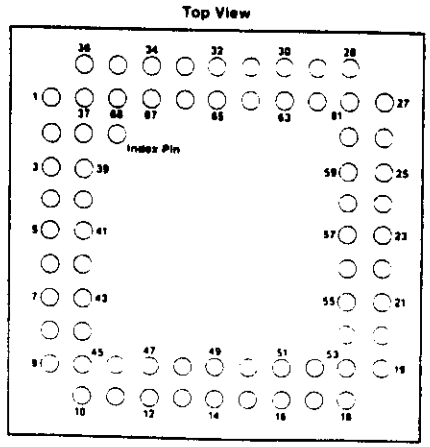
- ☐ General-purpose digital filtering (FIR, IIR, FFT)
- ☐ High-speed data modems
- ☐ Adaptive equalization (CCITT)
- ☐ Echo cancelling
- ☐ High-speed controls
- ☐ Image processing
- ☐ Graphic transformations
- ☐ Instrumentation electronics
- ☐ Numerical processing
- ☐ Speech processing
- ☐ Sonar/radar signal processing
- ☐ Waveform generation

Floating-Point Performance Benchmark:

Second-order digital filter (biquad)	0.9 μ
32-tap finite impulse response filter	5.25 μ
Fast Fourier transform (FFT)	
32-point complex (radix 2)	0.15 m
512-point complex FFT	4.7 m
1024-point complex FFT	10.75 m
Square root	6.0 μ

Pin Configuration

68-Pin PGA



PHILIPS

PCB5010 and PCB5011 at a glance

The PCB5010 and PCB5011 are programmable digital signal processors (DSPs) belonging to our new CMOS SP 50 family of DSP ICs. With their highly parallel architecture and extensive I/O capabilities, they can provide all the processing power you could want. The PCB5010 has on-chip RAM, plus on-chip mask programmable ROM for data and program storage, giving a low chip-count by eliminating the need for external memories.

The PCB5011, a ROMless version, is available for small-scale production runs, and for special situations where a large program memory (up to 64 K) is required.

There are also PCF versions of these chips which are identical except for a higher operating temperature range.

Features of the PCB5010 and PCB5011

- Harvard architecture with two 16-bit data buses
- Up to 6 basic operations performed simultaneously during each instruction. The execution of a new instruction can begin every 125 ns.
- 16 × 16-bit multiplications in a 40-bit hardware multiplier/accumulator plus barrel-shifter/format adjuster
- 2-operand, 31-operation ALU
- Three-port scratchpad file containing fifteen 16-bit registers
- Two 128 × 16-bit static RAMs for data
- Three independent address computation units for the data memories
- Two independent externally-controlled serial input and output channels (up to 4 million bits/s)
- 16-bit bidirectional parallel data port (up to 8 million words/s)
- 16-bit address port including 4 page-bits (64 K × 16-bit external data memory address range)
- Maskable interrupts
- Pipelined mode (P) and non-pipelined mode (NP)
- CMOS technology
- Single 5 V power supply
- Operating temperature range:
  - 0 to 70 °C for PCB versions
  - −40 to 85 °C for PCF versions

Specific features of the PCB5010

- 512 × 16-bit on-chip data ROM (mask programmable)
- 987 × 40-bit on-chip program ROM (mask programmable)
- 32 × 40-bit on-chip program RAM
- 5 × 40-bit on-chip ROM containing a 'load-RAM' program
- 68-pin PLCC package

Specific features of the PCB5011

- A second 16-bit bidirectional parallel data port with a 9-bit address port
- Direct access to 1024 × 40-bit external program memory (or 64 K × 40-bit when some external logic is added)
- 144-pin grid array package

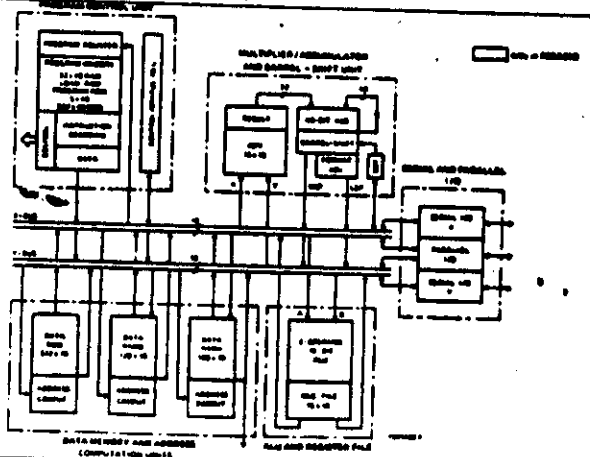


Fig. 1 Simplified block diagram of the PCB5010 and PCB5011 illustrating the double bus structure, powerful arithmetic units and on-board memory.

MARKET APPLICATIONS

- Radar/Sonar
- Image Processing
- Communications
- Image/Data Compression
- Spectral Analysis
- Speech Processing

FUNCTIONAL APPLICATIONS

- 1-D and 2-D FFT
- 1-D and 2-D DCT
- Auto/Cross Correlation
- Convolution/Filtering
- Modulation/Demodulation
- Vector Multiply/Add

PERFORMANCE BENCHMARKS

Function	time (μsec)
1024-point block-floating complex FFT	
—Single VSP	3300
—Two parallel VSPs	1900
—Four parallel VSPs	1200
1024-point integer complex FFT	2600
128-point block-floating complex FFT	237
16 x 16 DCT	1100
8 x 8-point 2-D complex FFT	164
64 x 64-point 2-D block-floating complex FFT	18000
128-point x 128-point complex vector multiply	53
128-point magnitude-square/accumulate	26
128-point complex modulation or demodulation	52
4 x 4 matrix multiplication	33

DESCRIPTION

The ZR34161 Vector Signal Processor (VSP™) is a member of Zoran's family of high-performance Systems Processors™. The VSP is a 16-bit processor which introduces algorithm-level and vector-oriented instructions to digital signal processing (DSP) system design. It functions as the key element in high-performance DSP applications. In coordination with a host controller, the VSP performs computation-intensive tasks while making minimum demands on host resources and system I/O capacity.

The block diagram shown in Figure 1 illustrates a system which is useful in a number of different applications, such as doppler frequency estimation or "zoom" FFT spectral analysis. It also serves to illustrate the processing power of the VSP. This type of high-performance system is implemented very efficiently by the VSP; most of the blocks in the diagram can be implemented using a single instruction.

WHY THE VSP IS UNIQUE

The VSP achieves its high performance through a number of unique architectural features. Firstly, the VSP uses a "high-level", vector-oriented instruction set; for instance, "FFT" is a single instruction within the VSP. These types of instructions can greatly simplify the amount of programming effort required to implement signal processing algorithms. Secondly, the VSP provides a block floating-point arithmetic capability which will help retain the original dynamic range of an input signal when performing FFT operations. This typically provides dynamic-range performance significantly greater than that of 16-bit fixed-point integer machines. Finally, the nature of its architecture and instruction set allows additional VSPs to be paralleled on the same bus to increase the signal processing throughput well beyond that of a single VSP.

GENERAL FEATURES

- High-performance 16-bit digital signal processor
- Architecture optimized for DSP operations
- High-level Vector-oriented instructions
- Block floating-point arithmetic for FFT
- Concurrent I/O and arithmetic processing
- Easily paralleled for greater throughput
- Fabricated in two micron DLM CMOS
- < 300mW power dissipation
- Powerful hardware and software development tools

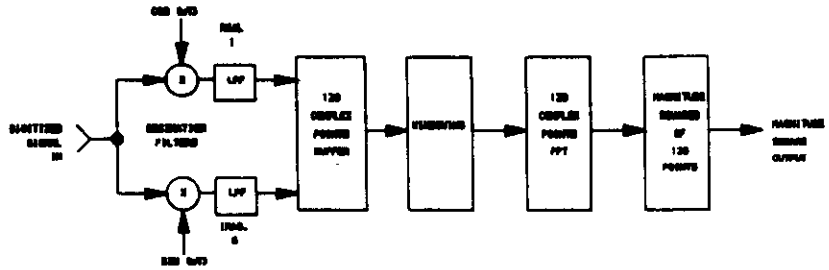


FIGURE 1. BLOCK DIAGRAM OF GENERAL COMPLEX DEMODULATION PROCESS IMPLEMENTED EFFICIENTLY BY THE VSP.

DSP IN ACCELERATOR CONTROL

Diagnostic and corrections on operating parameters of an Accelerator require the analysis and processing of analog signals.

Sophisticated signal processing functions can be implemented using digital techniques

Digital systems are inherently more reliable, more compact, programmable and less sensitive to enviromental conditions and component ageing than analog systems.

Depending on whether the parameter to control has or does not have a correlation with other signals, then we may see requirements to inplement a Real-Time Control Function for:

- a stand alone signal filtering or frequency analysis, or

- a linked DSP system that monitors and corrects parameters that have some degree of interdependence.

What then could be the reality for the future in the short term ?

STAND ALONE SYSTEM
(for a short-term solution)

for very high performance Real-Time applications

- Based on VME

- DSP: INMOS A-100 or
  Motorola DSP56000+DSP56200

- Interfaced to VAX, uVAX (DRQ11)
  or to IBM-PC
  or UNIX drivers for the Loughborough
  VME56k board.

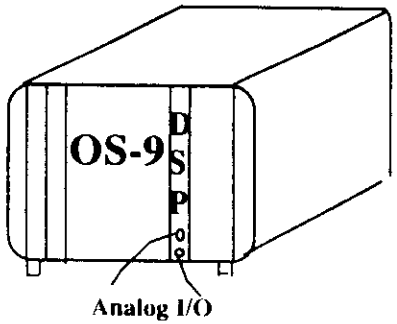
Present hardware and software standardization in the Accelerator Control.

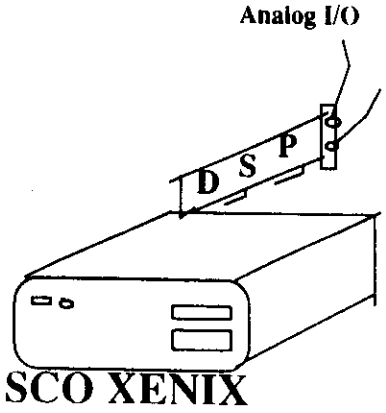
- OS-9 VME
- SCO XENIX for IBM PC AT

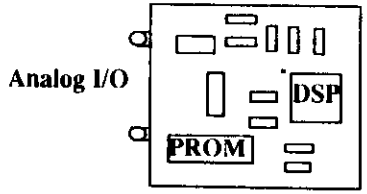
DSP

STAND-ALONE SOLUTIONS

IN THE ACCELERATOR CONTROL SYSTEM







HOW DOES THE DSP FIT IN WITH THE ACCELERATOR CONTROL SYSTEM ?

- STAND ALONE SYSTEM
  - Transp.-DSP
- NETWORK SYS.
  - RISC-DSP

Is there an aperture to the following hard-ware and software systems in the future ?

- VME - VXI
- UNIX
- ORKID (Open Real-Time Kernel Interface
- . Definition)
- (Occam, Helios ??)

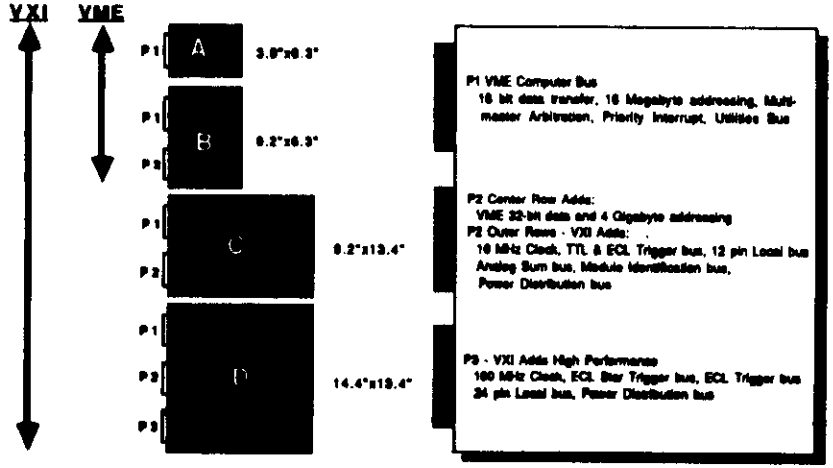


FIGURE 2
The VXIbus Specification uses the VMEbus specification as a base and adds two board sizes and backplane instrumentation signals.

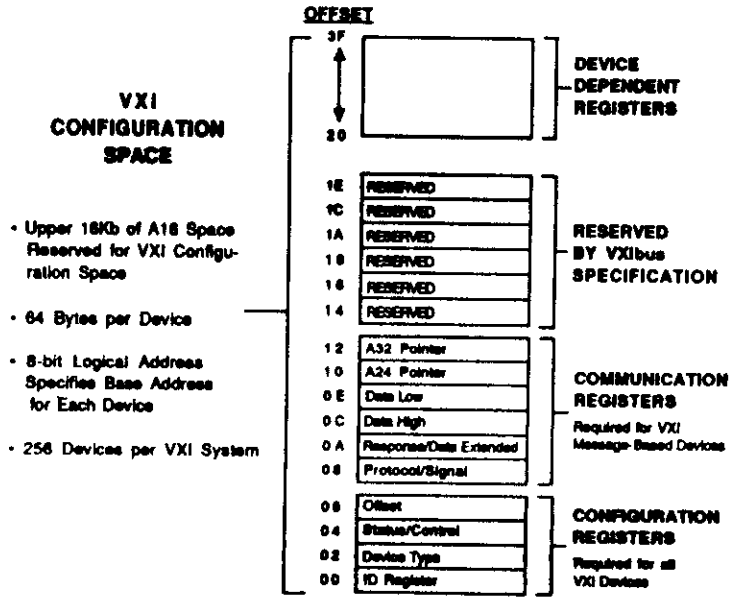


FIGURE 3
VXIbus modules must have a minimum set of registers located at specific addresses A module with only configuration registers is called a Register Based Device while

How to Get a Copy of the Draft

- leave your name with L. Hevle
- or contact VITA
- in Europe:

  VITA Europe
  P.O.Box 192
  NL-5300 AD Zaltbommel
  The Netherlands

- in the US:

  VITA
  10229 N. Scottsdale Rd., Suite E
  Scottsdale, AZ 85253-1437
  USA

- price for a single copy:

MCF-256 FUNCTIONAL SPECIFICATIONS

The following highlights the functional specifications of the MCF-256 system

NO. OF INPUTS (NC)	1-512
NO. OF TAPS/CHANNEL (NT)	4-4096
NO. OF DECIMATION (ND)	1-1024
CHANNEL SAMPLING RATE (fs)	see table 2
INPUT DATA WORD LENGTH	16-Bit
COEF. DATA WORD LENGTH	16-Bit
OUTPUT DATA WORD LENGTH	16-bit Mantissa and 4-bit Exponent
HOST PROGRAMMABLE PARAMETERS	NC,NT,ND, Filter Taps Output Exponents, Coef. Memory Bank
HOST COMMUNICATION DATA RATE	125 K words/sec.
DYNAMIC RANGE	>90dB

Note that because of the internal memory size limitation, the following restrictions apply;

$NC \times NT \leq 65536$

$NC \times ND \leq 16384$

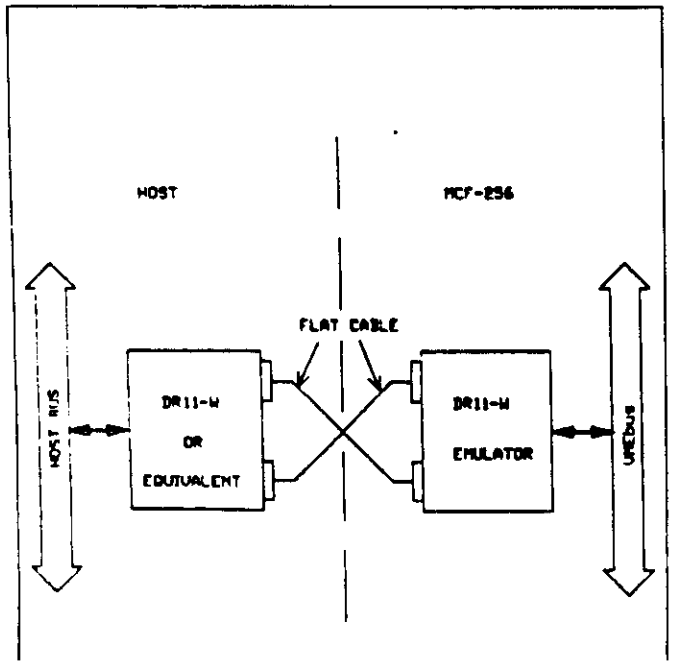
Table 2
MCF-256 Channel Sampling Rate
(No Decimation: ND=1)

No. of INPUTS (NC)	No. of TAPS/CHAN (NT)	MAX. SAMPLE RATE (MAX. fs)
1-16	4-4096	12.5/(NT)MHZ
17-32	4-2048	12.5/(2xNT)MHz
33-64	4-1024	12.5/(4xNT)MHz
65-128	4-512	12.5/(8xNT)MHz
129-256	4-256	12.5/(16xNT)MHz

Note: For ND > 1, the input sample rate is ND x fs. For example, with decimatio 256 channels and 256 taps per channel the maximum sample rate is 6KHz.

The MCF-256 system includes a VMEbus DR-11 emulator board for interfacing to a host computer (MICROVAX, VAX, or PC AT). The host sends commands and data to the MCF-256, and also receives data from the MCF-256 via a 16-bit parallel link. The link is terminated at both ends by a DR-11 or equivalent card as shown in Fig. 8. All communication is initiated by the host computer. The MCF-256 system only responds to the host requests. All transfers over the link are DMA transfers for high speed communication. Each transfer is initiated and terminated by interrupting the receiving device. All communications are controlled by a link driver software. A library of user callable Fortran subroutines have been developed for the VAX and PC AT host computer in order to communicate with the MCF-256 system.

All parameters downloaded from the host are stored in the battery backed memory board. These parameters are then entered in the filter and the I/O boards. The CPU board holds the entire system software in PROMs. Fig. 9 shows a simple flow chart of the system firmware which shows the sequence of important operations following a power up.



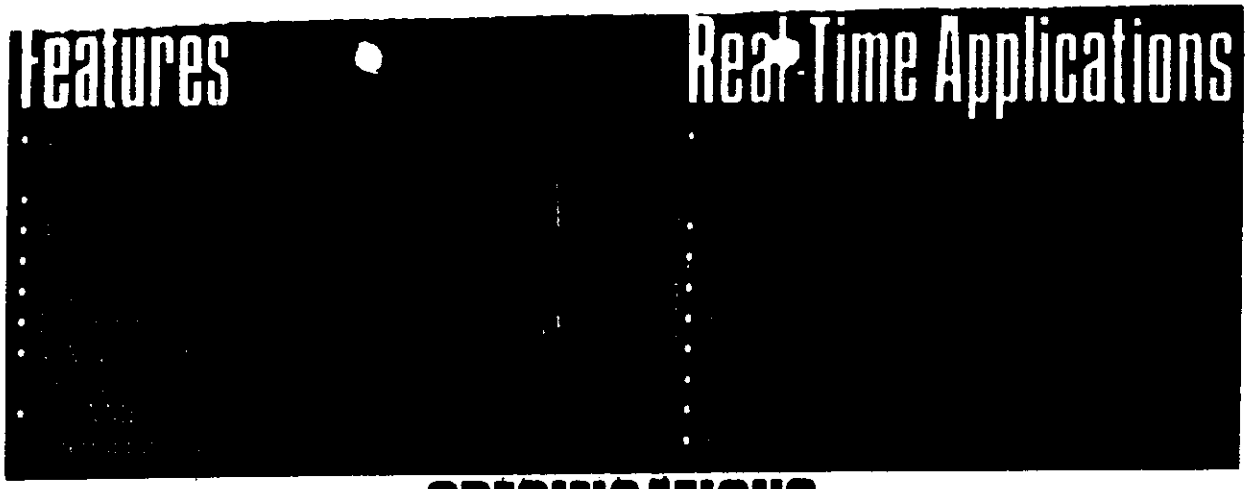
HOST COMMANDS

The host software developed for the MCF-256 system allows the host to download and upload all system parameters, run off-line diagnostics and read system status by using a set of simple commands. A list of important commands are given below:

	COMMAND	ACTION
1	MCF_SET_CONFIG	configure the MCF-256 system in terms of No. of Inputs, No. of Taps, No. of Decimation and Sampling Frequency
2	MCF_SET_TAPS	download filter taps
3	MCF_SET_BANK	select the active coefficient bank
4	MCF_SET_EXPONENT	set the filter output exponents
5	MCF_SET_ANALOG	set analog outputs and the cut-off frequency of the reconstruction filters
6	MCF_SET_DIAGNOSTICS	run off-line diagnostics
7	MCF_GET_CONFIG	return current configuration
8	MCF_GET_TAPS	upload filter taps
9	MCF_GET_BANK	return active coefficient bank number
10	MCF_GET_EXPONENT	upload filter output exponent values
11	MCF_GET_ANALOG	upload analog channel numbers and the cut-off frequency of the reconstruction filters
12	MCF_GET_STATUS	return the system status register
13	MCF_GET_OFF_ERR	return off-line diagnostics error codes
14	MCF_GET_ON_ERR	return on-line diagnostics error codes

Features

Real-Time Applications



SPECIFICATIONS

(Typical at 25° C)

INPUT SECTION:

No. of Analog Inputs	4 (Simultaneous Sampling)
Input Resolution	12-bit (2's Complement)
Max. Sampling Rate	500 KHz, Single Channel 125 KHz, Four Channel
Input Impedance	>5MΩ
Input Range	± 5V Full Scale

OUTPUT SECTION:

No. of Analog Outputs	4 (Simultaneous Updating)
Resolution	12-bit (2's complement)
Output Swing	+10V

DIGITAL FILTER SECTION:

No. of Digital Filter Channels	**4 for Real Data** **2 for Complex Data**
Input Data Word Length	**16-bit (2's Complement)**
Tap Data Word Length	**16-bit (2's Complement)**
Output Data Word Length	**16-bit (2's Complement)**
No. of Taps/Channel	4 to 255
Throughput Rate (N Taps, M Channel)	10/ [M (N+2)] MHz for Real Data 10/ [M (4N+4)] MHz for Complex Data
Decimation Interval (DI)	Max. 16 Filter Cycles or Sampling Periods
Interrupt Interval	Max. (255 x DI) Filter Cycles or Sampling Periods

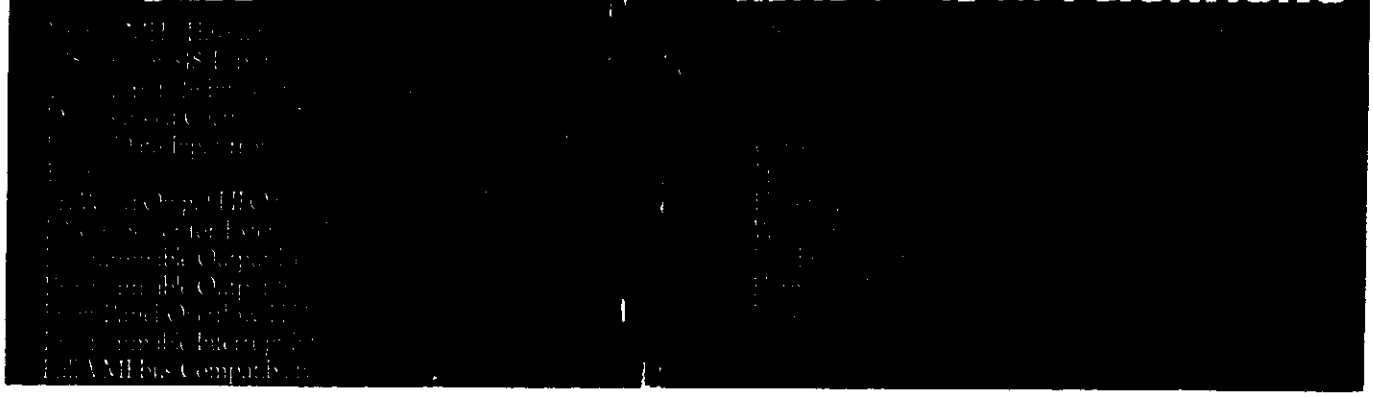
1 VME board = 3800 MIPS

Video Bandwidth FIR Filter

1 MIPS INMOS A-100 is not comparable to 1 MIPS Motorola 96000 or AT&T 3

FEATURES

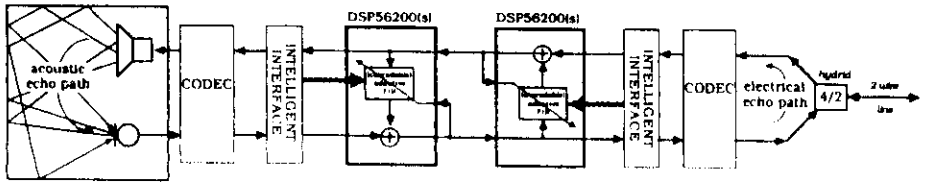
REAL-TIME APPLICATIONS



SPECIFICATIONS

Input Data Word Length	16-bit
Coef. Data Word Length	8, 12 or 16-bit
Output Data Word Length	24-bit (User Scaled)
Max. Throughput Rate	5, 3.3 or 2.5 MHz for 8, 12 or 16-bit Coefficients
Max. No. of Taps	128, 256 or 384 for the L, M or H Version
Decimation Interval (DI)	1-128 Sample Periods
Interrupt Interval	Up to 256xDI sample periods in steps of 2xDI
Power Input	+5V ±5% at 3.5A (H) 3.0A (M) 2.5A (L)
Operating temp.	Commercial Range
Board Size	233mm(W)x160mm(L)

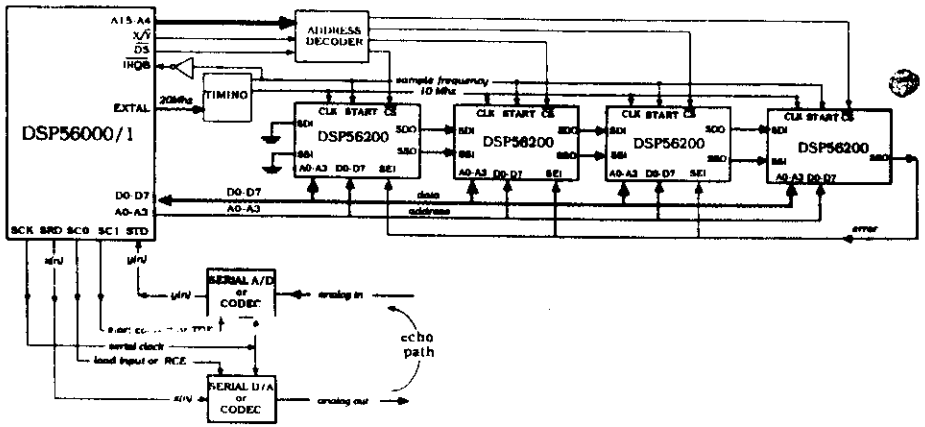
ACOUSTIC/TELEPHONE ECHO CANCELLER (A/TEC)



ACOUSTIC ECHO CANCELLING

ELECTRICAL ECHO CANCELLING

FOUR-CHIP ADAPTIVE FILTER SYSTEM



One DSP56000/1 & Four DSP56200s implement a 1024 tap echo-canceller up to 19 khz or a 128 tap echo-canceller up to 120 khz

DSP56200 PERFORMANCES

Maximum sampling frequency is a function of the number of taps and of the number of DSP56200's used.

MODE	TOTAL NUMBER OF TAPS					
	32	64	128	256	512	1024
FIR FILTER • Standalone • 4 chip in cascade	227khz	132khz	71khz 222khz	37khz 132khz	71khz	37khz
ADAPTIVE FIR • Standalone • 4 chip in cascade	123khz	69khz	37khz 120khz	19khz 69khz	37khz	19khz
DUAL FIR FILTER • Standalone • 4 chip in cascade	122khz	68khz	36khz no cascade in dual mode			

The DSP56200 is a 28 pin algorithm specific, digital signal processing peripheral designed to perform computationally intensive tasks associated with digital filtering. A flexible chip-cascading scheme enables the user to build filters with extended tap lengths and/or increased speed. Its performance, features and simple interface make the DSP56200 a natural solution for problems such as echo cancelling, telephone line equalization, noise cancelling, conventional filtering and many other DSP applications.

The high performance 10.25 MHz internal operation of the DSP56200 allows many DSP algorithms to be implemented in one chip.

The core of the chip consists of a 16 x 24 → 40-Bit multiply accumulation unit, 256 x 16-Bit data RAM and 256 x 24-Bit coefficient RAM. The interface is extremely versatile providing an 8-Bit I/O port with 4 address pins and 3 control pins, and 5 serial I/O pins to allow true cascadability.

The operating modes, set by the user, determine how the DSP56200 will operate. The key features of the mode-select are:

- single channel
- dual channel (non-cascadable, non-adaptive)
- stand alone/cascadable
- fixed coefficients/LMS coefficient update

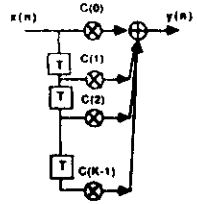
With a programmable tap-length of up to 256 taps in single channel mode and 128 taps in dual channel mode the DSP56200 offers a very high degree of user flexibility. Features such as programmable gain and leakage, as well as a DC tap option further highlight the chip's capability. The adaption algorithm can be disabled during "double-talk" situations allowing the coefficients to remain constant, and indeed unused coefficient and data memory is available as "scratch-pad" memory. The DSP56200 utilizes an ultra-low power stand-by mode to reduce power consumption.

FIR FILTERS: FINITE IMPULSE RESPONSE FILTERS

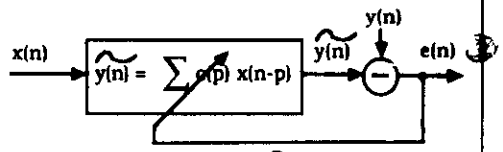
The filtered output signal y(n) is obtained by weighted summation of a finite set of input samples. Coefficients of the weighted sum constitute the impulse response of the filter.

FIR filters can be linear phase filters and this structure is generally chosen for adaptive filtering.

$$y(n)=\sum_{p=0}^{k-1} c(p)x(n-p)$$



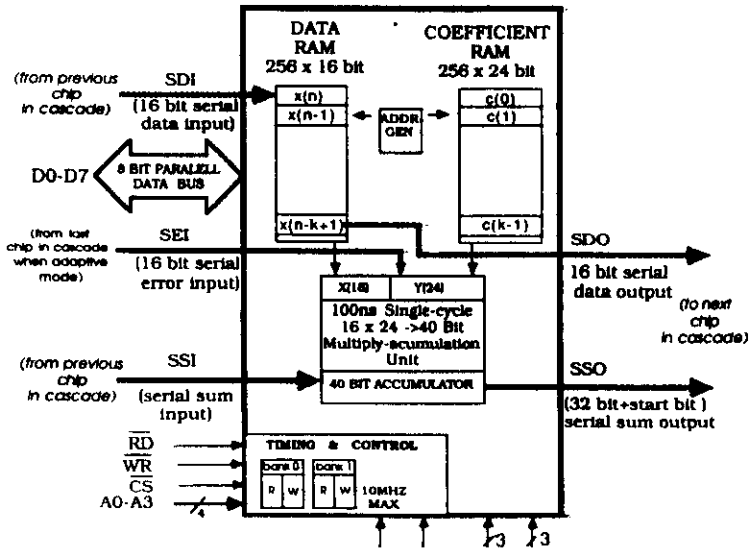
ADAPTIVE FILTERING



The filter approximates a signal y(n) with the input sequence x(n). Using the error e(n) between the filtered output y(n) and the desired signal y(n), an adaptive algorithm adjusts the filter coefficients, altering its response in order to minimize a measure of the error. In the DSP56200, coefficients are adapted by the Least Mean Square (LMS) algorithm.

$$c_{n+1}(p) = c_n(p) + K\,e(n)\,x(n-p) + \mathrm{sgn}\,(c_n(p))L$$

DSP56200 GENERAL BLOCK DIAGRAM



each device can
ution time. Some
ad of looping) for
verhead looping.
relatively small,
e for in-line code

ry use
hods to minimize
line code to keep
DSP320C10 from
liate-multiply in-
of the operands
ed the full 16-bit
tialization. Those
e 16-bit word use
d do not consume
e any extra mem-
are embedded in
difference in this
entations is the
64-tap FIR filter,
t coefficients out
fficients required
ld jump from 178

ble of performing
ssistance of some
nal memory, ad-
TMS320C10 and
address counters
both set those
h time external
ically increments
peration. This is
ave limited inter-
ny external data
of address count-
r presence when
calculating large

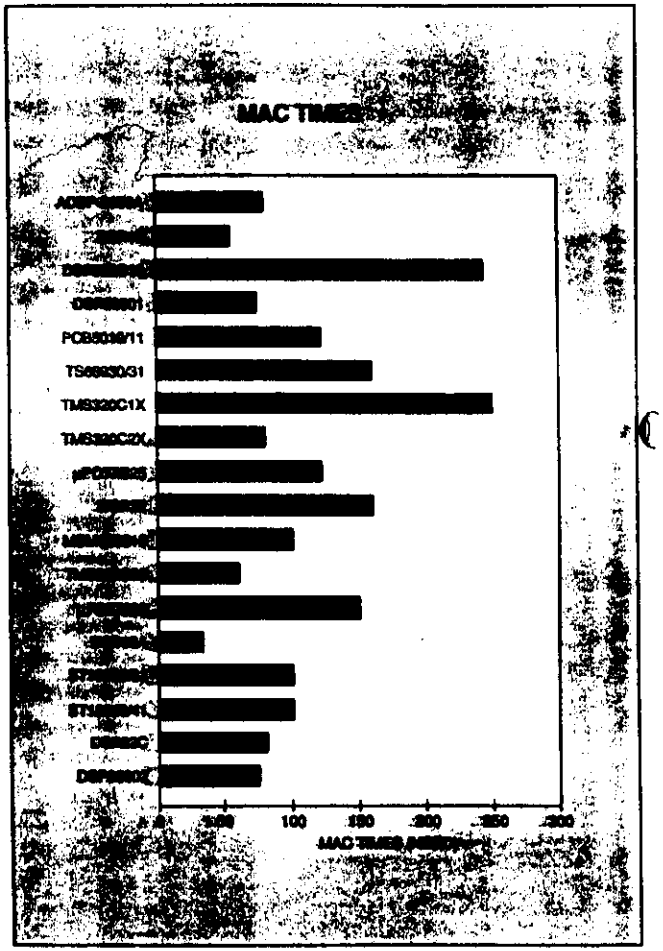
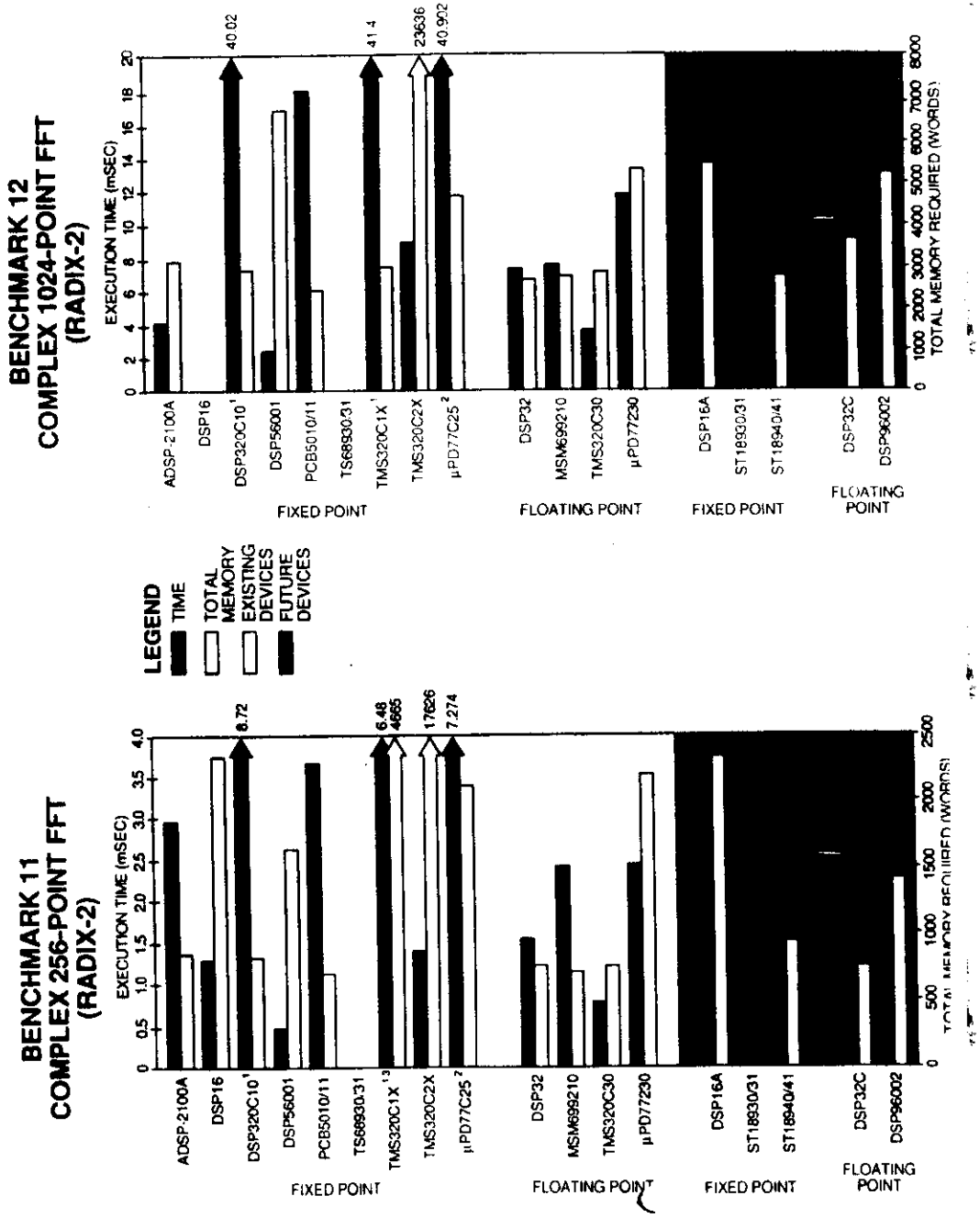


Fig 1—The multiply-accumulate (MAC) times for the DSPs match the execution times for the filters. Filters, after all, are largely a series of MACs.

quickly. SGS-Thomson's ST18940/41 has a complex MAC instruction that executes in two cycles. It also has on-chip bit-reversal and three on-chip data memories. The Motorola DSP56000/1 and DSP96001/2 also have three separate memories and hardware bit-reversal. They also have special instructions (ADDR, ADDL, SUBR, and SUBL) written specifically for FFTs.



This benchmark will perform a complex, radix-2 FFT with complex data that is already available in memory. The FFT's results will be a set of complex values with bits in normal order (include bit reversal in the benchmark). The routine may destroy your original data during the processing tasks. The size of this FFT is 1024 points. Precalculation of coefficients and lookup tables are allowed but their size must be included in the memory size. For example, if you use a lookup-table algorithm for bit-reversal, the number of words occupied by the lookup table must be included in the memory size you report.

This benchmark will perform a complex, radix-2 FFT with complex data that is already available in memory. The FFT's results will be a set of complex values with bits in normal order (include bit reversal in the benchmark). The routine may destroy your original data during the processing tasks. The size of this FFT is 256-point. Precalculation of coefficients and lookup tables are allowed but their size must be included in the memory size. For example, if you use a lookup-table algorithm for bit-reversal, the number of words occupied by the lookup table must be included in the memory size you report.

Application Note

AT&T

Implementation of Digital Filters With the *WE*® DSP32 Digital Signal Processor

Contents

Contributed by: *J. Tow*

Introduction

Filtering is one of the most commonly used operations in signal processing. In many applications, the use of digital filters is preferred to analog filters. Among the advantages of digital filters are:

- Digital filters can be designed to meet more stringent requirements
- Digital filters offer high reliability, high performance, and high accuracy
- Digital filter characteristics do not change with temperature and time
- Programmable digital filters can be easily modified to meet changing filter characteristics.

Implementation of digital filters, however, was difficult and expensive until the recent advances in VLSI technologies and the availability of programmable single-chip digital signal processors e.g., the *WE* DSP32 Digital Signal Processor [1].*

In this application note, digital filtering is reviewed first from the concepts of the more familiar continuous time domain analog filters. Described next are the two common types of digital filters, the finite impulse response (FIR) and the infinite impulse response (IIR) filters. This is followed by a discussion of the practical considerations on the implementation of these filters and, in particular, the advantages of using floating-point processors over fixed-point processors. The remainder of the application note covers

* [ ] indicates a reference listed at the conclusion of this application note.

the major topic of implementing these digital filters with the DSP32.

The DSP32 is a single-chip 32-bit floating-point programmable digital signal processor that is capable of executing 6 million instructions per second, or 12 million floating-point operations per second with a 25 MHz DSP32 input clock. These same filter routines can also be used with the DSP32C Digital Signal Processor which is capable of executing 12.5 million instructions per second, or 25 million floating-point operations per second with a 50 MHz input clock. Complete FIR/IIR filtering routines using the DSP32 are given. These routines are chosen for their efficiency in minimizing memory requirement or execution time. The routines are written in such a way so as to facilitate their incorporation with a minimum of effort into the users' application programs. Readers who have prior knowledge of digital filtering can go directly to the sections on the DSP32 implementation.

Overview of Digital Filtering

A system for digital filtering of analog signals is shown in Figure 1. The incoming analog signal is first passed through an analog lowpass filter, known as an anti-aliasing filter. This filter bandlimits the frequency components of the input signal to less than one-half of the sampling rate in order to minimize errors due to frequency aliasing. The filtered signal is then sampled by the A/D converter to produce a sequence of quantized samples or digital words. The sequence of digital words can be operated upon numerically by a digital processor (under software control) to produce another sequence of digital words. The altered sequence of digital samples is converted back to an analog signal by the D/A converter. The final analog lowpass filter, or reconstruction filter, is needed to compensate for the characteristics of the D/A converter and to recover a smooth analog output signal.

The operation of the digital signal processor as described above implements the system transfer function by converting an input sequence into another sequence by means of some numerical computation algorithms. The computation algorithms are called digital filters when the system transfer function corresponds to filtering functions.

For more detailed discussion, readers are referred to the many standard texts on digital signal processing [2,3].

Digital FIR/IIR Filters

Most digital filtering functions can be accomplished by programming the digital signal processor to implement the following difference equation.

$$1)\quad y(n) = -\sum_{k=1}^{N} d[k]\cdot y(n-k) + \sum_{k=0}^{M} n[k]\cdot x(n-k)$$

where x(n) is the input sequence and y(n) the output sequence, and d[k] and n[k] are constant coefficients.

Equation 1 implies that the present output value y(n) can be computed from the present and past M input values and past N output values. If past output values are actually used in the computation of the present output, i.e., if the filter implementation contains feedback, then the implementation is said to be recursive. Otherwise, the filter implementation is nonrecursive.

An alternate representation of equation 1 can be derived by taking the z-transform [2] of both sides of equation 1, which produces

$$2)\quad H(z) = \frac{Y(z)}{X(z)} = \frac{\sum_{k=0}^{M} n[k]\cdot z^{-k}}{1 + \sum_{k=1}^{N} d[k]\cdot z^{-k}}$$

where X(z) and Y(z) are the z-transforms of x(n) and y(n), respectively. H(z) is called the system function (or transfer function).

FIR Filters

For FIR filters, all d[k]'s in equation 1 are zero, hence

$$3)\quad y(n) = \sum_{k=0}^{M} n[k]\cdot x(n-k)$$

where M+1 is the length of the FIR filter. The computation of each value of the output sequence requires M+1 multiplications and M additions. Since y(n) does not depend on the past output values, the FIR filter is of the nonrecursive type. It is readily seen that the coefficients, n[k]'s in equation 3, are

Figure 1. Digital Filtering of Analog Signals

the unit-pulse response of the filter; i.e., they correspond to the output sequence y(n) when the input to equation 3 is 0 for all x(n) except x(0)=1. Therefore, the impulse response (equivalently, unit-pulse or unit-sample response) of a FIR filter is of finite duration.

The system function for the FIR filter is:

$$3a)\quad H(z) = \sum_{k=0}^{M} n[k]\cdot z^{-k}$$

Some of the properties of the FIR filters are:

- FIR filters can be constrained to be linear phase or linear plus 90 degrees phase response, corresponding to even or odd symmetry, respectively, in its impulse response. Linear phase filters are important in applications where frequency dispersion due to nonlinear phase is intolerable, e.g., in data communication and speech processing.
- FIR filters have no poles or feedback paths, hence their realizations are inherently stable.
- FIR filters require long filter length to approximate sharp frequency selective filters; hence, they require more computational effort than other filter types, e.g., the IIR filters.
- FIR filters have no analog counterparts; hence, no analog filter design techniques can be used. Several design methods are available: the window function technique, the frequency-sampling technique, and the equiripple design or Parks-McClellan design [2,3]. These iterative techniques, however, allow FIR filters to approximate rather arbitrary frequency response characteristics, compared to those obtainable from the classical analog filter design techniques.

IIR Filters

For IIR filters, at least one of the coefficients d[k]'s in equation 1 is nonzero. Hence the computation of the output sequence depends on some past output values and, therefore, the IIR filter is of the recursive type. Furthermore, because of the recursion, it can be readily shown that the unit-pulse response of an IIR filter is of infinite duration.

Some of the properties of IIR filters are:

- With both poles and zeros, IIR filters can approximate closely a quickly varying magnitude response characteristic with many fewer terms than FIR filters and are therefore more efficient in some applications. It is difficult to maintain phase linearity in an IIR filter, so IIR filters find applications where good phase response is of secondary importance, such as in voice and sound communications.
- Due to the recursive or feedback nature, IIR filter implementations can become unstable. This is especially true with a fixed-point arithmetic implementation.
- IIR filter designs are usually obtained by a transformation (e.g., the bilinear z-transformation) of their counterpart analog filter designs [2,3]. Hence, no new techniques need to be learned.

Block Diagram Representation of FIR/IIR Filters

Block diagrams are useful for depicting the computational procedure to implement a digital filter. Figure 2 shows a network structure whose input and output satisfy the difference equation (equation 1) and the system function (equation 2). The constant coefficient multipliers n's and d's are shown as branch transmittances. The figure assumes M=N. If not, then some of the branch transmittances can be equated to zero. For example, in FIR filters, all of the d[k]'s are zero. Hence, the left column of adders can be removed. Since the coefficients in equation 1 and the transmittances in the network structure in Figure 2 have a one-to-one correspondence, the structure is called the direct form implementation. More precisely, Figure 2 corresponds to the direct form II structure.

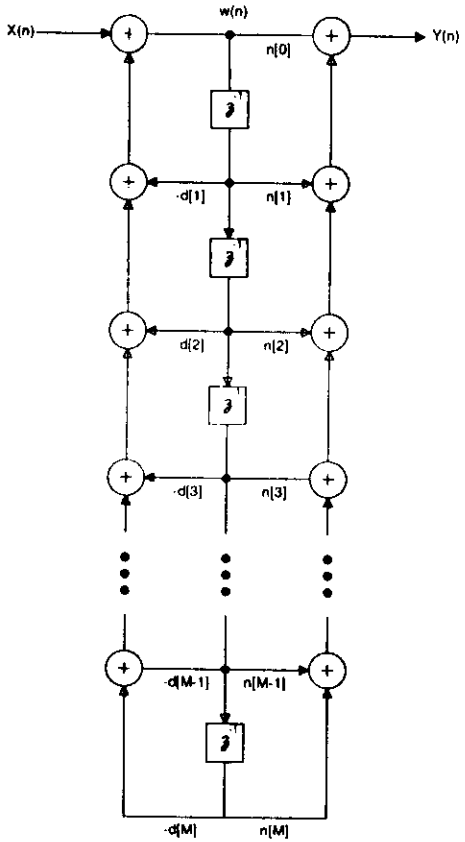


Figure 2. Direct Form II Structure

Many other network structures are available for a given system function. Our emphasis is to deal with the most practical and commonly employed structures. Of these the direct form I (or transposed form) is shown in Figure 3. The transposed form is obtained from the direct form II by applying the transposition theorem (Chapter 4 of [2]).

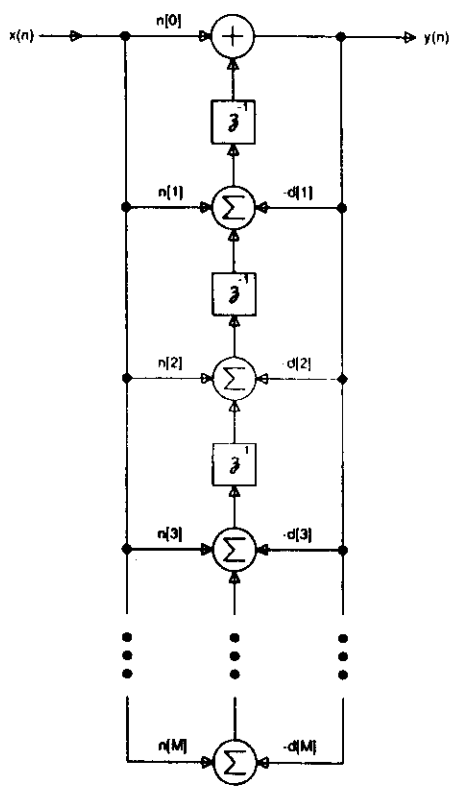


Figure 3. Direct Form I (or Transposed) Structure

Cascade Second-Order Direct Form Representation of IIR Filters

As shown in later sections, the DSP32 implementation of FIR filters employs the direct form II structure or, equivalently, a direct computation of the convolution sum of equation 3. On the other hand, high-order direct forms are seldom used in the implementation of IIR filters.

Instead, the cascade of second-order direct forms are used. In the cascade form, the system function (equation 2) is represented by a product of second-order terms as follows:

$$4)\quad H(z) = \frac{Y(z)}{X(z)} = \prod_{i=1}^{N} K_i \cdot \frac{n[i,0] + n[i,1] \cdot z^{-1} + n[i,2] \cdot z^{-2}}{1 + d[i,1] \cdot z^{-1} + d[i,2] \cdot z^{-2}} = K_1 \cdot \prod_{i=1}^{N} \frac{n[i,0] + n[i,1] \cdot z^{-1} + n[i,2] \cdot z^{-2}}{1 + d[i,1] \cdot z^{-1} + d[i,2] \cdot z^{-2}}$$

The K_i's are scaling factors for the sections. They are chosen to avoid overflow and to maximize signal-to-noise ratio.

Note that in the actual implementations, the K_i's are absorbed into the second-order sections as shown in the second equation in equation 4. In particular, for direct form II, the K_i's are absorbed into the preceding section; hence K_1 is usually not equal to 1. For direct form I (transposed form), the K_i's are absorbed into its corresponding section, hence $K_1 = 1$. As an example, Figures 4 and 5 show, respectively, the cascade structures for the direct form II and I where M=N=4.

Practical Considerations in Digital Filter Implementation

Before implementing a digital filter, the structure and coefficients must be determined from the filter specifications. Digital filter design involves obtaining the set of coefficients n[k] for the FIR filter, or n[k] and d[k] for the IIR filter (equation 1). Equivalently, for the cascade second-order IIR filter structures, the set of coefficients are K_1, n[i,0], n[i,1], n[i,2], d[i,0], and d[i,2].

As previously mentioned, many design techniques are available for designing digital filters [2,3]. Computer programs are also available for these designs [4–8]. As an example, [6] describes a digital filter design package (DFDP) from Atlanta Signal Processor, Inc. (ASPI). The package consists of modules for designing Butterworth, Chebyshev, and Elliptic IIR filters and linear phase FIR filters using Kaiser window and Parks-McClellan method. The IIR filter designs are suitable for direct form I implementation. On the other hand, the filter design program of [8] can be used to design both cascade direct forms I and II IIR filters. These interactive design programs run on the AT&T PC6300 Personal Computer and other IBM PC-compatible personal computers.

For hardware prototyping of digital filters, the section titled "Prototyping Digital Filters With the *WE* DSP32-DS Digital Signal Processor Development System" describes the procedure for obtaining and downloading the filter design data generated from DFDP and the FIR/IIR filter routine code to the development system [9] or a commercially available DSP32 PC6300 plug-in card. This capability allows digital filter designers to achieve a quick turnaround time from filter specification to hardware prototype, and permits the freedom to explore the many tradeoffs in implementing digital filters.

There are several important considerations when implementing recursive IIR filters with finite precision arithmetic digital signal processors. These considerations have led to the general practice of implementing IIR filters using the cascaded second-order sections. In addition, a set of recommended procedures exists on how the poles and zeros should be paired to form the second-order sections, how these second-order sections should be ordered, how to assign scale factors to the sections, and often the number of bits required to represent the filter coefficients and/or the dynamic (or state) variables of the filter. An excellent treatment on these practical implementation considerations, and in particular for the fixed-point processors, can be found in Chapter 11 of [3].

The above practical considerations are of utmost importance for fixed-point arithmetic processors since their strict adherence in the implementation is the only way to guarantee the successful operation of the filter. However, the considerations have negligible effects on implementation using 32-bit floating-point arithmetic, such as with the *WE* DSP32 Digital Signal Processor. For IIR filter implementations, the 24-bit fractional part of the DSP32 corresponds to a signal-to-noise ratio of 144 dB, which is sufficiently accurate to neglect the filter coefficient quantization error. The 8-bit exponent field of the DSP32 has a dynamic range of over 1500 dB, which is large enough to eliminate signal quantization noise and overflow problems.

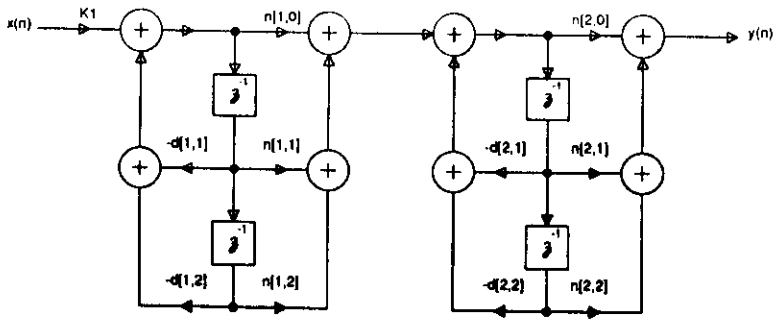


Figure 4. Cascade Direct Form II with M=N=4

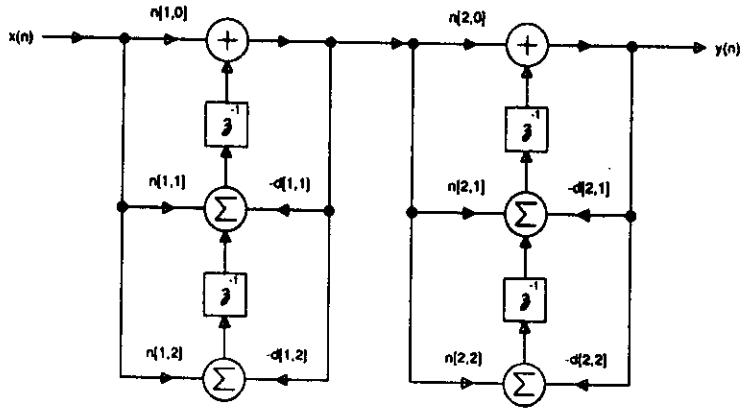


Figure 5. Cascade Direct Form I (Transposed) with M=N=4

WE DSP32 Digital Signal Processor Routine Coding Conventions

Review of Digital Filter Representation

An FIR filter is represented by the following convolution sum where a slight change of notation from equation 3 is employed:

$$5)\quad y(n) = \sum_{k=0}^{N-1} h[k] \cdot x(n-k)$$

where x(n) and y(n) are the input and output sequences, respectively. The filter length is N and h[k]'s are constant coefficients and are the sample values of the unit-pulse response of the filter.

A cascade second-order IIR filter is represented by

$$6)\quad H(z) = \frac{Y(z)}{X(z)} =$$

$$K_1 \cdot \prod_{i=1}^{N} \frac{n[i,0] + n[i,1] \cdot z^{-1} + n[i,2] \cdot z^{-2}}{1 + d[i,1] \cdot z^{-1} + d[i,2] \cdot z^{-2}}$$

where X(z) and Y(z) are, respectively, the z-transform of the input and output sequence, x(n) and y(n). H(z) is the system or transfer function.

For the transposed or direct form I, K_1 also equals 1. This representation requires 5 multiplications per section. Since scaling is not necessary with the 32-bit floating point DSP32 processor, a reduction from 5 to 4 multiplications per section can be realized by a normalization of all of the coefficients n[i,0] to 1. This effectively changes the overall constant multiplier K_1. In either case, i.e., whether n[i,0] equals 1 or not, equation 6 is used to represent the cascade second-order IIR filters with 4 or 5 coefficients per section.

Filter Routine Coding Conventions

Each implementation of the FIR/IIR filter in the DSP32 assembly language is given in the format of a callable routine. This convention facilitates its use in other application programs. Each routine includes two distinct portions of DSP32 instructions. The first portion consists of instructions for passing the arguments from the calling program to the routine. The second portion is the actual implementation of the filter computation. The arguments generally consist of the order of the filter (e.g., the length N of the FIR filter or the number of second-order sections of the IIR filter), the location of the filter coefficients, input, output, and intermediate variables. The intermediate variables are often called the state variables and are required in the calculation of subsequent output values. For the FIR filters, these variables are the (N–1) most recent input sample values.

The convention also assumes that upon returning from the routine call, all output values, as well as the static variables (unless otherwise noted for the latter), are available for immediate use by the calling program.

WE DSP32 Digital Signal Processor Routines for FIR Filters

Three DSP32 routines are described for the implementation of FIR filters. The routines differ in the amount of program storage required and the speed of execution. DSP32 application programmers can choose any of the routines to suit their particular needs.

For each of the routines, the following is given: the calling procedure from the main program, a description of the arguments, and the actual DSP32 code for the routine.

FIR Routine 1 (fir)

The fir routine uses a fixed amount of program memory space to implement an arbitrary length FIR filter. The routine code is listed in Program 1. Notice that line numbers, e.g., M1, M2, ..., 1, 2, ..., etc, are added in the first column to facilitate discussion in the text; they are not part of the routine.

Detailed Explanation of the fir Routine

An instruction-by-instruction description of the fir routine is given below. Readers who have prior knowledge of the DSP32 assembly language may want to skip over this section.

In the main program, instruction M1 calls the fir routine and stores the return address in the DSP32 control arithmetic unit (CAU) register r14. The nop following the subroutine call is a latent instruction that is executed before the branch is taken. Execution of the program then branches to line 1 of the fir routine. At this time, r14 stores the address of memory location M3. Since each DSP32 instruction requires 4 bytes, M3 equals the address at M1+8.

Loughborough Sound Images Ltd

DSP56001 VME BOARD

User Manual

Preliminary January 1989

UNIX drivers for VME56K.

The UNIX drivers support the following features.

- Select memory space (P, X or Y)
- Set memory address
- Read memory
- Write memory
- Start program running
- Stop program running
- Set handshake flag in interface
- Clear handshake flag in interface
- Read handshake flag in interface

These routines are all implemented via the host interface of the DSP56001 which is mapped as a slave port onto the VME bus. All transfers take place over the bus as byte transfers.

These drivers support the 'open', 'close', 'IOCTL', 'read' and 'write' funtions, which can be called from programs running under UNIX and SUN workstations.

2.5 DSP56200 Configuration

The board is supplied with two DSP56200 FIR filter devices. The DSP56200 can be configured as an FIR filter or as an adaptive filter. More than one DSP56200 can be cascaded to form a filter of more taps, at a given sample rate, than would be possible using just one device. The two devices supplied are mapped into the on board I/O peripheral space. The first device (A) occupies locations from Y:$FFD0 to Y:$FFDF and the other device (B) is at Y:$FFE0 to Y:$FFEF. Link LK19 can be used to select the type and length of the filters required (figure 2.10). In the single/cascaded FIR or single/cascaded adaptive mode device A has the first half of the coefficients and device B the second half. The START signal is produced from the on board sample rate generator.

1.6 Software

A monitor program is supplied that runs on an IBM-PC or compatible. Full details are in section 3. It controls the VMEbus based board via the RS232 link provided. It supports accesses to all available memory, including on-chip memory, and to on-chip registers. Programs can be run at full speed, up to breakpoints, or be single stepped. An option is available to time user code via the on-board interval timer. Further details in Section 3.
The monitor reserves for its use one address register set (R7,N7,M7), and program memory addresses $40 to $7F and $E000 to $E100.

If the Motorola software has been purchased, it includes their DSP56000 Macro Cross Assembler and DSP56000 Cross Linker. These provide the facility to process source program code and generate an object code file for use with the DSP56001. The simulator program is provided as a software tool to develop programs and algorithms for the DSP56001. The Motorola DSP56000 'C' Compiler software is available, and can be purchased as a separate item.

1.7 Additional Documentation

2.9.2 Sample Rates for Analog I/O

An on board sample rate generator or an external trigger source may generate the sample rate for the ADCs and DACs. The source for this trigger is selected with link LK13. If link Lk13a is inserted, and link LK13b removed the external trigger is selected. Otherwise, if link LK13b is inserted and link LK13a removed the on board sample rate generator is selected. On shipping the on-board sample rate generator is selected.

TRANSPUTER MODULES (TRAMs)

IMS B401 TRAM

FEATURES

- IMS T414 or IMS T800 transputer
- 32 KBytes no-wait-state static RAM
- Stackable
- Size 1

The IMS B401 (TRAM) is a low cost, high performance, high density, 16 pin transputer ideal for applications where 2 KBytes or 4 KBytes of on-chip RAM is not quite enough. The 32 KBytes of off-chip RAM is more than was sold on many PDP8's, and is ideal for systolic processing, signal processing, feature extraction etc. The IMS B008, fitted with ten IMS B401-3 TRAMSs, offers 40 MWhetstones/s in a single slot of an IBM PC (*). In the INMOS ITEM, 160 IMS B401-2s would offer the user 1.6 GIPS (1600 MIPS) and 5 MBytes.

(*) For IBM PC read. Original PC, XT, AT, PS2 Model 30 and most clones.

IMS B402 TRAM

FEATURES

- IMS T212 transputer
- 8 KBytes no-wait-state static RAM
- Stackable
- Size 1

The IMS B402 (TRAM) is useful for similar applications as the IMS B401 (TRAM), as well as communications. The faster multiplication of the 16 bit transputer and the faster external memory interface give the IMS B402 higher performance than the IMS B401 for fixed point signal processing and feature extraction. Even with programmed floating point, the IMS B402 delivers 127 KWhetstones/s. Occam makes it easy to program 16 bit integers on this 16 bit processor, for high precision arithmetic and operations on sets. The IMS B402 is also ideal for message switching, for intelligent control of IMS C004 link switches, and for protocol conversion.

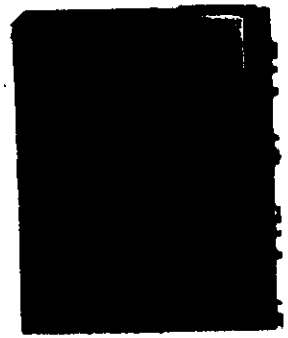
IMS B403 TRAM

FEATURES

- IMS T414 or IMS T800 transputer
- 1 MByte no-wait-state dynamic RAM
- Stackable
- Size 4

The IMS B403 (TRAM) provides the ultimate performance for a full 1 MByte of RAM, offering 4 MWhetstones/s with the IMS T800-20 transputer option. Rather than use expensive 256K SRAM's, which would take up much more board area, the IMS B403 uses 60 ns access time DRAM's, designed by INMOS.

Using IMS B403's, the INMOS ITEM (IMS B211) will hold 40 transputers, offering a total of 40 MBytes, 3 cycle memory and fully flexible network configurations.



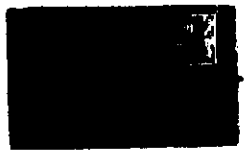
IMS B404 TRAM

FEATURES

- IMS T800 transputer
- 2 MBytes single wait-state dynamic RAM
- 128 KBytes zero wait-state static RAM
- Stackable
- Size 2

The IMS B404 (TRAM), of all INMOS board level products, has the highest packing density of silicon; 11 cm² on a board the size of a credit card. Its speed has been enhanced by extending the principle of fast on-chip RAM to include 128 KBytes of SRAM.

Four IMS B404s fit on to the IMS B008 in a single slot of the IBM PC (*). Eighty IMS B404's fit into an INMOS ITEM (IMS B211), to give 160 MBytes, 800 MIPS, 320 MWhetstones, with space to spare for other modules.



(*) For IBM PC read. Original PC, XT, AT, PS2 Model 30 and most clones.

TRANSPUTER BOARDS

The following products have been designed to provide the necessary tools to make evaluation and development of transputer systems inexpensive and straightforward.

All products come with [illegible] and run-time software for each item. INMOS customers receive their own personal registration number which they subsequently quote if requiring support from their local INMOS representative office. This ensures [illegible]

[illegible] products can be upgraded [illegible] INMOS and support terms.

Products available for transputer development fall into 3 major categories.

The Transputer Development System (T.D.S.)

FEATURES

- Fully integrated editor and development tools
- Transputer add in board
- Compiles and loads occam II for: evaluation boards other transputer networks
- All application development at occam source level
- Hierarchical program structure with separate compilation
- Source level network debugger
- Allows the inclusion of in-line transputer assembly code

Stand alone tool sets

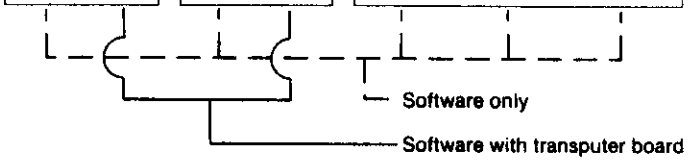
- Commands invoked from host operating system level prompt
- Supports mixed occam and scientific language compilations
- Permits multi-user access
- Simplifies version control management
- Allows the inclusion of in-line transputer assembly code

Scientific Languages

- C, PASCAL and FORTRAN compilers meeting industry standards are available for transputers.

Inmos Development Tools — Nomenclature

	T.D.S.		Toolsets		Scientific languages		
					C	PASCAL	FORTRAN
SUN	—	—	D505*	D525*	D511*	D512*	D513*
VAX	D600	—	D605*	—	D611*	D612*	D613*
IBM PC	D700	D701	D705	D725	D711	D712	D713
NEC PC	D700	D801	D705	D825	D711	D712	D713



* *Advanced information.*

Please contact Inmos representative for availability.

DEVELOPMENT TOOLS

Fast Digital Parallel Processing module

[redacted]

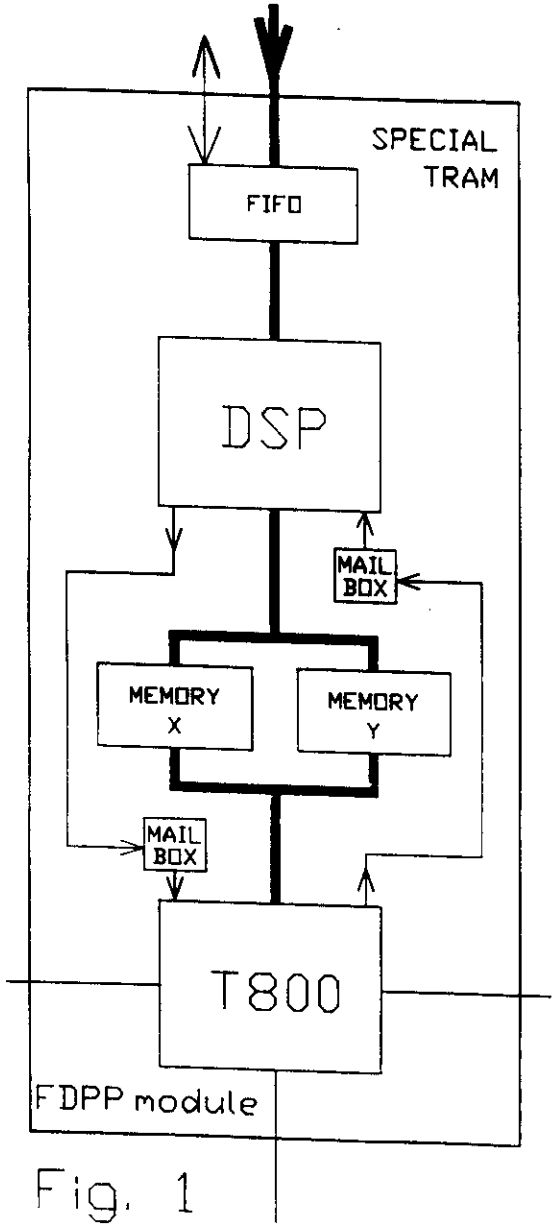
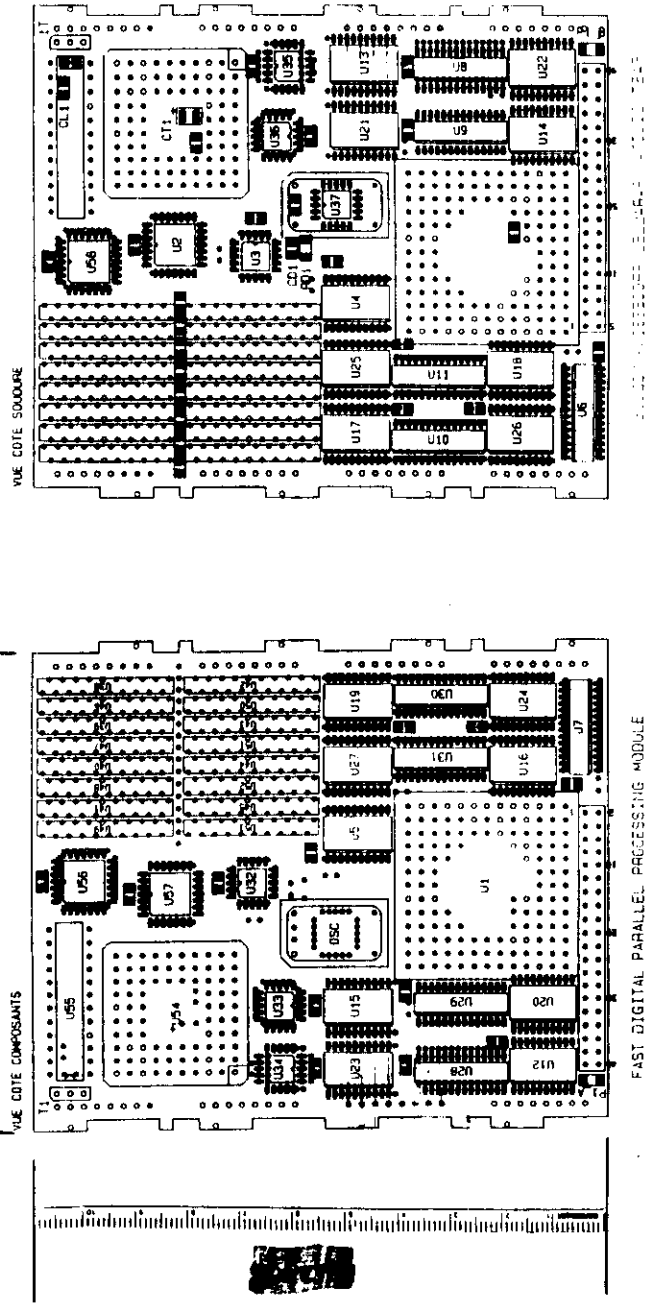


Fig. 1

Fast Digital Parallel Processing module

[redacted]



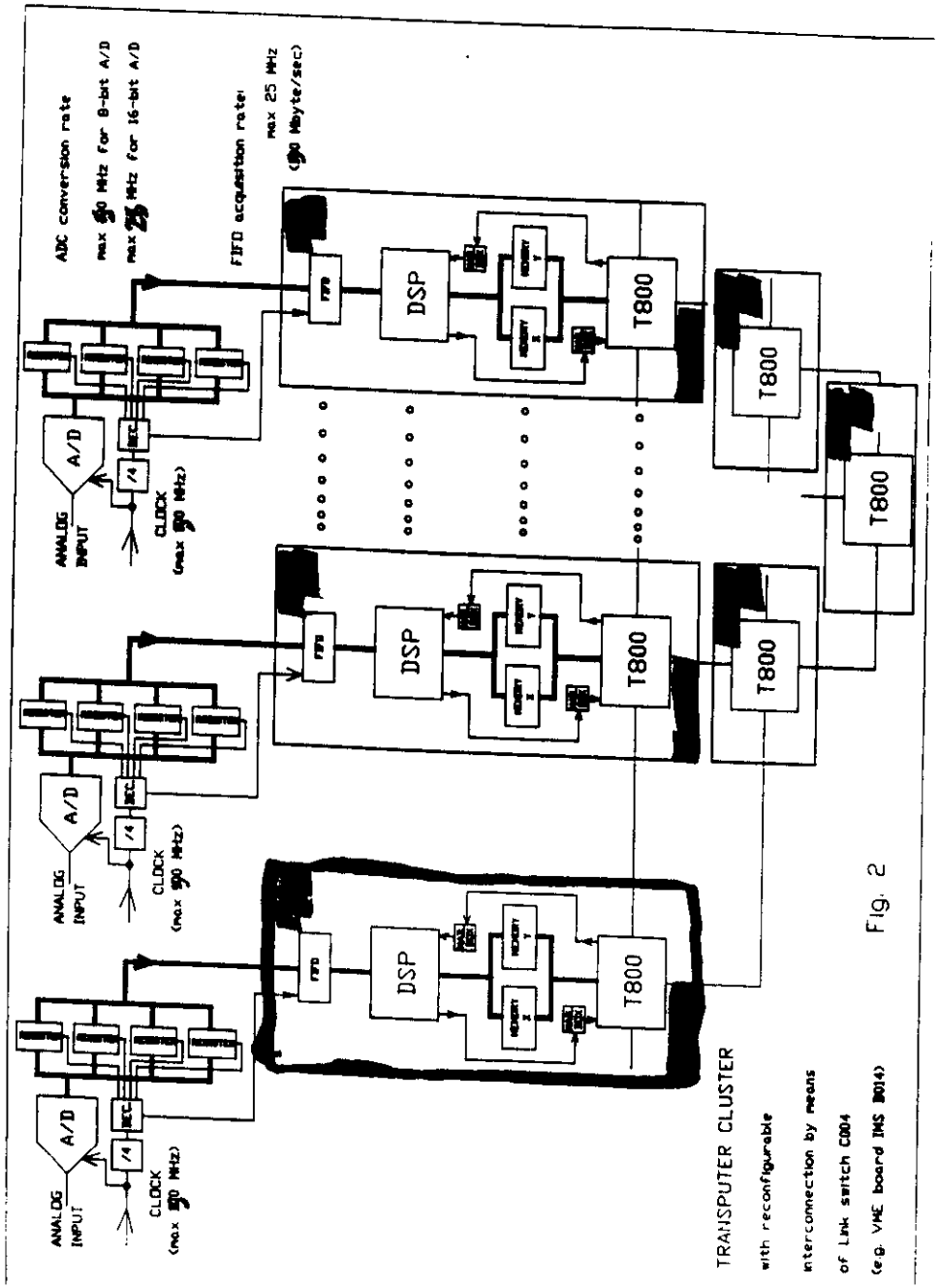


Fig. 2

In/Out	Function	Pin No.
	Power supply and return	3,14
in	5MHz clock signal	8
in	Transputer reset	10
in	Transputer error analysis	9
out	Transputer error indicator (inverted)	11
in	INMOS serial link inputs to transputer	13,5,2,16
out	INMOS serial link outputs from transputer	12,4,1,15
in	Transputer link speed selection	6,7
out	Subsystem reset	1b
out	Subsystem error analysis	1c
in	Subsystem error indicator	1a

Table 1: IMS B404 Pin designations

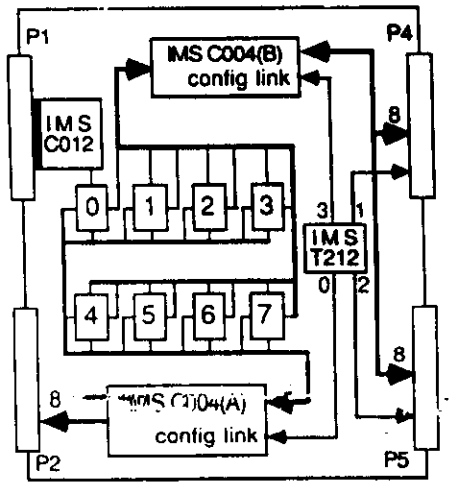


Figure 1: IMS B014 Block Diagram

FEATURES

- Compatible with VMEbus Specification Rev. C.1
- Accomodates 8 standard transputer modules (TRAMs)
- Static or dynamic link configuration using two IMS C004 link switches
- Expandable to form arbitarily large systems
- Suitable for use as VMEbus–transputer interface with IMS D505 SUN based development system

Figure 2.
8 x TRANSPUTERS & DSPs CLUSTER
in a single VME board

Fast Digital Parallel Processing module (FDPP)

The FDPP is equivalent to an INMOS "TRAM" B404 with in addition: a FIFO memory to input data from an analog data acquisition module and a DSP that is not intended to be used as a co-processor, but more likely as a pocket calculator to execute long expressions among a large set of data.

[redacted]

- DSP32C [redacted]
- T800 [redacted]

- DSP32C 6Kbytes of internal fast RAM

- 128kbytes x 2 of Dual-Port RAM (35nsec), switching technique

- T800 4Kbytes of internal RAM

- T800 2Mbytes of external dynamic RAM (100 nsec)

- 2K x 16bit of FIFO memory to buffer the input data

- [redacted] FDPP acquisition rate [redacted]

- [redacted] FDPP acquisition rate [redacted] in DMA without interrupting execution of programs on the DSP.

[redacted]

- 3 serial Link lines from the Transputer at 20 Mbit/sec

- 16 bit input to FIFO memory from an analog data acquisition module.

HANDSHAKE LINES:

-to FDPP

write to FIFO

reset FIFO

- from FDPP

FIFO full

reset analog data acq. module

[redacted]

- Initialization and monitoring of the DSP activity is done by the Transputer through the Parallel I/O of the DSP.

- Short messages are transferred using Transputer Link 0 at 20 Mbit/sec (MAIL BOX), Interrupt or Polling technique is provided.

- Large blocks of data are communicated using switchable memory banks (two banks of 128Kbytes each, switchable from the Transputer).

[redacted]

[redacted] (in DMA without disturbing execution of programs on the DSP and on the Transputer)

[redacted]

(DSP internal 6Kbyte of RAM is configured in mode 5)

extermal memory and I/O map

MEMX	200.000
MEMY	400.000
MAILBOX	800.000
FIFORESET	A00.000

[redacted]

The total memory available on the Transputer is 2 Mbytes,

- 128KBytes of External Static RAM and 4KByte of Internal Static RAM are overlapped to the total Dynamic memory space of 2 Mbytes.

- by addressing the lower 128 Kbytes of this memory space one has the access to the fast external dual port Static RAM,

- by addressing the lower 4Kbytes of this memory space, one has the access to the very fast Internal Static RAM.

Internal RAM	8000.0000 - 8000.0FFF	4K
SRAM	8000.1000 - 8001.FFFF	124K
DRAM1	8002.0000 - 800F.FFFF	896K
DRAM2	8010.0000 - 801F.FFFF	1024K

Dual Port Mem. switch 4008.0000 D0=0 memory "X" sel
(memory "X" & memory "Y") D0=1 memory "Y" sel

I/O switch 400C.0000 D0=0 TRANSP. <> PIO
D0=1 FIFO > PIO

Transp. < FIFO sel. 4010.0000

Transp. <>PIO Select. 4014.0040 - 4014.007F

DSP RESET 4018.0000 D0=0 DSP ACTIVE
D0=1 DSP RESET

[redacted]

The same FDPP module (printed circuit board) can house different transputers (T800 or T425) and different DSP (DSP32C or DSP32), thus giving the possibility to have modules with [redacted]

LOWER PRICE and LOWER PERFORMANCE FDPP	HIGHER PRICE and HIGHER PERFORMANCE FDPP
- DSP32 (160 nsec)	DSP32C (80 nsec)
- 12.5 Mflops - 16-bit address space - 8-bit parallel I/O - 25 MHz operating freq. - NMOS - -	25 Mflops 24-bit address space 16-bit parallel I/O port 50 MHz operating freq. CMOS Lower power requ. Interrupt capability IEEE 754 floating-point format conversion
IMS T425	IMS T800
- 30 MIPS - 33 nsec internal cycle - 4 Kbytes on-chip RAM - 4 Serial Links 20 Mbit/se - -	30 MIPS 33 nsec internal cycle 4 Kbytes on-chip RAM 4 Serial Link 20 Mbit/sec 4.3 Mflops (peak) 64 bit on-chip floating point which conforms to IEEE 754

[redacted] (in alphabetical order) from H I & I Library

[redacted] Calculates the arccosine (inverse cosine) function of the argument, which must be in the range of −0.966 to 0.966.

2.24 µs _alog10 Calculates the base 10 anti-logarithm of the argument, which must be in the range of −38 to 38.

2.14 µs _alog2 Calculates the base 2 anti-logarithm of the argument, which must be in the range of −127 to 127.999.

2.24 µs _aloge Calculates the base e anti-logarithm of the argument, which must be in the range of −88 to 88.

[redacted] Calculates the arcsine (inverse sine) function of the argument, which must be in the range of −0.966 to 0.966.

[redacted] Calculates the arctangent (inverse tangent) function of the argument, which must be greater than zero.

2.02 µs _cos Calculates the cosine of the argument, which must be an angle measured in radians and in the range of $-\pi/2$ to $\pi/2$.

[redacted] Calculates the value of argument N divided by argument D (divide routine).

1.52 µs _divf Faster but less accurate divide routine.

0.64+0.16N µsec _dsp32 Converts an array of numbers in the IEEE 32-bit floating-point format to the interna DSP32/DSP32C floating-point format.

0.8+0.08N µsec _dsp32c Converts an array of numbers in the IEEE 32-bit floating-point format to the interna DSP32/DSP32C floating-point format. The maximum size of the array is 2048.

12.68 µs _gran Returns a random number with float value between 0 and 1. The random numbers have a Gaussian distribution with zero mean and unity variance.

0.72+0.16N µs _ieee32 Converts an array of numbers in the DSP32/DSP32C floating-point format to the [redacted] floating point format.

0.88+0.08N µs _ieee32e Converts an array of numbers in the DSP32/DSP32C floating-point format to the [redacted] floating point format. The maximum size of the array is 2048.

1.82 µs _inv Calculates the inverse (reciprocal) value of the argument.

1.34 µs _invf Faster but less accurate inverse routine.

2.52 µs _invsqr Calculates the inverse of the square root of the argument.

2.74 µs _log10 Calculates the base 10 logarithm of the argument, which must be greater than zero.

2.66 µs _log2 Calculates the base 2 logarithm of the argument, which must be greater than zero.

2.74 µs _loge Calculates the base e logarithm of the argument, which must be greater than zero.

1.64 µs _ran Returns a random number with float value uniformly distributed between 0 and 1.

1.64 µs _ran24 Returns a random number with float value uniformly distributed between 0 and 1. The period of the random number sequence is 16,777,216.

1.64 µs _sin Calculates the sine of the argument, which must be an angle measured in radians and in the range of $-\pi/2$ to $\pi/2$.

[redacted] Fastest and most accurate square root routine.

4.32 µs _sqrt Calculates the square root of the argument.

2.90 µs _sqrtf Faster but less accurate square root routine.

2.26 µs _sqrtq Fastest but least accurate square root routine.

2.82 µs _tan Calculates the tangent of the argument, which must be an angle measured in radians and in the range of $-\pi/4$ to $\pi/4$.

4.4 µs _xtoy Calculates the value of argument x to the power of the argument y. The value of the argument x must be greater than zero.

APPENDIX B. [redacted] (in alphabetical order)

[redacted]	[redacted]	Multiply two 2×2 matrices.
3.2 μs	_mat3x3	Multiply two 3×3 matrices.
2.08 μs	_mat4x1	Multiply a 4×4 matrix by a 4×1 matrix.
2.0 μs	_mat4x1f	Faster multiplying routine for a 4×4 matrix by a 4×1 matrix.
[redacted]	[redacted]	Multiply two 4×4 matrices.
11.04 μs	_mat5x5	Multiply two 5×5 matrices.
2.98 μs	_matin2	Calculates the inverse of a 2×2 matrix.
n=4; 50 μs	_matinf	Faster but less accurate (than _matinv) matrix inverse routine. (N=20; 2640 μs)
n=4; 70 μs	_matinv	General-purpose square matrix inversion routine. (N=20; 3060 μs)
1.2+0.08 m[(n+6)p+4] μs	_matmul	Multiply two general matrices, A and B, and stores result in matrix C. The dimension of matrix A is m × n, B is n × p, and C is m × p. (n=2; 1.88 μsec) (n=5; 11.04 μsec)

APPENDIX C. [redacted] (in alphabetical order)

1.18+0.08N μsec	_fir	Calculates the output of the finite impulse response (FIR) filter.
	_fir5	Calculates the output of the finite impulse response (FIR) filter where filter length must be an integer multiple of 5.
1.32+0.32N μs	_firc	Calculates the output of the complex finite impulse response (FIR) filter.
1.44+0.36N μs	_iir	Calculates the output of an infinite impulse response (IIR) filter. The IIR filter corresponds to the direct form II cascade of second-order sections with four multiplications per section.
1.62 μs	_iir2	Calculates the output of a 2-section, 4-multiply per section infinite impulse response filter.
1.92 μs	_iir3	Calculates the output of a 3-section, 4-multiply per section infinite impulse response filter.
2.34 μs	_iir4	Calculates the output of a 4-section, 4-multiply per section infinite impulse response filter.
1.24+0.4N μs	_iird	Calculates the output of an infinite impulse response (IIR) filter. The IIR filter corresponds to the direct form II cascade of second-order sections with five multiplications per section.
1.0+0.4N μs	_iirdN	Calculates the output of an infinite impulse response (IIR) filter. N indicates the number of sections, which can be 1, 2, 3, or 4. The IIR filter corresponds to the direct form II cascade of second order sections with five multiplications per section.
1.06+0.44N μs	_iirt	Calculates the output of an infinite impulse response (IIR) filter. The IIR filter corresponds to the direct form I cascade of second order sections with five multiplications per section.
1.26 μs	_iirt1	Calculates the output of a 1-section, 5-multiply per section (direct form I) IIR filter.
1.7 μs	_iirt2	Calculates the output of a 2-section, 5-multiply per section (direct form I) IIR filter.
2.14 μs	_iirt3	Calculates the output of a 3-section, 5-multiply per section (direct form I) IIR filter.
2.58 μs	_iirt4	Calculates the output of a 4-section, 5-multiply per section (direct form I) IIR filter.
0.82 μs	_osc	Returns a sample value from a digital sinusoidal oscillator. The oscillator must first be initialized by the routine _oscinit.
0.84+0.36N μs	_oscN	Returns N sample values from a digital sinusoidal oscillator. The oscillator must first be initialized by the routine _oscinit.
2.48 μs	_oscamp	Returns an estimate of the square amplitude of a digital oscillator from its state variables.
3.36 μs	_oscinit	Initializes a digital sinusoidal oscillator. After this initialization, the routine _osc can be used to generate digital oscillator samples.
1.08 μs	_tone	Returns a sample value from a digital sinusoidal tone-pair. The tone-pair must first be initialized by the routine _toneini.
0.88+0.52N μs	_toneN	Returns N combined sample values from a pair of digital sinusoidal oscillators. The oscillators must first be initialized by the routine _toneini.
4.72 μs	_toneamp	Returns an estimate of the square amplitude of a pair of digital oscillators from the state variables of the tone-pair.
6.96 μs	_toneini	Initializes the start-up of a pair of digital sinusoidal oscillators (tone). After this initialization, the routine _tone can be used to generate tone-pair samples.

APPENDIX D. [redacted] (in alphabetical order)

2.08+0.16N μs	_lms	Implements the real, least-mean-square (LMS) algorithm.
2.2+0.8N μs	_lmsc	Implements the complex, least-mean-square (LMS) algorithm.
2.02+0.26N μs	_lmsl	Implements the leaky, real, least-mean-square (LMS) algorithm.

APPENDIX E. [redacted] (in alphabetical order)

Summary: This documentation compares the execution time and code size requirements for the various FFT routines contained in this appendix.

	_chamm()	Multiplies an array of complex data by the Hamming window.
1.86+0.44N μs	_chann()	Multiplies an array of complex data by the Hanning window.
1.78+0.44N μs	_fft	Calculates the fast Fourier transform (FFT) of a complex array. The size of the array must be a power of 2, and the maximum array size is 4096.
N=64 219 μs		
N=1024 4550 μs	_ffta	Calculates the fast Fourier transform (FFT) of a complex array. The size of the array must be a power of 2, and the maximum array size is 4096.
N=64 172 μs		
N=1024 3832 μs		

_ffta16k — Calculates the fast Fourier transform (FFT) of a complex array. The size of the array must be a power of 2, and the maximum array size is 16384. Only for the DSP32C device.

N=4096 17.75 ms
N=16384 79.86 ms

_ffta1k — Calculates a complex 1024-point fast Fourier transform (FFT). (N=512, 1.947 ms)

3.375 ms

_ffta512 — Calculates a complex 64-, 128-, 256, or 512 point fast Fourier transform (FFT)

N=64; 143 ms

_fftb — Calculates the fast Fourier transform (FFT) of a complex array. The size of the array must be a power of 2, and the maximum array size is 4096. Separate arrays are used to store the real and imaginary parts of the data.

N=64; 213 μs
N=4096; 20.399 ms

_fftc — Calculates the fast Fourier transform (FFT) of a complex array. The size of the array must be a power of 2, and the maximum array size is 4096. Separate arrays are used to store the real and imaginary parts of the data.

N=64 ; 165 μs
N=4096; 17.53 ms

_fftc16k — Calculates the fast Fourier transform (FFT) of a complex array. The size of the array must be a power of 2, and the maximum array size is 16384. Separate arrays are used to store the real and imaginary parts of the data. Only for the DSP32C device.

N=4096; 17.75 ms
N=16384; 79.68 ms

[redacted] — Calculates a complex 1024-point fast Fourier transform (FFT). Separate arrays are used to store the real and imaginary parts of the data.

_fftc512 — Calculates a complex 64-, 128-, 256, or 512-point fast Fourier transform (FFT). Separate arrays are used to store the real and imaginary parts of the data.

N=64; 140 μs
N=512; 1.488 ms

_hamm — Multiply an array of real data by the hamming window.

1.3+0.72 * int(N/2) μs

_hamm0 — Multiply an array of real data by the hamming window. The length of the array must be a power of 2, and the maximum array size is 4096.

1.78+0.36N μs

_hamm1 — Multiply an array of real data by the hamming window. The length of the array must be a power of 2 plus 1, and the maximum array size is 4097.

1.94+0.22N μs

_hann — Multiply an array of real data by the hanning window.

1.3+0.72 * int(N/2) μs

_hann0 — Multiply an array of real data by the hanning window. The length of the array must be a power of 2, and the maximum array size is 4096.

1.7+0.36N μs

_hann1 — Multiply an array of real data by the hanning window. The length of the array must be a power of 2 plus 1, and the maximum array size is 4097.

1.94+0.22N μs

_ifft — Calculates the inverse fast Fourier transform (IFFT) of a complex array. The size of the array must be a power of 2, and the maximum array size is 4096.

N=64; 219 μs
N=1024; 4550 μs

_iffb — Calculates the inverse fast Fourier transform (IFFT) of a complex array. The size of the array must be a power of 2, and the maximum array size is 4096. Separate arrays are used to store the real and imaginary parts of the data.

N=64; 213 μs
N=1024; 4356 μs

_rffta — Calculates the fast Fourier transform (IFFT) of an array of real data. The size of the array must be a power of 2, and the maximum array size is 8192.

N=64; 97 μs
N=1024; 2085 μs
N=8192; 19.988 ms

APPENDIX F. [redacted] (in alphabetical order)

_grey — Converts an array of color pixels represented by 16 bits per pixel (5 bits per color) to an array of pixels with a 5-bit grey scale.

0.72+8.1N μs

_histeq — Implements the histogram equalization algorithm on a grey scale image with each pixel represented by a 5-bit value.

27.7+2.82N μs

[redacted]

Arrays of TRANSPUTERS are already in use in High Energy Physics Data Aquisition Systems.

[redacted] that can replace that existing in a "Transputer Array" with the following advantages:

- [redacted] with the previous "Transputer Array System".

- [redacted] o be used

  a) [redacted] during acquisition at the rate of
  - 50 Mbytes/sec for bursts smaller then 2048 words of 16-bit and
  - 10 Mbytes/sec for larger bursts of words.

  b) [redacted] " by making use of the processing power of the DSP [redacted] (sin, cos, log, sqr, etc. given in the AT&T software library) [redacted]. Between event acquisition the DSP can execute typical algorithms such as: peak-finding, pattern recognition, missing energy, cluster energy and cluster center of gravity in calorimeters, etc.

Until today the processing power at the front end of a data acquisition system (microcontrollers, microprocessors, etc.) was used only during acquisition time because [redacted] from the detector to a central computer.

By means of the flexibility of the Transputer link connections, the data can flow upword to a central Transputer that will correlate it and [redacted] linked this time in an array form.

Low cost

DSP Summary

It uses discrete algorithms instead of analog functions.

Is concerned with real-time processing of digitalized analog signals, which are discrete both in amplitude and time

- Harvard Architecture
  - separation between Program memory and data memory
  - pipelining
- Fast Data input channels
  - DMA from I/O peripherals
    - serial (~ 10 Mbit/sec)
    - Parallel (~ 10 Mbyte sec)
- Particular Instruction Set (for Filters, FFT, etc)
  - Multiply Accumulate
  - Hardware Do Loop
- Single Cycle instructions (similar to RISC)
- On chip memory and peripherals (Serial and parallel I/O, DMA channels, etc)
- hardware functions
  - barrell shifter, parallel multiplier, floating point unit

NETWORK SYSTEM

TRANSPUTER

for communication

DSP

for data acquisition

- filtering
- pattern recognition
- etc.

Good support for IBM PC systems (hardware and software) can be found from the firm: DEFINICON INC.

The market is offering Transputer support for:

- IBM PC & compatible
- VME
- Apple Macintosh
- Q-Bus (DEC)
- SUN
- Apollo

Besides hardware and software compatibility with the systems mentioned before, there are developments on Transputers in the fashion of "building blocks" using Transputers modules daughter boards as piggy-back of a mother board support system (VME, Double Eurocard, ecc).

NETWORK SYSTEM

The ideal is to find an existing expandable parallel processing system with standard software and hardware support with which to interface DSP-Peripherals.

1. A parallel reconfigurable processing system based on Transputers.
(serial transfer rate: 20 Mbit/sec)
- each Transputer having as a cooprocessor a DSP (Motorola, or AT & T, or Texas) as a front-end to the peripherals: A/D, D/A, Serial I/O, Parallel I/O. (acquisition time up to 50 Mbyte/sec). See Fig. 3

A possible architecture based on a TRANSPUTER CONCURRENT CONTROL SYSTEM LINKING SEVERAL DSPs

The software and hardware should profit as much as possible from what is presently on offer in the market, because the cost, and in particular, the time necessary to develop a new product, would be overtaken within a few years.

TEXAS Instruments sell 1st and 2nd generation DSP at 1 US $ = 1 MIP

Motorola puts 64Mbytes of memory into a single VME board).

Among the software for TRANSPUTERS, I have found the following firms:

- ALSYS
- BRAINWARE GmbH
- CESYS GmbH
- COMPUTER SYSTEM ARCHITECTS
- CONTROL-C SOFTWARE
- DENSITRON COMPUTERS
- hema Elektronik
- ITHACA SOFTWARE INC.
- LEVCO INC.
- LOGICAL SYSTEMS INC.
- LPKF CAD/CAM SYSTEMS GmbH
- MAXWELL TECHNOLOGY LTD
- METACOMCO ST
- NAG LIMITED
- PARSYTEC
- PENTASOFT SOFTWARE INC.
- PERIHELION SOFTWARE LTD
- QUINTEK LTD
- REAL TIME ASSOCIATES
- SENSION LTD
- SPARTA INC.
- SYSTE M
- TOPEXPRESS LTD
- TRANSOFT INC.
- TU-BERLIN
- U-MICROCOMPUTERS LTD

In an efford to find the most popular Devel-

opment Tool or Operating System among these firms, T.D.S. (Transputer Development System) from INMOS and Helios (multiple processors and multiple users Operating System) from PERIHELION SOFTWARE LTD, turned out to be the firms that deserve more investigation.

OCCAM & TOOLSET

Several other firms are using, or are compatible with these Systems.

INMOS, MICROWAY, PARSYTEC, DEFINICON, ECC. are using the Helios Operating System.

Among the High Level Languages, there is Parallel-C from Parsytec that is better than "C" from INMOS. However, it is not difficult to find support for many HLL on Transputers.

- Occam (INMOS)
- C (many firms...)
- Fortran (INMOS, PENTASOFT,...)
- Pascal (from INMOS under Occam,...)
- ADA (from ALSYS)

Having made the important choice of the

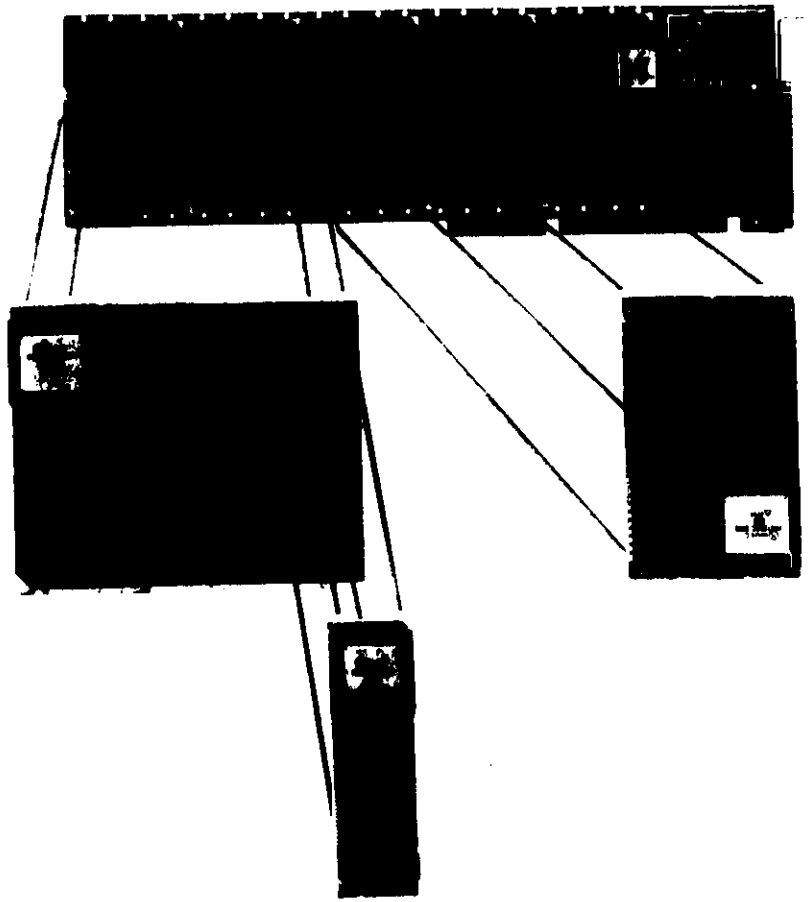
OPERATING SYSTEM

then comes the choice of the

HARDWARE

Because the existing IBM PC or IBM PC Compatible personal computers are more numerous than any other system, a lot of transputer developmen has been carried out on these boards, but for sure it is not the best mechanical and electrical system to support "modularity and expandibility" that a parallel processing Transputer system requires.

TRANSPUTER BOARDS



The INMOS TRAM and MOTHERBOARD family

The INMOS Transputer Modules (TRAMs)

- Standard technology
- Expandable
- Stackable
- Upgradeability
- Flexibility
- Cost effective
- Fast design cycles
- INMOS commitment to latest silicon products on TRAMs
- Third-Party vendors

INMOS provides a series of "TRAM" daughter boards that to housed by different mother boards (for IBM PC, VME, Double Eurocard, etc).

Another way of taking advantage of the particular interconnectivity facility in a Transputer system, has been implemented in busless boards, or in SuperCluster boards (MTM-EDC, Multi-processor Transputer Board with Error Detection and Correction) from PARSYTEC.

Among the firms that are producing hardware for Transputers are:

- **AG Electronics**
- **ARCHIPEL**
- **Caplin Cybernetics Corp.**
- **Cesys GmbH**
- **Computer System Architects**
- **Concurrent Techniques Ltd.**
- **Definicon Inc.**
- **Flight Electronics Ltd.**

- Gem of Cambridge Ltd.
- hema Elektronik
- Impuls Computer Systems
- Levco Corp.
- Megabyte Computers Inc.
- Microway Inc.
- Niche Data Systems Inc.
- Nth Graphics Inc.
- Parsytec GmbH
- Philips GmbH
- Proteus GmbH
- Quintek Ltd.
- Sang Computer Systems GmbH
- Sension Ltd.
- Sheldomberry Electronic GmbH
- Significat Inc.
- Syste M
- TU-Berlin
- U-Microcomputers Inc.
- Yberle