(NASA-CR-197255) DIGITAL SIGNAL PROCESSORS: MICROPROCESSORS, MICROCOMPUTER CHIPS AND DEVELOPMENT SYSTEMS (DSP Associates) 772 p

N95-70666

Unclas

Z9/60 0033447

NASA-CR-197255

[redacted]

Digital Signal Processors Microprocessors, Microcomputer Chips and Development Systems

[redacted]

February 22-24, 1988/Palo Alto, California
April 4-6, 1988/Cambridge, Massachusetts

Texas Instruments TMS32030, TMS32025, TMS32020, TMS 32010

AT&T WE DSP32C, WE DSP32, WE DSP16

Motorola DSP56000, DSP56200

NEC μpd77230, μpd77C25

Analog Devices ADSP-2100

Thomson/Mostek TS68930, TS68931

Fujitsu MB8764, MB86232, MB86220

INMOS A100

OKI MSM6992

GE/Intersil ISP9400, ISP9410

Zoran ZR33881, ZR34161, ZR36010, ZR34325

33447
OCT 1988

DSP ASSOCIATES

[redacted]

[redacted]

Digital Signal Processors Microprocessors, Microcomputer Chip and Development Systems

Texas Instruments TMS32030, TMS32025, TMS32020, TMS 32010

AT&T WE DSP32C, WE DSP16

Motorola DSP56000, DSP56200

NEC upd77230, upd77C25

Analog Devices ADSP-2100

SGS/Thomson TS68930, TS68931

Fujitsu MB8764, MB86232, MB86220

INMOS A100

OKI MSM6992

GE/Intersil ISP9400, ISP9410

Zoran ZR33881, ZR34161, ZR36010, ZR34325

DSP ASSOCIATES

TABLE OF CONTENTS

Texas Instruments

AT&T Technologies

Motorola

NEC Electronics

Analog Devices

SGS/Thomson

INMOS

OKI

Zoran

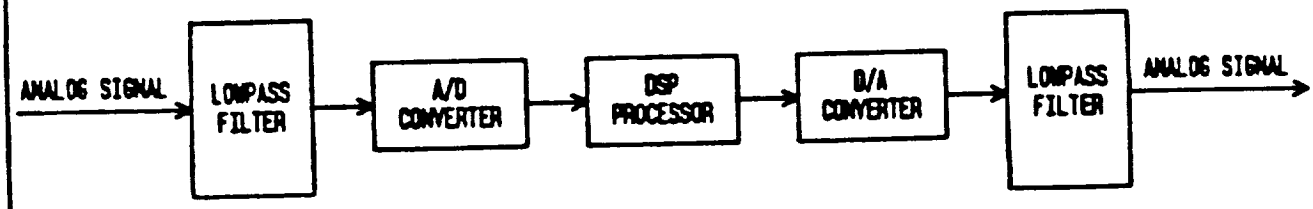
TEXAS INSTRUMENTS

TMS320 PRODUCTS AND DSP APPLICATIONS SEMINAR

AGENDA

- INTRODUCTION
- SECOND GENERATION 320 (320C25)
- FIRST GENERATION 320 (320C10)
- BREAK
- THIRD GENERATION 320 (320C30)
- TOOLS
- APPLICATIONS

THE DSP ENVIRONMENT: DEFINITIONS



- LPF: LIMITS THE SIGNAL BANDWIDTH TO ELIMINATE ALIASING DISTORTION
- A/D: SAMPLES AND QUANTIZES SIGNAL
- DSP: PROCESSES THE SIGNAL USING NUMERICAL TECHNIQUES TO EXTRACT INFORMATION, OR TO GENERATE CONTROL COMMANDS
- D/A: CONVERTS DIGITAL SIGNALS TO ANALOG VALUES
- LPF: SMOOTHS THE "STAIRCASE" WAVEFORM OUTPUT FROM THE D/A

(10-2-5)

DSP APPLICATION CHARACTERISTICS

APPLICATION REQUIREMENT		PROCESSOR ATTRIBUTES
● REAL-TIME PROCESSING	→	HIGH SPEED, HIGH THROUGHPUT
● LARGE ARRAY OF DATA	→	INSTRUCTIONS TO MOVE AND PROCESS LARGE DATA ARRAYS
● ALGORITHM INTENSIVE	→	FAST MATHMATICAL COMPUTATIONS, SINGLE CYLCE DSP OPERATIONS (MACD)
● SYSTEM FLEXIBILITY	→	GENERAL PURPOSE PROGRAMMABILITY, EPROM, MC/MP MODE

(10-2-91)

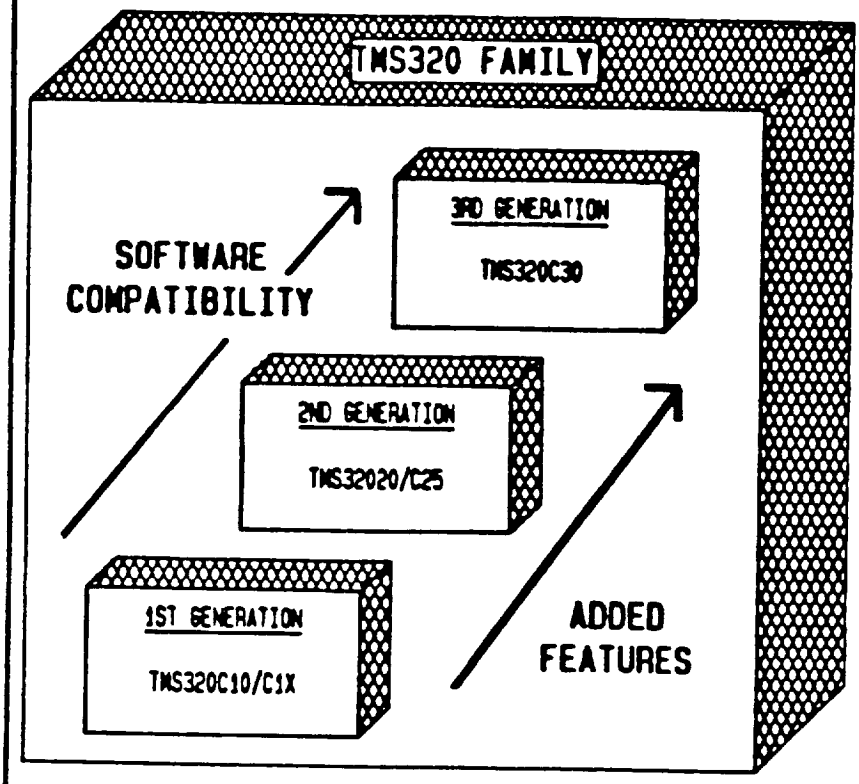
DSP IMPLEMENTATION APPROACHES

APPROACH	ADVANTAGE	DISADVANTAGE
BIT SLICE	• VERY FAST • HARDWARE MULTIPLIER	• HIGH CHIP COUNT/POWER/ COST • COMPLEX DESIGN & DIFFICULT TO PROGRAM
CUSTOM CHIPS	• LOWEST PRODUCTION COST • DEFINED PERFORMANCE	• EXTREMELY HIGH-VOLUME APPLICATIONS ONLY • HIGH INITIAL COST • HIGH RISK
GENERAL PURPOSE uPROCESSORS	• PROGRAMMABLE • EASY TO DESIGN WITH & PROGRAM • LOW COST	• NO HARDWARE MULTIPLIER • SLOW SPEED • NO SPECIAL DSP FUNCTIONS
GENERAL-PURPOSE DSP PROCESSORS	• PROGRAMMABLE • EASY TO DESIGN WITH & PROGRAM • LOW COST • FAST • HARDWARE MULTIPLIER • NOT APPLICATION LIMITED	

(10-2-4)

ORIGINAL PAGE IS
OF POOR QUALITY

TMS320 DSP FAMILY



FEATURES

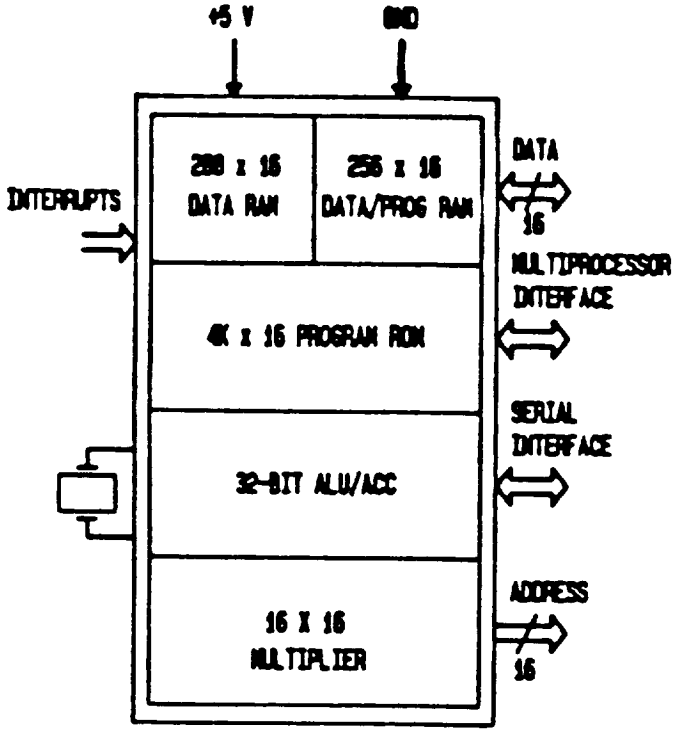
- FAST CYCLE TIME
- HARDWARE MULTIPLIER WITH 32-BIT PRECISION
- GENERAL-PURPOSE FEATURES
- DSP SPECIFIC FEATURES
- SOFTWARE COMPATIBILITY TO PROTECT YOUR INVESTMENT
- LONG-TERM TI COMMITMENT TO GROW THE FAMILY WITH YOUR REQUIREMENTS

(10-3-3)

AGENDA

* INTRODUCTION
* SECOND GENERATION 320 (320C25)
* FIRST GENERATION 320 (320C10)
* BREAK
* THIRD GENERATION 320 (320C30)
* TOOLS
* APPLICATIONS

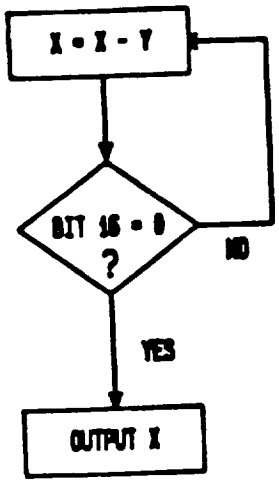
TMS320C25 KEY FEATURES



- 100 nS INSTRUCTION CYCLE TIME
- 128K-WORDS TOTAL MEMORY SPACE
- THREE PARALLEL SHIFTERS
- 133 GENERAL PURPOSE AND DSP INSTRUCTIONS
- S/W UPWARD COMPATIBLE WITH PREVIOUS FAMILY MEMBERS
- 1.8um CMOS; 68-PIN PLCC / PGA

(10-3-30)

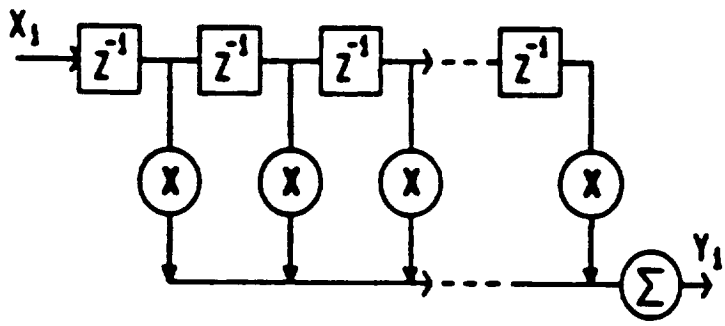
TMS320C25 GENERAL PURPOSE FEATURES



- COMPREHENSIVE INSTRUCTION SET -
  133 INSTRUCTIONS INCLUDING
  - NUMERICAL (34)
  - LOGICAL (15)
  - MEMORY MANAGEMENT (33)
  - BRANCHES (20)
  - PROGRAM/MODE CONTROL (31)
- EXTENDED-PRECISION ARITHMETIC
- SERIAL PORT (DOUBLE BUFFERED, STATIC)
- MULTIPROCESSOR INTERFACES
  (CONCURRENT DMA, GLOBAL DATA MEMORY)
- BLOCK MOVES (UP TO 10 M WORDS/SEC)
- ON-CHIP TIMER
- THREE EXTERNAL MASKABLE INTERRUPTS
- POWERDOWN MODE

(10-3-39)

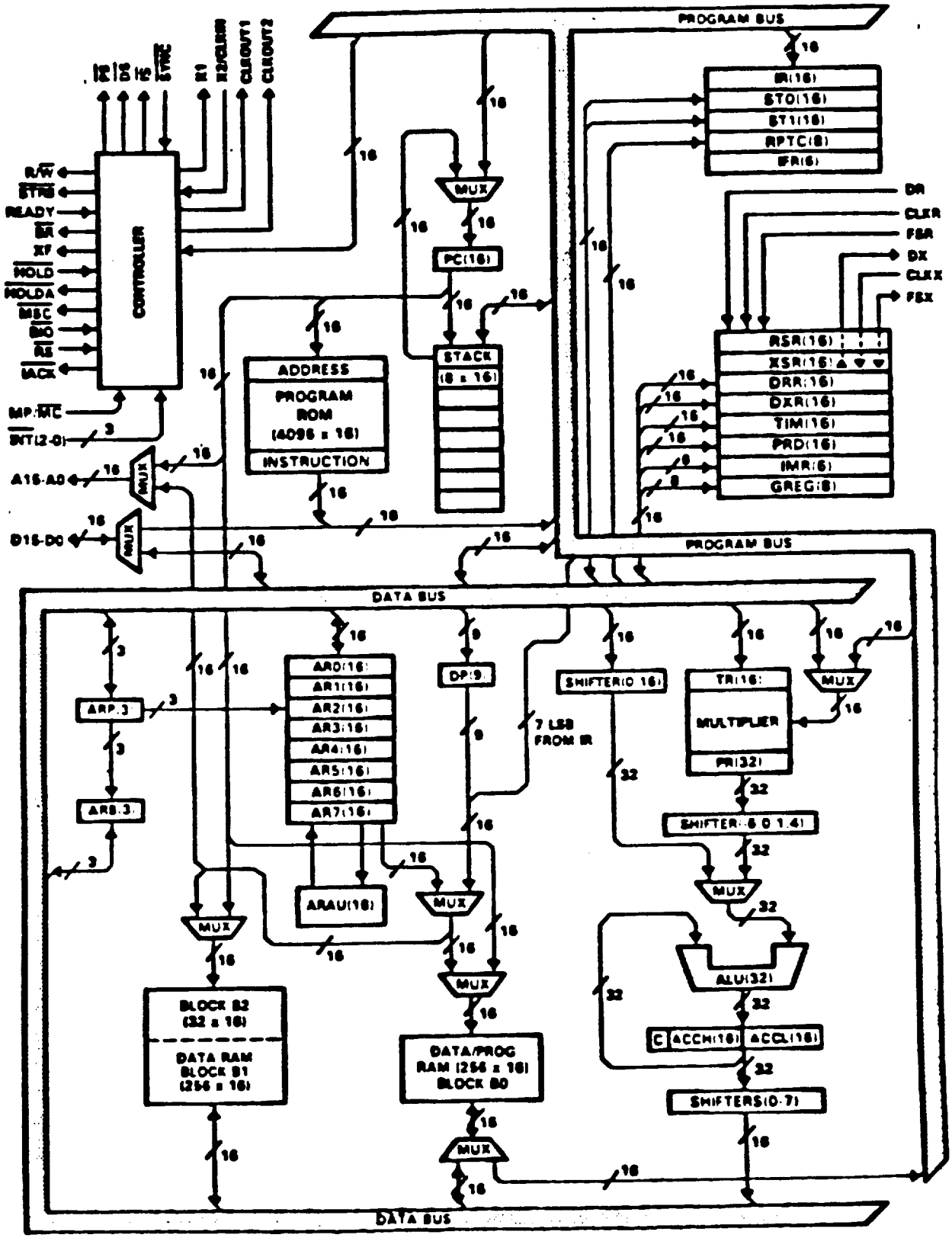
TMS320C25 SPECIAL PURPOSE FEATURES



- 0-16 BIT SCALING SHIFTER (SIGNED OR UNSIGNED)
- OVERFLOW MANAGEMENT
  - SATURATION MODE
  - BRANCH ON OVERFLOW
  - PRODUCT RIGHT SHIFT

- SINGLE-CYCLE MULTIPLY/ ACCUMULATE
- MULTIPLY/ACCUMULATE USING EXTERNAL PROGRAM MEMORY
- REPEAT INSTRUCTION
- ADAPTIVE FILTERING INSTRUCTIONS
- BIT-REVERSED ADDRESSING
- AUTOMATIC DATA-MOVE IN MEMORY (Z^{-n})

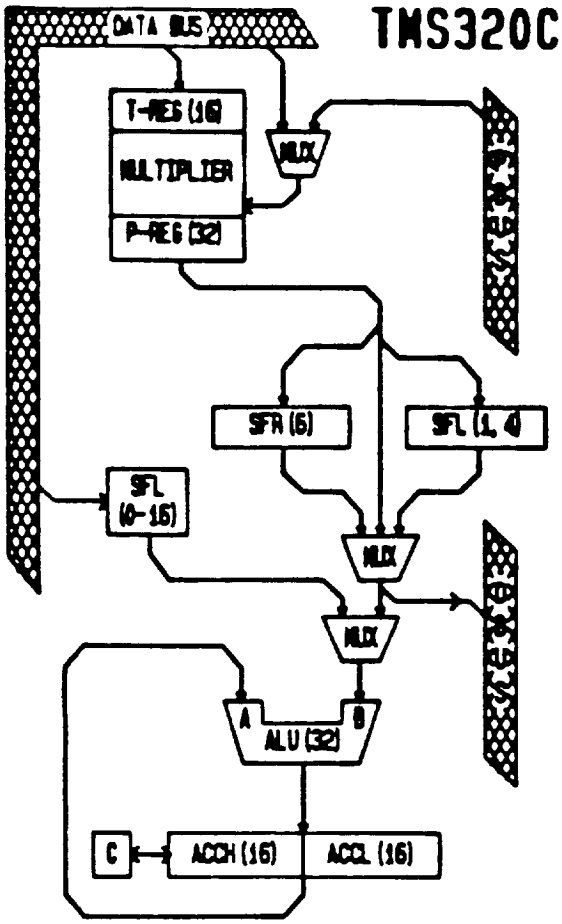
(10-3-40)



TMS320C25 Block Diagram

ORIGINAL PAGE IS
OF POOR QUALITY

TMS320C25 MULTIPLIER

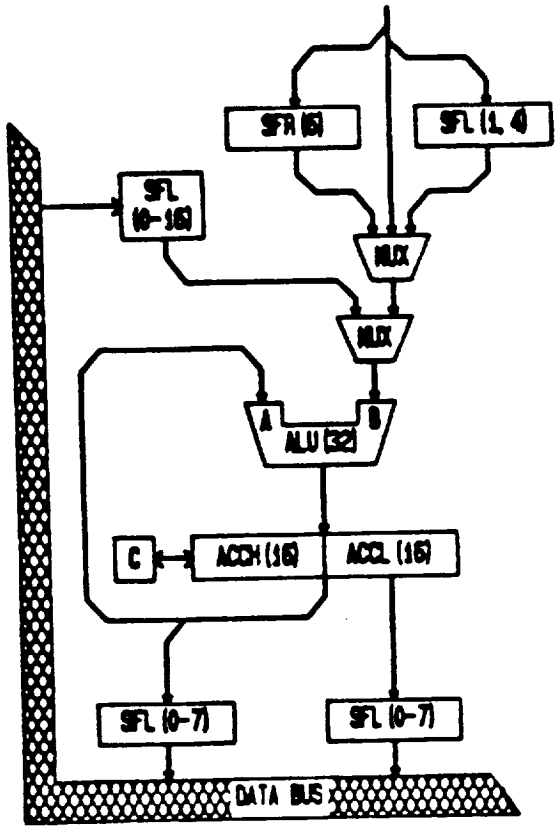


DESIGN & OPERATION

- SINGLE CYCLE 16 x 16-BIT MULTIPLIER W/32-BIT PRODUCT
- PARALLEL MULTIPLY/ACCUMULATE OPERATIONS
- DUAL BUS ACCESS FOR SINGLE CYCLE MULTIPLY/ACCUMULATES
- SQUARE AND ACCUMULATE FUNCTION
- PRODUCT OUTPUT SHIFTERS
- UNSIGNED MULTIPLY

(10-3-42)

TMS320C25 ALU

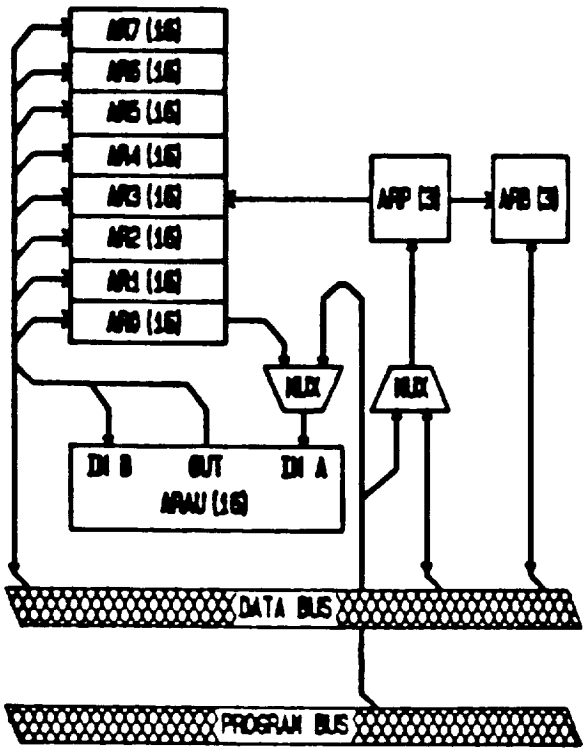


DESIGN & OPERATION

- 32-BIT ALU & ACCUMULATOR
- CARRY BIT FOR EXTENDED PRECISION
- OVERFLOW DETECTION & SATURATION
- SIGN EXTENSION OPTION
- 0-16 BIT PARALLEL SHIFTER FOR LOADS AND ARITHMETIC OPS
- SHIFTERS ON PRODUCT REGISTER OUTPUT DATA
- 0-7 BIT PARALLEL SHIFTER FOR ACCUMULATOR STORES

(10-3-43)

TMS320C25 AUXILIARY REGISTERS

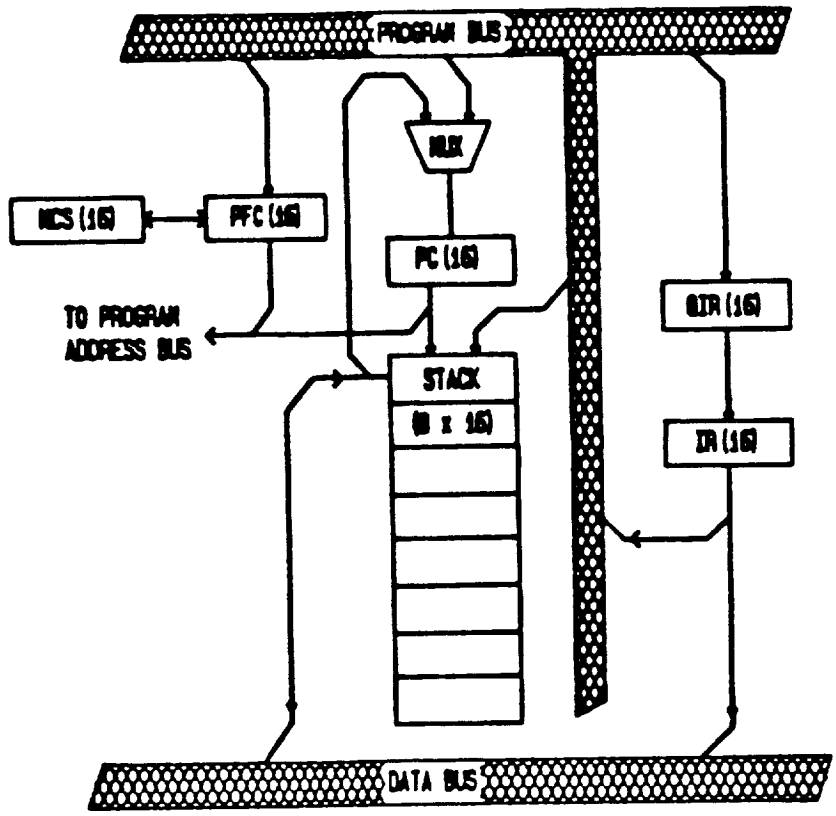


DESIGN & OPERATION

- EIGHT AUXILIARY REGISTERS FOR
  - INDIRECT ADDRESSING
  - LOOP COUNTING
  - TEMPORARY STORAGE
- 16-BIT UNSIGNED ARITHMETIC UNIT (ARAU)
- AUTO INCREMENT & DECREMENT
- AUTO INDEXING USING AR0
- REVERSE CARRY BIT PROPAGATION
- AUXILIARY REGISTER POINTER & POINTER BUFFER (ARP, ARB)

(10-3-44)

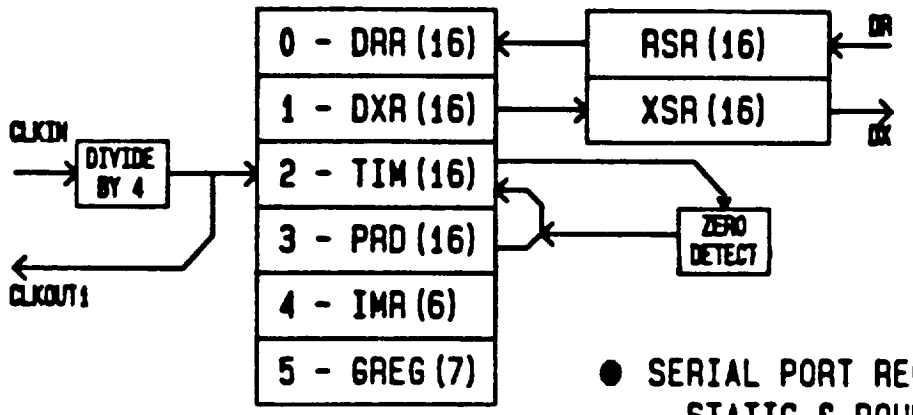
TMS320C25 PROGRAM CONTROL REGISTERS & STACK



- EIGHT LEVELS OF HARDWARE STACK
- SPECIAL FETCH COUNTER STORAGE
- INSTRUCTION REGISTER (IR)
- INSTRUCTION QUEUE REGISTER (QIR)
- REPEAT COUNTER

(10-3-85)

TMS320C25 MEMORY-MAPPED REGISTERS



- SERIAL PORT RECEIVE/TRANSMIT STATIC & DOUBLE BUFFERED
- INTERNAL TIMER REGISTERS
- INTERRUPT MASK REGISTER
- GLOBAL MEMORY REGISTER

(10-3-46)

TMS320C25 STATUS REGISTERS

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ST0	ARP			OV	OVM	1	INTM	DP								
ST1	ARB			CNF	TC	SXM	C	1	1	HM	FSM	XF	FO	TXM	PM	

ARP - AUXILIARY REGISTER POINTER (INDIRECT ADDRESSING)

ARB - AUXILIARY REGISTER POINTER BUFFER (CONTEXT SAVES)

DP - DATA PAGE POINTER (DIRECT ADDRESSING)

CNF - CONFIGURATION OF BLOCK B0 (DATA/PROGRAM)

INTM - GLOBAL INTERRUPT MASK

C - ALU/ACCUMULATOR CARRY BIT

OV - ALU OVERFLOW BIT (LATCHED SIGNAL)

OVM - OVERFLOW MODE BIT (SATURATION/NO SATURATION)

SXM - SIGN EXTENSION BIT (SIGN EXTENSION/NO SIGN EXTENSION)

PM - PRODUCT SHIFT MODE (0, 1, OR 4 LEFT/ 6 RIGHT)

TC - TEST/CONTROL BIT (BIT TESTS, COMPARES, NORMALIZATION)

HM - HOLD MODE (HALT/NO HALT OF INTERNAL EXECUTION)

XF - EXTERNAL FLAG PIN

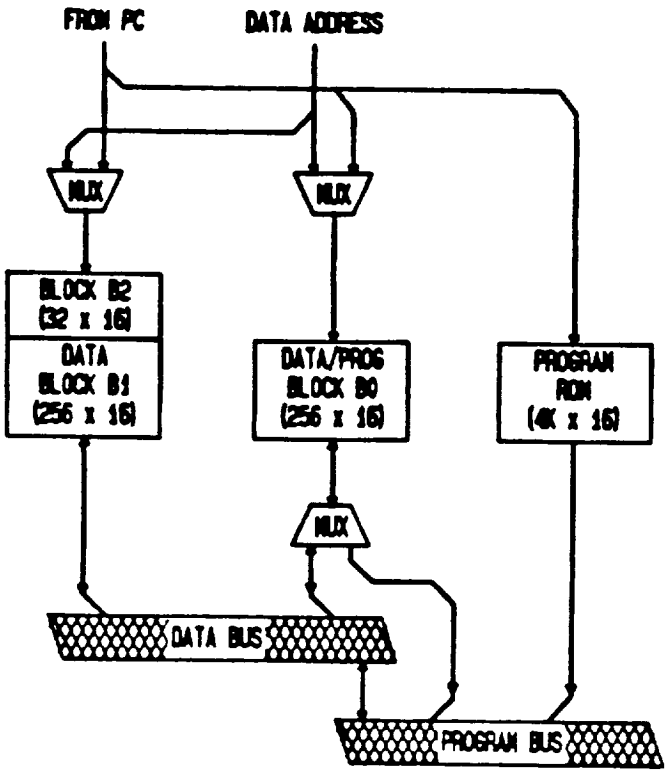
FSM - FRAME SYNCHRONIZATION MODE (USE/IGNORE FSR/FSX)

FO - SERIAL PORT FORMAT BIT (8/16 BIT TRANSFERS)

TXM - TRANSMIT MODE BIT (TRANS BY 'C25)

(10-3-86)

TMS320C25 ON-CHIP MEMORY

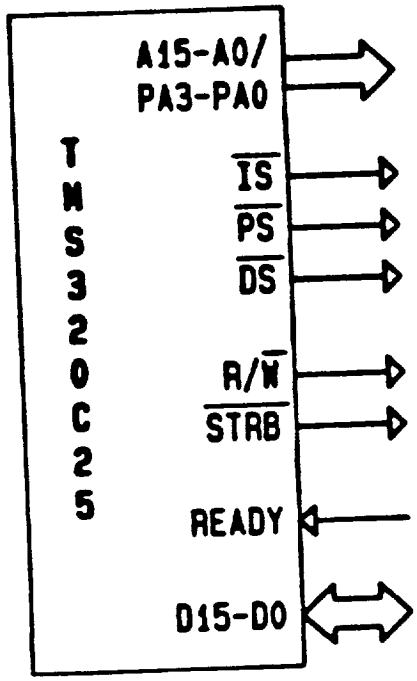


MEMORY ORGANIZATION

- 4K WORDS ON-CHIP MASKED ROM
- 544 WORDS ON-CHIP DATA RAM
- 256 WORDS ON-CHIP RAM RECONFIGURABLE AS DATA/ PROGRAM MEMORY
- BLOCK TRANSFERS IN MEMORY
- DIRECT, INDIRECT, AND IMMEDIATE ADDRESSING MODES

(10-3-47)

TMS320C25 INTERFACE TO OFF-CHIP DEVICES

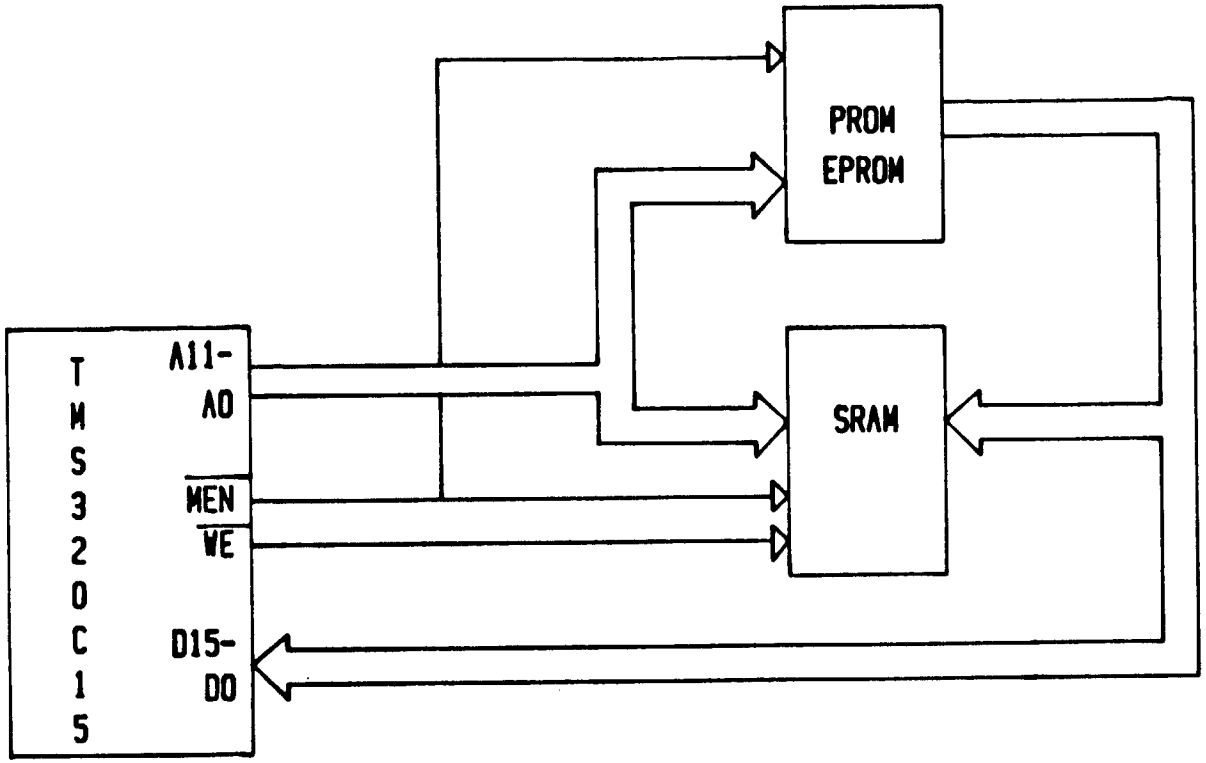


- MICROPROCESSOR-LIKE 16-BIT INTERFACE OVER A SINGLE BUS
  -- MULTIPLEXED FOR PROGRAM, DATA, OR I/O
- WAIT-STATES FOR COMMUNICATION WITH SLOW MEMORIES OR PERIPHERALS
- 16 INPUT/16 OUTPUT CHANNELS CAPABLE OF SINGLE-CYCLE I/O
- BLOCK MOVES CAPABLE OF MOVING ONE 16-BIT WORD PER CYCLE

TMS320C25 MEMORY INTERFACE REQUIREMENTS

WAIT STATES	ACCESS TIME 32020	ACCESS TIME 320C25	EXAMPLE DEVICE TMS320C25
0	85nS	35nS	TMS29C192-35
1	285nS	135nS	
2	485nS	235nS	
3	685nS	335nS	

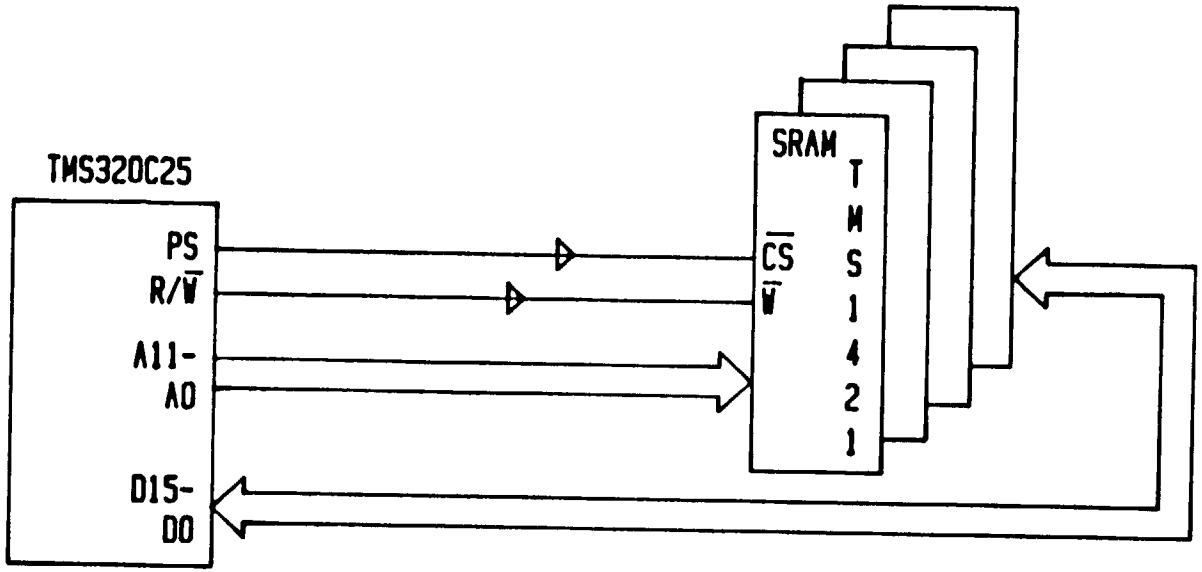
(10-4-16)



TMS320C10/C15 DIRECT MEMORY EXPANSION

		Access Time	Vendor	P/N
1st Gen -20 MHz				
SRAM	(4Kx4)	70 ns	INMOS	IMS1420M-70
PROM	(2Kx8)	45 ns	Texas Instruments	TBP38L165-45
EPROM	(8Kx8)	70 ns	Wafer Scale Integration	WS57C49-70
	(8Kx8)	70 ns	Wafer Scale Integration	WS57C64F-70
1st Gen -25 MHz				
SRAM	(4Kx4)	55 ns	INMOS	IMS1423M-55
PROM	(2Kx8)	45 ns	Texas Instruments	TBP38L165-45
EPROM	(2Kx8)	50 ns	Texas Instruments	TBP27C292-50

TMS320C25 INTERFACE TO STATIC RAM/PROM



		ACCESS TIME	VENDOR	P/N
2nd Gen - 20 MHz				
SRAM	(4Kx4)	50 ns	INMOS	IMS1421-50
PROM	(2Kx8)	45 ns	Texas Instruments	TBP38L165-45
EPROM	(8Kx8)	70 ns	Wafer Scale Integration	WS57C64F-70
	(8Kx8)	70 ns	Wafer Scale Integration	WS67C64F-70
2nd Gen -40 MHz				
SRAM	(4Kx4)	25 ns	Cypress	CY7C169-25
PROM	(2Kx8)	35 ns	Texas Instruments	TBP38L165-35
EPROM	(2Kx8)	35 ns	Texas Instruments	TMS27C292-35

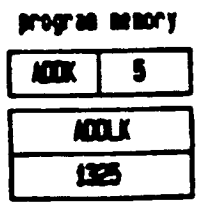
(10-4-11)

TMS320C25 ADDRESSING MODES

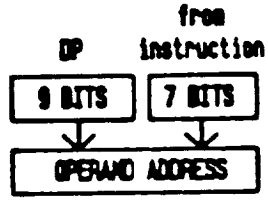
- **IMMEDIATE ADDRESSING**
  - BOTH LONG AND SHORT CONSTANTS
  - EXAMPLES:

    ADDK 5

    ADLK >1325



- **DIRECT ADDRESSING**
  - SAME AS TMS320C10 BUT DP IS 9 BITS
  - 512 'BANKS' OF 128 WORDS
  - USED OFTEN FOR LONG SEQUENCES OF IN-LINE CODE



- **INDIRECT ADDRESSING**
  - 8 AUXILIARY REGISTERS
  - USED OFTEN IN PROGRAM LOOPS WITH AUTO INC/DEC OPTIONS
  - MORE OPTIONS THAN WITH TMS320C10

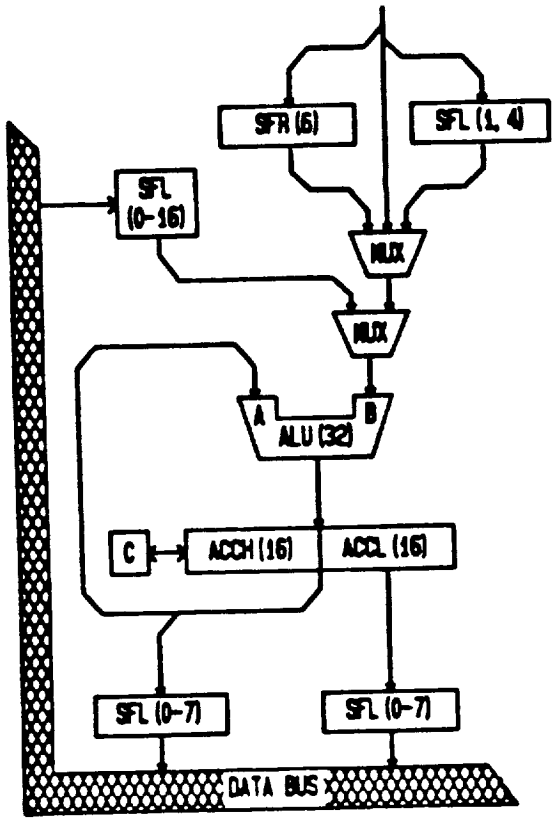
(10-3-48)

TMS320C25 INDIRECT ADDRESSING OPTIONS

OP *[,NARP]	NO CHANGE TO CURRENT AR
OP *+[,NARP]	CURRENT AR IS INCREMENTED
OP *-[,NARP]	CURRENT AR IS DECREMENTED
OP *0+[,NARP]	AR0 IS ADDED TO CURRENT AR
OP *0-[,NARP]	AR0 IS SUBTRACTED FROM CURRENT AR
OP *BR0+[,NARP]	AR0 IS ADDED TO CURRENT AR (W/REVERSE CARRY PROPAGATION)
OP *BR0-[,NARP]	AR0 IS SUBTRACTED FROM CURRENT AR (W/REVERSE CARRY PROPAGATION)

(10-3-49)

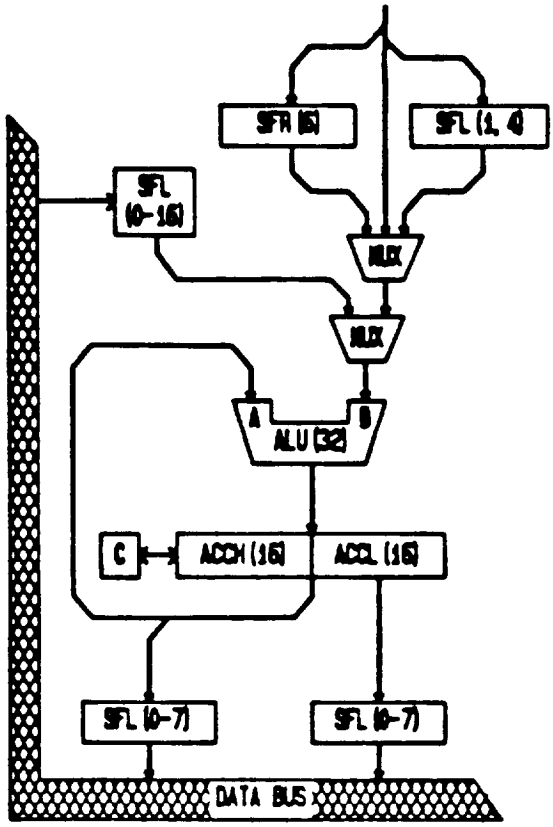
TMS320C25 ALU - LOAD INSTRUCTIONS



LAC	Load ACC w/shift
LACT	Load ACC w/shift specified by T-register
LACK	Load ACC w/short constant
LALK	Load ACC w/long constant
ZAC	Zero ACC
ZALH	Zero ACC & load high
ZALR	Zero ACC & load high w/rounding
ZALS -	Zero ACC & load low w/o sign extension

(10-3-50)

TMS320C25 ALU - STORE INSTRUCTIONS

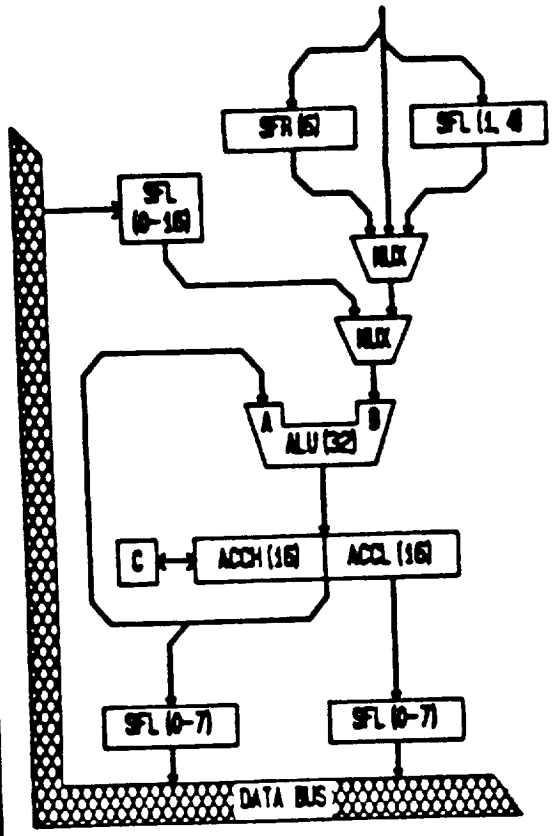


SACH Store ACC high w/shift

SACL Store ACC low w/shift

(10-3-51)

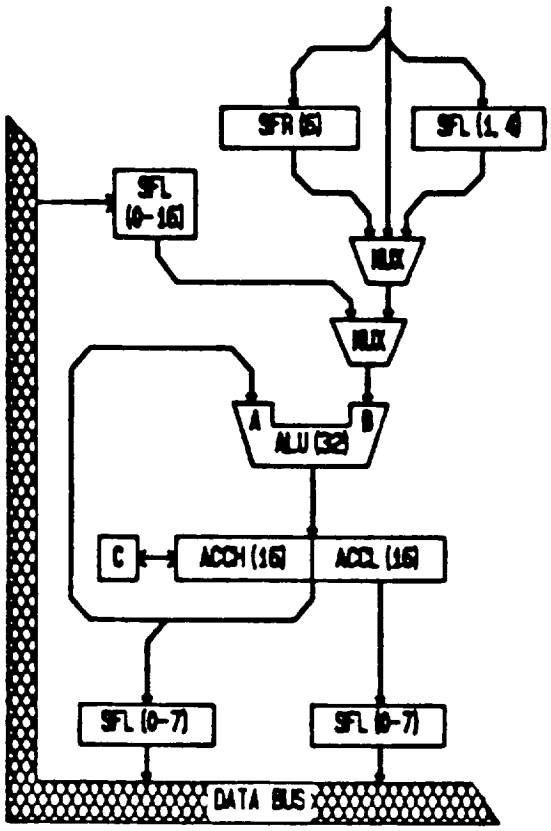
TMS320C25 ALU - LOGICAL INSTRUCTIONS



AND	AND ACC w/16-bit data
ANDK	AND ACC w/long constant, w/shift
OR	OR ACC w/16-bit data
ORK	OR ACC w/long constant, w/shift
XOR	Exclusive-OR ACC w/16-bit data
XORK	Exclusive-OR ACC w/long constant, w/shift
CMPL	Complement ACC

(10-3-52)

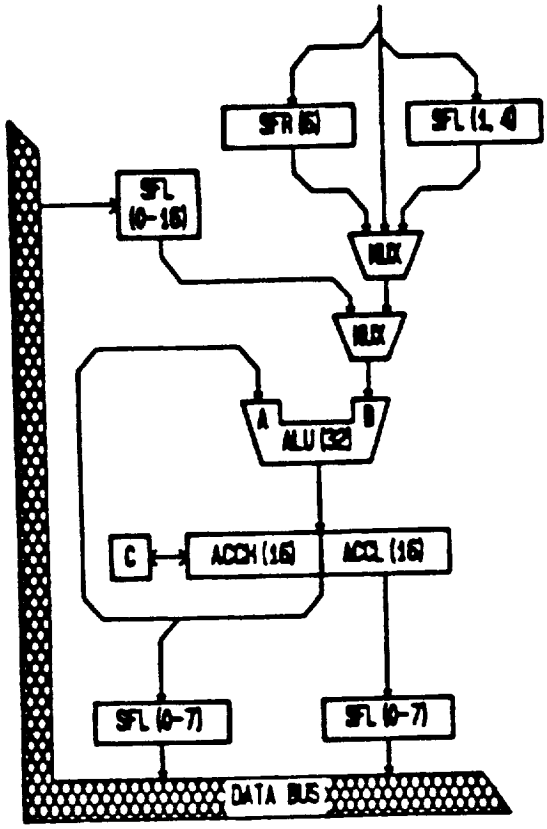
TMS320C25 ALU - SHIFT INSTRUCTIONS



NORM	Normalize ACC 1-bit conditional left shift
ROL	Rotate ACC left 1 bit
ROR	Rotate ACC right 1 bit
SFL	Shift ACC left 1 bit
SFR	Shift ACC right 1 bit

(10-3-53)

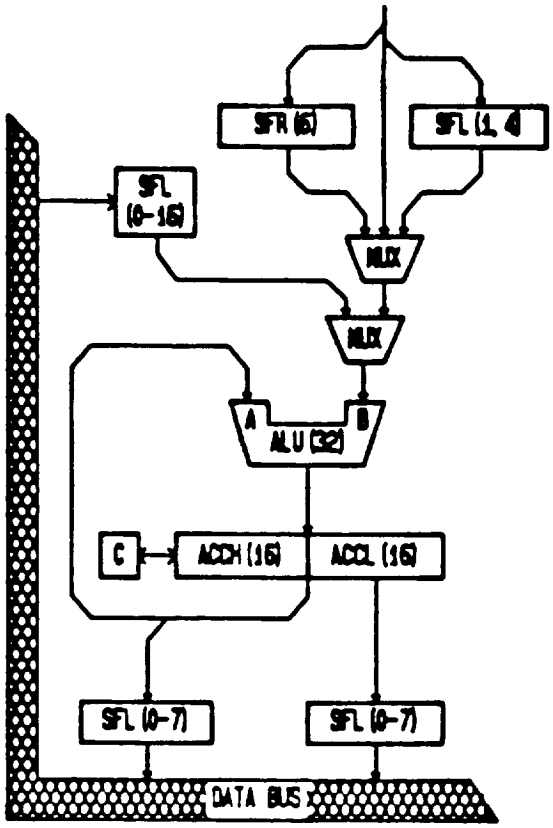
TMS320C25 ALU - ARITHMETIC INSTRUCTIONS I



ADD	Add to ACC w/shift
ADDC	Add to ACC low w/carry
ADDH	Add to ACC high
ADDS	Add to ACC low w/o sign extension
ADDT	Add to ACC w/shift specified by T-register
ADDK	Add short constant to ACC
ADLK	Add long constant to ACC

(10-3-54)

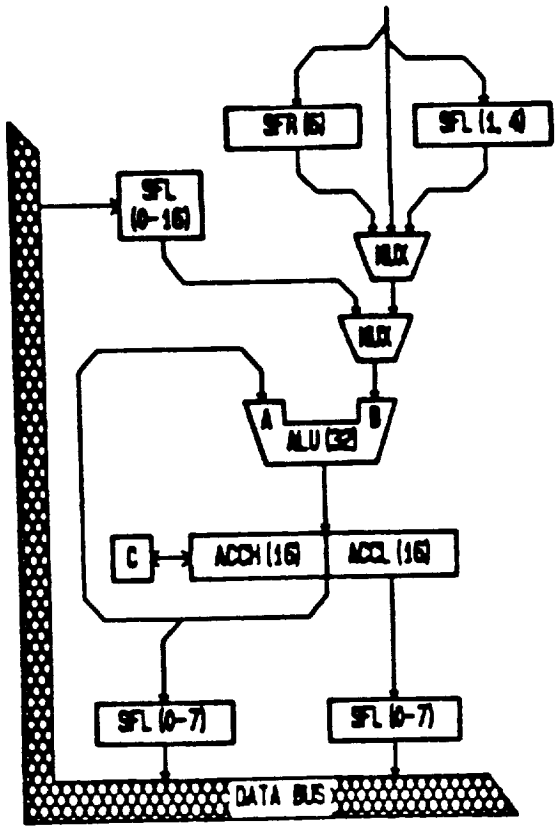
TMS320C25 ALU - ARITHMETIC INSTRUCTIONS II



SUB	Subtract from ACC w/shift
SUBB	Sub from ACC low w/borrow
SUBH	Subtract from ACC high
SUBS	Subtract from ACC low w/o sign extension
SUBT	Subtract from ACC w/shift specified by T-register
SUBK	Sub short constant from ACC
SBLK	Sub long constant from ACC
SUBC	Conditional subtract

(10-3-85)

TMS320C25 ALU - ARITHMETIC INSTRUCTIONS III

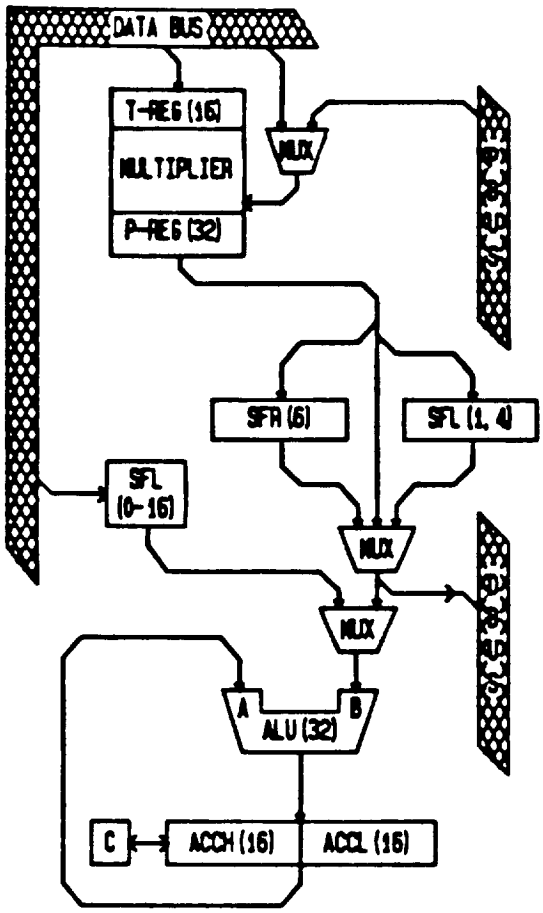


ABS Absolute value of ACC

NEG Negate ACC

(10-3-56)

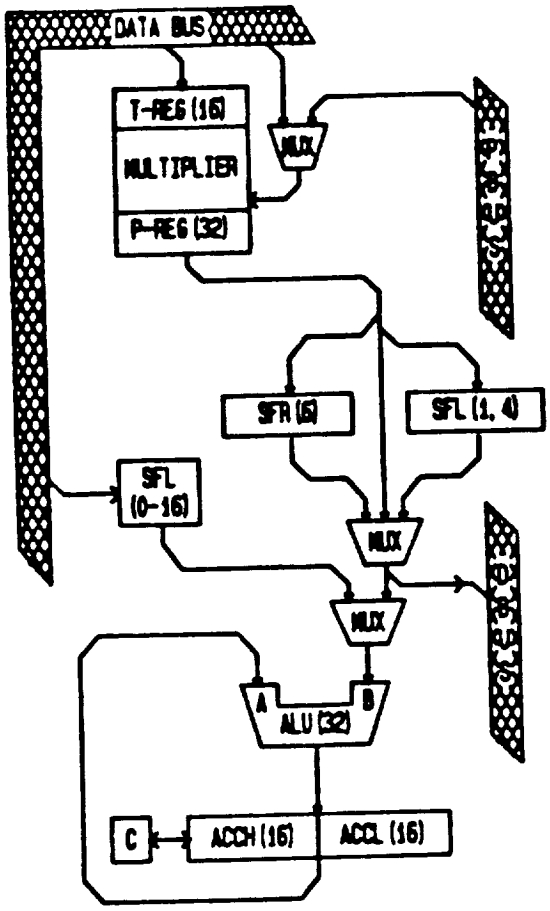
TMS320C25 - P-REGISTER INSTRUCTIONS



PAC	Load ACC w/P-register
APAC	Add P-Reg to ACC
SPAC	Subtract P-Reg from ACC
LPH	Load P-Reg high
SPH	Store P-Reg high
SPL	Store P-Reg low

(10-3-57)

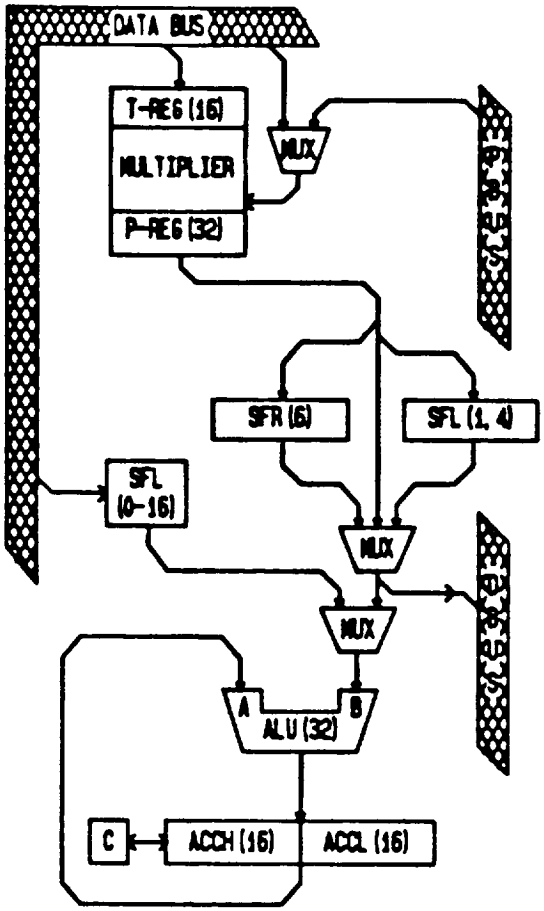
TMS320C25 - T-REGISTER INSTRUCTIONS



LT	Load T-Reg
LTP	Load T-Reg & move P-Reg to ACC
LTA	Load T-Reg & add P-Reg to ACC
LTS	Load T-Reg & subtract P-Reg from ACC
LTD	Load T-Reg, add P-Reg to ACC, & move data to next memory location

(10-3-58)

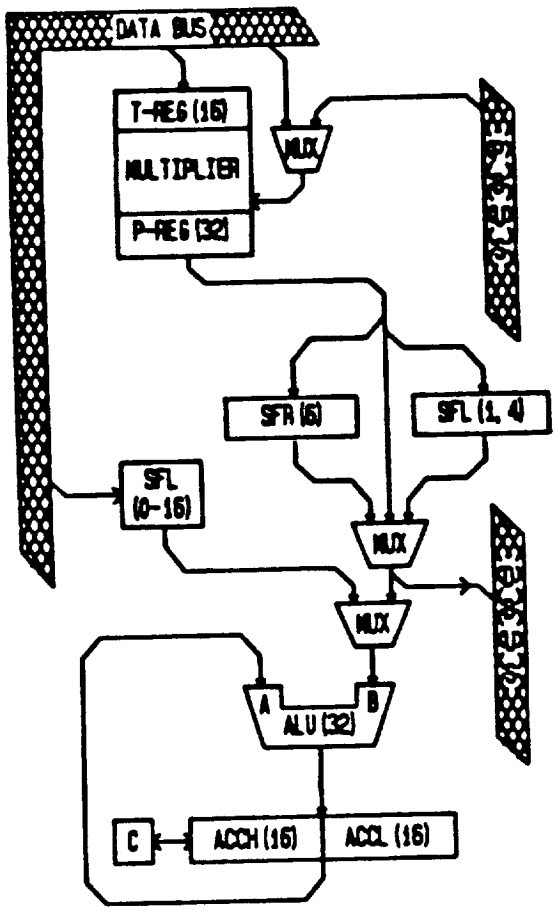
TMS320C25 - MULTIPLY INSTRUCTIONS I



MPY	Multiply T-Reg w/data
MPYK	MPY T-Reg w/13-bit constant
MPYU	MPY T-Reg w/unsigned data
MPYA	MPY T-Reg w/data & add past P-Reg to ACC
MPYS	MPY T-Reg w/data & subtract past P-Reg from ACC

(10-3-59)

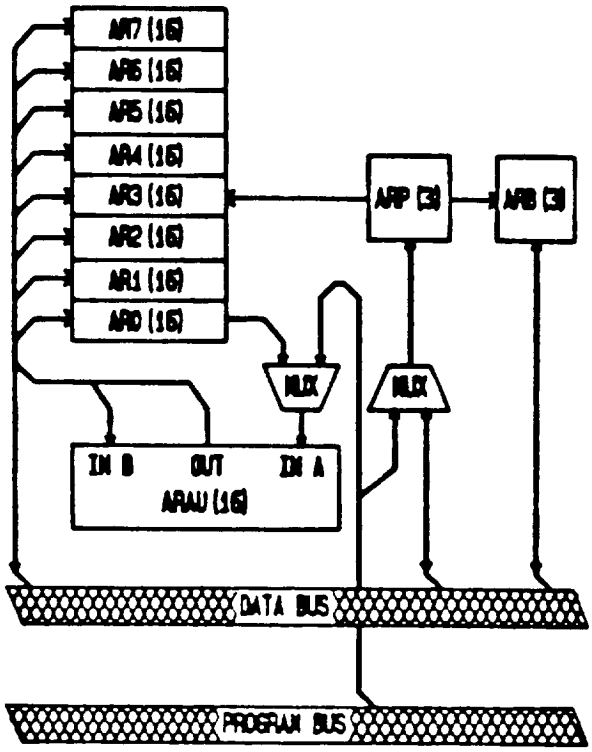
TMS320C25 - MULTIPLY INSTRUCTIONS II



MAC	MPY data memory * program memory & add past P-Reg to ACC
MACD	MPY data memory * program memory, add past P-Reg to ACC, & move data memory
SQRA	Square data memory value & add past P-Reg to ACC
SQRS	Square data memory value & sub past P-Reg from ACC

(10-3-60)

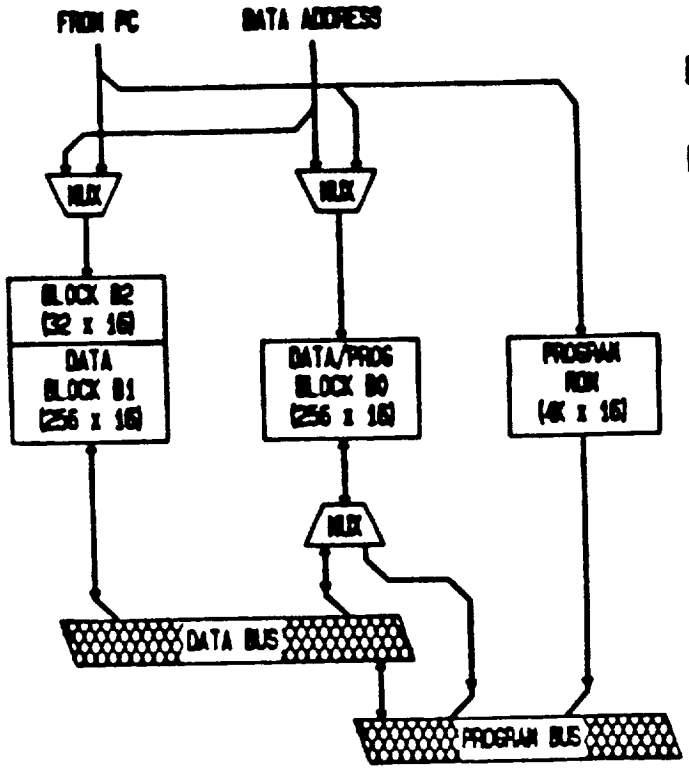
TMS320C25 AUXILIARY REGISTER INSTRUCTIONS



LAR	Load aux-reg w/data
LARK	Load AR w/8-bit constant
LRLK	Load AR w/16-bit constant
MAR	Modify auxiliary register
SAR	Store auxiliary register
ADRK	Add 8-bit constant to AR
SBRK	Sub 8-bit constant from AR
LARP	Load auxiliary register pointer

(10-3-61)

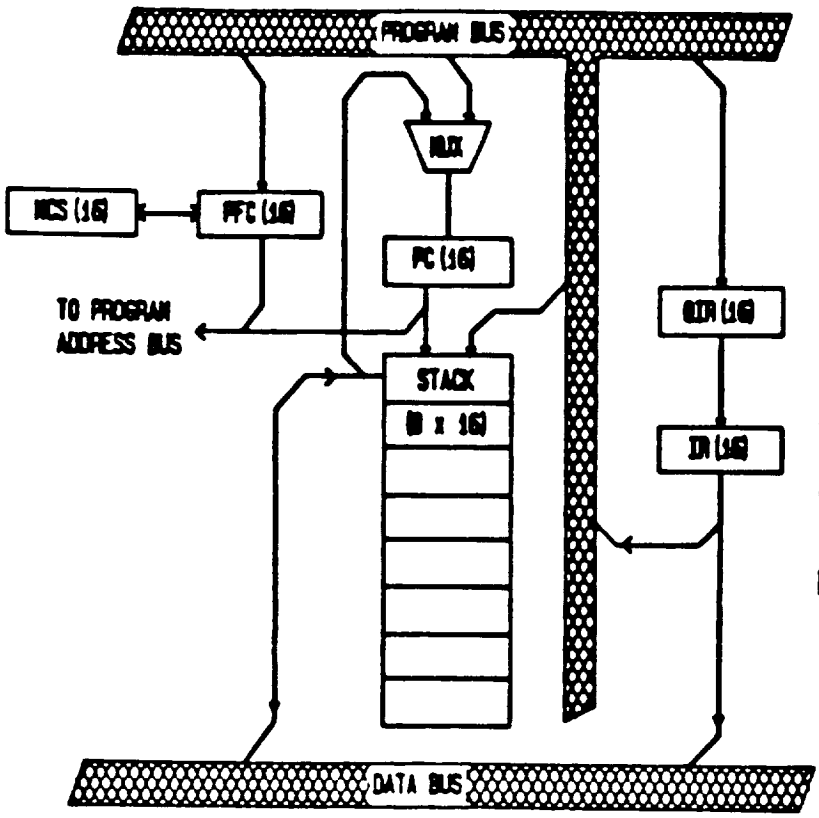
TMS320C25 - MEMORY MOVE INSTRUCTIONS



DMOV	Data move in data mem
BLKD	Block move from data mem to data mem
BLKP	Block move from program mem to data mem
TBLR	Table read from program mem to data mem
TBLW	Table write to program mem from data mem

(10-3-82)

TMS320C25 - BRANCH INSTRUCTIONS



B	Branch immediate, unconditional
BACC	Branch by ACC
Bcnd	Branch conditional
CALL	Call immediate
CALA	Call by ACC
RET	Return from interrupt or subroutine

(10-3-63)

TMS320C25 - CONDITIONAL BRANCH INSTRUCTIONS

BGZ - BRANCH ON ACC > 0

BGEZ - BRANCH ON ACC >= 0

BZ - BRANCH ON ACC = 0

BNZ - BRANCH ON ACC <> 0

BLZ - BRANCH ON ACC < 0

BLEZ - BRANCH ON ACC <= 0

BC - BRANCH ON CARRY

BNC - BRANCH ON NO CARRY

BV - BRANCH ON OVERFLOW

BNV - BRANCH ON NO OVERFLOW

BBZ - BRANCH IF TC = 0

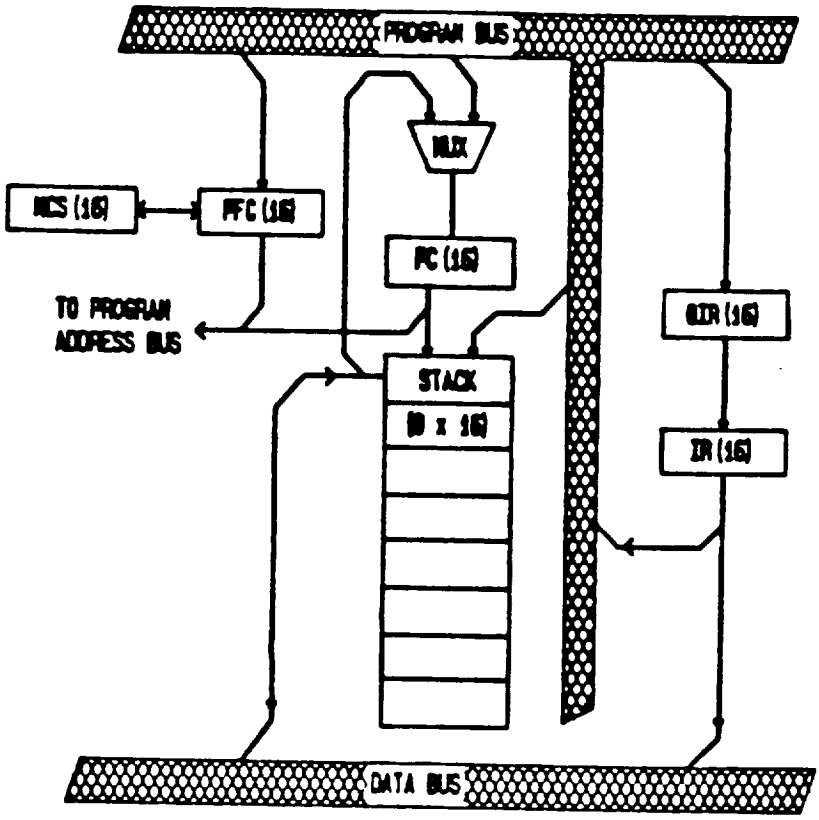
BBNZ - BRANCH IF TC <> 0

BANZ - BRANCH ON AUXILIARY REGISTER NOT ZERO

BIOZ - BRANCH ON BIO = 0

(10-3-60)

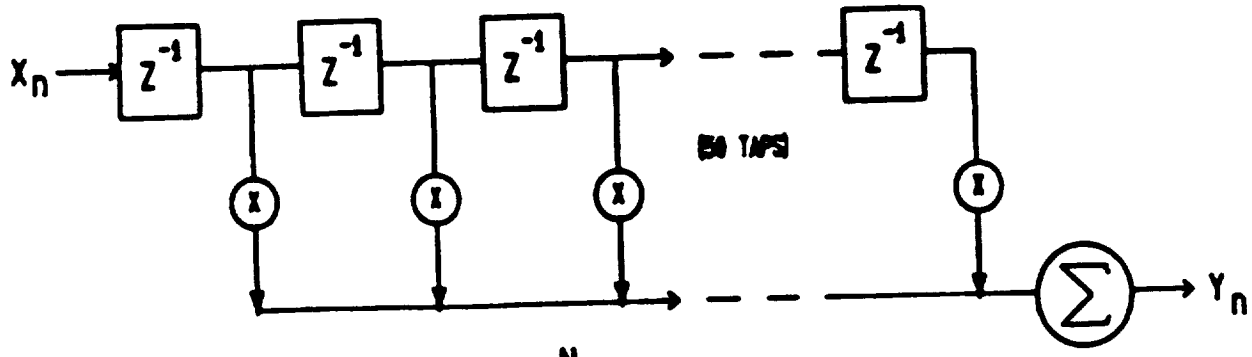
TMS320C25 - STACK INSTRUCTIONS



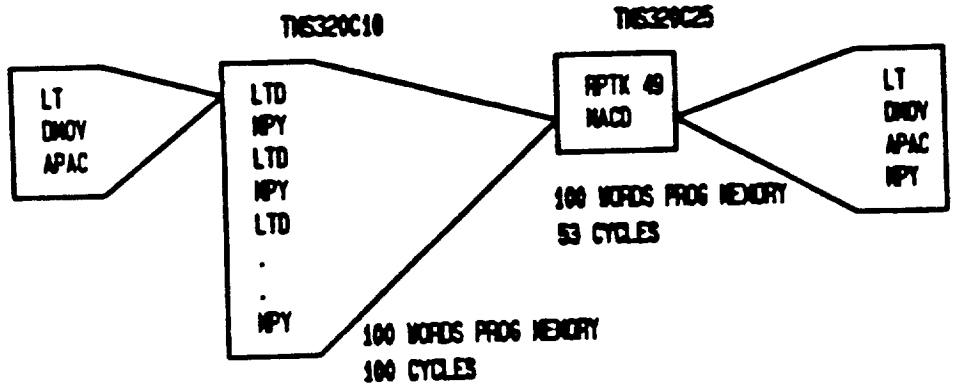
POP	Pop to low ACC
POPD	Pop to data mem
PSHD	Push data mem
PUSH	Push low ACC

(10-3-85)

TMS320C25 - HIGHER PERFORMANCE AT LESS CODE SPACE



$$Y_n = \sum_{k=0}^{N} b_k X(n-k)$$

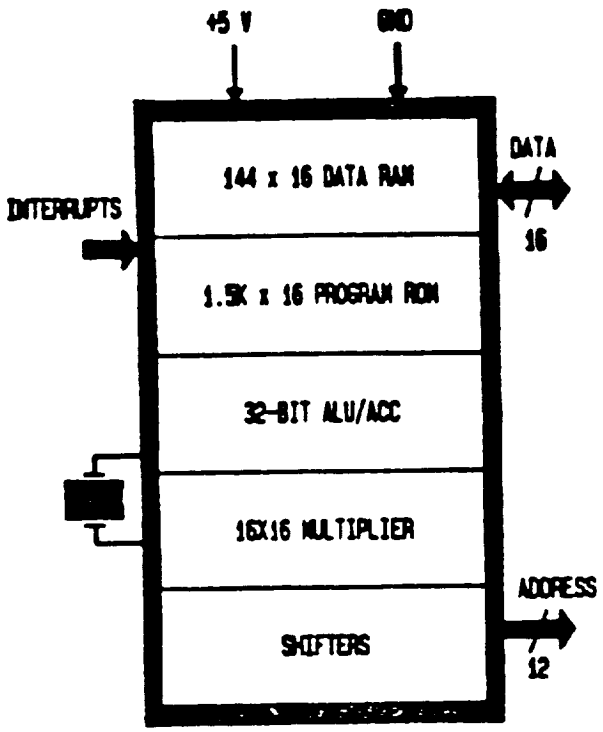


(10-3-87)

AGENDA

* INTRODUCTION
* SECOND GENERATION 320 (320C25)
* FIRST GENERATION 320 (320C10)
* BREAK
* THIRD GENERATION 320 (320C30)
* TOOLS
* APPLICATIONS

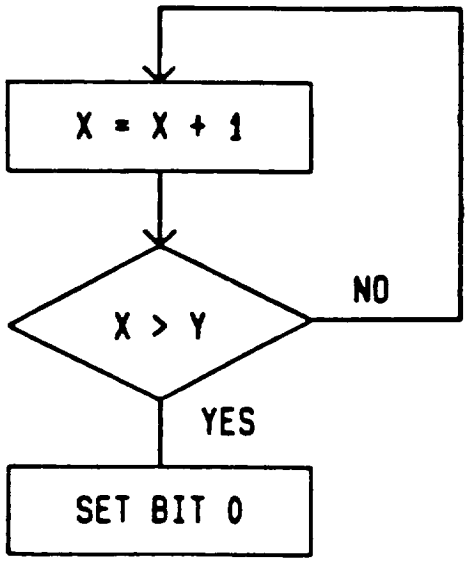
TMS320C10 KEY FEATURES



- 160/200-NS INSTRUCTION CYCLE
- EXTERNAL MEMORY EXPANSION TO 4K WORDS AT FULL SPEED
- 60 GENERAL PURPOSE AND ARITHMETIC INSTRUCTIONS
- POWER DISSIPATION: 165 mW
- 2 um CMOS; 40-PIN DIP, 44-PIN PLCC

(10-3-5)

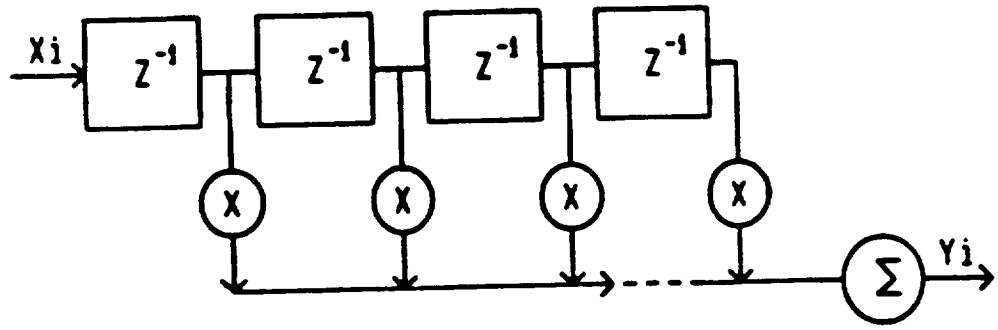
TMS320C10 GENERAL-PURPOSE FEATURES



- 60 INSTRUCTIONS INCLUDING:
  - SIMPLE BUT POWERFUL ADDRESSING MODES
  - FULL SET OF BRANCH INSTRUCTIONS
  - LOGICAL OPERATIONS FOR BIT TEST
- BIT EXTRACTION VIA SHIFTERS
- MULTIPLE SYSTEM CONFIGURATION INCLUDING STANDALONE

(10-3-6)

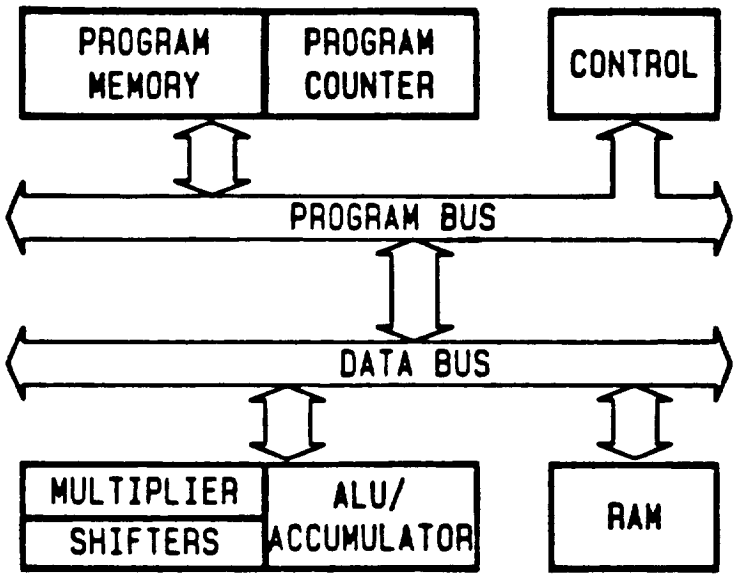
TMS320C10 SPECIAL-PURPOSE FEATURES



- FAST 16X16 HARDWARE MULTIPLIER (160/200 NS)
- ACCUMULATOR SATURATION MODE
- OVERFLOW FLAG
- AUTOMATIC DATA MOVEMENT
- EXTERNAL BIT TEST AND BRANCH (SAMPLING CLOCK)
- 0-16 BIT SCALING SHIFTER

(10-3-7)

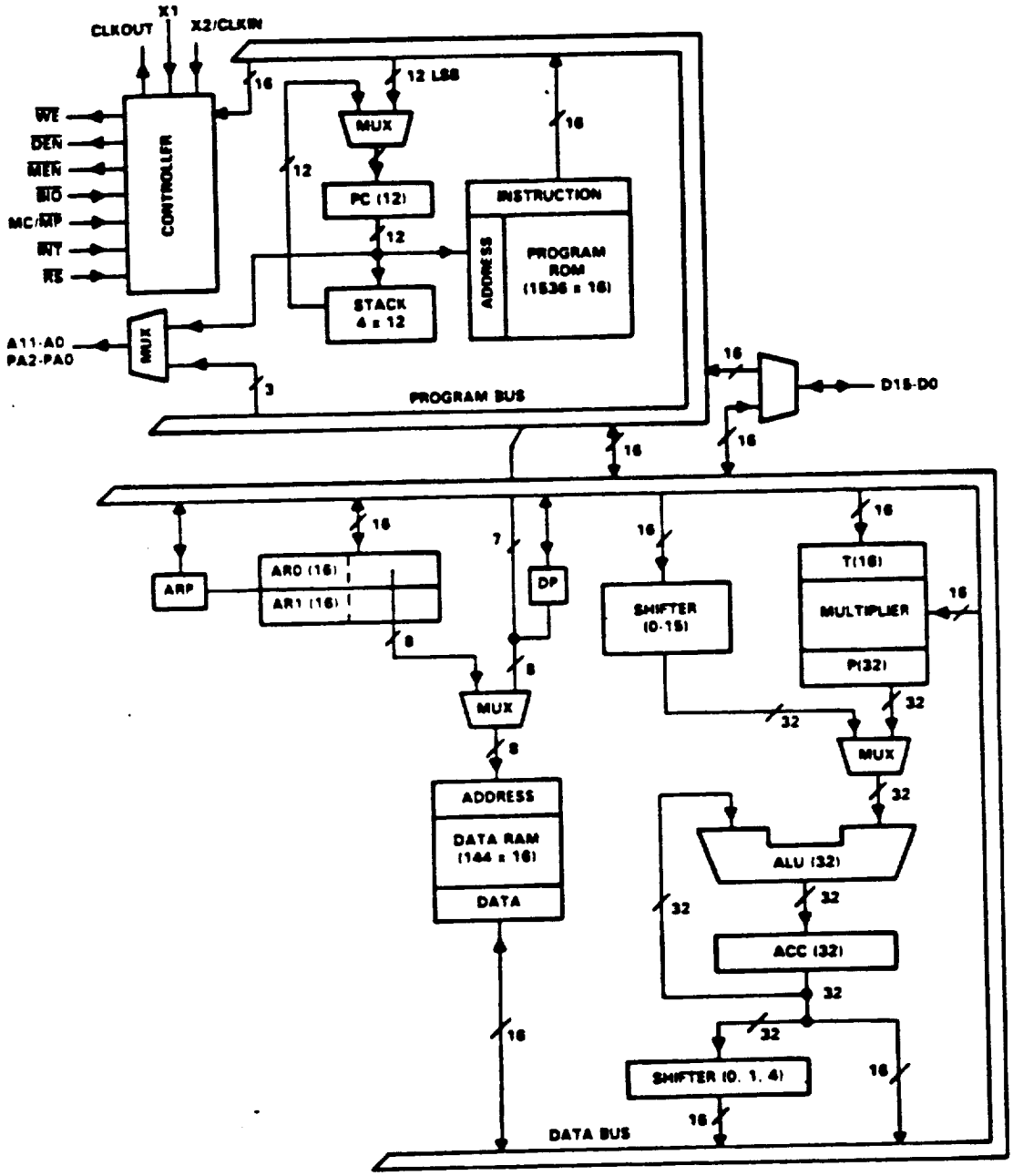
TMS320C10 HIGH-SPEED ARCHITECTURE



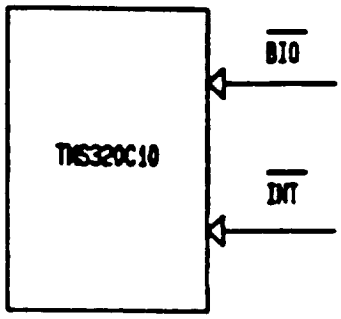
- ADVANCED HARVARD ARCHITECTURE
  - SEPARATE DATA/PROGRAM BUSES
  - "PARALLEL" INSTRUCTION FETCH, OPERAND FETCH, EXECUTE
  - FAST CYCLE TIME
- PARALLEL MULTIPLIER
- MULTIPLE SHIFTERS
- AUTOMATIC INCREMENTING/ DECREMENTING REGISTERS

(10-3-8)

TMS32010 FUNCTIONAL DIAGRAM



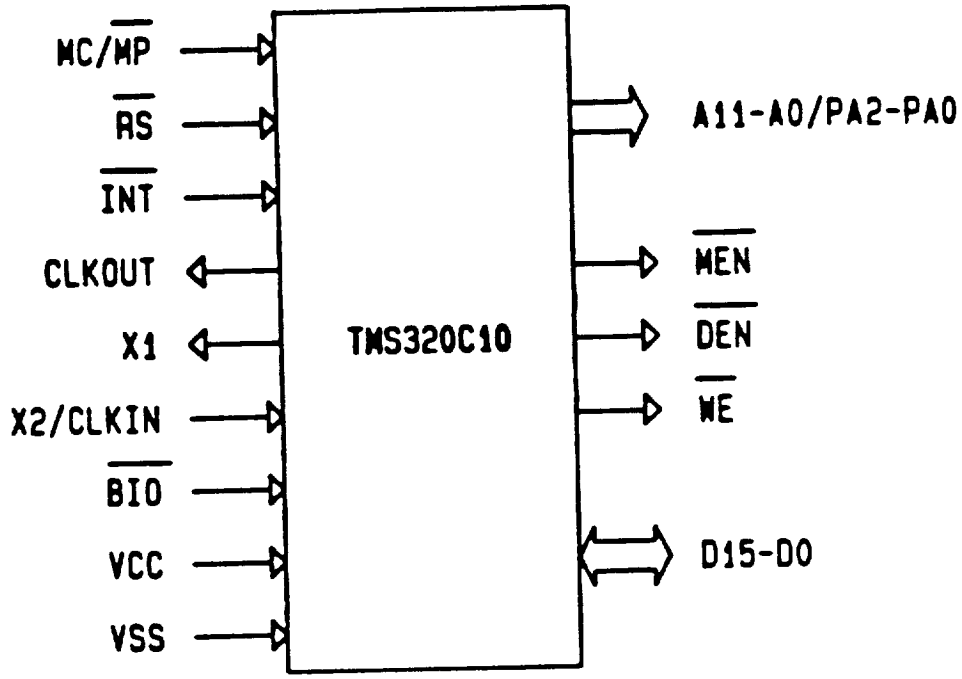
TMS320C10 INTERRUPTS



- HARDWARE AND SOFTWARE INTERRUPT AVAILABLE
- HARDWARE INTERRUPT
  - MASKABLE
  - PULSED OR LEVEL
  - UPON GRANT, INTERRUPTS AUTOMATICALLY DISABLED
  - HARDWARE PROTECTION AGAINST INTERRUPT DURING MULTICYCLE INSTRUCTIONS
  - SPECIAL PROTECTION
- SOFTWARE INTERRUPT
  - BIO=0 DETECTED BY BIOZ INSTRUCTION
  - SUITABLE FOR I/O CONTROL
  - SYNCHRONIZATION FOR MULTIPROCESSING

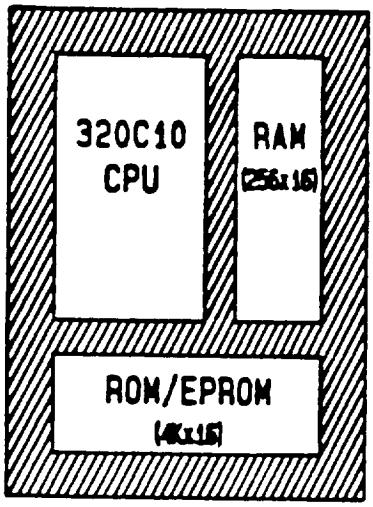
(10-3-14)

TMS320C10 PIN CONFIGURATION



(10-3-16)

TMS320C15/E15 KEY FEATURES

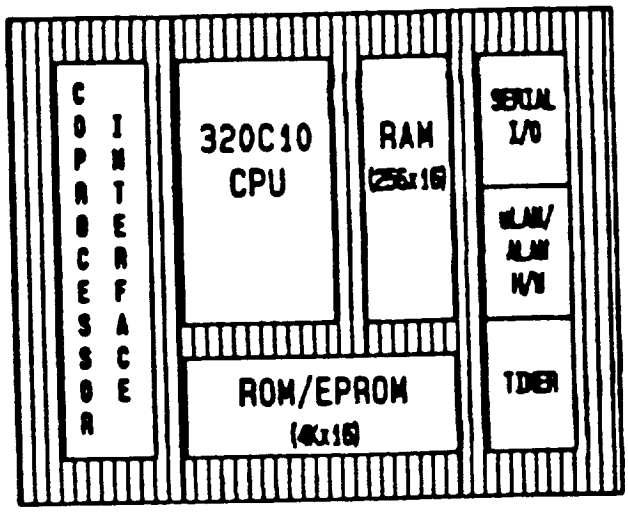


- 100% OBJECT CODE/PLUG-IN COMPATIBLE WITH TMS320C10
- 160/200-NS INSTRUCTION CYCLE
- EXPANDED ON-CHIP RAM/ROM
- MASKED ROM OR EPROM VERSION
- 2u CMOS TECHNOLOGY
  40 PIN DIP & 44 PIN PLCC

DEVICE	RAM	ROM	EPROM
320C15	256	4K	-
320E15	256	-	4K

(10-3-17)

TMS320C17/E17 KEY FEATURES



- 100% OBJECT CODE/PLUG-IN COMPATIBLE WITH TMS320C10
- 160/200-NS INSTRUCTION CYCLE
- EXPANDED ON-CHIP RAM/ROM
- MASKED ROM OR EPROM VERSION
- 2u CMOS TECHNOLOGY 40 PIN DIP & 44 PIN PLCC

DEVICE	RAM	ROM	EPROM	SERIAL I/O	COPROC I/F
320C17	256	4K	-	2	YES
320E17	256	-	4K	2	YES

(10-3-18)

PROGRAMMING THE TMS320E15/E17 EPROM CELL

FEATURES

- 4K X 16 INDUSTRY STANDARD EPROM (TMS27C64) FOR PROTOTYPING AND EARLY FIELD TESTING
- EPROM ADAPTOR SOCKET PROVIDES 40-PIN TO 28-PIN CONVERSION (PART # RTC/PGM320A-06)
- EPROM INCLUDES SECURITY FEATURE FOR CODE PROTECTION

(10-3-20)

SUMMARY - 1ST GENERATION TMS320

	PROCESS	SPEED (MHz)	ON-CHIP PROG ROM (WORDS)	ON-CHIP PROG EPROM (WORDS)	ON-CHIP DATA RAM (WORDS)	SERIAL PORT	TIMER	CO-PROCESSOR I/F
TMS32010	3u NMOS	20	1.5K	—	144	—	—	—
TMS32010-25	3u NMOS	25	1.5K	—	144	—	—	—
TMS32010-14	3u NMOS	14	1.5K	—	144	—	—	—
TMS32011	3u NMOS	20	1.5K	—	144	2	1	—
TMS320C10	2u CMOS	20	1.5K	—	144	—	—	—
TMS320C10-25	2u CMOS	25	1.5K	—	144	—	—	—
TMS320C15	2u CMOS	20	4K	—	256	—	—	—
TMS320C15-25	2u CMOS	25	4K	—	256	—	—	—
TMS320E15	2u CMOS	20	—	4K	256	—	—	—
TMS320C17	2u CMOS	20	4K	—	256	2	1	YES
TMS320C17-25	2u CMOS	25	4K	—	256	2	1	YES
TMS320E17	2u CMOS	20	—	4K	256	2	1	YES

(10-3-36)

TMS320C10 INSTRUCTION SET CHARACTERISTICS

- A TOTAL OF 60 INSTRUCTIONS INCLUDING NUMERICAL, LOGICAL, BRANCHES, AND CONTROL INSTRUCTIONS
- 16-BIT INSTRUCTION WORD - MAJORITY OF INSTRUCTIONS ARE SINGLE WORD, ONLY BRANCH INSTRUCTIONS ARE TWO WORDS
- MAJORITY OF INSTRUCTIONS CAN BE EXECUTED IN A SINGLE MACHINE CYCLE - ONLY BRANCHES AND TRANSFERS BETWEEN MEMORY SPACES REQUIRE MULTIPLE CYCLES
- INSTRUCTIONS CAN BE EXECUTED AT FULL SPEED FROM ON-CHIP OR OFF-CHIP PROGRAM MEMORY

(10-3-21)

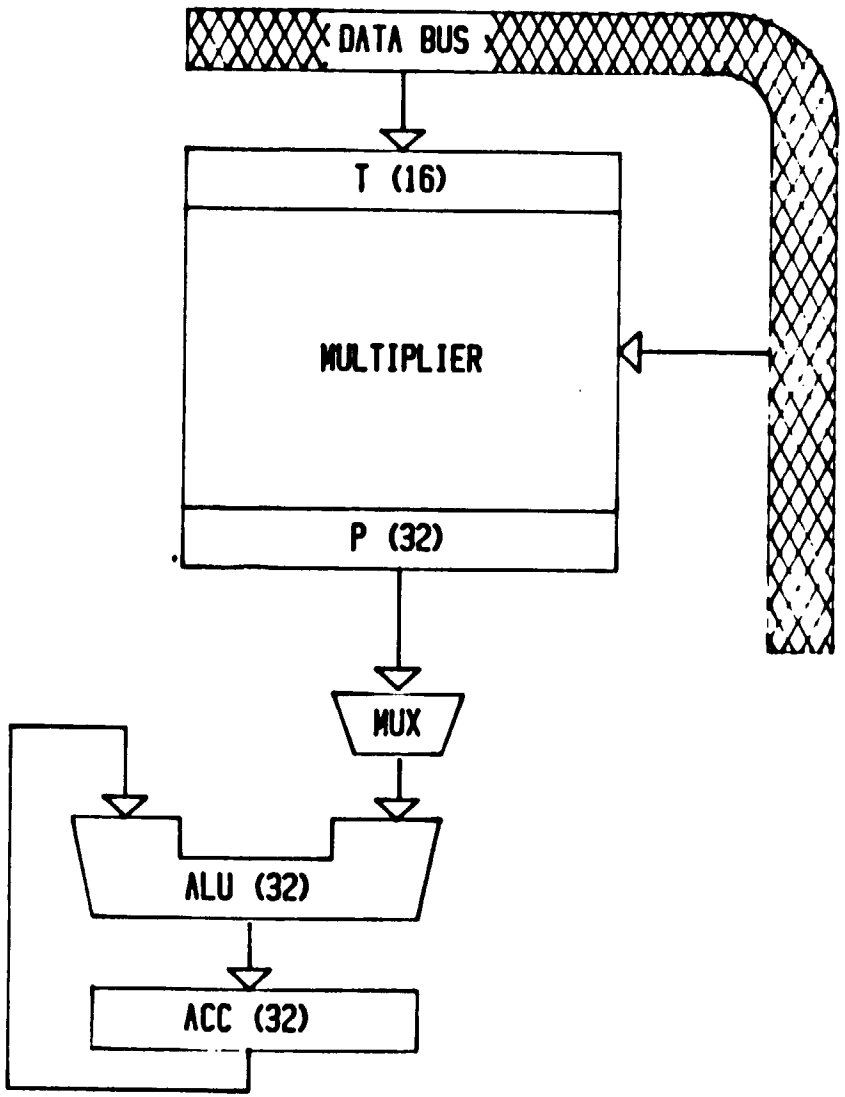
TMS320C10 ADDRESSING MODES

- IMMEDIATE ADDRESSING
  - SHORT CONSTANT (PART OF THE INSTRUCTION) IS THE OPERAND
- DIRECT ADDRESSING (OPERAND IN DATA MEMORY)
  - DP REGISTER PLUS MEMORY ADDRESS IN THE INSTRUCTION FORMS OPERAND ADDRESS
- INDIRECT ADDRESSING (OPERAND IN DATA MEMORY)
  - USES AUXILIARY REGISTER CONTENTS AS OPERAND ADDRESS

(10-3-22)

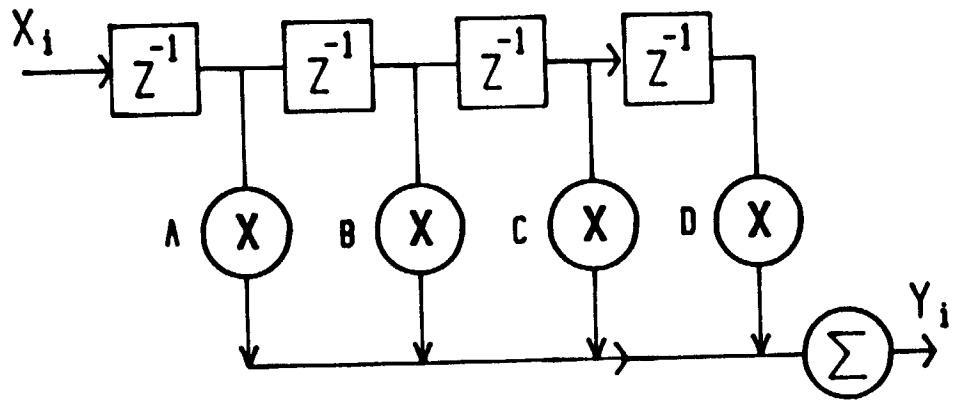
TMS320C1X EXAMPLE - SUM OF PRODUCTS

COMPUTE Y = AX1 + BX2 + CX3 + DX4



ZAC		ACC=0
LT	X1	T=X1
MPY	A	P=AX1
LTA	X2	ACC=AX1; T=X2
MPY	B	P=BX2
LTA	X3	ACC=AX1+BX2; T=X3
MPY	C	P=CX3
LTA	X4	ACC=AX1+BX2+CX3; T=X4
MPY	D	P=DX4
APAC		ACC=AX1+BX2+CX3+DX4
SACH	Y1	STORE 32-BIT RESULT
SACL	Y2	AT LOCATIONS Y1, Y2

TMS320C1X EXAMPLE - FIR FILTER



$$Y(n) = AX(n-1) + BX(n-2) + CX(n-3) + DX(n-4)$$

```
START   IN      X1, PA0     INPUT NEW SAMPLE
        ZAC
        LT      X4
        MPY     D
        LTD     X3          ACC = DX4; X3 -> X4
        MPY     C
        LTD     X2          ACC = DX4 + CX3; X2 -> X3
        MPY     B
        LTD     X1          ACC = DX4 + CX3 + BX2; X1 -> X2
        MPY     A
        APAC                ACC = DX1 + CX3 + BX2 + AX1
        SACH    Y
        OUT     Y, PA0
        B START
```

SUMMARY - 1ST GENERATION TMS320

	PROCESS	SPEED (MHz)	ON-CHIP PROG ROM (WORDS)	ON-CHIP PROG EPROM (WORDS)	ON-CHIP DATA RAM (WORDS)	SERIAL PORT	TIMER	CO-PROCESSOR I/F
TMS32010	3u NMOS	20	1.5K	---	144	---	---	---
TMS32010-25	3u NMOS	25	1.5K	---	144	---	---	---
TMS32010-14	3u NMOS	14	1.5K	---	144	---	---	---
TMS32011	3u NMOS	20	1.5K	---	144	2	1	---
TMS320C10	2u CMOS	20	1.5K	---	144	---	---	---
TMS320C10-25	2u CMOS	25	1.5K	---	144	---	---	---
TMS320C15	2u CMOS	20	4K	---	256	---	---	---
TMS320C15-25	2u CMOS	25	4K	---	256	---	---	---
TMS320E15	2u CMOS	20	---	4K	256	---	---	---
TMS320C17	2u CMOS	20	4K	---	256	2	1	YES
TMS320C17-25	2u CMOS	25	4K	---	256	2	1	YES
TMS320E17	2u CMOS	20	---	4K	256	2	1	YES

(10-3-38)

TMS320 BENCHMARKS

	TMS320C10	TMS320C25
64 POINT COMPLEX FFT		
RADIX 4, STRAIGHT LINE	555 us (*)	217 us
256 POINT COMPLEX FFT		
RADIX 4, STRAIGHT LINE	N/A	1.22 ms
RADIX 2, STRAIGHT LINE	N/A	2.15 ms
RADIX 2, LOOP	20 ms (*)	4.03 ms
1024 POINT COMPLEX FFT		
RADIX 4, STRAIGHT LINE	N/A	7.1 ms
RADIX 2, STRAIGHT LINE	N/A	15.1 ms
RADIX 4, LOOP	42.3 ms (*)	17.5 ms
RADIX 2, LOOP	69.4 ms (*)	22.5 ms
FIR FILTER TAP	400 ns	100 ns
BIQUAD FILTER TAP	2 us	1 us
LMS ADAPTIVE FILTER TAP	1.4 us	0.4 us

(*) Source: DFT/FFT and Convolution Algorithms (Burris and Parks)
All other sources: TMS320 DSP Applications Book

(10-5-11)

TMS320 FAMILY COMPARISON MATRIX - I

FEATURES	1ST GENERATION	2ND GENERATION
ARCHITECTURE	MODIFIED HARVARD	MODIFIED HARVARD
INSTRUCTIONS	60	133
DATA SPACE	256 x 16	64K x 16
PROGRAM SPACE	4K x 16	64K x 16
DATA RAM (ON-CHIP)	256 x 16	544/288 x 16
PROGRAM RAM (ON-CHIP)	0	0/256 x 16
PROGRAM ROM (ON-CHIP)	4K x 16	4K x 16
AUX. REGISTERS	2 x 16 bit	8 x 16 bit
H/W STACK	4 x 12 bit	8 x 16 bit
MULTIPLY	16 x 16 = 32 bits	16 x 16 = 32 bits
ALU	32-bit	32-bit

(10-3-88)

TMS320 FAMILY COMPARISON MATRIX - II

FEATURES	1ST GENERATION	2ND GENERATION
ACCUMULATOR	32-bits	32-bits
REPEAT?	NO	YES
ON-CHIP TIMER	YES	YES
INTERFACE		
EXT. HW INT.	1	3
SERIAL PORT	2	SYNCH. I/O, STATIC, DOUBLE BUFFERED
I/O PORTS	8 x 16 bits	16 x 16 bits
DMA INTERFACE	NO	YES
WAIT STATES	NO	YES
GLOBAL MEMORY	NO	YES

(10-3-69)

TMS320 FAMILY COMPARISON MATRIX - III

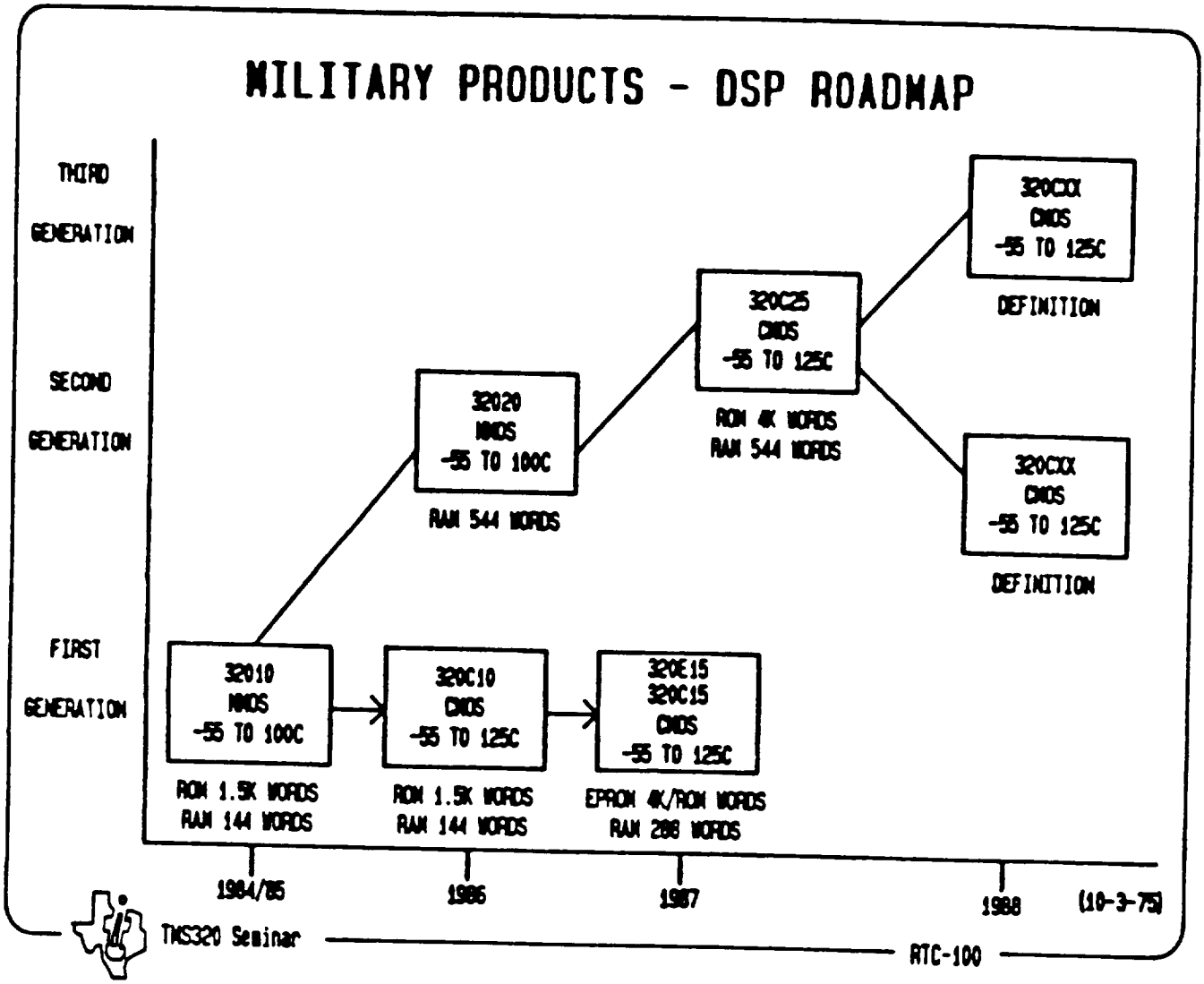
PERFORMANCE	1ST GENERATION	2ND GENERATION
CYCLE TIME	160/200 NS	100 NS
MULTIPLY/ACCUMULATE	320/400 NS	100 NS
RELATIVE PERFORMANCE	1X	4-6X
INDIRECT ADDRESSING MODE OPTIONS	INCREMENT/DECREMENT	INCREMENT/DECREMENT/ INDEXED/BIT REVERSED
TECHNOLOGY		
PROCESS	1.8u CMOS	1.8u CMOS
TYPICAL POWER	165mW	500mW
PACKAGE	40-PIN DIP/PLCC	68-PIN PLCC & PGA

MEMORIES FOR THE TMS320 FAMILY

	Access Time	Vendor	P/N
1st Gen -20 MHz			
SRAM (4Kx4)	70 ns	INMOS	IMS1420M-70
PROM (2Kx8)	45 ns	Texas Instruments	TBP38L165-45
EPROM (8Kx8)	70 ns	Wafer Scale Integration	WS57C49-70
(8Kx8)	70 ns	Wafer Scale Integration	WS57C49-70
1st Gen -25 MHz			
SRAM (4Kx4)	55 ns	INMOS	IMS1423M-55
PROM (2Kx8)	45 ns	Texas Instruments	TBP38L165-45
EPROM (2Kx8)	50 ns	Texas Instruments	TBP27C292-50
2nd Gen - 20 MHz			
SRAM (4Kx4)	50 ns	INMOS	IMS1421-50
PROM (2Kx8)	45 ns	Texas Instruments	TBP38L165-45
EPROM (8Kx8)	70 ns	Wafer Scale Integration	WS57C64F-70
(8Kx8)	70 ns	Wafer Scale Integration	WS67C49-70
2nd Gen -40 MHz			
SRAM (4Kx4)	25 ns	Cypress	CY7C169-25
PROM (2Kx8)	35 ns	Texas Instruments	TBP38L165-35
EPROM (2Kx8)	45 ns	Texas Instruments	TBP27C292-45

(10-4-6)

ORIGINAL PAGE IS
OF POOR QUALITY


MILITARY PRODUCTS - DSP ROADMAP
THIRD GENERATION
SECOND GENERATION
FIRST GENERATION
32010 NMOS -55 TO 100C
ROM 1.5K WORDS RAM 144 WORDS
32020 NMOS -55 TO 100C
RAM 544 WORDS
320C10 CMOS -55 TO 125C
ROM 1.5K WORDS RAM 144 WORDS
320C25 CMOS -55 TO 125C
ROM 4K WORDS RAM 544 WORDS
320E15 320C15 CMOS -55 TO 125C
EPROM 4K/ROM WORDS RAM 288 WORDS
320CXX CMOS -55 TO 125C
DEFINITION
320CXX CMOS -55 TO 125C
DEFINITION
1984/85
1986
1987
1988
(10-3-75)
TMS320 Seminar
RTC-100

AGENDA

* INTRODUCTION
* SECOND GENERATION 320 (320C25)
* FIRST GENERATION 320 (320C10)
* BREAK
* THIRD GENERATION 320 (320C30)
* TOOLS
* APPLICATIONS

A HIGH SPEED

FLOATING POINT DSP

FROM

TEXAS INSTRUMENTS

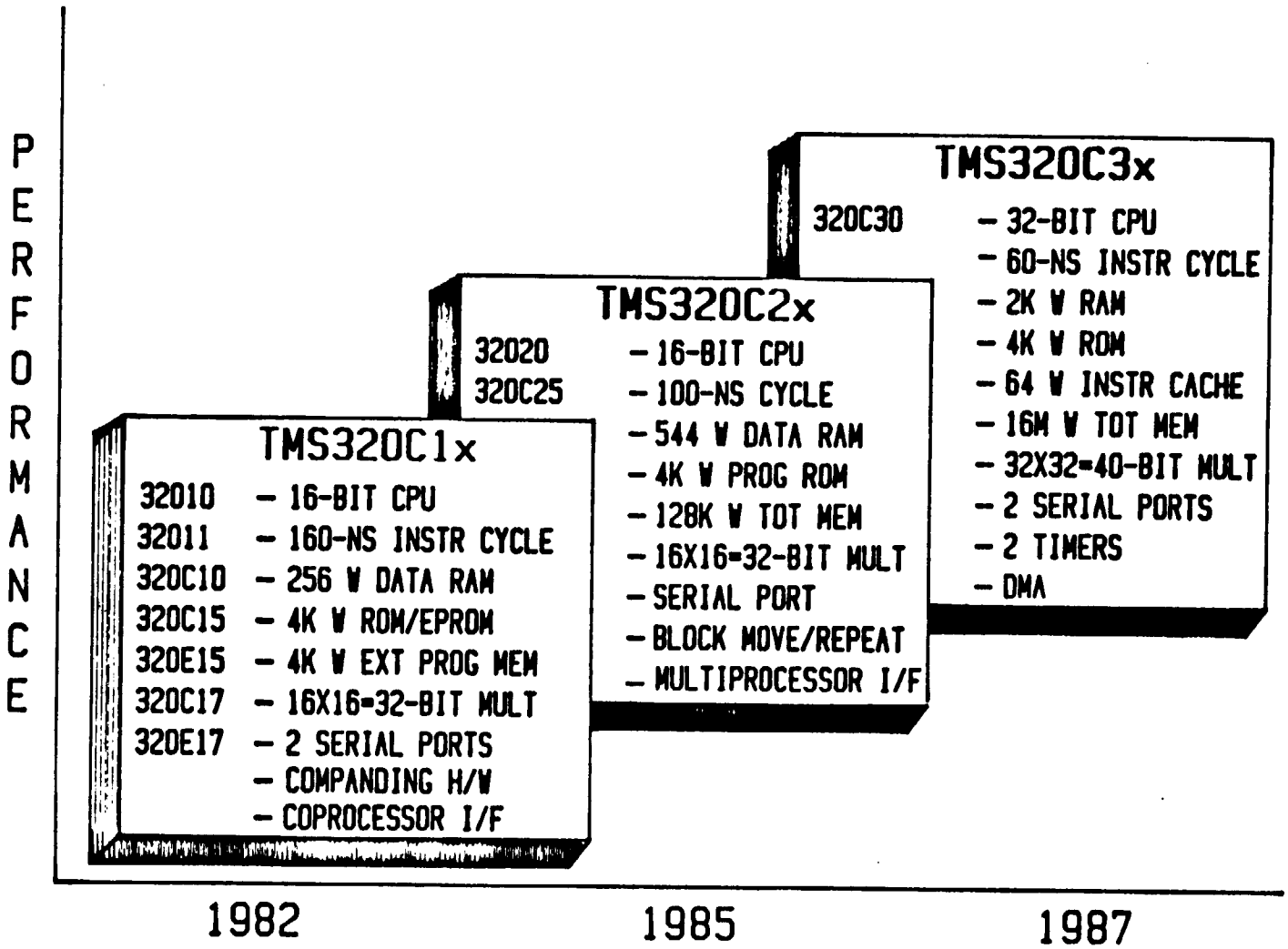
THE

TMS320C30

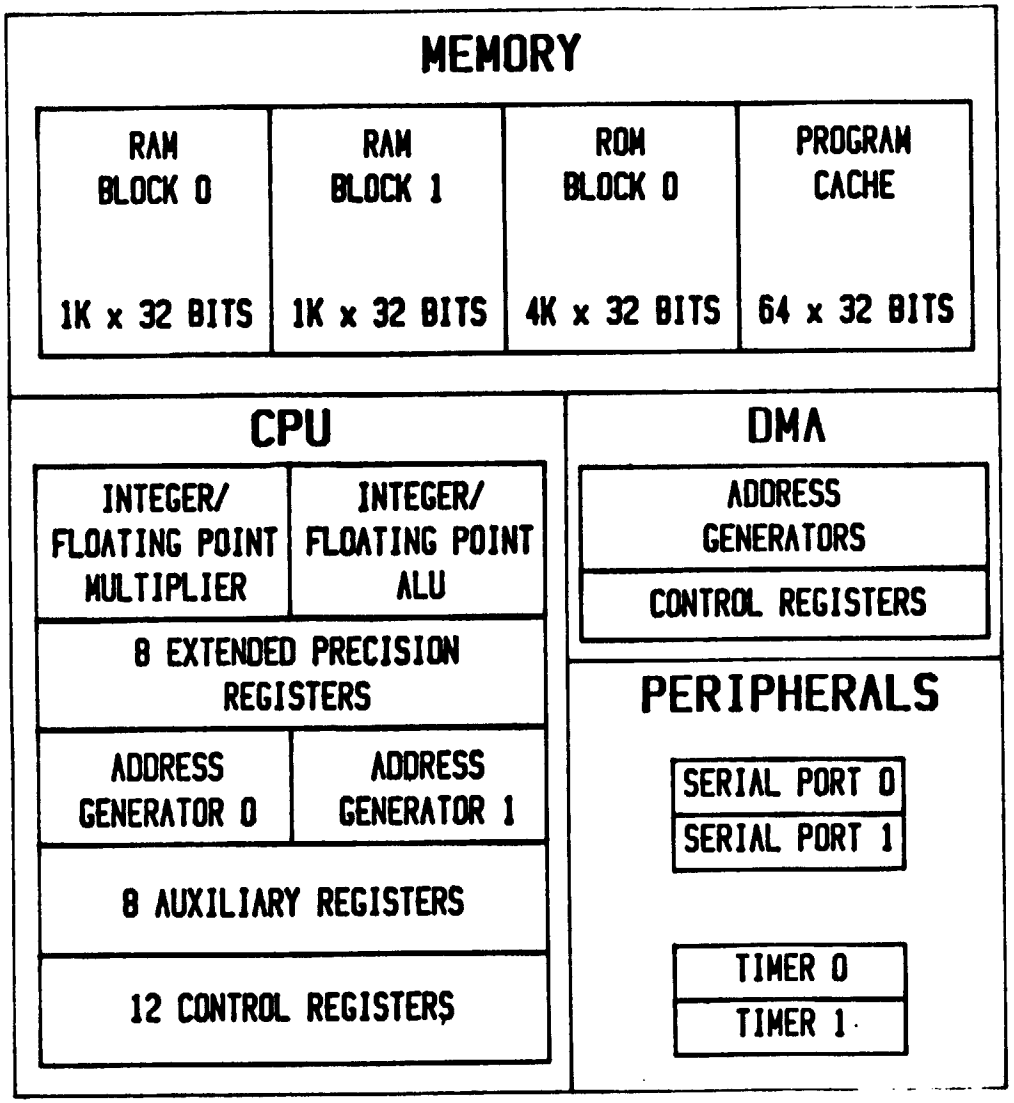
TMS320C30 PRESENTATION OUTLINE

- TMS320 FAMILY MIGRATION PATHS
- TMS320C30 KEY FEATURES.
- TMS320C30 BLOCK DIAGRAM.
- DETAILS OF TMS320C30 ARCHITECTURE.
- TMS320 FAMILY BENCHMARKS.
- TMS320C30 SUPPORT TOOLS.
- TMS320C30 PROGRAM SCHEDULE.

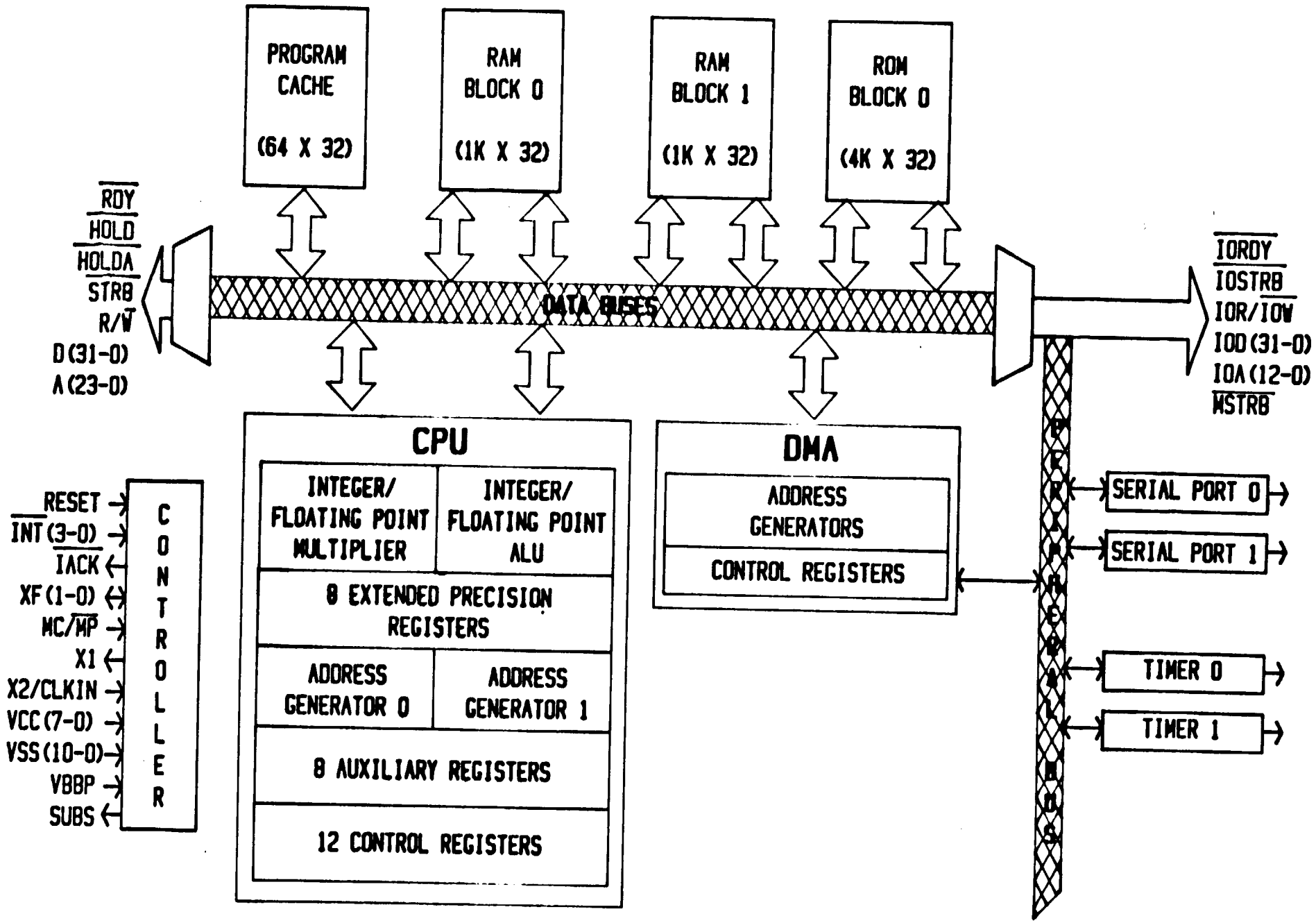
TMS320 FAMILY MIGRATION PATH



TMS320C30 KEY FEATURES

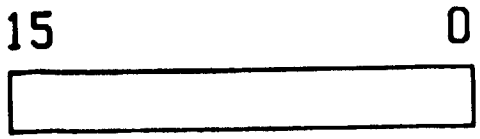


- 60 ns SINGLE-CYCLE EXECUTION TIME, 1u CMOS.
- TWO 1K x 32-BIT SINGLE-CYCLE DUAL-ACCESS RAM BLOCKS.
- ONE 4K x 32-BIT SINGLE-CYCLE DUAL-ACCESS ROM BLOCK.
- 64 x 32-BIT INSTRUCTION CACHE.
- 32-BIT INSTRUCTION AND DATA WORDS, 24-BIT ADDRESSES.
- 32/40 BIT FLOATING-POINT AND INTEGER MULTIPLIER.
- 32/40 BIT FLOATING-POINT, INTEGER, AND LOGICAL ALU.
- 8 EXTENDED-PRECISION REGISTERS.
- TWO ADDRESS-GENERATORS WITH 8 AUXILIARY REGISTERS.
- DMA CONTROLLER FOR CONCURRENT I/O AND CPU OPERATION.
- TWO SERIAL PORTS AND TWO TIMERS.
- HIGH-LEVEL LANGUAGE SUPPORT.
- INTERLOCKED INSTRUCTIONS SUPPORT MULTIPROCESSING.
- ZERO-OVERHEAD LOOPS AND SINGLE-CYCLE BRANCHES.
- 176-PIN MICROPROCESSOR VERSION.
- 100-PIN MICROCOMPUTER VERSION.


PROGRAM CACHE (64 X 32)
RAM BLOCK 0 (1K X 32)
RAM BLOCK 1 (1K X 32)
ROM BLOCK 0 (4K X 32)
RDY
HOLD
HOLDA
STRB
R/W
D(31-0)
A(23-0)
DATA BUSES
IORDY
IOSTRB
IOR/IOW
IOD(31-0)
IOA(12-0)
MSTRB
CPU
INTEGER/ FLOATING POINT MULTIPLIER
INTEGER/ FLOATING POINT ALU
8 EXTENDED PRECISION REGISTERS
ADDRESS GENERATOR 0
ADDRESS GENERATOR 1
8 AUXILIARY REGISTERS
12 CONTROL REGISTERS
DMA
ADDRESS GENERATORS
CONTROL REGISTERS
PERIPHERAL BUS
SERIAL PORT 0
SERIAL PORT 1
TIMER 0
TIMER 1
RESET
INT(3-0)
IACK
XF(1-0)
MC/MP
X1
X2/CLKIN
VCC(7-0)
VSS(10-0)
VBBP
SUBS
CONTROLLER

TMS320C30 INTEGER FORMATS

- SHORT INTEGER FORMAT.

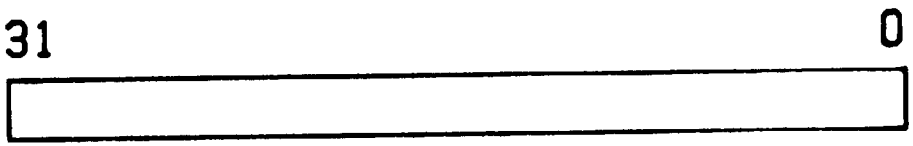


* RANGE $-2^{15} \leq si \leq 2^{15}-1$

– 16-BIT FORMAT USED FOR IMMEDIATE SIGNED OR UNSIGNED INTEGER OPERANDS

si - SIGNED INTEGER

- SINGLE-PRECISION INTEGER FORMAT.



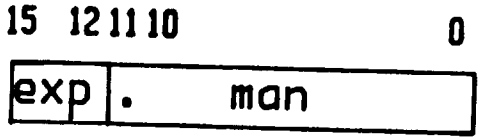
* RANGE $-2^{31} \leq si \leq 2^{31}-1$

– 32-BIT SINGLE-PRECISION SIGNED OR UNSIGNED INTEGERS

si - SIGNED INTEGER

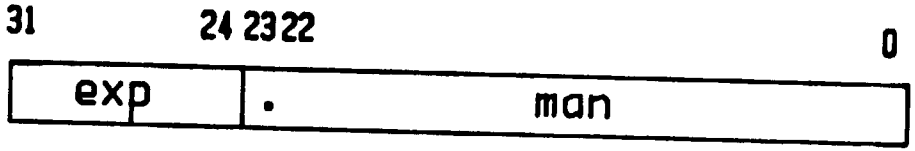
TMS320C30 FLOATING POINT FORMATS

0 SHORT FLOATING-POINT FORMAT.



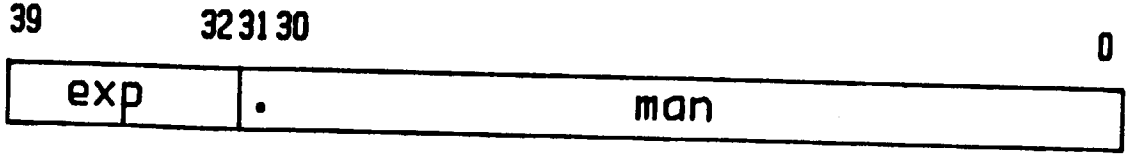
MOST POSITIVE	2.5594×10^{2}
LEAST POSITIVE	7.8125×10^{-3}
LEAST NEGATIVE	-7.8163×10^{-3}
MOST NEGATIVE	-2.5600×10^{2}

0 SINGLE-PRECISION FLOATING-POINT FORMAT.



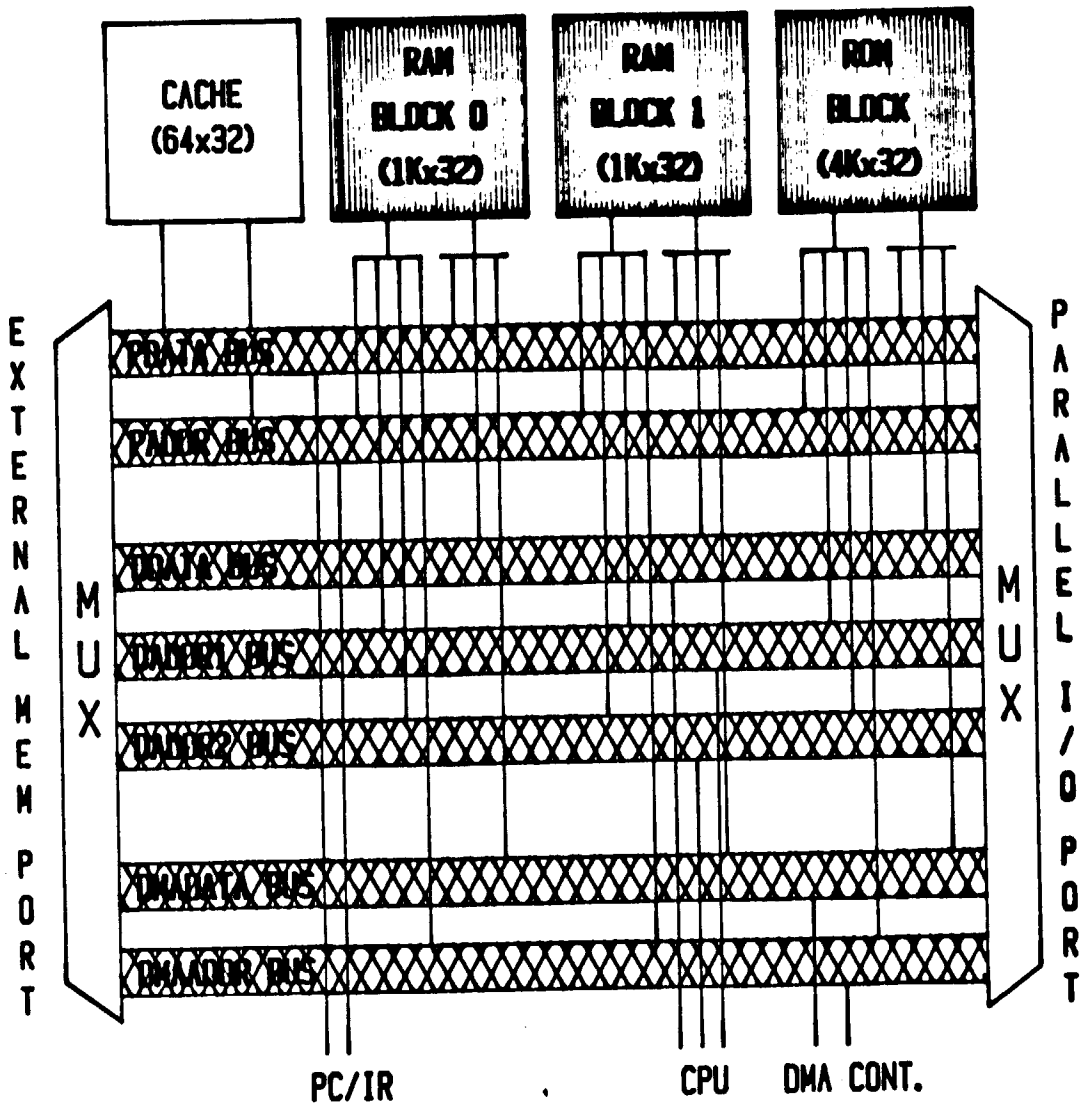
MOST POSITIVE	3.4028234×10^{38}
LEAST POSITIVE	$5.8774717 \times 10^{-39}$
LEAST NEGATIVE	$-5.8774724 \times 10^{-39}$
MOST NEGATIVE	$-3.4028236 \times 10^{38}$

0 EXTENDED-PRECISION FLOATING-POINT FORMAT.



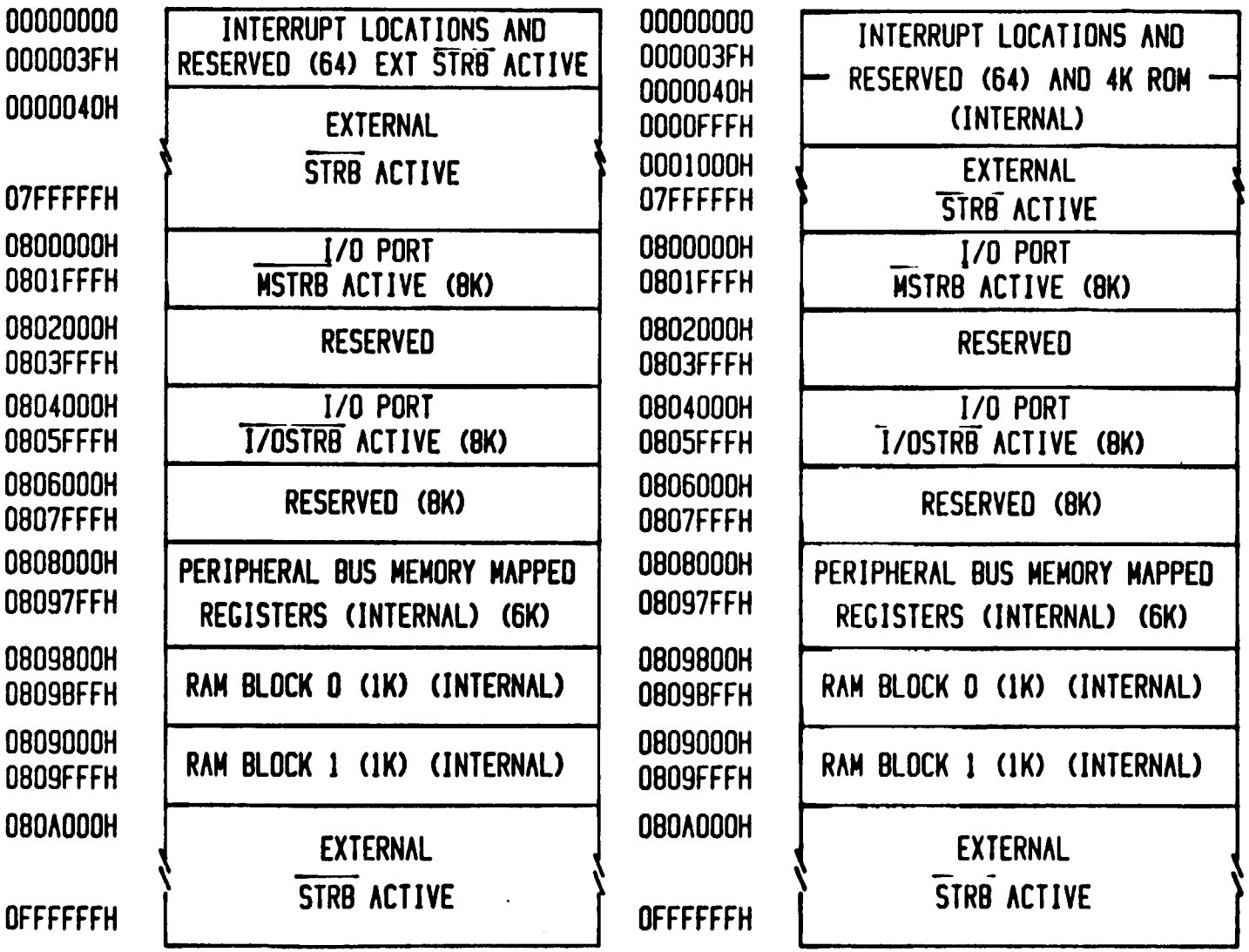
MOST POSITIVE	$3.4028236683 \times 10^{38}$
LEAST POSITIVE	$5.8774717541 \times 10^{-39}$
LEAST NEGATIVE	$-5.8774717569 \times 10^{-39}$
MOST NEGATIVE	$-3.4028236691 \times 10^{38}$

TMS320C30 MEMORY SPACE



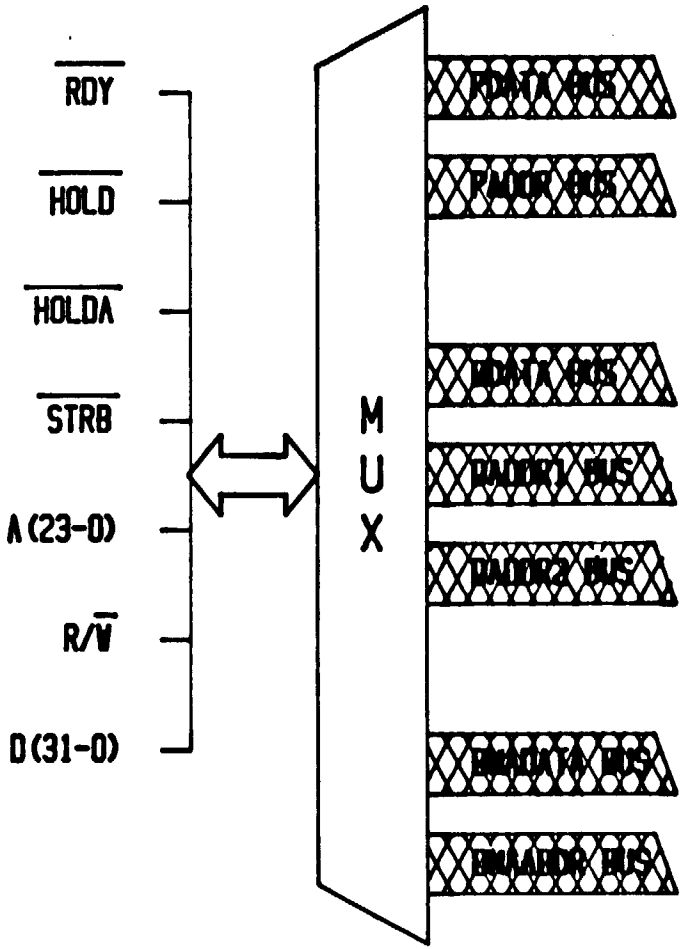
- 16-MEGAWORD SINGLE LOGICAL MEMORY SPACE
  - PROGRAM
  - DATA
  - I/O
- TWO 1Kx32 BIT ON-CHIP RAM BLOCKS
- ONE 4Kx32 BIT ON-CHIP ROM BLOCK
- EACH RAM OR ROM BLOCK SUPPORTS TWO ACCESSES PER SINGLE CYCLE
- MULTIPLE BUSES SUPPORTS MULTIPLE PHYSICAL ADDRESS SPACES
- TABLES, COEFFICIENTS, PROGRAM CODE, OR DATA CAN BE STORED IN RAM OR ROM

TMS320C30 MEMORY MAP



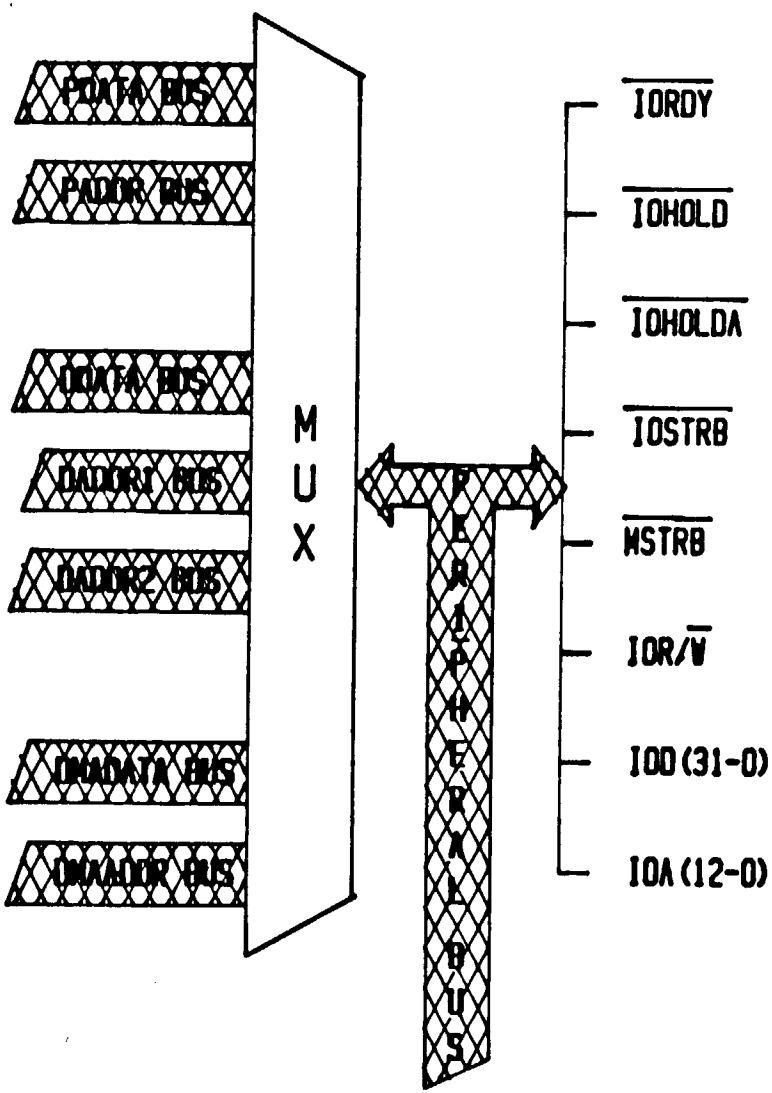
- SINGLE LOGICAL 16M-WORD MEMORY THAT CONTAINS PROGRAM, DATA AND I/O SPACE
- MEMORY MAP CONTAINS
  - ALL ON-CHIP MEMORY
  - ALL ON-CHIP PERIPHERALS
  - ALL EXTERNAL INTERFACES
- ON-CHIP ROM IS ENABLED IN MICROCOMPUTER MODE

TMS320C30 EXTERNAL MEMORY PORT



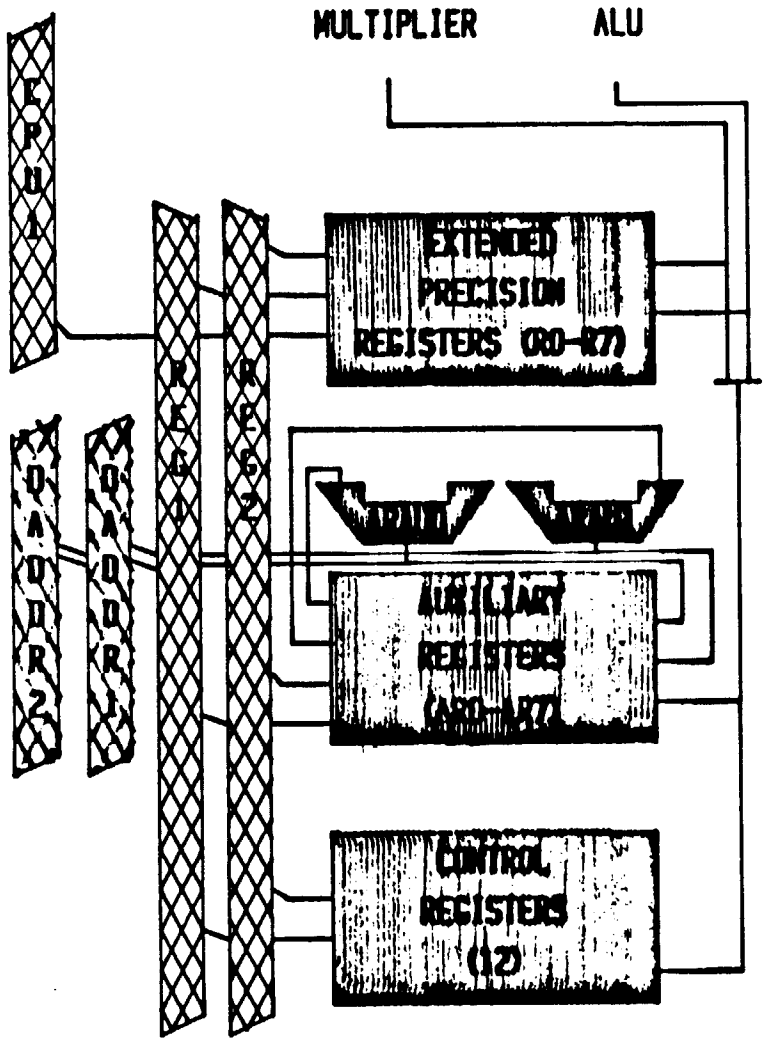
- 32-BIT EXTERNAL DATA, 24-BIT EXTERNAL ADDRESS BUSES
- SUPPORTS SINGLE CYCLE ACCESSES
- SOFTWARE PROGRAMMABLE WAIT STATES PLUS READY LINE
- HOLD LOGIC FOR TRI-STATING PORT
- 16 MEGAWORDS TOTAL ADDRESS SPACE
- MEMORY-MAPPED ACCESS
- BANK-SWITCH DETECTION

TMS320C30 PARALLEL I/O PORT



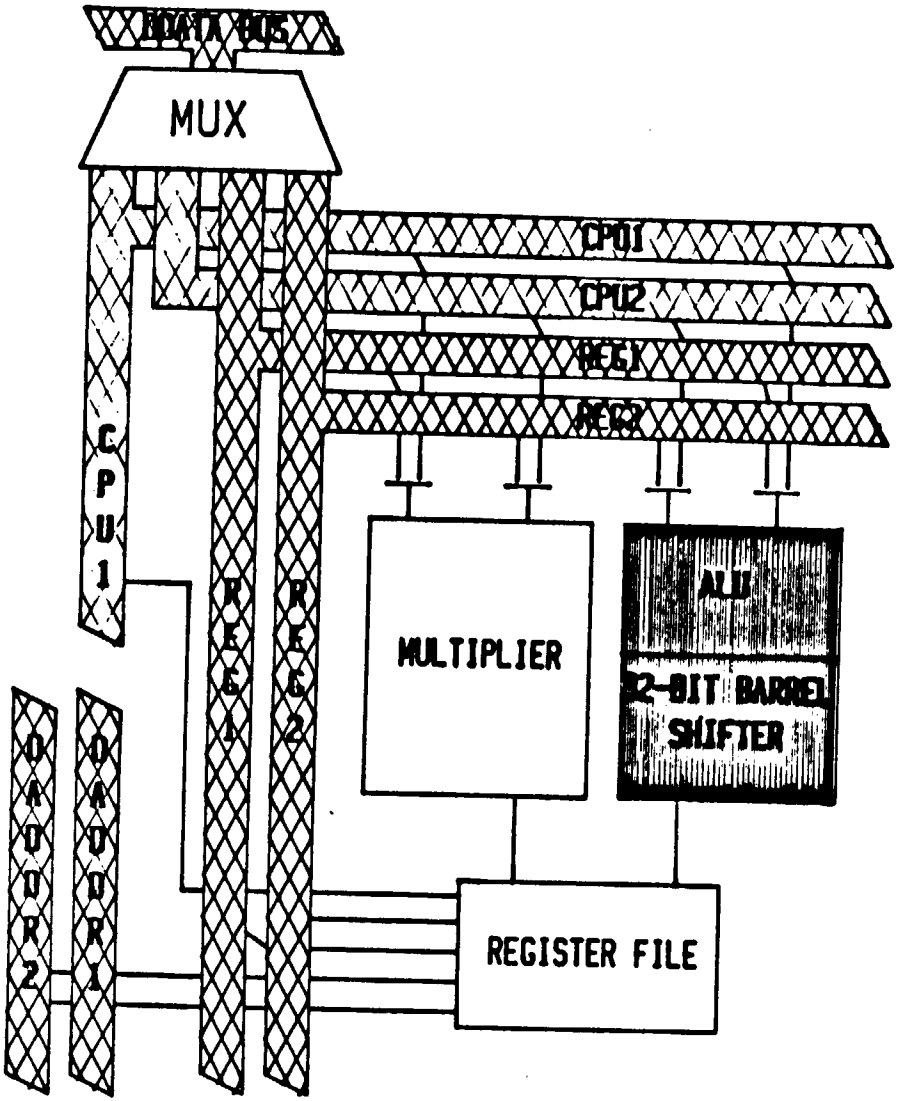
- 32-BIT EXTERNAL DATA, 13-BIT EXTERNAL ADDRESS BUSES
- SUPPORTS SINGLE CYCLE ACCESSES
- SOFTWARE PROGRAMMABLE WAIT STATES PLUS READY LINE
- MEMORY-MAPPED ACCESS
- SUPPORT FOR I/O INTERFACES

TMS320C30 REGISTER FILE



* EXTENDED PRECISION REGISTERS
  - SUPPORTS OPERATIONS ON 40-BIT FLOATING POINT AND 32-BIT INTEGER NUMBERS
* AUXILIARY REGISTER ARITHMETIC UNITS (ARAU0 AND ARAU1)
  - CAN GENERATE TWO ADDRESSES IN ONE CYCLE
  - OPERATE IN PARALLEL WITH THE MULTIPLIER AND ALU
  - SUPPORT DISPLACEMENTS AND INDEXES FOR LINEAR, CIRCULAR, AND BIT-REVERSED ADDRESSING
* CONTROL REGISTERS
  - STACK POINTER
  - INTERRUPT MASK AND FLAGS
  - BLOCK REPEAT REGISTERS
  - STATUS REGISTER
  - INDEX REGISTERS

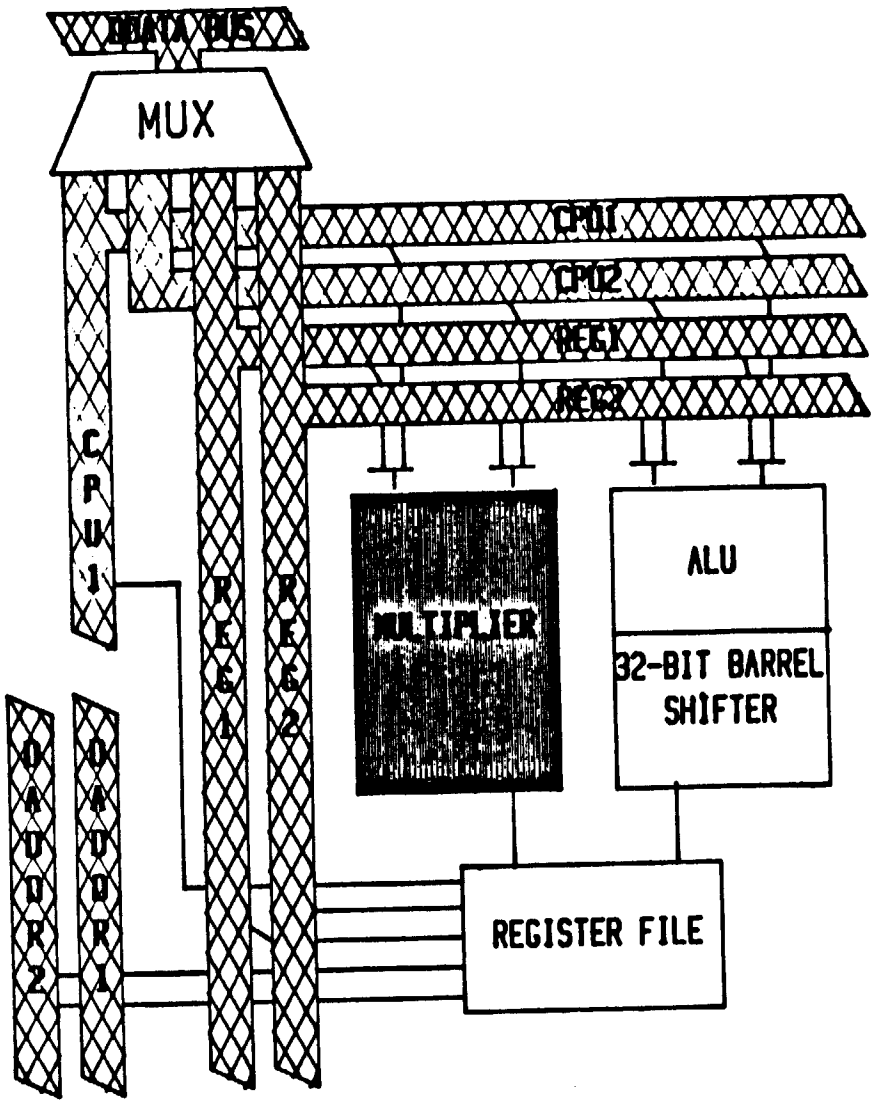
TMS320C30 ALU FEATURES



* SINGLE CYCLE ALU OPERATIONS
* FLOATING POINT OPERATIONS
  - (8-BIT EXP, 32 BIT MAN)+/-(8-BIT EXP, 32-BIT MAN)
    = (8-BIT EXP, 32-BIT MAN)
  - AUTOMATIC HANDLING OF MANTISSA OVERFLOW
  - FLAG SET ON EXPONENT OVERFLOW
* INTEGER OPERATIONS
  - 32-BIT INTEGER OPERATIONS
  - FLAG SET ON OVERFLOW
* LOGICAL OPERATIONS
  - 32-BIT LOGICAL OPERATIONS
* BARREL SHIFTER
  - SHIFTS UP TO 32-BIT RIGHT OR LEFT

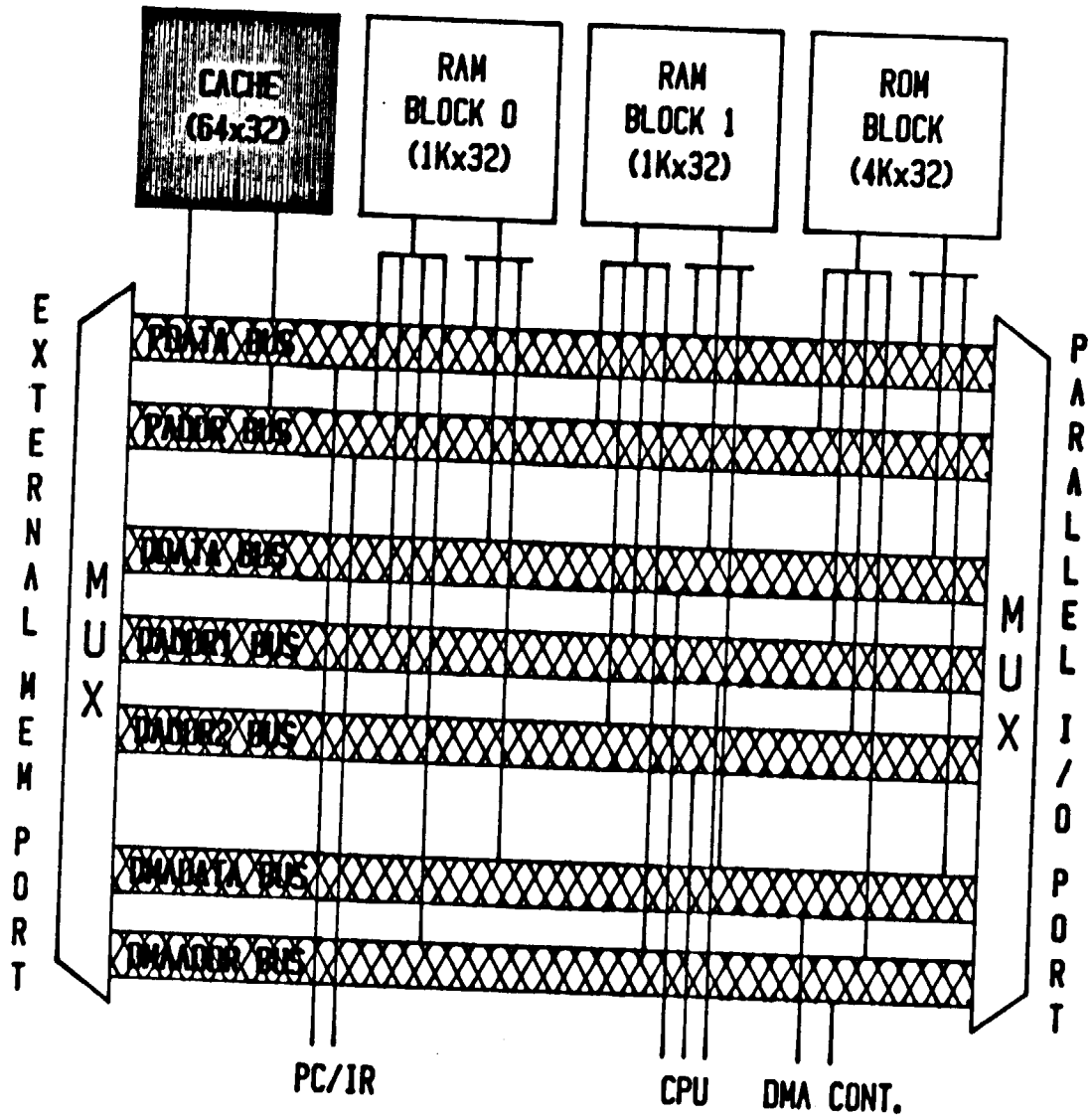
ORIGINAL PAGE IS
OF POOR QUALITY

TMS320C30 MULTIPLIER FEATURES



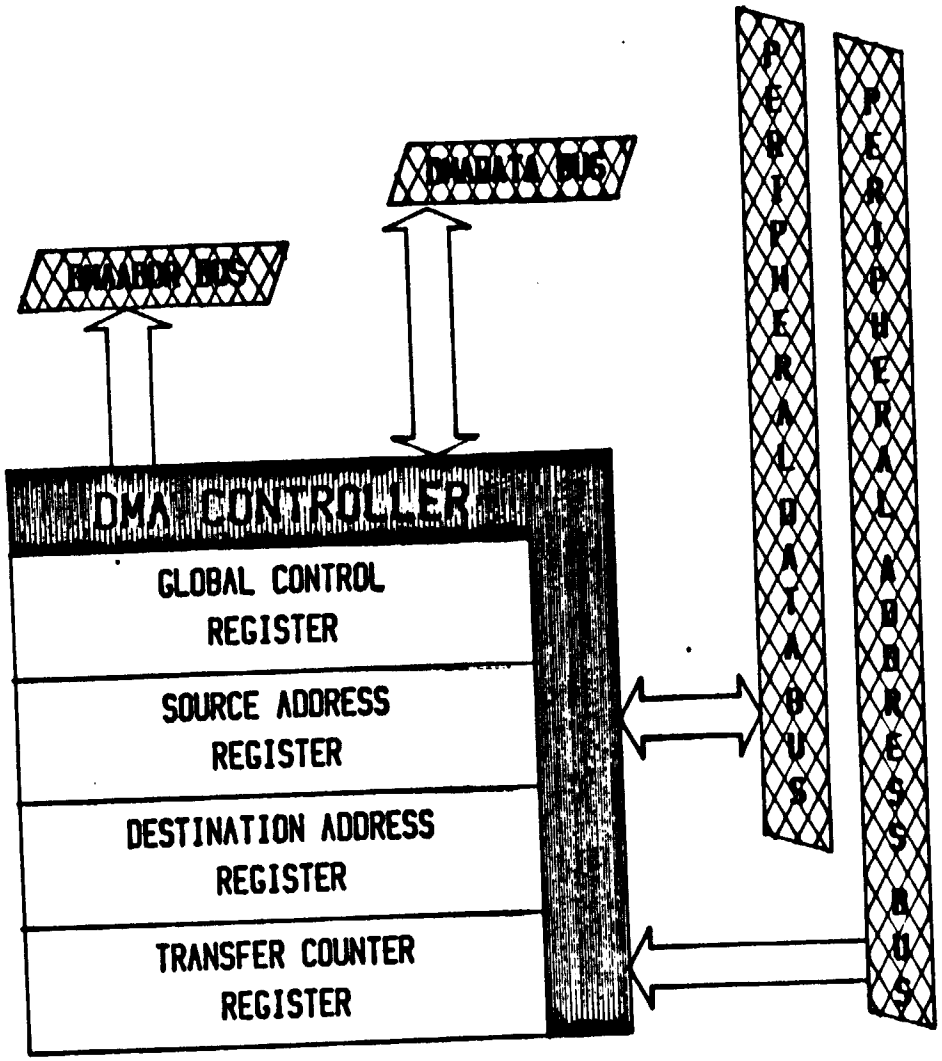
* SINGLE CYCLE MULTIPLICATIONS
* FLOATING POINT OPERATIONS
  - (8-BIT EXP, 24 BIT MAN) x (8-BIT EXP, 24-BIT MAN) = (8-BIT EXP, 32-BIT MAN)
  - AUTOMATIC HANDLING OF MANTISSA OVERFLOW
  - FLAG SET ON EXPONENT OVERFLOW
* INTEGER OPERATIONS
  - (24-BIT INT) x (24-BIT INT) --> (32-BIT INT)
  - FLAG SET ON OVERFLOW

TMS320C30 INSTRUCTION CACHE FEATURES



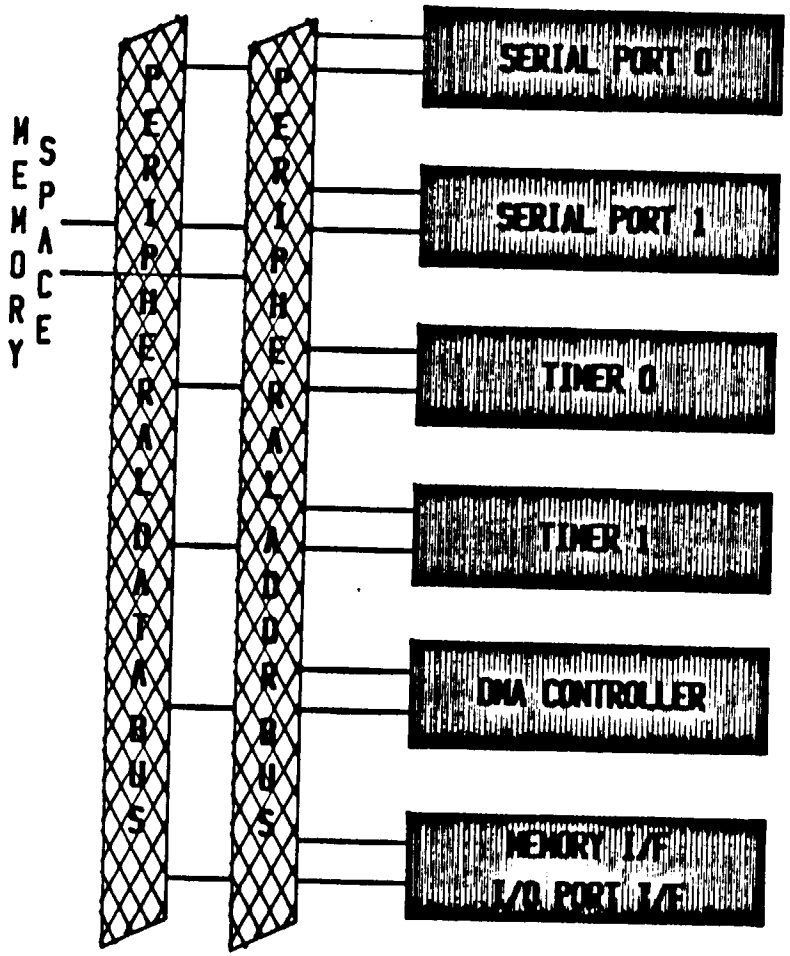
- 64 X 32-BIT GENERAL PURPOSE INSTRUCTION CACHE.
- TWO 32-WORD SEGMENTS UPDATED ON A LEAST-RECENTLY USED BASIS
- USER CONTROL FUNCTIONS:
  - CACHE DISABLE
  - CACHE FREEZE
  - CACHE CLEAR

TMS320C30 DMA CONTROLLER



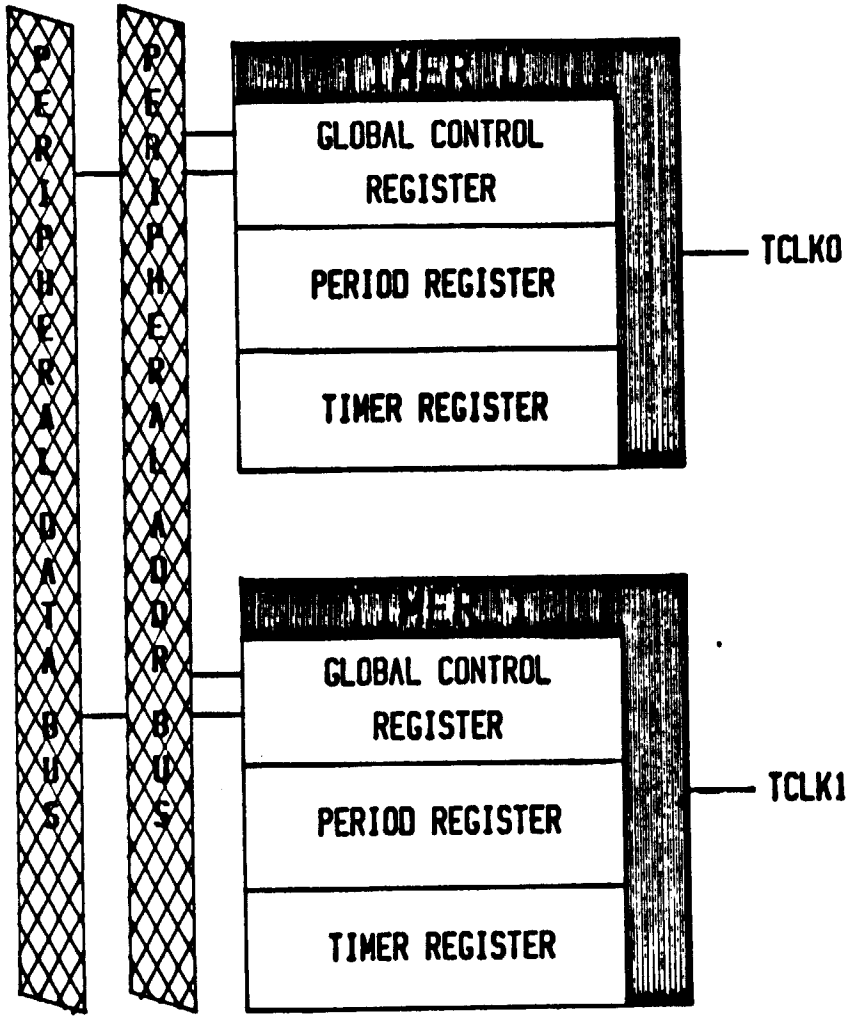
- ON-CHIP DMA CONTROLLER FOR CONCURRENT I/O
- ANY SIZE DATA TRANSFERS AND DATA RATES
- MEMORY TO MEMORY TRANSFERS
- CYCLE-STEAL TRANSFER MODE
- FIVE MEMORY-MAPPED REGISTERS FOR DMA CONTROL:
  - CONTROL AND INTERRUPT MASK REGISTERS
  - SOURCE ADDRESS
  - DESTINATION ADDRESS
  - TRANSFER COUNTER
- PROGRAMMABLE INCREMENTING OR DECREMENTING OF ADDRESSES
- PROGRAMMABLE I/O SYNCHRONIZATION

TMS320C30 PERIPHERAL BUS



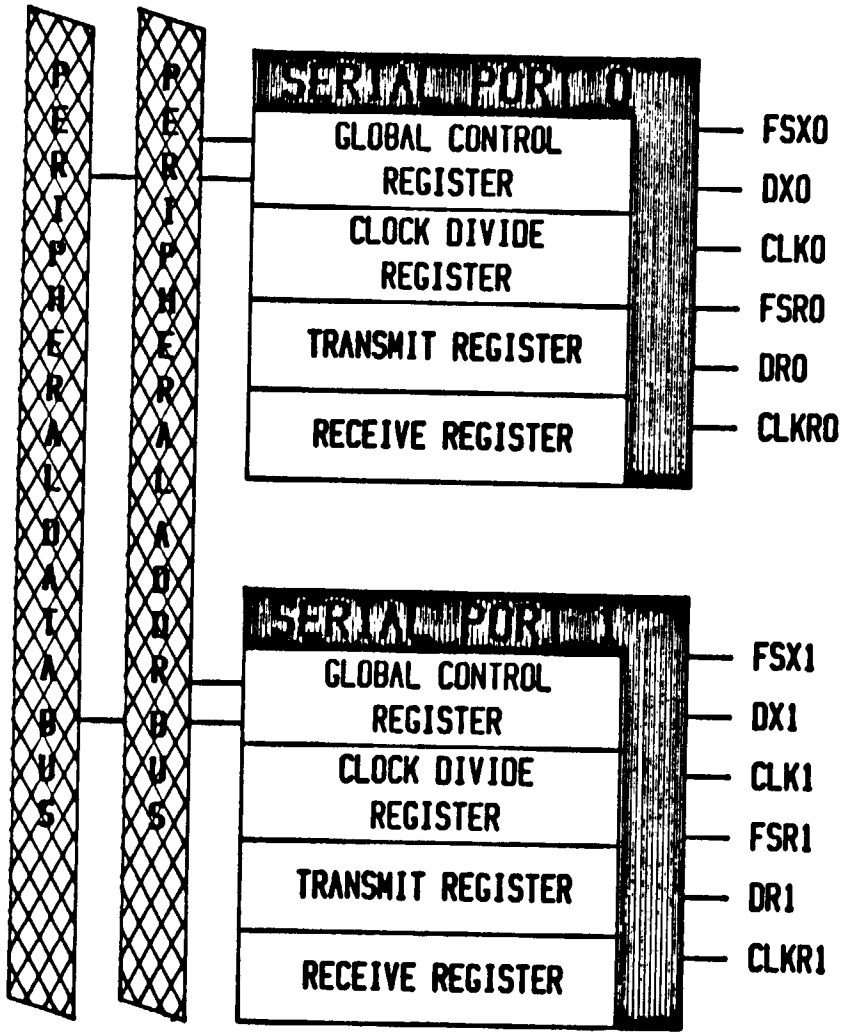
- DEDICATED INTERNAL ADDRESS AND DATA BUSES FOR PERIPHERALS
- CURRENT PERIPHERAL FUNCTIONS INCLUDE:
  - DIRECT MEMORY ACCESS CONTROLLER
  - TWO SERIAL PORTS
  - TWO TIMERS
  - EXTERNAL MEMORY INTERFACE CONTROL
  - PARALLEL I/O PORT CONTROL

TMS320C30 TIMER MODULES



- TWO SEPARATE, INDEPENDANT TIMERS/COUNTERS
- MEMORY-MAPPED REGISTERS FOR EACH TIMER:
  - TIMER CONTROL
  - PERIOD REGISTER
  - COUNTER
- ONE I/O PIN PROGRAMMABLE AS CLOCK INPUT OR STATUS OUTPUT
- 32-BIT COUNTER - MAXIMUM CLOCKING = f(CLKOUT)/2
- INTERNAL OR EXTERNAL CLOCK SOURCE
- PROGRAMMABLE RESET, COUNTER ENABLE, AND OUTPUT CONTROL

TMS320C30 SERIAL PORTS



- TWO SEPARATE, INDEPENDENT PORT MODULES
- 8, 16, 24 AND 32-BIT WORD LENGTHS
- MEMORY MAPPED REGISTERS FOR EACH MODULE
  - CONTROL, TRANSMIT AND RECEIVE BUFFER, CLOCK PERIOD
- PROGRAMMABLE INTERNAL OR EXTERNAL CLOCK SOURCE
- FIXED OR VARIABLE DATA RATE SUPPORT
- HANDSHAKE MODE FOR ZERO-GLUE INTERCONECT
- SIX EXTERNAL PINS PER MODULE:
  - TRANSMIT AND RECEIVE DATA, FRAME SYNC AND CLOCK PINS
- MAXIMUM CLOCK FREQUENCY = f(CLKOUT)/2
- MAXIMUM TRANSFER RATE > 8MHZ
- DOUBLE BUFFERED TRANSMITTER AND RECEIVER

TMS320C30 PIPELINE

INSTRUCTION	CYCLE 1	2	3	4	5	6	7
1	FETCH	DECODE	READ	EXECUTE			
2		FETCH	DECODE	READ	EXECUTE		
3			FETCH	DECODE	READ	EXECUTE	
4				FETCH	DECODE	READ	EXECUTE
5					FETCH	DECODE	READ

* PIPELINE SEQUENCE HANDLED TO GUARANTEE CORRECT EXECUTION SEQUENCE

* "DELAYED" BRANCHES ALLOWED

TMS320C30 ADDRESSING MODES

MODE	EXAMPLE
REGISTER	ADDF R0, R1
DIRECT	ADDF @DMA, R1
SHORT IMMEDIATE	ADDF 3.14, R1

TMS320C30 ADDRESSING MODES

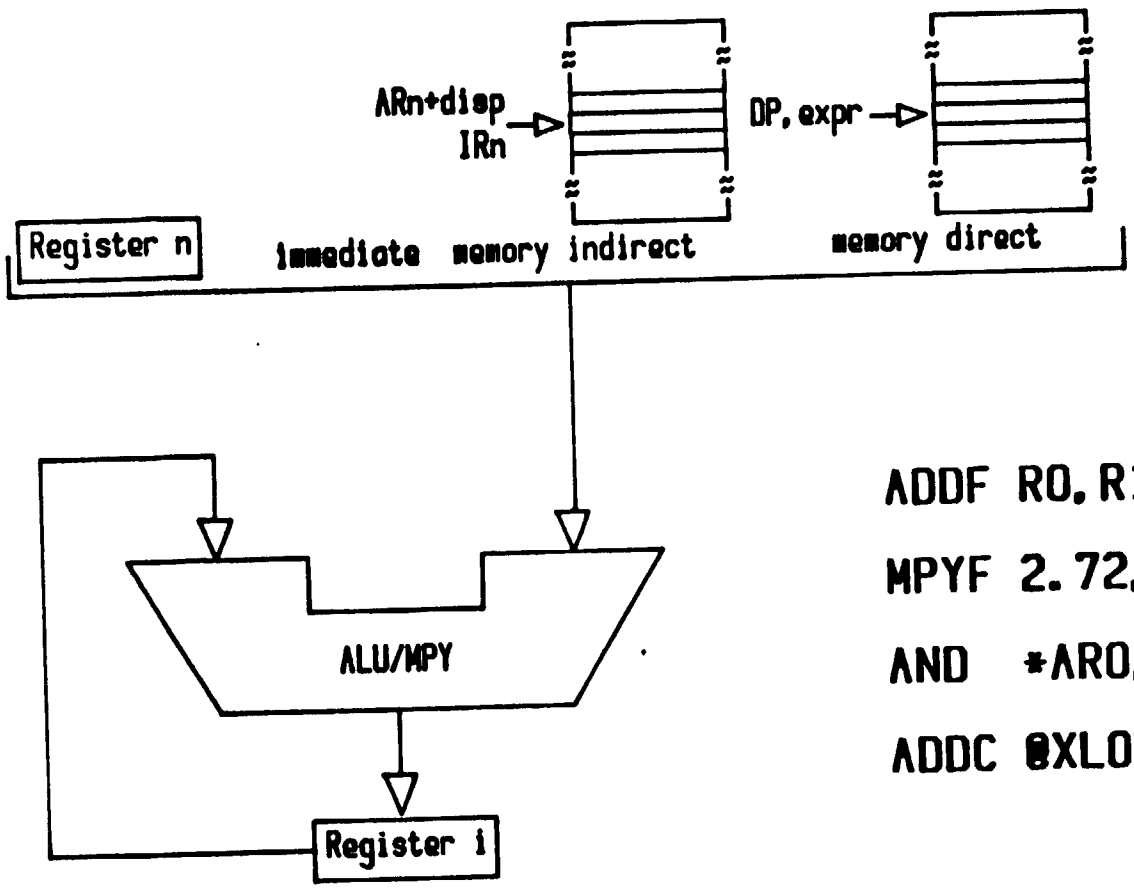
MODE	EXAMPLE
LONG IMMEDIATE	B ADDR
PC RELATIVE	BV LABEL (ASSEMBLER WILL CALCULATE DISPLACEMENT)

TMS320C30 INDIRECT ADDRESSING MODES

MODE	EXAMPLE
INDIRECT	ADDF *+AR1(15), R1
PRE	*+AR0(di) *-AR4(di)
PRE WITH UPDATE	*++AR7(di) *--AR1(di)
POST WITH UPDATE	*AR3++(di) *AR4--(di)
CIRCULAR	*AR5++(di)% *AR6--(di)%
BIT REVERSAL	*AR2++(IR0)B

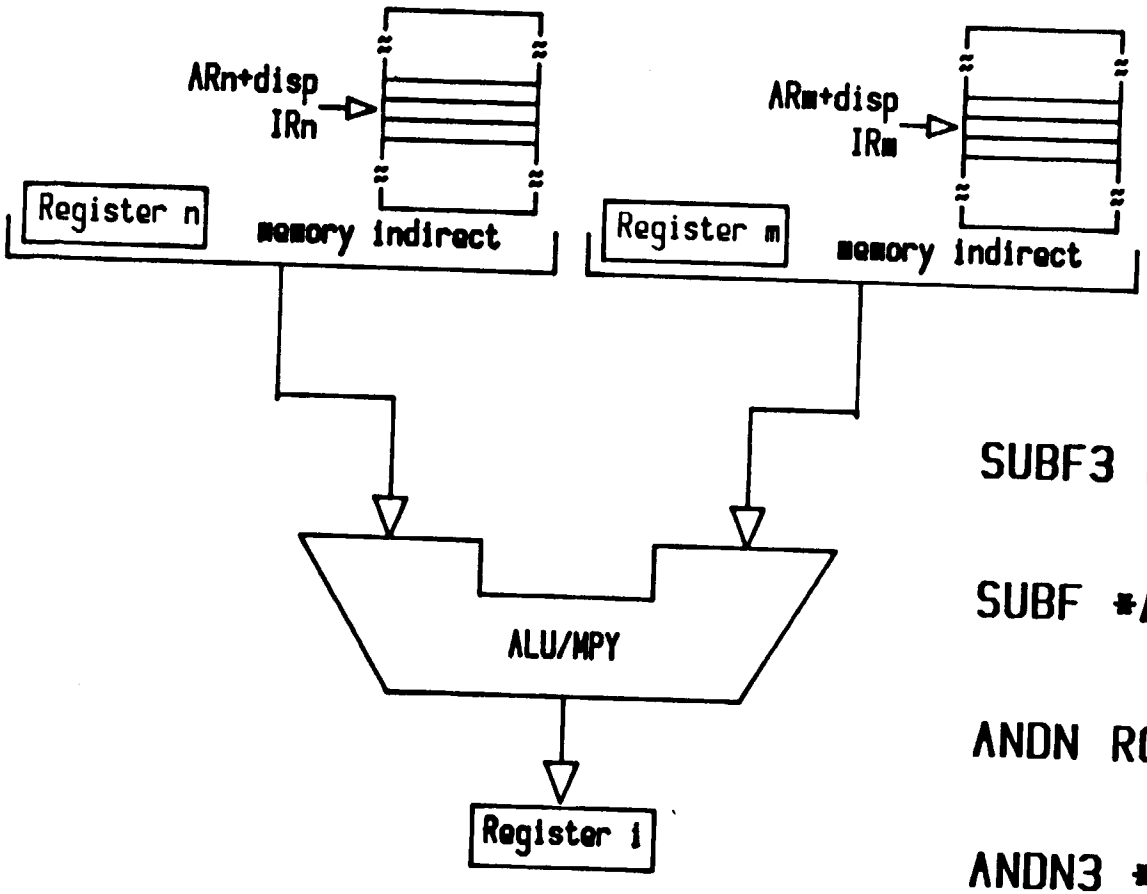
('di' CAN BE A DISPLACEMENT OR AN INDEX REGISTER (IR0 OR IR1))

TMS320C30 TWO-OPERAND INSTRUCTIONS



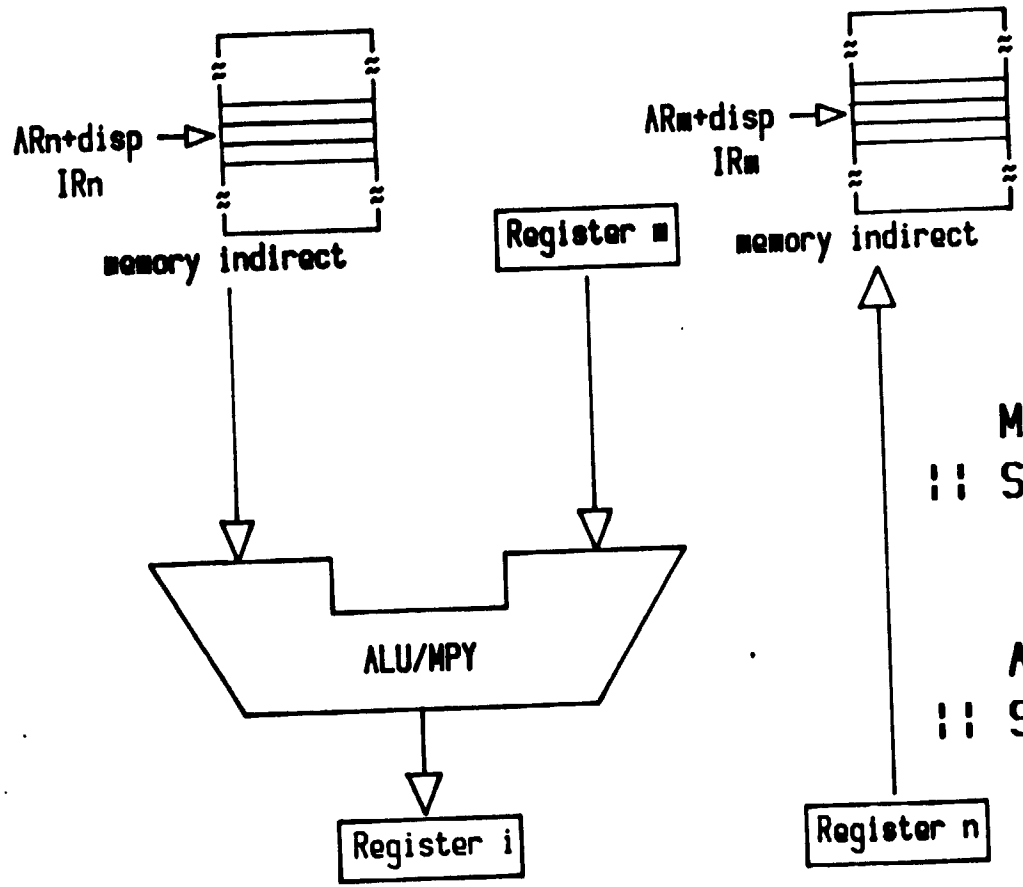

```
ADDF R0, R1          ; REGISTER MODE
MPYF 2.72, R7        ; IMMEDIATE MODE
AND  *AR0, R2        ; INDIRECT MODE
ADDC @XLOW, AR1      ; DIRECT MODE
```

TMS320C30 THREE-OPERAND INSTRUCTIONS



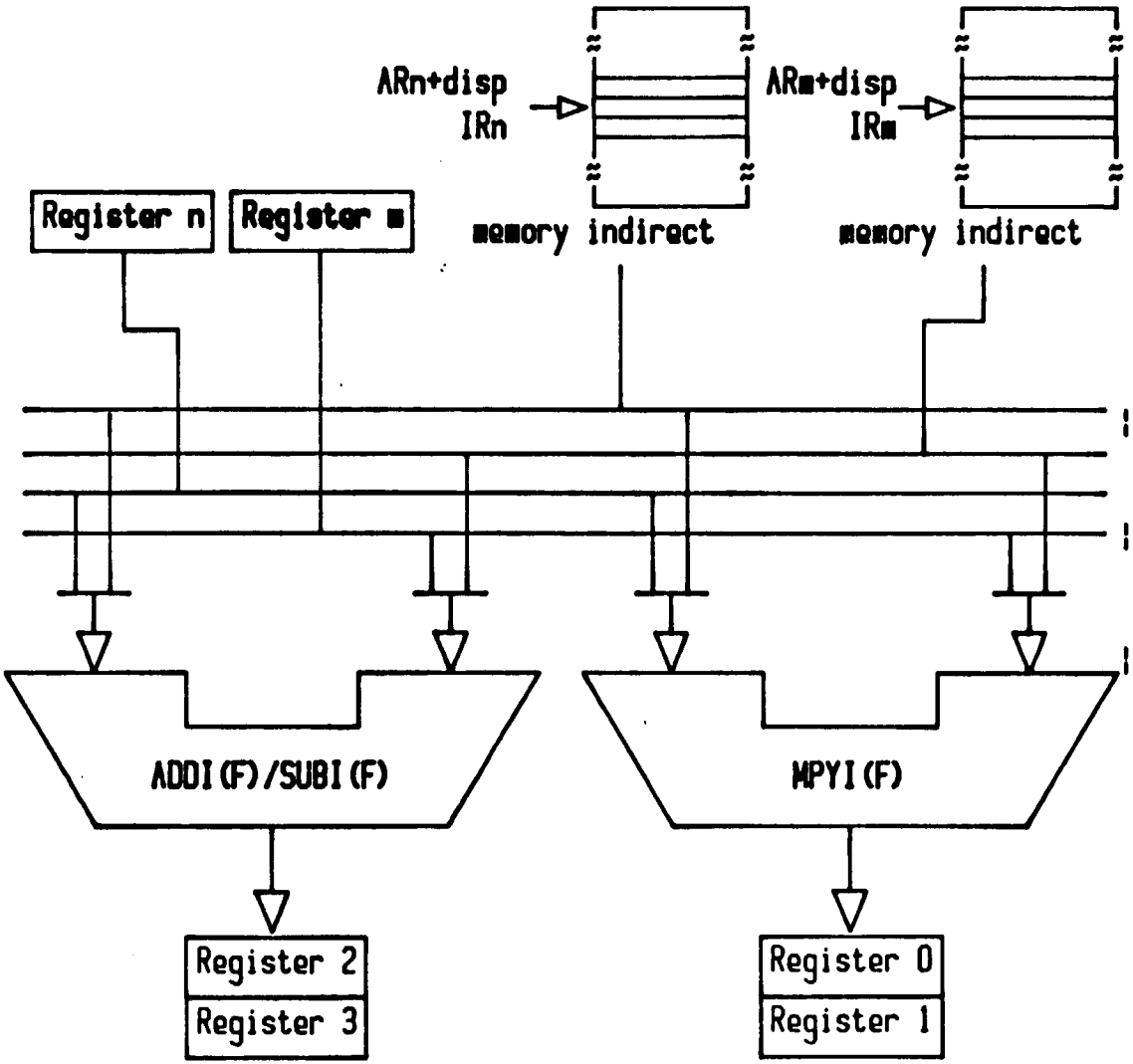

```
SUBF3 R0, R1, R2          ; REGISTER AND
                          ; REGISTER

SUBF  *AR2, R1, R2        ; INDIRECT AND
                          ; REGISTER

ANDN  R0, *AR3, R4        ; REGISTER AND
                          ; INDIRECT

ANDN3 *AR2, *AR3, R1      ; INDIRECT AND
                          ; INDIRECT
```

TMS320C30 ALU/MPY INSTRUCTIONS WITH STORE




```
   MPYF R0, *AR0++, R2        ; MPYF IN
|| STF  R3, *AR1--            ; PARALLEL
                              ; WITH STF

   AND  R2, *AR2++(IR0), R3   ; AND IN
|| STI  R3, *AR1--(IR1)       ; PARALLEL
                              ; WITH STI
```

TMS320C30 PARALLEL ADD AND MPY INSTRUCTIONS




```
MPYF *AR0,*AR1++,R0    ;MPYF
|| ADDF R0,R1,R2       ;IN PARALLEL WITH ADD

SUBF R0,R2,R2          ;SUBI
|| MYPF *AR1++,*AR2--,R1;IN PARALLEL WITH MPY

MPYI *AR1,R0,R1        ;MPYI
|| ADDI *AR2,R2,R3     ;IN PARALLEL WITH ADD
```

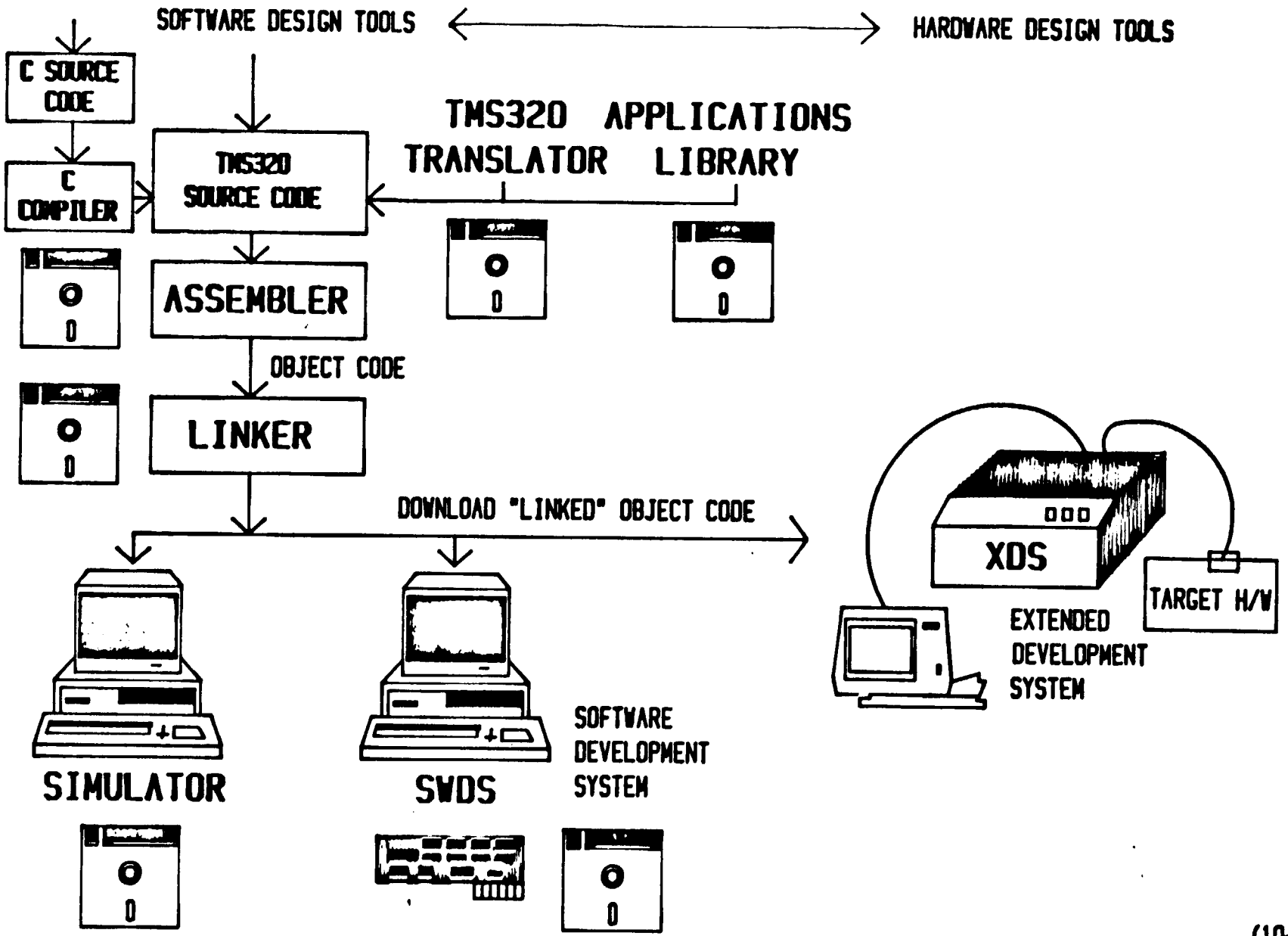
C-2.

TMS320C30 FAMILY BENCHMARKS

DSP APPLICATIONS	TIMING
FIR FILTER TAP	60 NS
LMS ADAPTIVE FIR FILTER TAP	180 NS
BI-QUAD FILTER ELEMENT (FIVE MULTIPLES)	360 NS

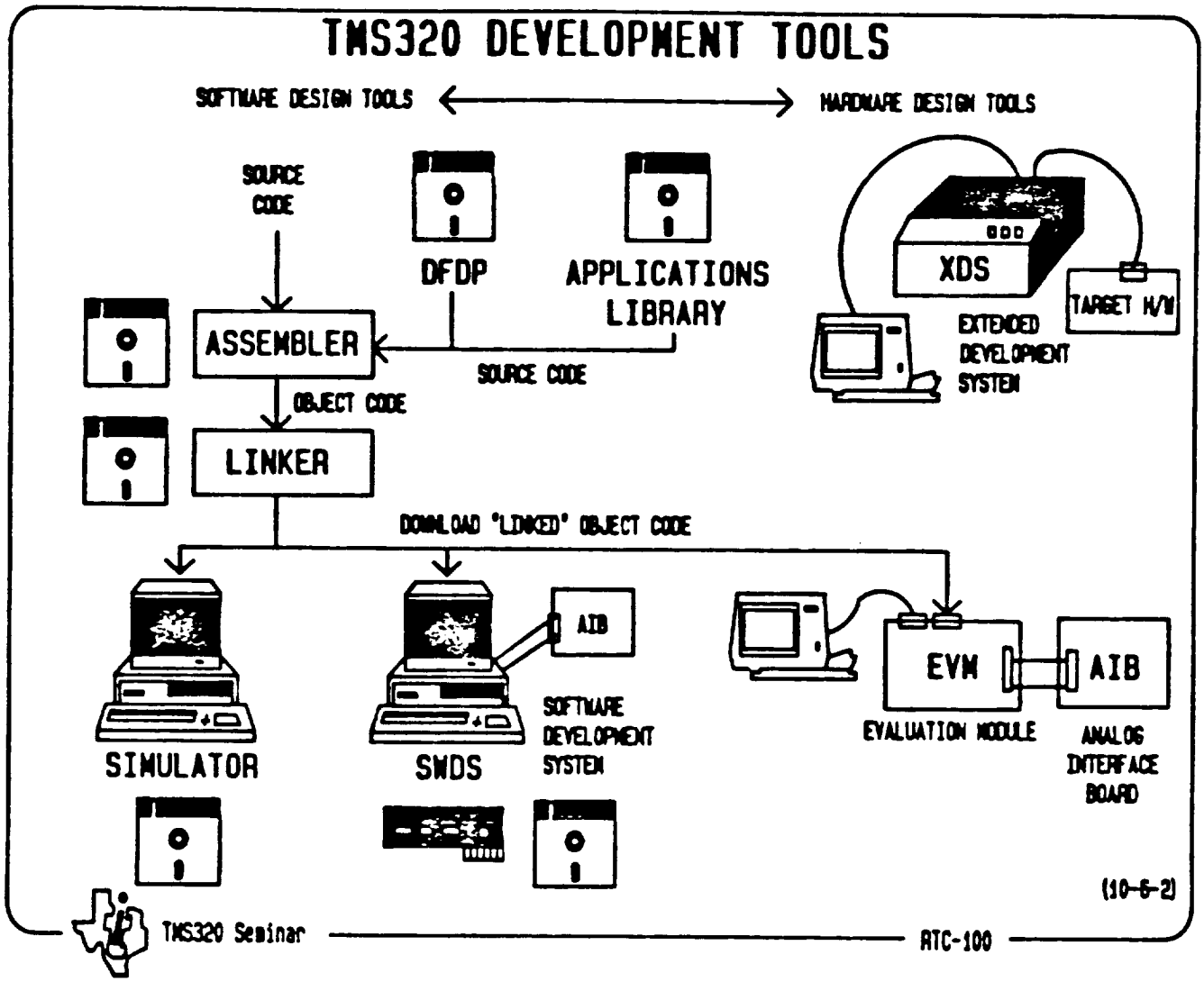
FFT TIMING		RADIX-2 (REAL)	RADIX-4 (CMPLX)
NUMBER OF POINTS	64	0.075 ms	0.123 ms
	256	0.354 ms	0.624 ms
	1024	1.670 ms	3.040 ms

TMS320C30 DEVELOPMENT TOOLS



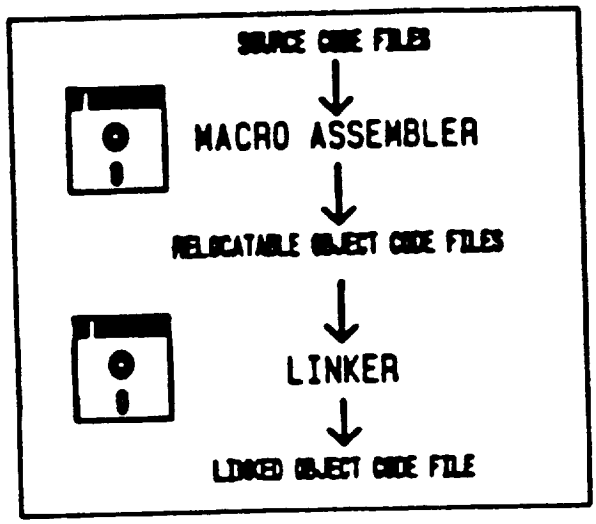
AGENDA

* INTRODUCTION
* SECOND GENERATION 320 (320C25)
* FIRST GENERATION 320 (320C10)
* BREAK
* THIRD GENERATION 320 (320C30)
* TOOLS
* APPLICATIONS


TMS320 DEVELOPMENT TOOLS
SOFTWARE DESIGN TOOLS
HARDWARE DESIGN TOOLS
SOURCE CODE
DFDP
APPLICATIONS LIBRARY
XDS
TARGET H/W
EXTENDED DEVELOPMENT SYSTEM
ASSEMBLER
SOURCE CODE
OBJECT CODE
LINKER
DOWNLOAD 'LINKED' OBJECT CODE
AIB
EVM
AIB
SIMULATOR
SWDS
SOFTWARE DEVELOPMENT SYSTEM
EVALUATION MODULE
ANALOG INTERFACE BOARD
(10-6-2)
TMS320 Seminar
RTC-100

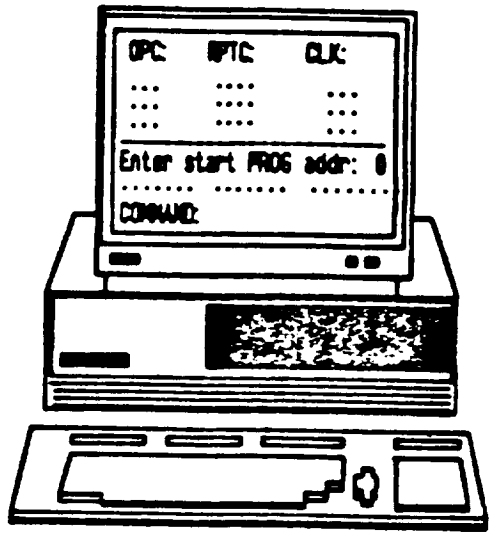
TMS320 MACRO ASSEMBLER/LINKER

- MACRO CAPABILITIES
- LIBRARY FUNCTIONS
- CONDITIONAL ASSEMBLY
- RELOCATABLE MODULES
- AVAILABLE ON VAX/VMS AND PC/MS-DOS SYSTEMS



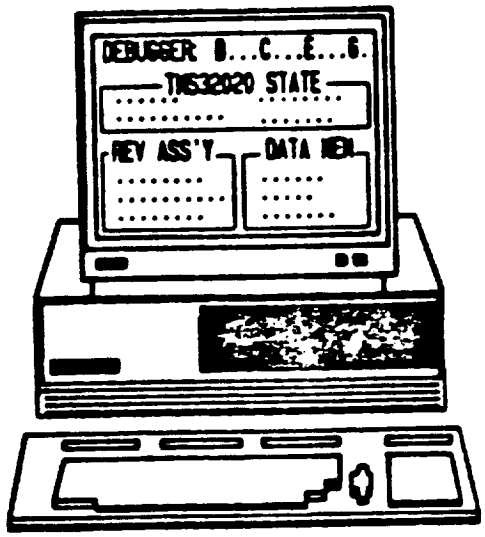
TMS320 SOFTWARE SIMULATOR

- ABILITY TO SIMULATE EITHER MICROPROCESSOR OR MICROCOMPUTER MODE
- BREAKPOINT/TRACE CAPABILITIES
- MODIFY AND DISPLAY BOTH PROGRAM MEMORY OR DATA MEMORY
- MODIFY AND INSPECT REGISTERS
- SINGLE-STEP EXECUTION
- ASSOCIATE FILES TO I/O
- AVAILABLE ON VAX/VMS AND PC/MS-DOS SYSTEMS



SWDS SOFTWARE DEVELOPMENT SYSTEM

- IBM-PC/AT ADD-IN BOARD
- REALTIME DEBUG FOR TMS320C25 AND TMS32020
- 24K WORDS PROG/DATA MEMORY
- MENU-DRIVEN COMMAND ENTRY
- DIRECT ACCESS TO OTHER PC UTILITIES INCLUDING ASSEMBLER/ LINKER/TEXT EDITORS
- EXTERNAL CABLE AND ADAPTERS:
  - PGA CONNECTOR CONNECTS SWDS TO 68-PIN GRID ARRAY FOOTPRINT (TMS320C0/C25 TARGET SYSTEM)
  - ANALOG INTERFACE BOARD ADAPTER CONNECTOR CONNECTS SWDS TO TMS32010 AIB



(10-6-5)

DFDP DIGITAL FILTER DESIGN PACKAGE

- INTERACTIVE MENU-DRIVE PACKAGE FROM ATLANTA SIGNAL PROCESSORS INC (ASPI)
- DESIGN FIR FILTERS
  - KAISER WINDOW
  - PARKS-MCCLELLAN
- DESIGN IIR FILTERS
  - BUTTERWORTH
  - CHEBYCHEV I AND II
  - ELLIPTIC
- DISPLAYS FILTER RESPONSES
- GENERATES TMS320C10/C25 ASSEMBLY CODE FULLY COMMENTED
- RUNS on IBM PC/XT/AT

USER ENTERS FILTER SPEC

↓

DFDP ESTIMATES REQ'D FILTER ORDER OR IMPULSE RESPONSE LENGTH

↓

USER SELECTS FILTER LENGTH TO BE USED

↓

DFDP CALCULATES COEFFICIENTS AND RESPONSE CHARACTERISTICS

↓

USER CAN EXAMINE/PLOT RESPONSES

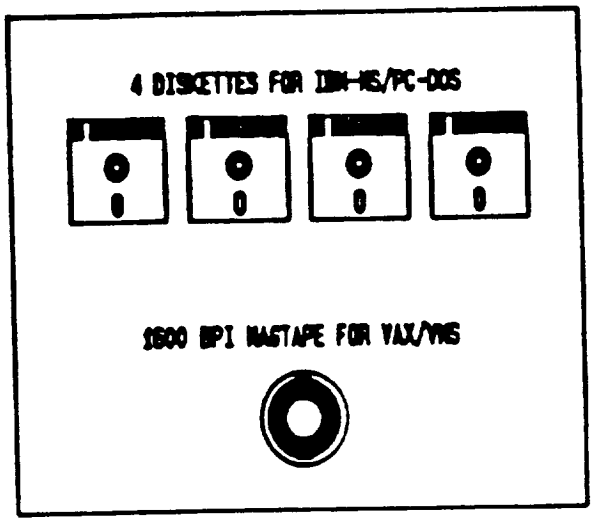
↓

DFDP GENERATES TMS320C10/C25 CODE

ORIGINAL PAGE IS
OF POOR QUALITY

TMS320 APPLICATIONS LIBRARY

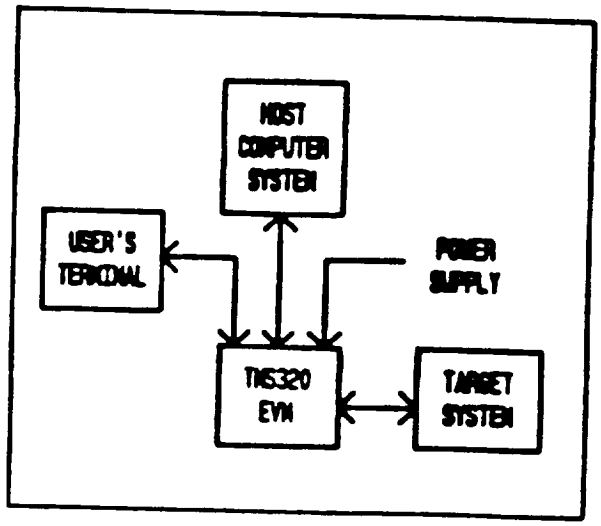
- SOFTWARE LIBRARY CONTAINING MAJOR DSP ROUTINES
  - FFT
  - FIR/IIR FILTERING
  - FLOATING-POINT OPERATIONS
- APPLICATION ALGORITHMS
  - ECHO CANCELLATION
  - ADPCM
  - DTMF ENCODING/DECODING
- MACROS FOR THE TMS320C10
- SEE TMS320 DSP BULLETIN BOARD FOR UPDATES



(10-6-7)

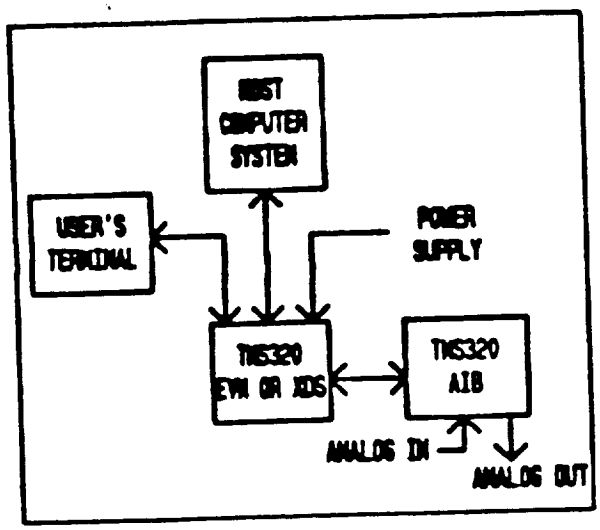
TMS320 EVALUATION MODULE

- INEXPENSIVE EVALUATION TOOL
- RESIDENT EDITOR/DEBUG MONITOR
- ASSEMBLER/REVERSE ASSEMBLER
- S/W SINGLE-STEP/BREAKPOINT
- TARGET CONNECTOR FOR IN-CIRCUIT EMULATION
- DUAL SERIAL RS232 PORTS
- HOST CPU UPLOAD/DOWNLOAD
- BUILT-IN EPROM PROGRAMMER



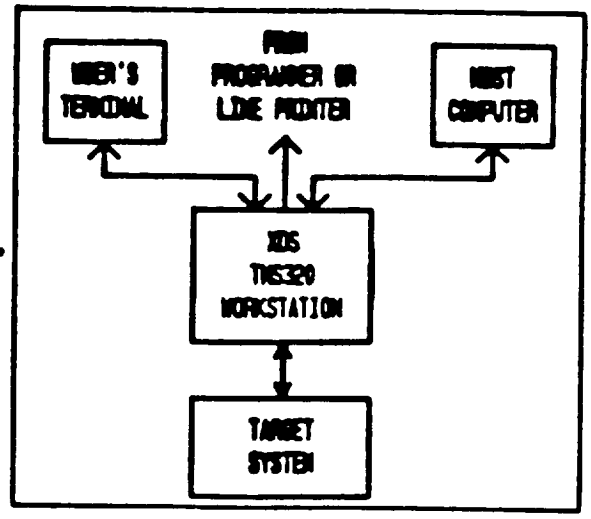
TMS320 ANALOG INTERFACE BOARD

- 12-BIT A/D, D/A CONVERTERS
- TWO USER-DEFINABLE 16-BIT INPUT LATCHES FOR A/D'S WITH A/D STATUS REGISTER
- ONE USER-DEFINABLE 16-BIT OUTPUT LATCH FOR D/A'S
- PROGRAMMABLE SAMPLE-RATE CLOCKS
- TWO ACTIVE LOWPASS FILTERS
- IC AUDIO AMPLIFIER WITH VOLUME CONTROL
- WIRE-WRAP AREA
- ADAPTER BOARD TO CONVERT TO TMS32020/C25 DEVELOPMENT TOOLS



XDS TMS320 EMULATOR

- FULL-SPEED IN-CIRCUIT EMULATION
- HOST-INDEPENDENT
- SERIAL I/O PORTS ALLOW INTERCONNECT TO HOST COMPUTER, TERMINAL, PRINTER, OR PROM PROGRAMMER
- FEATURES
  - UPLOAD/DOWNLOAD CAPABILITY
  - ASSEMBLER/REVERSE ASSEMBLER
  - DEBUG MONITOR
  - SINGLE-STEP OPTION
  - SOFTWARE BREAKPOINTS
- FULL-SPEED HARDWARE BREAKPOINT AND TRACE CAPABILITIES
- MULTIPROCESSING CAPABILITY



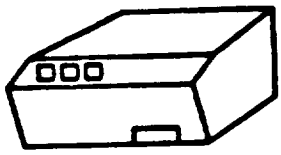
(10-6-10)

ORIGINAL PAGE IS OF POOR QUALITY

XDS UPGRADES

- UPGRADE KITS EXTEND FUNCTIONALITY OF EXISTING DEVELOPMENT SYSTEMS AT A MINIMUM COST
- INSTALLATION OPTIONS
  - FACTORY UPGRADE
  - CUSTOMER UPGRADE (LOWER COST, FASTER TURNAROUND)

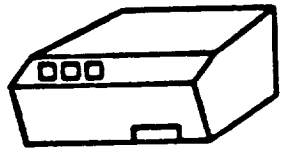
TMS32010 XDS/22



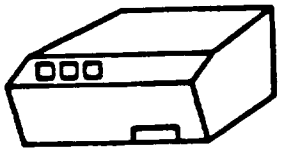
UPGRADE

+ EMULATOR BOARD, 40 MHZ BKPT/TRACE/TIMING, DIP TARGET CONNECTOR

TMS320C10 1ST GENERATION XDS-22



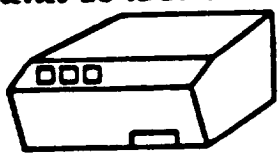
TMS32020 XDS/22



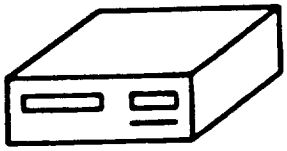
UPGRADE

+ 2 PALS, 2 EPROMS, 2 RAMS, CRYSTAL, TMS320C25, 40 MHZ BKPT/TRACE/TIMING

TMS320C25 2ND GENERATION XDS-22



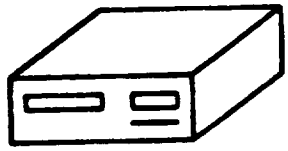
TMS32020 XDS-11



UPGRADE

SAME AS ABOVE WITHOUT 40 MHZ BKPT/TRACE BD

TMS32020/C25 XDS-11



(10-6-11)

TMS320 DEVELOPMENT TOOLS - FEATURE MATRIX

	EVM	SIMULATOR	SWDS	XDS
DEVELOPMENT PURPOSE:				
EVALUATION/BENCHMARKING	YES	YES	YES	YES
SOFTWARE DEVELOPMENT	NO	YES	YES	YES
HARDWARE DESIGN	NO	NO	NO	YES
LINE-BY-LINE ASSEMBLER/REVERSE ASSEMBLER	YES	NO	YES	YES
MODIFY/DISPLAY MEMORY AND REGISTERS	YES	YES	YES	YES
SINGLE-STEPPING	YES	YES	YES	YES
BREAKPOINT ON ADDRESS	YES	YES	YES	YES
BREAKPOINT ON MEMORY ACCESS/READ/WRITE	NO	YES	NO	YES
TIME-STAMPING/CLOCK COUNTER	NO	YES	YES	YES
REAL-TIME TRACE SAMPLES	NO	NO	NO	YES
FULL-SPEED IN-CIRCUIT EMULATION				
FROM ON-BOARD MEMORY	YES	N/A	YES	YES
FROM TARGET MEMORY	NO	N/A	NO	YES
MULTI-USER SYSTEM	NO	YES	NO	NO
BUILT-IN SYSTEM INTERFACE	NO	NO	YES	NO
FILES ASSOCIATED TO I/O PORTS	NO	YES	YES	NO
AMOUNT OF MEMORY				
ON-BOARD PROGRAM/DATA (TMS32010/C10)	4K/-	N/A	N/A	4K/-
ON-BOARD PROGRAM/DATA (TMS32020/C25)	4K/4K	N/A	24K	4K/4K
PROGRAM/DATA EXPANSION (TMS320C25)	16K/16K	N/A	N/A	64K

(10-6-12)

ORIGINAL PAGE IS
OF POOR QUALITY

SUMMARY - TMS320 DEVELOPMENT TOOLS

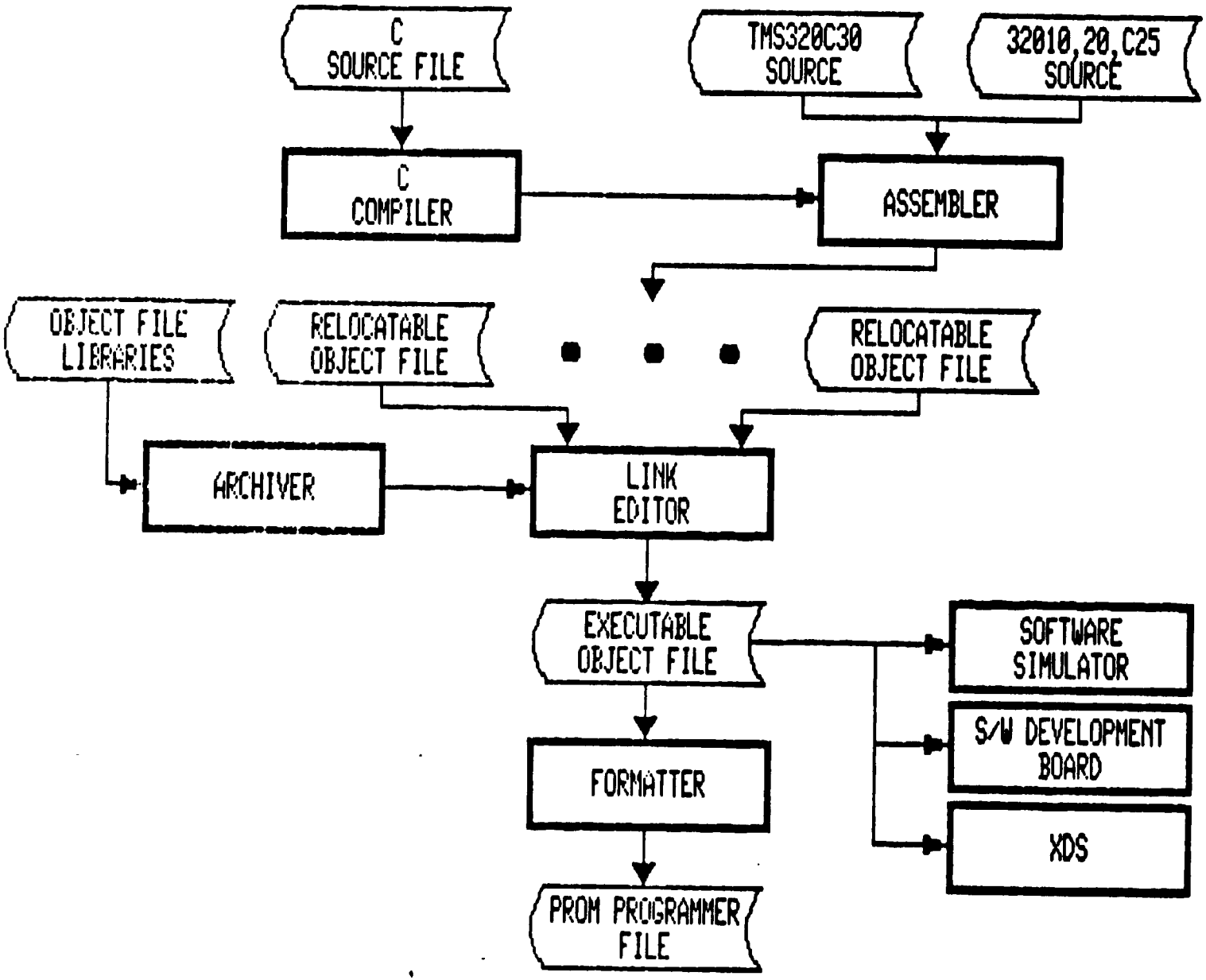
SOFTWARE	1ST GEN	2ND GEN	TMS320 FAMILY
MACRO ASSEMBLER/LINKER			
VAX VMS	X	X	
IBM-PC MS-DOS	X	X	
SIMULATOR			
VAX VMS	X	X	
IBM-PC MS-DOS	X	X	
SWDS - S/W DEVELOPMENT SYSTEM		X	
DFDP - DIGITAL FILTER DES PKG			X
DSP SOFTWARE LIBRARY			
VAX VMS			X
IBM-PC MS-DOS			X
BELL 212A MODEM S/W	X		

HARDWARE	1ST GEN	2ND GEN	TMS320 FAMILY
EVALUATION MODULE - EVM	X	X	
XDS/22 EMULATOR	X	X	
XDS/22 UPGRADE			
FACTORY	X	X	
CUSTOMER	X	X	
XDS/11 UPGRADE			
FACTORY		X	
CUSTOMER		X	
ANALOG INTERFACE BOARD - AIB			X
AIB ADAPTOR		X	X
TMS320 DESIGN KIT			X

x - SEE TABLE A-2 IN TMS320 FAMILY DEVELOPMENT SUPPORT REFERENCE GUIDE FOR PART NUMBERS

(10-6-13)

TMS320C30 DEVELOPMENT TOOLS



TMS320 SUPPORT TOOLS

- DOCUMENTATION
- NEWSLETTER/BULLETIN BOARD/HOTLINE
- DSP TEXTBOOKS
- THIRD-PARTY SUPPORT
- HANDS-ON TRAINING

(10-6-14)

TMS320 DOCUMENTATION

- PRODUCT DESCRIPTIONS/BULLETINS
  - TMS320 DSP PRODUCT BULLETIN
  - TMS320C25 PRODUCT DESCRIPTION
- USER'S GUIDES
  - TMS32010 USER'S GUIDE
  - TMS32011 USER'S GUIDE
  - TMS320 FAMILY DEVELOPMENT SUPPORT REFERENCE GUIDE
  - TMS32020 USER'S GUIDE
  - TMS320C25 USER'S GUIDE
- DATA SHEETS
  - TMS32010 DATA SHEET
  - TMS32010-14 DATA SHEET
  - TMS32011 DATA SHEET
  - TMS320C10 DATA SHEET
  - TMS32020 DATA SHEET
  - TMS320C25 DATA SHEET
- DSP APPLICATIONS BOOK

(10-6-15)

TMS320 DSP TEXTBOOKS

TITLE	AUTHOR	PUBLISHER
Theory and Implementation of DFT/FFT and Convolution Algorithms	C.S. Burrus, T.W. Parks	John Wiley and Sons
Digital Filter Design	T.W. Parks, C.S. Burrus	John Wiley and Sons
Practical Approaches to Speech Coding	P. Papamichalis	Prentice-Hall
A Practical Guide to Adaptive Filter Design	J.R. Treichler, C.R. Johnson, Jr., M.G. Larimore	John Wiley and Sons
A DSP Laboratory Using the TMS32010	T.W. Parks, D.L. Jones	Prentice-Hall

(10-6-17)

TMS320 NEWSLETTER/BULLETIN BOARD/HOTLINE

- TMS320 NEWSLETTER
  - CURRENT AND NEW TMS320 EVENTS/PRODUCTS/UPDATES
  - PUBLISHED QUARTERLY
  - CALL 800-232-3200 TO BE ADDED TO MAILING LIST
- BULLETIN BOARD
  - EXCELLENT COMMUNICATIONS TOOL
  - CONTAINS SOURCE CODE FROM THE DSP APPS BOOK
  - PHONE 713-879-2091 (BELL 212A MODEM, 300 OR 1200 BPS)
  - USER ID CAN BE OBTAINED FOR FULL ACCESS
  - 60-MINUTE PER DAY ACCESS LIMIT
- HOTLINE
  - HOURS ARE 8:00 AM - 4:30 PM (CENTRAL TIME)
  - PHONE 713-TRY-A-320
    713-879-2320

(10-6-16)

TMS320 THIRD-PARTY SUPPORT

- FOR MORE DETAILS, SEE THE TMS320 FAMILY DEVELOPMENT SUPPORT REFERENCE GUIDE

SOFTWARE TOOLS

- ASSEMBLERS
- SIMULATORS
- FILTER DESIGN S/W
- ALGORITHM DEVELOPMENT
- SUBBAND CODER
- COMPILERS
- DATA COMMUNICATIONS S/W
- DSPL COMPILER/LIBRARY
- SPEECH ALGORITHMS
- DSP S/W PROTOTYPING SYSTEM

SUPPORTED ON

- IBM-PC/XT/AT
- CP/M-80
- CP/M-86
- HP/M
- VAX UNIX
- VAX VMS
- PDP-11
- RT-11
- RSX-11
- ILS, SPICE, ISMP
- ECLIPSE S130 AOS
- TURBODOS
- APPLE II/E/C
- AMIGA
- OLIVETTI M20/21/24 PC
- TEKTRONIX 8560

HARDWARE TOOLS

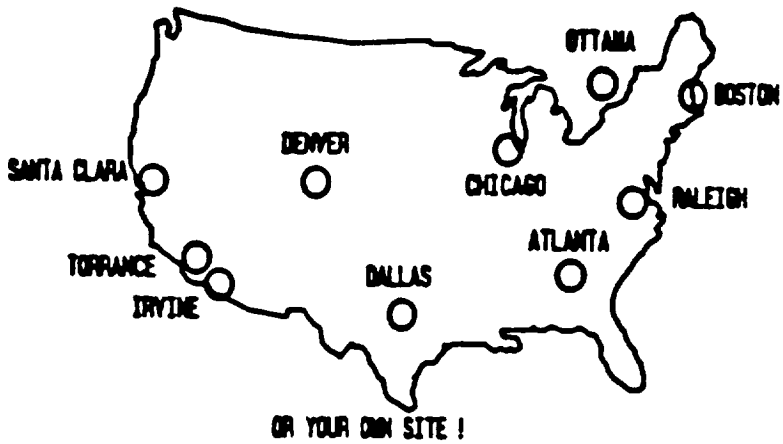
- ANALOG I/O BOARDS
- DSP BOARDS
- DSP LAB
- EMULATORS
- ANALYZERS
- EPROM PROGRAMMERS
- ENGINEERING WORKSTATIONS
- LOGIC ANALYZER, DISASSEMBLER
- TEST PROGRAMS
- DSP BOARDS (WITH C-COMPILER)
- SIMULATOR SYSTEMS

SUPPORTED ON

- VMEBUS
- Q-BUS
- IBM-PC/XT/AT
- HP54000 LDS
- VALID
- SENTRY 7/8/20/21

(10-6-18)

TMS320 HANDS-ON TRAINING



- HALF-DAY SEMINARS
- 3-DAY 1ST GENERATION DSP WORKSHOP ($995)
  - EVM/AIB DEVELOPMENT TOOLS
  - TMS320C15/C17 ARCHITECTURE/INSTRUCTION SET/APPLICATIONS
- 3-DAY 2ND GENERATION DSP WORKSHOP ($995)
  - SMDS AND/OR XDS DEVELOPMENT TOOLS
  - TMS320C25 ARCHITECTURE/INSTRUCTION SET/APPLICATIONS

15% DISCOUNT FOR WHEN 3 OR MORE FROM SAME COMPANY ENROLL IN A WORKSHOP

(10-6-198

SUMMARY - TMS320 DEVELOPMENT SUPPORT

- DEVELOPMENT TOOLS
  - XDS (EMULATOR)
  - EVM (EVALUATION MODULE)
  - AIB (ANALOG INTERFACE BOARD)
  - DESIGN KIT
  - SDS (SOFTWARE DEVELOPMENT SYSTEM)
  - ASSEMBLER/LINKER
  - SIMULATOR
  - DFDP (DIGITAL FILTER DESIGN PACKAGE)
- SUPPORT TOOLS
  - DOCUMENTATION
  - NEWSLETTER
  - BULLETIN BOARD
  - HOTLINE
  - TEXTBOOKS
- THIRD-PARTY SUPPORT
- TRAINING
  - 3-DAY 1ST GENERATION DSP WORKSHOP
  - 3-DAY 2ND GENERATION DSP WORKSHOP

(10-6-29)

AGENDA

* INTRODUCTION
* SECOND GENERATION 320 (320C25)
* FIRST GENERATION 320 (320C10)
* BREAK
* THIRD GENERATION 320 (320C30)
* TOOLS
* APPLICATIONS

TMS320 APPLICATIONS - I

GENERAL-PURPOSE DSP

- DIGITAL FILTERING
- CONVOLUTION
- CORRELATION
- HILBERT TRANSFORMS
- FAST FOURIER TRANSFORMS
- ADAPTIVE FILTERING
- WINDOWING
- WAVEFORM GENERATION

VOICE/SPEECH

- VOICE MAIL
- SPEECH VOCODING
- SPEECH RECOGNITION
- SPEAKER VERIFICATION
- SPEECH ENHANCEMENT
- SPEECH SYNTHESIS
- TEXT-TO-SPEECH

GRAPHICS/IMAGE

- 3-D ROTATION
- ROBOTIC VISION
- IMAGE TRANSFORMATION/COMPRESSION
- PATTERN RECOGNITION
- IMAGE ENHANCEMENT
- HOMOMORPHIC PROCESSING
- WORKSTATIONS
- ANIMATION/DIGITAL MAP

CONTROL

- DISK CONTROL
- SERVO CONTROL
- ROBOT CONTROL
- LASER PRINTER CONTROL
- ENGINE CONTROL
- MOTOR CONTROL

INSTRUMENTATION

- SPECTRUM ANALYSIS
- FUNCTION GENERATION
- PATTERN MATCHING
- SEISMIC PROCESSING
- TRANSIENT ANALYSIS
- DIGITAL FILTERING
- PHASE-LOCKED LOOP

MILITARY

- SECURE COMMUNICATIONS
- RADAR PROCESSING
- SONAR PROCESSING
- IMAGE PROCESSING
- NAVIGATION
- MISSILE GUIDANCE
- RF MODEMS

(10-5-2)

TMS320 APPLICATIONS - II

TELECOMMUNICATIONS

- ECHO CANCELLATION
- ADPCM TRANSCODERS
- DIGITAL PBXs
- LINE REPEATERS
- CHANNEL MULTIPLEXING
- 1200 TO 19200-bps MODEMS
- ADAPTIVE EQUALIZERS
- DTMF ENCODING/DECODING
- DATA ENCRYPTION
- FAX
- CELLULAR TELEPHONES
- SPEAKER PHONES
- DIGITAL SPEECH INTERPOLATION (DSI)
- x.25 PACKET SWITCHING
- VIDEO CONFERENCING
- SPREAD SPECTRUM COMMUNICATIONS

AUTOMOTIVE

- ENGINE CONTROL
- VIBRATION ANALYSIS
- ANTISKID BRAKES
- ADAPTIVE RIDE CONTROL
- GLOBAL POSITIONING
- NAVIGATION
- VOICE COMMANDS
- AUDIO PROCESSING

CONSUMER

- RADAR DETECTORS
- POWER TOOLS
- DIGITAL AUDIO/TV
- MUSIC SYNTHESIZERS
- EDUCATIONAL TOYS

INDUSTRIAL

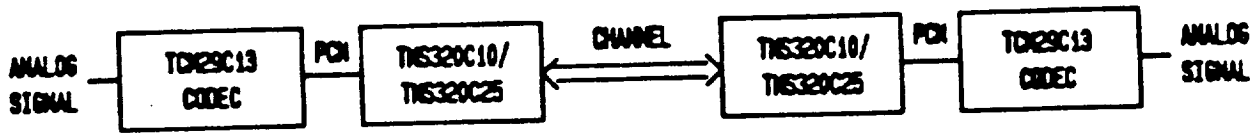
- ROBOTICS
- NUMERIC CONTROL
- SECURITY ACCESS
- POWER-LINE MONITORS
- MOTOR CONTROL
- PROGRAMMABLE CONTROLLERS

MEDICAL

- HEARING AIDS
- PATIENT MONITORING
- ULTRASOUND EQUIPMENT
- FETAL MONITORS
- DIAGNOSTIC TOOLS
- PROSTHETICS

(10-5-3)

TRANSCODER BLOCK DIAGRAM

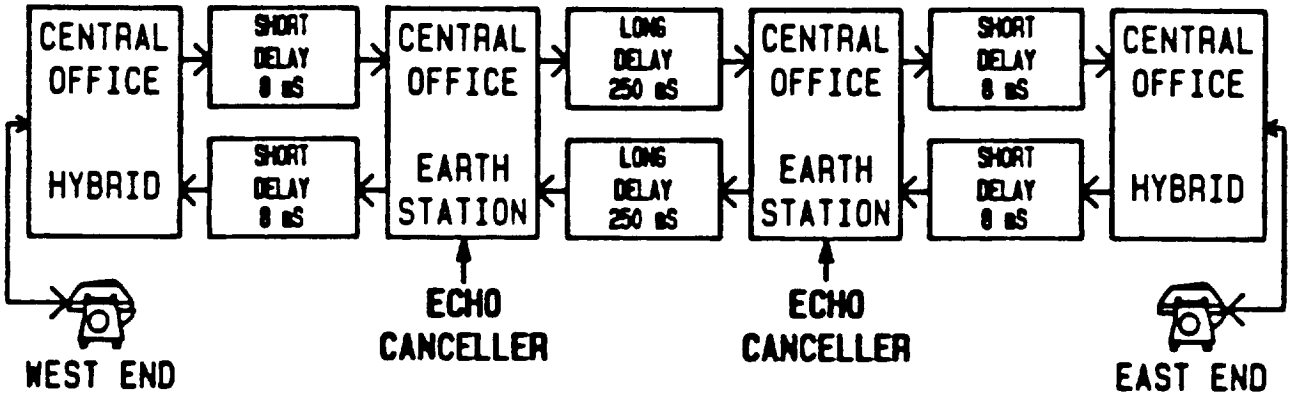


TMS320 TRANSCODER ALGORITHM	PERFORMANCE STANDARD	TMS320C10	TMS320C25	SOURCE
● 32KBPS ADPCM (ANSI CCITT)	% CPU LOADING (HALF-DUPLEX)	100%	50%	TI ROM Code (ANSI) DSP Apps Book (CCITT)
● 32KBPS ADPCM (NON-CCITT)	% CPU LOADING (FULL DUPLEX)	80%	45%	DSP Apps Book
● 16KBPS SUBBAND CODER	% CPU LOADING (FULL DUPLEX)	80%*	35%	ASPI License

* REQUIRES 1K WORDS EXTERNAL DATA MEMORY

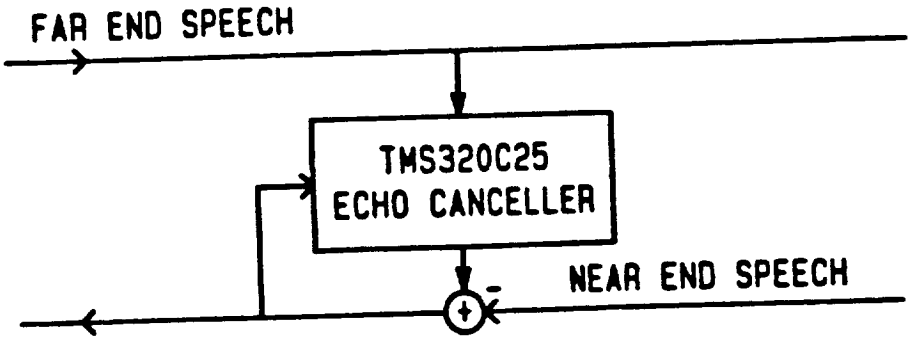
(10-5-15)

ECHO CANCELLER SYSTEM BLOCK DIAGRAM



(10-5-16)

ECHO CANCELLER BLOCK DIAGRAM

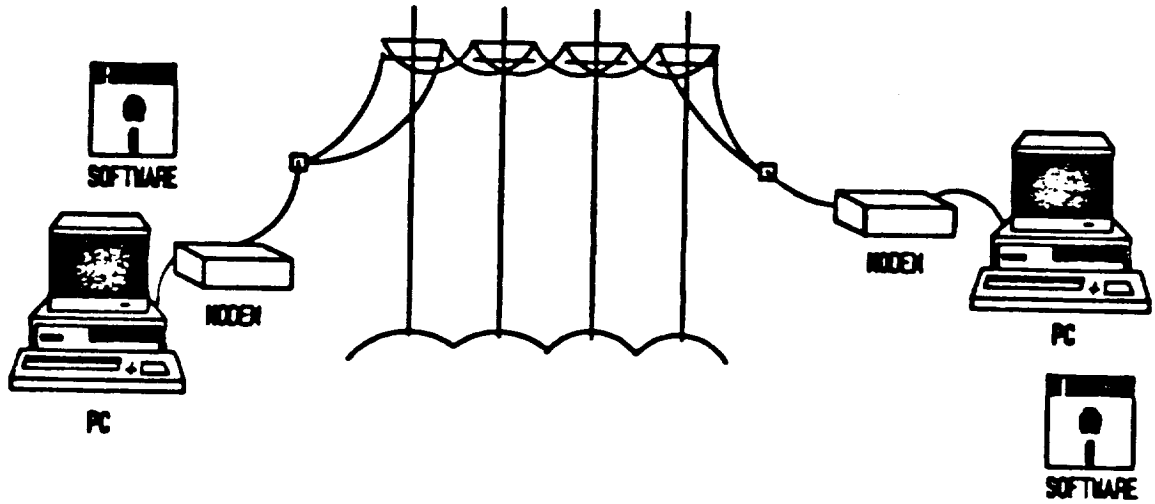


TMS320 APPLICATION	PERFORMANCE STANDARD	TMS320C25*	SOURCE
ECHO CANCELLATION (CCITT G.165)	%CPU LOADING (16 MS ECHO LENGTH)	50%	DSP Apps Book

* WITH ON-CHIP ROM AND RAM

(10-5-17)

DATA COMMUNICATIONS



(10-5-20)

DSP2400 MODEM OVERVIEW

WHAT IS IT?

A 2400 bps, HAYES COMPATIBLE, 2-WIRE FULL-DUPLEX MODEM AVAILABLE FOR OEMS TO TAKE TO PRODUCTION.

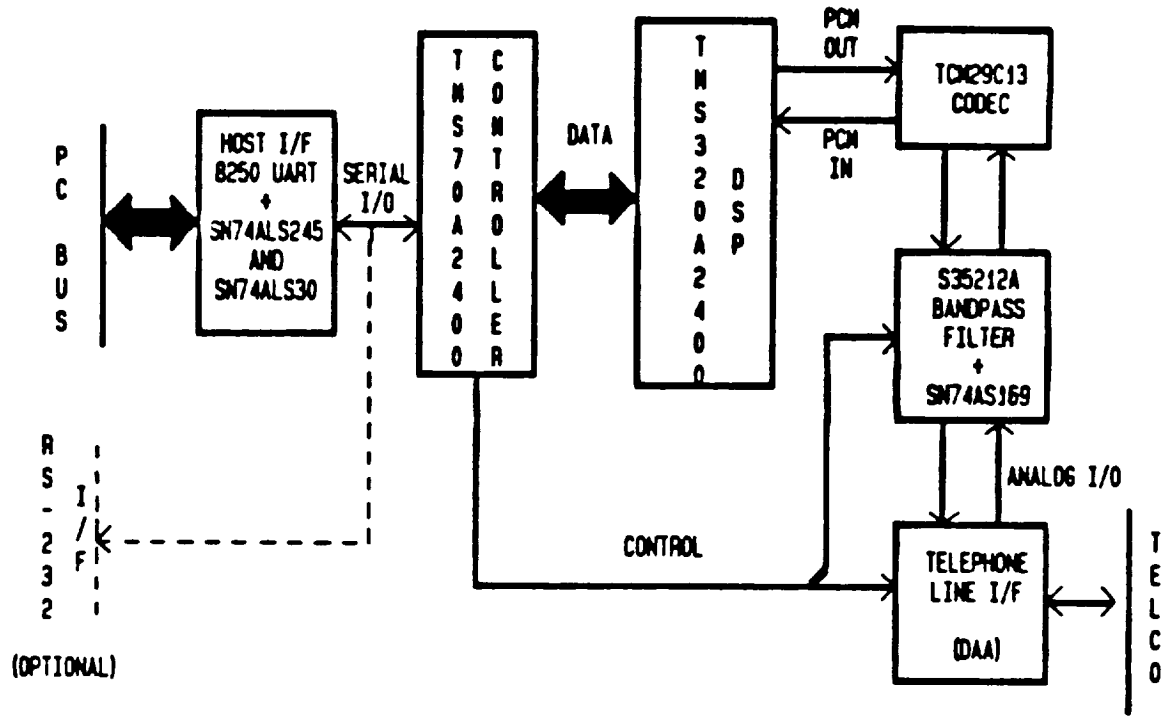
WHY IS IT DIFFERENT?

1. IT BRINGS THE COST OF 2400 bps DOWN TO THE 1200 bps RANGE.

2. IT ELIMINATES SILICON RISK BECAUSE ONLY STANDARD, INDUSTRY PROVEN COMPONENTS ARE USED IN THE DESIGN.

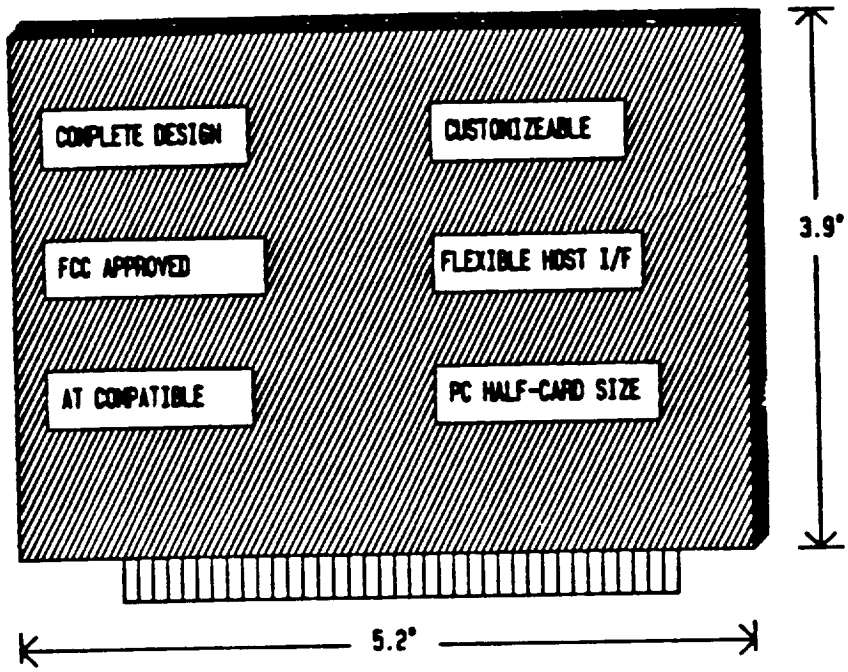
(10-5-22)

DSP2400 MODEM BLOCK DIAGRAM



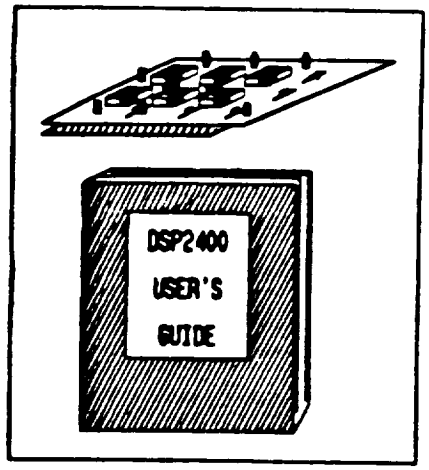
(10-5-23)

DSP2400: A TOTAL SYSTEM SOLUTION



(10-5-24)

DSP2400 PROTOTYPE KIT



CONTENTS:

- FUNCTIONAL DSP2400 ASSEMBLY ON PC HALF-CARD
- DSP2400 USER'S GUIDE
  - INSTALLATION & OPERATION
  - SCHEMATIC
  - ASSEMBLY DRAWING
  - LIST OF MATERIALS
  - PERFORMANCE DATA
- ARTWORK AVAILABLE ON REQUEST

P/N: TMDSP2400PK

(10-5-25)

DSP2400 PRODUCT SUMMARY

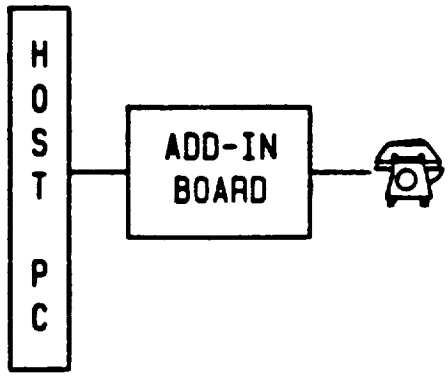
PART NUMBER	DESCRIPTION
1. TMDSP2400PK	DSP2400 PROTOTYPE KIT
2. TMS320A2400NL	DSP2400 MODEM DSP
3. TMS70A2400NL	DSP2400 MODEM CONTROLLER
4. TMDS3240814-12 TMDS3240214-18	TMS320A2400 SOURCE CODE (PC-MSDOS) TMS320A2400 SOURCE CODE (VAX/VMS)
5. TMDS3240815-12 TMDS3240215-18	TMS70A2400 SOURCE CODE (PC-MSDOS) TMS70A2400 SOURCE CODE (VAX/VMS)

ALL PRODUCTS ARE AVAILABLE THROUGH YOUR LOCAL TI SALES OFFICE
OR AUTHORIZED TI DISTRIBUTER

(10-5-26)

PC ADD-IN BOARD

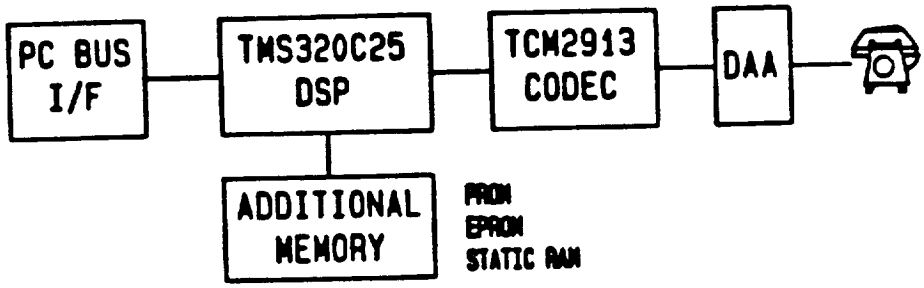
SYSTEM BLOCK DIAGRAM



- TYPICAL APPLICATIONS
  - VOICE MAIL
  - SPEECH I/O
  - GRAPHICS ACCELERATOR
  - MODEM
  - NUMERIC PROCESSOR

(10-5-30)

PC ADD-IN BOARD BLOCK DIAGRAM

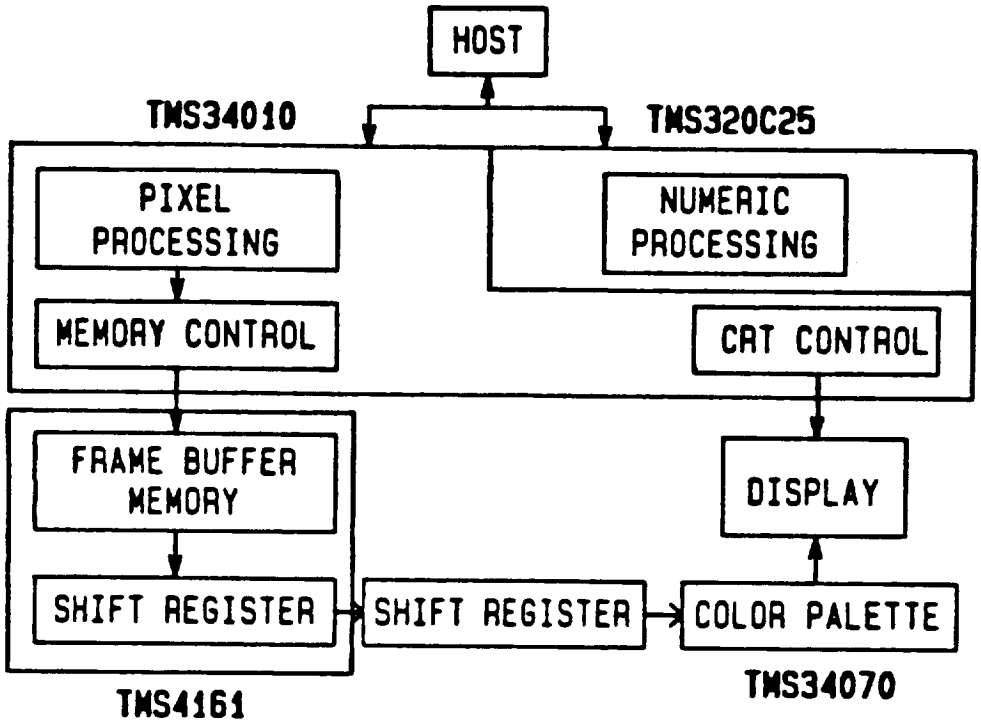


APPLICATION	TMS320 ALGORITHM	PERFORMANCE STANDARD	TMS320C25	SOURCE
VOICE MAIL	2400 bps LPC-10 (DOD)	%CPU LOADING (HALF DUPLEX)	40% +	TI LICENSE
	16K bps SUBBAND CODER	%CPU LOADING (HALF DUPLEX)	20%	ASPI LICENSE
MODEM	1200 bps SPLIT-BAND (CCITT V.22/212A)	%CPU LOADING (FULL-DUPLEX)	30%	$250 FROM TI
	2400 bps SPLIT-BAND (V.22 bis)	%CPU LOADING (FULL DUPLEX)	30%	TI LICENSE

+ requires external data memory

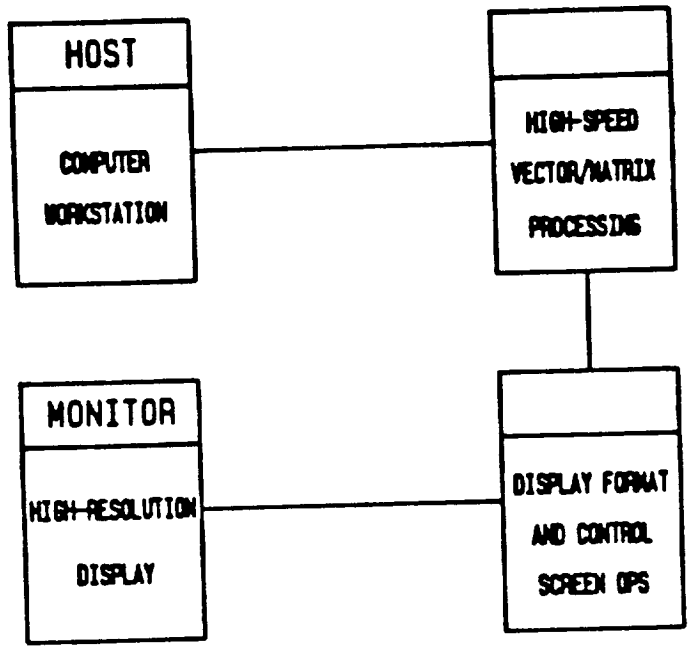
(10-5-31)

GENERIC GRAPHICS SYSTEM



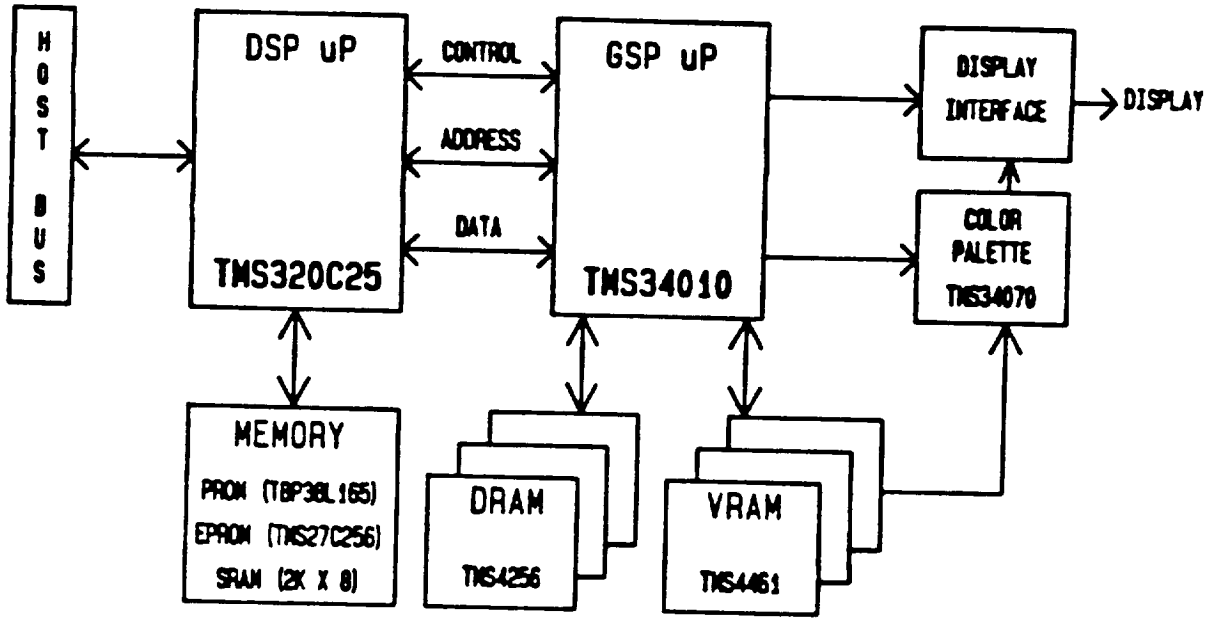
(10-5-33)

GRAPHICS WORKSTATION SYSTEM BLOCK DIAGRAM



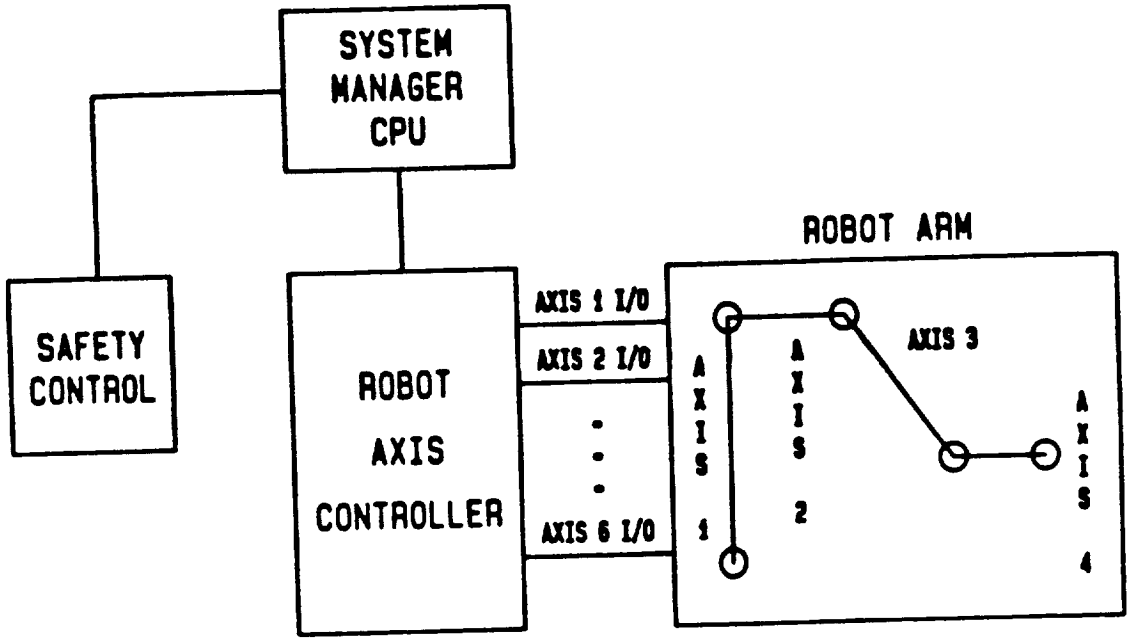
(10-5-34)

GRAPHICS SYSTEM BLOCK DIAGRAM



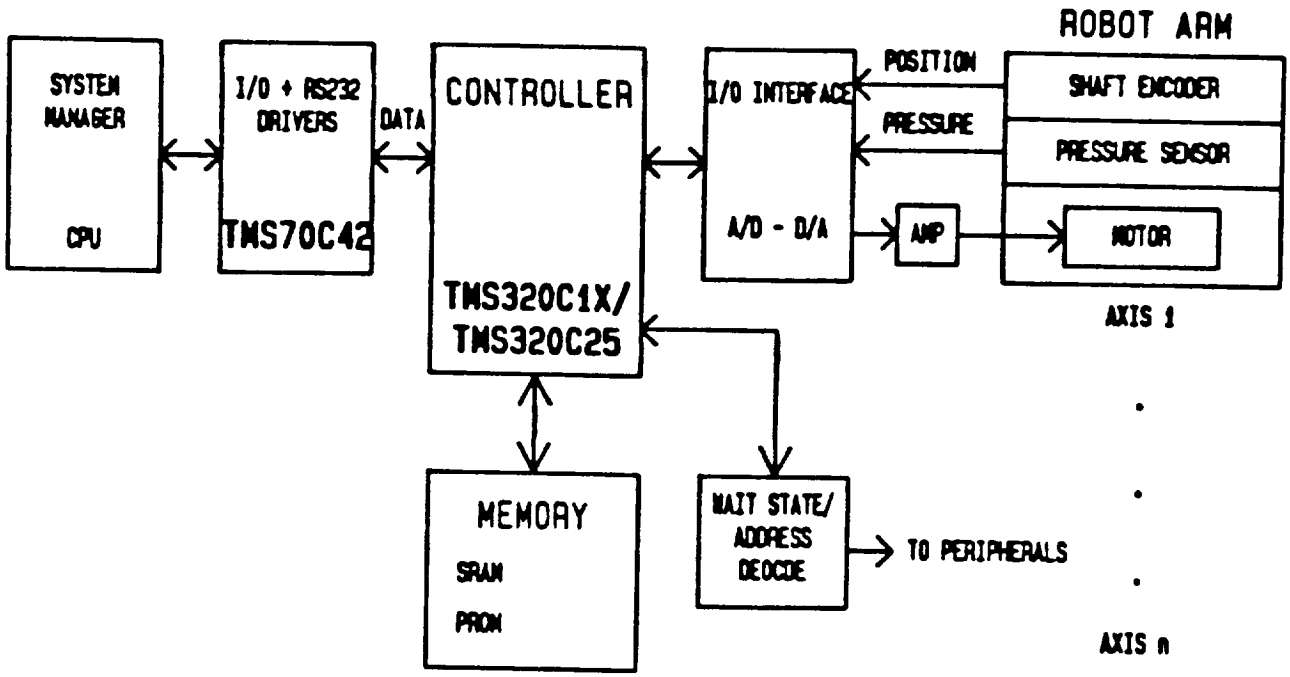
(10-5-35)

ROBOTICS SYSTEM BLOCK DIAGRAM



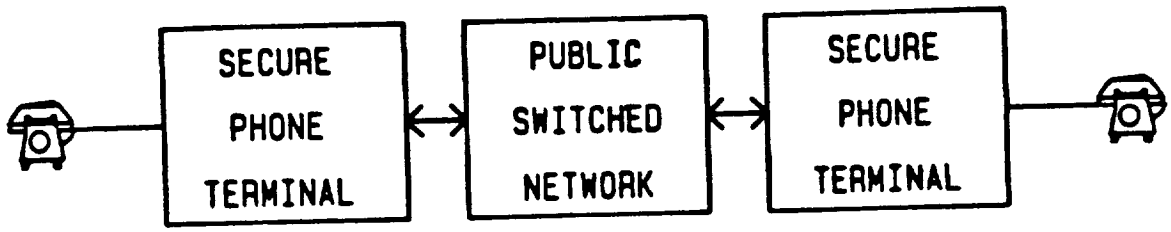
(10-5-40)

ROBOT AXIS CONTROLLER SUBSYSTEM



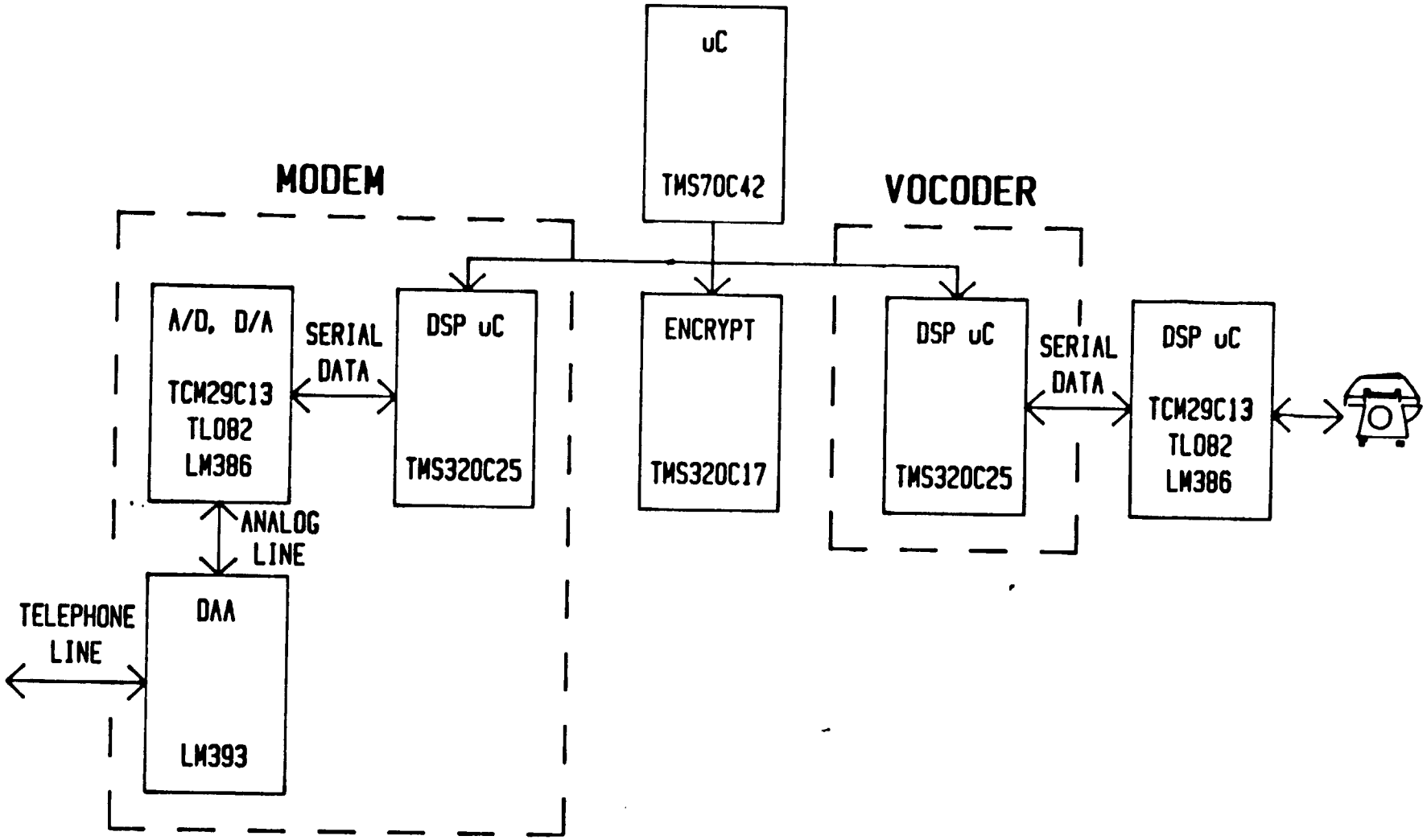
(10-5-41)

SECURE PHONE SYSTEM BLOCK DIAGRAM



(10-5-49)

SECURE PHONE BLOCK DIAGRAM



(10-5-50)

AT&T PRESENTS

THE WE® DSP32 AND WE® DSP16 DIGITAL SIGNAL PROCESSOR FAMILIES

AT&T

TECHNICAL SEMINAR

75922.01

© 1987 AT&T Technologies, Inc.

Notes

PRECEDING PAGE BLANK NOT FILMED

AGENDA

- Introduction
- DSP32
  - Features
  - Architecture
  - I/O Interfacing

    Break
  - Instruction set
  - Benchmarks
- Development Tools
  - Software library
  - Development system
- DSP32C
  - Features
  - Architecture
  - Instruction Set
  - Support
- DSP16
  - Features
  - Architecture
  - I/O Interfacing

    Break
  - Instruction set
  - Benchmarks
- Development Tools
  - Software library
  - Development system
- Concluding Remarks
  - Support

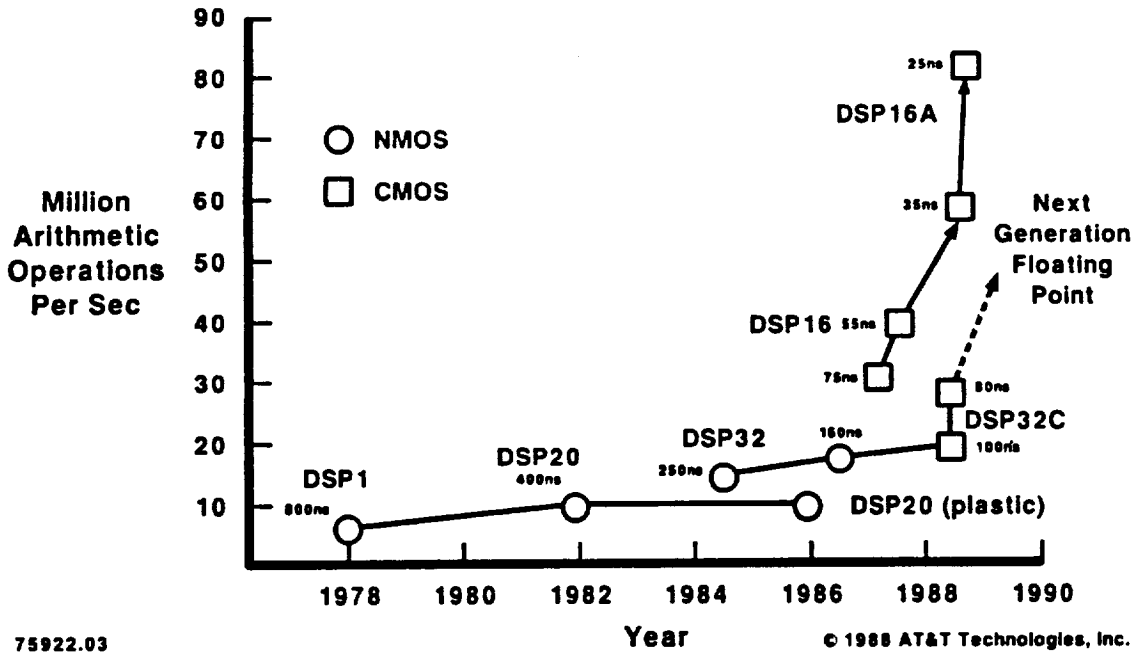
75922.02

© 1987 AT&T Technologies, Inc.

Notes

PRECEDING PAGE BLANK NOT FILMED

DSP PRODUCT EVOLUTION



75922.03

© 1988 AT&T Technologies, Inc.

Notes

DSP APPLICATIONS

Processor Size	DSP Device	Applications Markets:
32-Bit floating point	DSP32C DSP32	Computing (PC, minis) - Main processing element - Math accelerator Work stations - Graphics (high end) Image processing Advanced speech Advanced telecom Robotics Digital audio Graphics Servo control
16-Bit fixed point	DSP16A DSP16	Speech - Low bit rate Graphics (low end) Controllers Telecom - PBX - Transmission - Modems Consumer

75922.04

© 1988 AT&T Technologies, Inc.

Notes

WHY FLOATING POINT?

- Higher precision
- Wider dynamic range
- Ease of use (auto-scaling)

65169.08

© 1986 AT&T Technologies, Inc.

Notes

WHY FIXED POINT?

- High speed
- Low power
- Low cost

65169.10

© 1986 AT&T Technologies, Inc.

Notes

WHY SINGLE-CHIP?

- Reduced power
- Reduced board area
- Reduced cost
- Reduced development time

65169.11

© 1986 AT&T Technologies, Inc.

Notes

AT&T PRESENTS

THE WE® DSP32 DIGITAL SIGNAL PROCESSOR

AT&T

65169.12

© 1986 AT&T Technologies, Inc.

Notes

WE® DSP32

- Features
  - Technology
  - Architecture
  - Instruction set
- Architecture
  - High speed
  - Functional diagram
  - Data arithmetic unit
  - Floating point format
  - Floating point advantages
  - Control arithmetic unit
  - Memory
  - Memory map
  - Serial interface
  - Parallel interface
  - Error control logic
- I/O interfacing
  - Microcomputer
  - External memory
  - CODEC
  - Multiprocessing
- Instruction set
  - Data arithmetic unit instructions
  - Control arithmetic unit instructions
- Addressing modes
  - Immediate
  - Memory direct
  - Register direct
  - Register indirect
- Filter example
- Benchmarks

© 1986 AT&T Technologies, Inc.

Notes

DSP32 FEATURES
Technology

- 1.5 micron NMOS
- 155,000 transistors
- Chip area-81 square mm, (12.7mm x 6.35mm)
- 40-pin DIP and 100-pin PGA
- 16 MHz clock (250ns instruction cycle)
- 25 MHz clock (160ns instruction cycle)
- Power dissipation
  - 1.8W (Typ), 2.3W (max) for DIP
  - 2.0W (Typ), 2.6W (max) for PGA
- Single +5V power supply

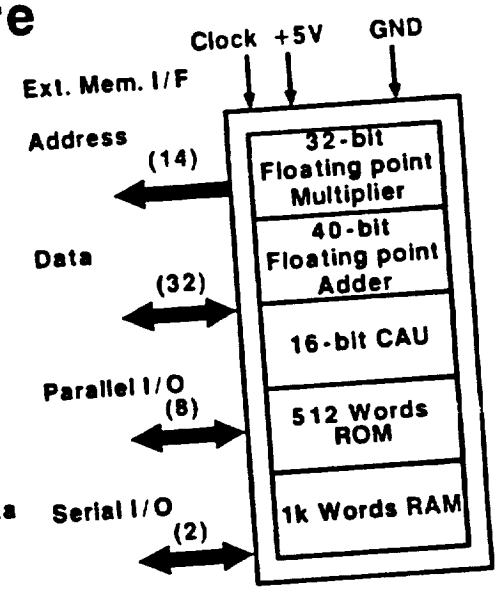
75922.06

© 1987 AT&T Technologies, Inc.

Notes

DSP32 FEATURES
Architecture

- Full 32-bit floating point arithmetic
- All instructions are single-cycle
- Two execution units
  1) DAU 32-bit floating point
     - Parallel multiplier and adder
     - Automatic post-normalization
     - Four 40-bit accumulators
  2) CAU 16-bit RISC μ P
     - 16-bit ALU
     - Address generation for DAU
     - Conditional operation
- Memory
  - Flexible. Programs or data in RAM or ROM
  - No speed penalty for off-chip program or data
  - 4 accesses per instruction cycle
  - Byte Addressable
- Serial and parallel I/O ports with DMA

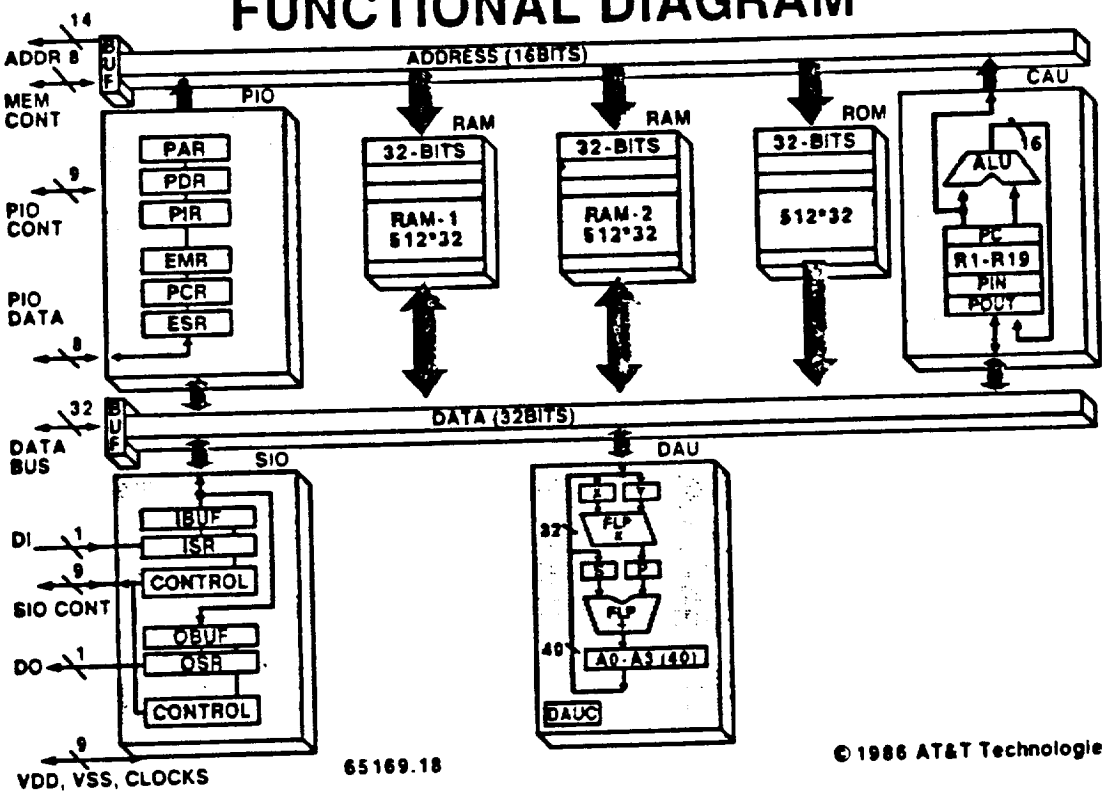


© 1987 AT&T Technologies, Inc.

75922.07

Notes

FUNCTIONAL DIAGRAM



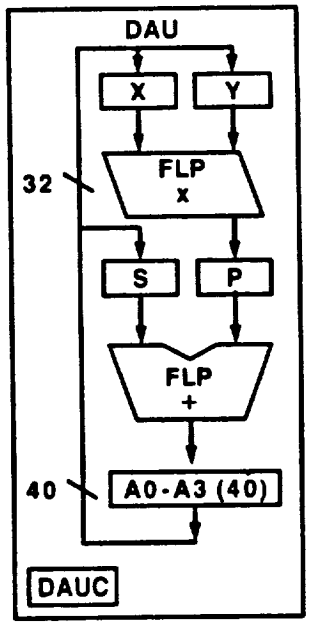
© 1986 AT&T Technologies, Inc.

Notes

DATA ARITHMETIC UNIT (DAU)

- Executes over six million instructions per second
  - Instructions have the form a=b+cxd
  - Each instruction includes two floating point operations
  - Inputs are 32-bit floating-point numbers with 24-bit mantissa and 8-bit exponent
- Multiplier and adder input sources:
  - Memory
  - Accumulator (A0-A3)
  - SIO Register
- Multiplier and adder operate in parallel
- Adder is 40 bits wide
- Accumulators maintain eight guard bits for extra precision
- Data type conversions performed in DAU
  - Floating point to-and-from integer
  - Floating point to-and-from μ-law and A-law
- All normalization is automatic with no added pipeline stages

75922.09

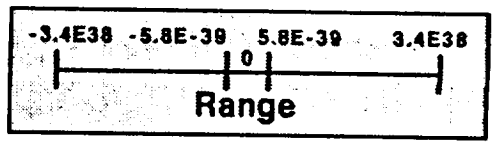


© 1988 AT&T Technologies, Inc.

Notes

ADVANTAGES OF FLOATING POINT ARITHMETIC

- Provides large dynamic range (>1500 dB)
- Provides high precision independent of magnitude
- Eliminates need for scaling after intermediate computations
  - Simplifies software development
  - Reduces program size & complexity
- Algorithms can be developed on larger floating-point machines
- Advanced Applications



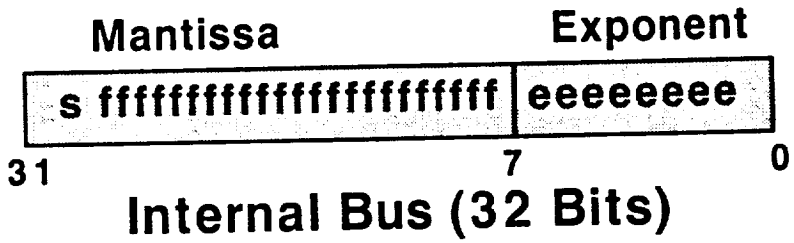
65169.21

© 1986 AT&T Technologies, Inc.

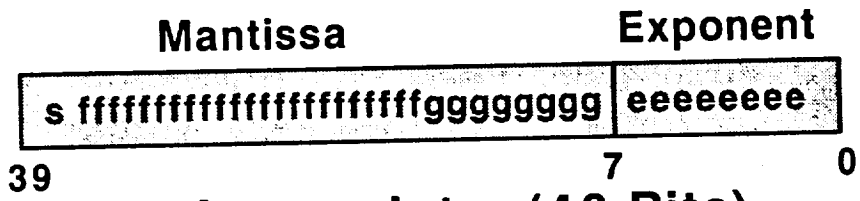
Notes

FLOATING POINT DATA FORMAT

2's Complement Data



Internal Bus (32 Bits)



Accumulator (40 Bits)

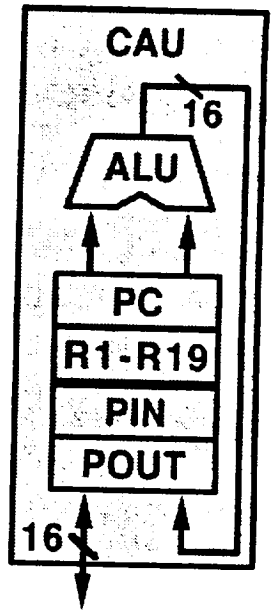
65169.22

© 1986 AT&T Technologies, Inc.

Notes

CONTROL ARITHMETIC UNIT (CAU)

- Generates and modifies addresses
- Executes microprocessor-like instructions on 16-bit data:
  - 2's complement fixed-point arithmetic
  - Logic operations
  - Conditional branches
  - Data moves between memory or I/O registers and any of the CAU registers
- Contains twenty-one 16-bit general purpose static registers, also used as:
  - DAU Memory pointers, R1-R14
  - DAU Increment pointers, R15-R19
  - SIO DMA pointers, R20-R21



65169.23

© 1986 AT&T Technologies, Inc.

Notes

DSP32 MEMORY

- All instructions are 32 bits
- All memory is byte-addressable
- RAM and ROM can be used for either
  - Program
    - or -
  - Data
- Zero chip interface to standard byte-wide memory chips
- No speed penalty for using off-chip memory

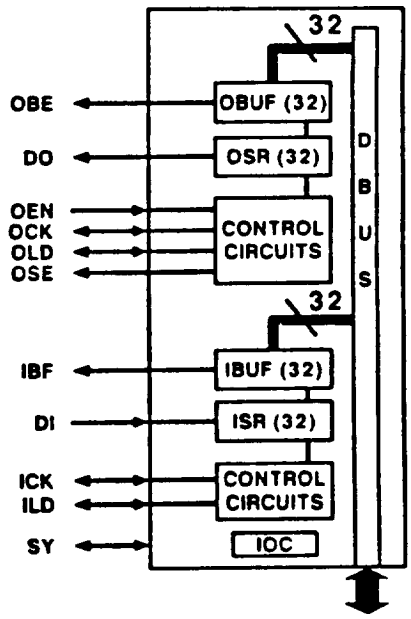
© 1987 AT&T Technologies, Inc.

75922.10

Notes

SERIAL INTERFACE

- Separate transmit and receive ports
- Double buffering for back-to-back transfers
- 8-, 16-, or 32-bit data width modes available
- Data transfers using on-chip or external clock
  - 12.5 Mb/s data rate with external clock
  - 3.2 Mb/s data rate with on-chip clock
- Provides external control signals for:
  - Direct TDM line interface
  - Direct T7500 or T7520 CODEC interface
- Zero chip interface to other DSP32's
- Allows for serial DMA transfers
  - Useful for multichannel applications



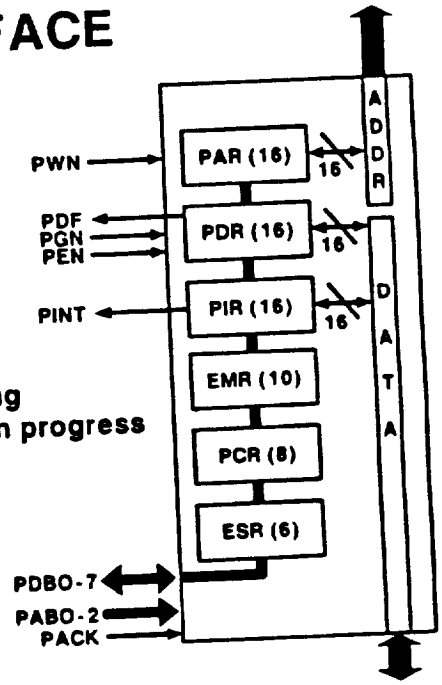
75922.11

© 1987 AT&T Technologies, Inc.

Notes

PARALLEL INTERFACE

- Microprocessor-like 8-bit bidirectional parallel port
- 9-pin control interface
- Port allows for both program and DMA controlled data transfers
- PIO DMA allows:
  - Downloading an application program/uploading results without interrupting another program in progress
  - Memory addresses can be auto-incremented
  - 2.3 Mbyte/sec
- Error control logic



© 1987 AT&T Technologies, Inc.

75922.12

Notes

INTERFACING THE *WE*® DSP32

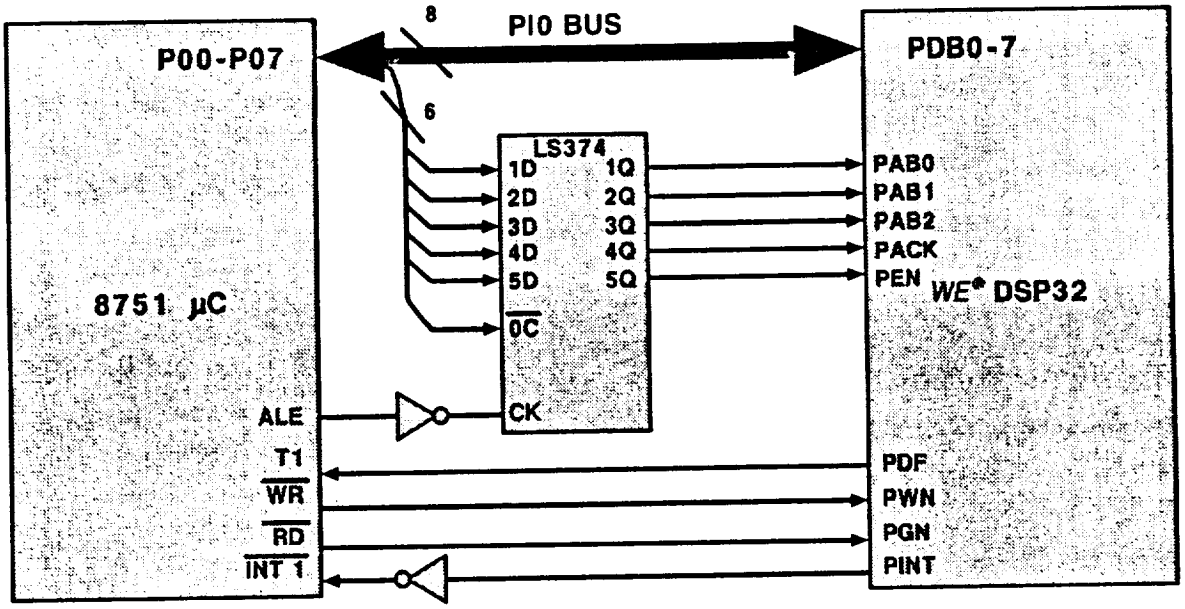
- Microcomputer
- External memory
- CODEC
- Multiprocessing

65169.31

© 1986 AT&T Technologies, Inc.

Notes

TYPICAL MICROCOMPUTER INTERFACE

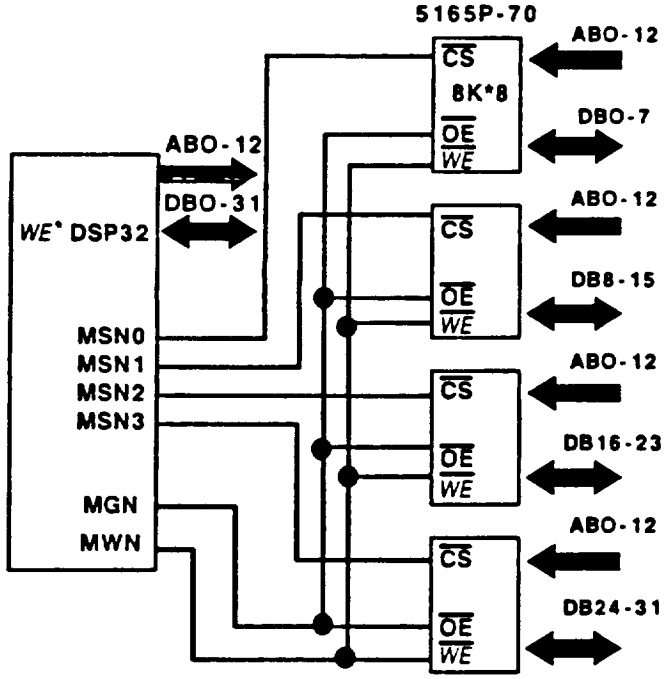


65169.32

© 1986 AT&T Technologies, Inc.

Notes

EXTERNAL MEMORY EXPANSION (32 K BYTES)

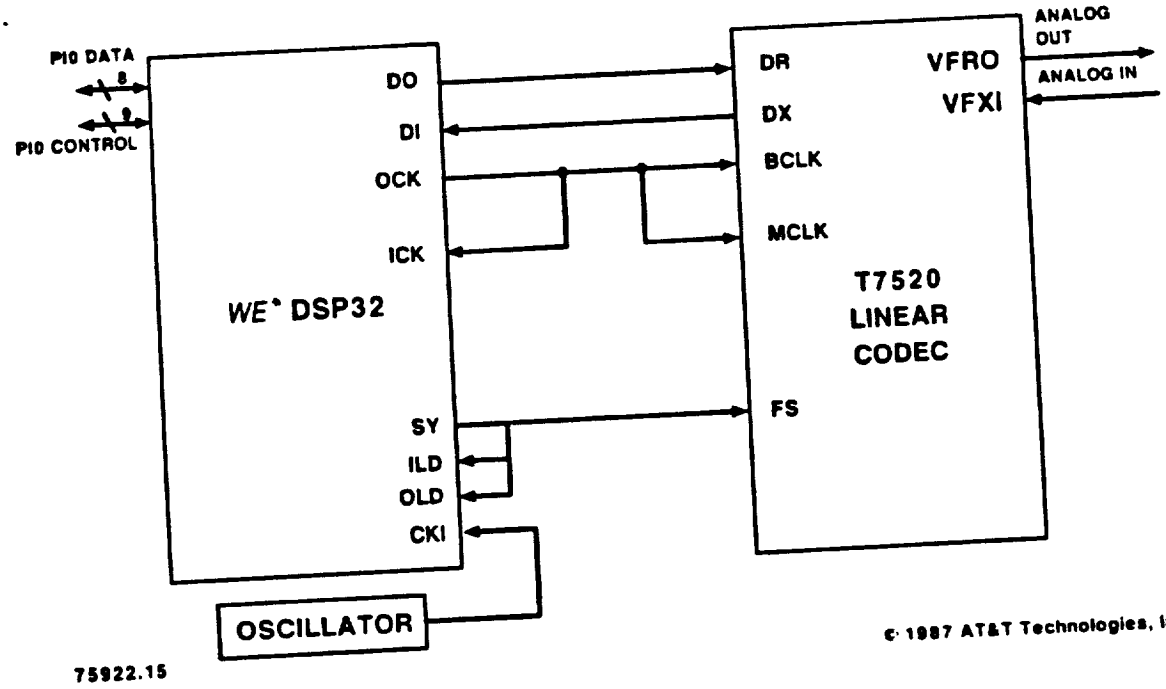


75922.14

© 1987 AT&T Technologies, Inc.

Notes

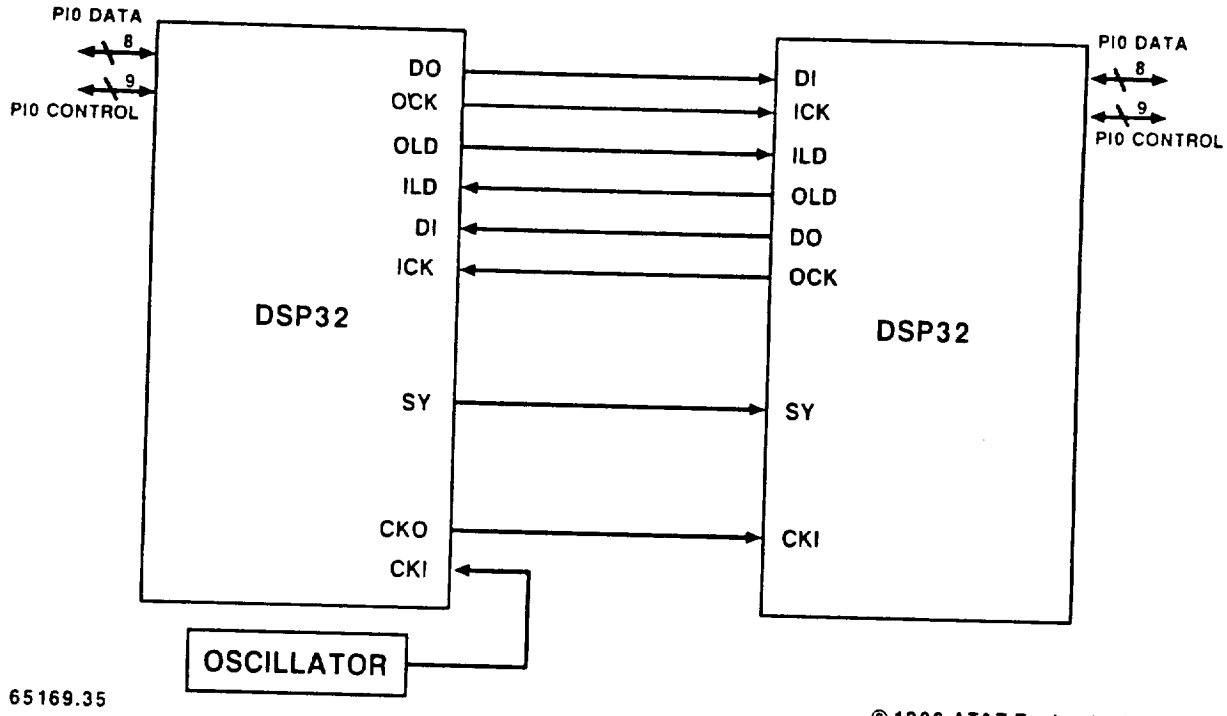
CODEC INTERFACE



© 1987 AT&T Technologies, Inc.

Notes

MULTIPROCESSING USING SERIAL COMMUNICATION



65169.35

© 1986 AT&T Technologies, Inc.

Notes

WE® DSP32 INSTRUCTION SET

- Data arithmetic unit (DAU) instructions
  - Multiply / accumulate
  - Special functions
- Control arithmetic unit (CAU) instructions
  - Arithmetic & logic
  - Data move
  - Control group

65169.38

© 1986 AT&T Technologies, Inc.

Notes

WE® DSP32 DAU INSTRUCTIONS

Mult/accumulate: [Z=] aN = aM + Y * X
[Z=] aN = Y + aM * X
aN = aM + (Z = Y) * X

Type conversions: [Z=] aN = function (Y)

where:

aN, aM is one of the four 40-bit accumulators
X, Y, Z fields can be:

- Memory locations
- I/0 registers
- Accumulators

[Z=] indicates that the result is optionally sent to Z

65169.39

© 1986 AT&T Technologies, Inc.

Notes

Example:

DAU MULTIPLY-ACCUMULATE GROUP

$$[Z =]\ aN = aM + Y * X$$

$$\underbrace{*r5{+}{+}}_{Z} = \underbrace{a1}_{aN} = \underbrace{a0}_{aM} + \underbrace{*r7}_{Y} * \underbrace{*r10\ {+}{+}\ r17}_{X}$$

65169.40

© 1986 AT&T Technologies, Inc.

Notes

DSP32 CAU INSTRUCTIONS

ALU:	rD=rD op(rS,N)
Control:	if (cond) goto label
	Call label (rM)
	Return (rM)
Data Move:	(mem, I/O reg)=rS
	rD=(mem, I/O reg)

where:

rD,rS,rM are selected from the 21 16-bit CAU registers,
rH is any of the 21 CAU registers or the pc register,
N is a 16-bit operand contained in the 32-bit instruction.

75922.18

© 1988 AT&T Technologies, Inc.

Notes

CAU INSTRUCTIONS

Examples:

r11 = r11 & r17

*r7++ = r10

if(eq) goto r3+12

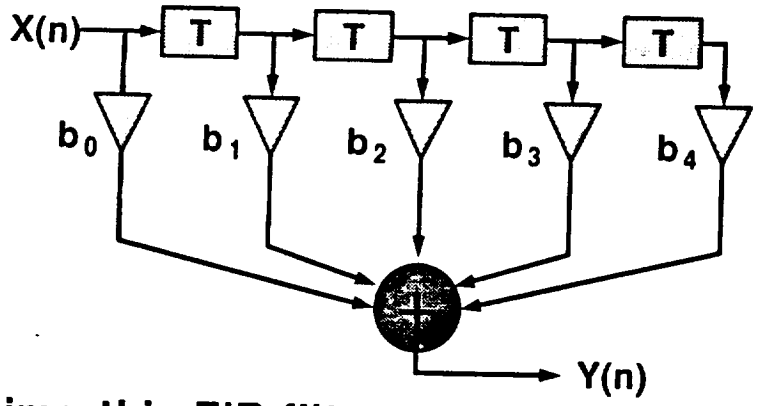
© 1986 AT&T Technologies, Inc.

65169.42

Notes

FIR FILTER EXAMPLE

$$Y(n) = b_0 X(n) + b_1 X(n-1) + b_2 X(n-2) + b_3 X(n-3) + b_4 X(n-4)$$



Each time this FIR filter equation is computed:

- A new data sample is input via the SIO port
- The oldest sample is overwritten
- The filter output value is output via the SIO port

65169.45

© 1986 AT&T Technologies, Inc.

Notes

FILTER EXAMPLE (cont.)

Code:	Comments:
. rsect "ROM"	/*ROM code*/
main: ioc = 0x547	/*Set the ioc and dauc*/
dauc = 0	/*Control registers*/
wait: if (ibe) goto wait	/*Wait for new data sample*/
r1 = data	/*Initialize*/
r3 = data	
r2 = coef	/*Pointers*/
a1 = ic (ibuf)	/*Input new data sample*/
a0 = *r1++* *r2++	/*b_4X(n-4)*/
a0 = a0+(*r3++ = *r1++) **r2++	/*a0 + b_3X(n-3)*/
a0 = a0+(*r3++ = *r1++) **r2++	/*a0 + b_2X(n-2)*/
a0 = a0+(*r3++ = *r1++) **r2++	/*a0 + b_1X(n-1)*/
a1 = a0+(*r3++ = a1)**r2++	/*a0 + b_0X(n)*/
goto wait	
obuf = a1 = oc(a1)	/*Filter output*/
coef: b_4,b_3,b_2,b_1,b_0	
.rsect "RAM"	/*Put data in RAM*/
data: x(n-4), x(n-3), x(n-2), x(n-1)	

65169.46

© 1986 AT&T Technologies, Inc.

Notes

DSP32 BENCHMARKS

Function	@160ns	@250ns	Units
• Complex FFT (includes bit reversal)			
- 256 point	1.90	2.98	ms
- 1024 point	8.96	14.0	ms
• Filters			
- FIR	160	250	ns/tap
- Adaptive FIR	320	500	ns/tap
- 5-Multiply IIR (SOS)	800	1250	ns
- 4-Multiply IIR (SOS)	640	1000	ns
• Numeric processing			
- Logarithm	3.1	4.8	µs
- Divide	2.9	4.5	µs
- Matrix multiplication (2x2) x (2x2)	1.28	2.0	µs
- Matrix multiplication (4x4) x (4x1)	2.56	4.00	µs
- Matrix multiplication (4x4) x (4x4)	10.2	16.0	µs

75922.19

© 1988 AT&T Technologies, Inc.

Notes

DEVELOPMENT TOOLS FOR THE WE® DSP32

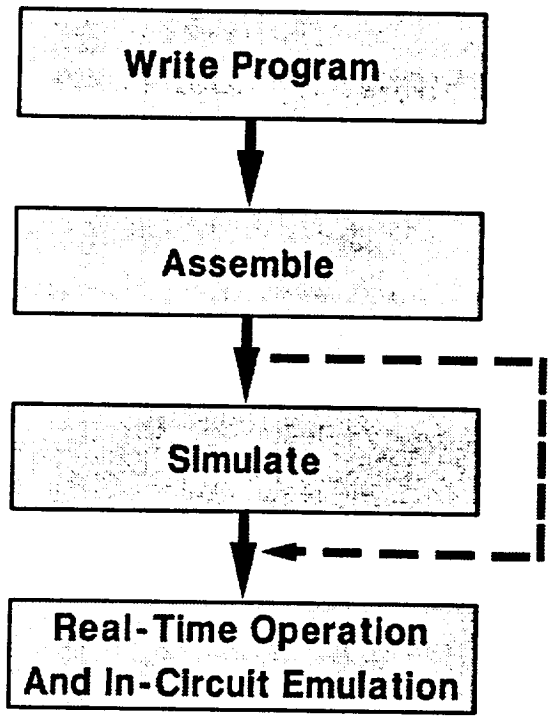
© 1986 AT&T Technologies, Inc.

65169.48

Notes

APPLICATIONS DEVELOPMENT SUPPORT

- Text editor
- Assembler / Link Editor
- Simulator
- Develpment system operating under Simulator control

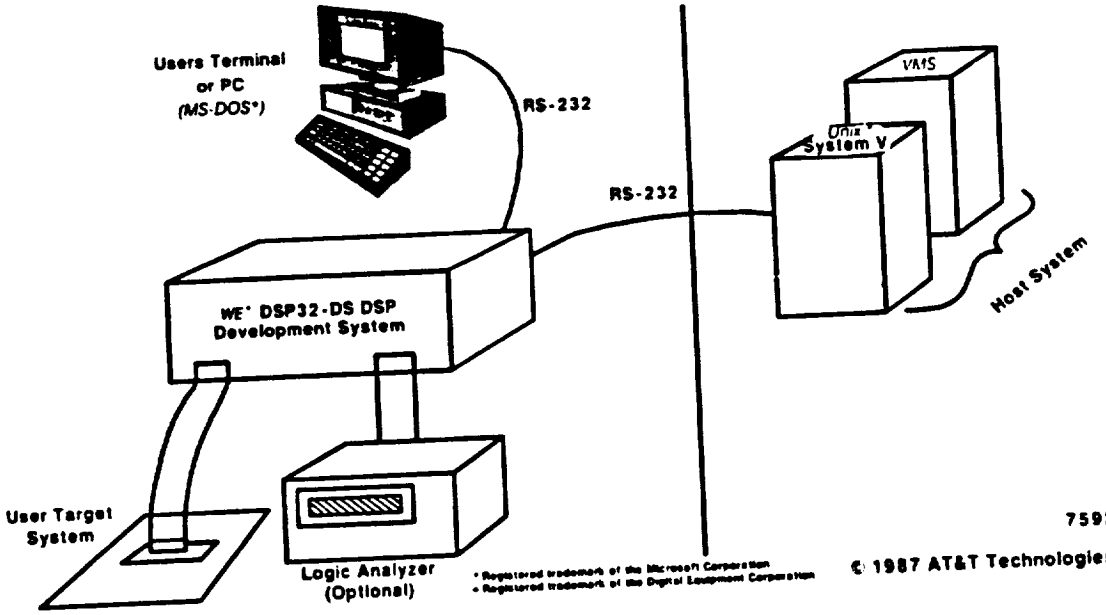


65169.49

© 1986 AT&T Technologies, Inc.

Notes

APPLICATION DEVELOPMENT ENVIRONMENT



© 1987 AT&T Technologies, Inc.

Notes

DSP32 DEVELOPMENT SOFTWARE

- Runs on the *UNIX*® or *MS-DOS** Operating Systems
- Assembler
  - High-level syntax
  - C preprocessor
- C-Compiler
- Simulator
  - Fast (1500 ins/s)
  - Register and memory access
  - Breakpoints
  - Symbolic debugging
  - User-defined functions
  - I/O files

* Registered trademark of the Microsoft Corporation

75922.22

© 1988 AT&T Technologies, Inc.

Notes

C-Compiler

- Full Kernighan and Ritchie specification
- Optimizer
  - Speed or memory optimization
- Compatibility with DSP32-SL
  - Allows linking of DSP32 source or object code

© 1987 AT&T Technologies, Inc.

75922.23

Notes

WE® DSP32 ASSEMBLER EXAMPLE

Equation:

$$A(k) = A(k) + U^* A(i) - V^* A(i+1)$$

DSP32 syntax:

```
#define i r1
#define k r2
#define U a2
#define V a3

          a0 = *k + U* *i++;
    *k =  a0 = a0 - V* *i;
```

65169.52

© 1986 AT&T Technologies, Inc.

Notes

WE® DSP32 DEVELOPMENT SYSTEM

- Full speed program and I/O interface debugging
- Display of registers and memory
- Software breakpoints
- Buffered I/O for in-circuit emulation
- Multiple DSP support
  - Emulation capabilities for seven DSP32's by cascading the development systems
- Convenient size
- Controlled by the Simulator
  - Can load and run the same program in both the DSP32-DS and the Simulator, and automatically compare the results

65169.57

© 1986 AT&T Technologies, Inc.

Notes

The *WE*® DSP32C Digital Signal Processor

75922.27

© 1987 AT&T Technologies, Inc.

Notes

DSP32C

Features

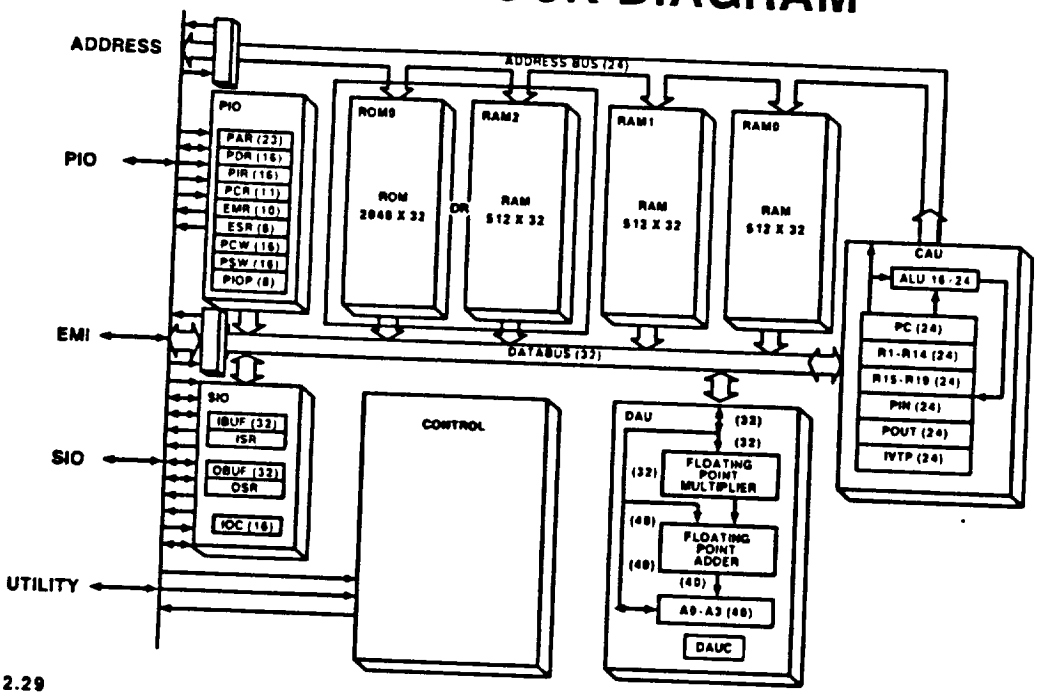
- 0.75 μm CMOS technology
- 80 ns (all instructions are single-cycle)
- 32-bit floating point format
- Four memory accesses per instruction
- Four 40-bit accumulators
- On-chip memory
  - Three banks of 512x32 RAM
  - *OR* Two banks of 512x32 RAM with 2Kx32 ROM
- 16M bytes external memory
- 24-bit μP like operations
- Low overhead looping
- 16-bit parallel port
- 16M bit/s serial port
- Bit reversed addressing
- IEEE conversion (single-cycle)
- 133-pin PGA

© 1987 AT&T Technologies, Inc.

75922.28

Notes

DSP32C BLOCK DIAGRAM



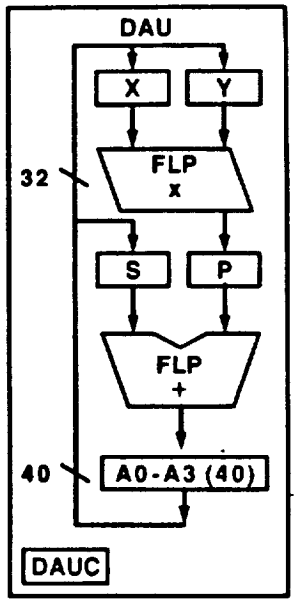
75922.29

© 1988 AT&T Technologies, Inc.

Notes

DSP32C
DATA ARITHMETIC UNIT (DAU)

- Executes 12.5 million instructions per second
  - Instructions have the form a=b+c*d
  - Each instruction includes two floating point operations
  - Inputs are 32-bit floating-point numbers with 24-bit mantissa and 8-bit exponent
- Multiplier and adder input sources:
  - Memory
  - Accumulators (A0-A3)
  - SIO Register
- Multiplier and adder operate in parallel
- Adder is 40-bits wide
- Accumulators maintain eight guard bits for extra precision
- Data type conversions performed in DAU
  - Floating point to-and-from integer (16 or 24 bits)
  - Floating point to-and-from μ-law and A-law
  - Floating point to-and-from 8 bit linear
  - Floating point to-and-from IEEE



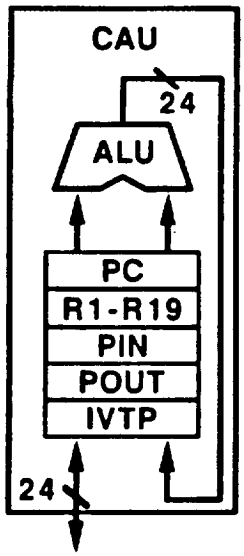
75922.30

© 1988 AT&T Technologies, Inc.

Notes

CONTROL ARITHMETIC UNIT (CAU)

- Generates and modifies addresses
- Executes microprocessor-like instructions on 16- or 24-bit data:
  - 2's complement fixed-point arithmetic
  - Logic operations
  - Conditional branches
  - Conditional ALU operations
  - Data moves between memory or I/O registers and any of the CAU registers
- Contains twenty-two 16-bit general-purpose static registers, also used as:
  - DAU Memory pointers, R1-R14
  - DAU Increment pointers, R15-R19
  - SIO DMA pointers, R20-R21
  - IVTP R22



75922.31

© 1987 AT&T Technologies, Inc.

Notes

DSP32C

Memory

- On-chip RAM 1.5K words
- Or 1K words of RAM with on-chip ROM
- On-chip ROM 2K words
- 16M bytes external memory
- All memory is byte addressable
- Wait states for slower memory
  - A wait state is one quarter instruction cycle (0, 1, 2, or 3 waits)
  - Synchronous ready
- Bus arbitration logic (Bus Request / Request Acknowledge)

75922.32

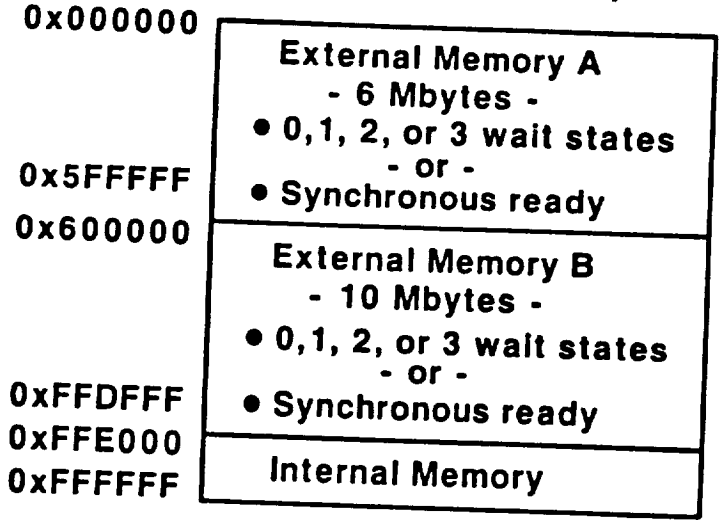
© 1988 AT&T Technologies, Inc.

Notes

DSP32C

Memory Map

- Two independent external speed partitions



75922.78

© 1988 AT&T Technologies, Inc.

Notes

DSP32C

Input / Output

- DMA, interrupt driven, and programmed I / O
- Software configurable (reduces glue logic)
- Serial I / O
  - 16M bits / second
  - 8, 16, 24, or 32-bit transfers
  - Zero glue logic to codes, TDMs, and other DSPs
- Parallel I / O
  - 8 or 16-bit parallel port
  - Microprocessor interface

75922.33

© 1988 AT&T Technologies, Inc.

Notes

DSP32C
Interrupts

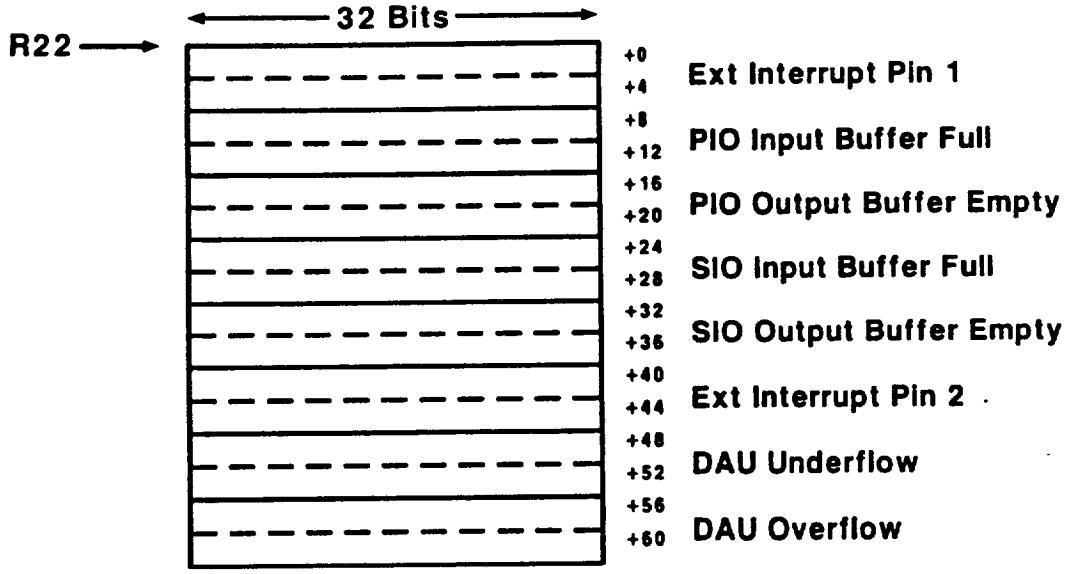
- 8 Vectored interrupts
  - Two external pins
  - SIO read or write
  - PIO read or write
  - DAU overflow or underflow

75922.34

© 1987 AT&T Technologies, Inc.

Notes

DSP32C
(Interrupt Vector Table)

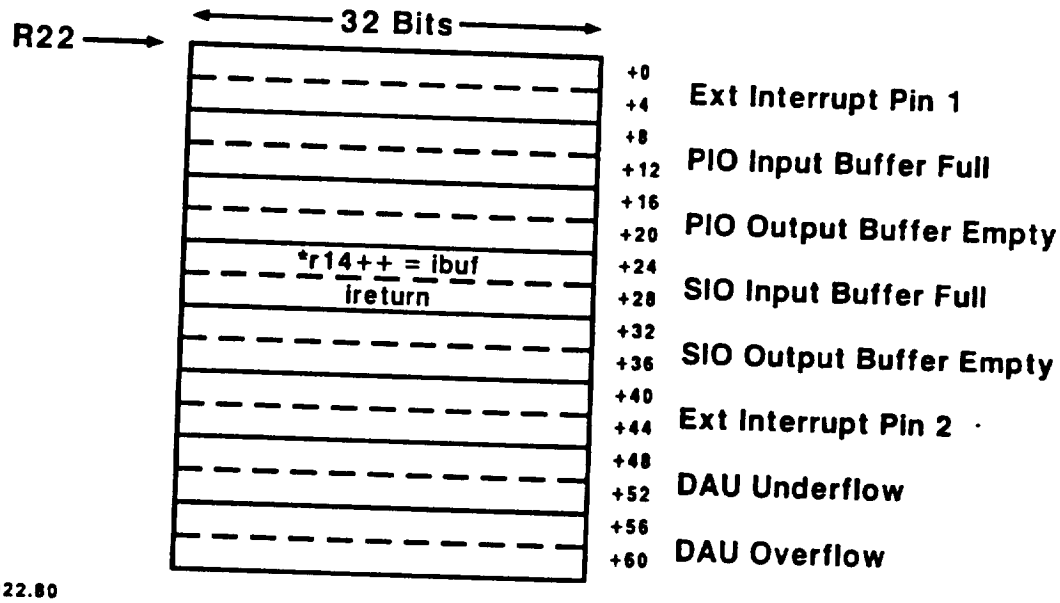


75922.79

© 1988 AT&T Technologies, Inc.

Notes

DSP32C
(Quick Interrupt Example)



75922.80

© 1988 AT&T Technologies, Inc.

Notes

DSP32C INSTRUCTION SET

- Data arithmetic unit (DAU) instructions
  - Multiply / accumulate
  - Special functions
- Control arithmetic unit (CAU) instructions
  - Arithmetic & logic
  - Data move
  - Control group

75922.35

© 1987 AT&T Technologies, Inc.

Notes

DSP32C
DAU Instructions

Multiply/Accumulate

[Z=] aN = aM + Y * X
aN = aM + (Z=Y) * X
aN = (Z=Y) + X

Special Functions

[Z=] aN = Function (Y)

Where:
aN,aM = One of the four accumulators,
X,Y,Z = Memory, I/O register, or accumulator,
[Z=] is an optional field.

75922.36

© 1988 AT&T Technologies, Inc.

Notes

DSP32C

CAU Instructions

Arithmetic and Logical

Examples:

```
r1 = r2 & r3
if (eq) r1 = r2 & r3
r1e = r1 # r2
```

Conditional branch

Example:

```
if (ibf) goto r1
```

Looping (low overhead)

Example:

```
do 71 {
      a0 = a0 + *r1++**r2
      }
```

75922.37

© 1988 AT&T Technologies, Inc.

Notes

DSP32C
Benchmarks

Benchmark	DSP32	DSP32C	Units
FIR	160	80	ns
AFIR	320	160	ns
SOS	800	400	ns
Lattice	320	160	ns
Mat. Mpy (3x3)(3x1)	1440	720	ns
256pt Window	40	20	μs
1024pt FFT	8.96	3.3	ms

75922.38

© 1988 AT&T Technologies, Inc.

Notes

DSP32C

Development Support

- Assembler, linker, and simulator (*UNIX*® and *MS-DOS**)
- Hardware development system
  - PC environment (plug-in card)
  - In-circuit emulation (ICE)
  - Real-time development
- C Compiler
- Application library
- Application guide
- Third party support
  - Atlanta Signal Processors, Inc (ASPI)
  - Burr-Brown
  - CAC
- University program

* Registered trademark of Microsoft Corp.

75922.39

© 1988 AT&T Technologies, Inc.

Notes

AT&T PRESENTS

THE WE® DSP16 DIGITAL SIGNAL PROCESSOR

AT&T

TECHNICAL SEMINAR

65169.61

© 1986 AT&T Technologies, Inc.

Notes

WE® DSP16

- Features
  - Technology
  - Architecture
  - Instruction set
- Architecture
  - Functional diagram
  - Data arithmetic unit
  - X address arithmetic unit
  - Y address arithmetic unit
  - Memory
  - Instruction cache
  - Serial interface
  - Multiprocessor mode
  - Parallel I/O
- I/O Interfacing
  - Microcomputer
  - External program memory
  - External data memory
  - Codec
  - Multiprocessing
- Instruction set
  - Multiply/accumulate
  - ALU
  - Data move
  - Control
  - Special functions
  - Cache
- Addressing modes
  - Immediate
  - Indirect
  - Compound
  - Modulo
- Interrupts
- Program example
- Benchmarks

© 1986 AT&T Technologies, Inc.

65169.62

Notes

DSP16 FEATURES
Technology

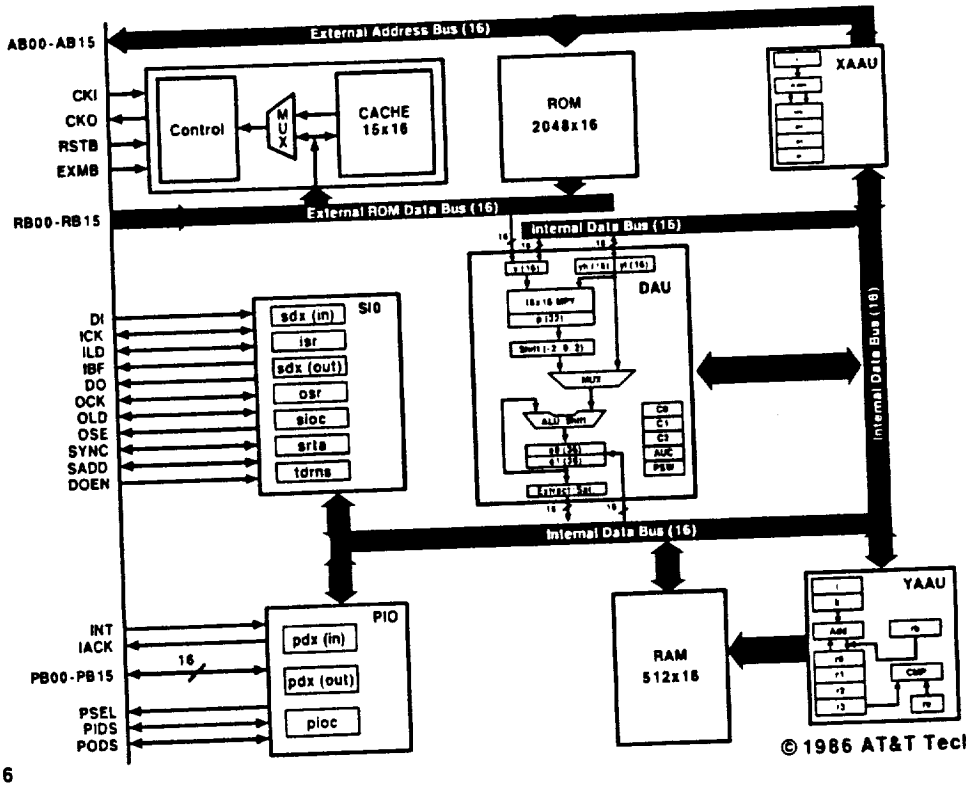
- 1.0 micron CMOS
- 140,000 transistors
- Chip area - 56 square mm
- Available in two packages:
  - 84-pin plastic leaded chip carrier
  - 84-pin ceramic PGA
- 26.6 MHz clock (75ns instruction cycle)
- 36.3 MHz clock (55ns instruction cycle)
- 350 milliwatts @ 75ns (worst case)
- 250 milliwatts @ 75ns (typical)
- Single +5 volt supply

75922.41

© 1987 AT&T Technologies, Inc.

Notes

WE® DSP16 BLOCK DIAGRAM



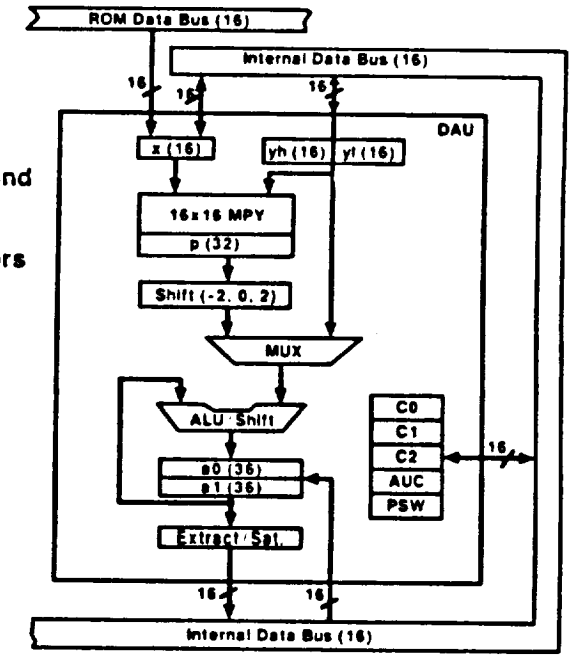
© 1986 AT&T Technologies, Inc.

65169.66

Notes

DATA ARITHMETIC UNIT (DAU)

- Two's complement arithmetic
- 16x16 multiplier with 32-bit product
- 18.2 million multiply/accumulates per second
- Two 36-bit general purpose accumulators
- Full precision load and store of accumulators in 16-bit words
- Programmable product alignment
- Operands to the multiplier come from memory or an accumulator
- Programmable saturation on overflow
- ALU for arithmetic and logical operations
  - Operations on 16-bit or 32-bit data
  - ALU operation results can be tested for branching



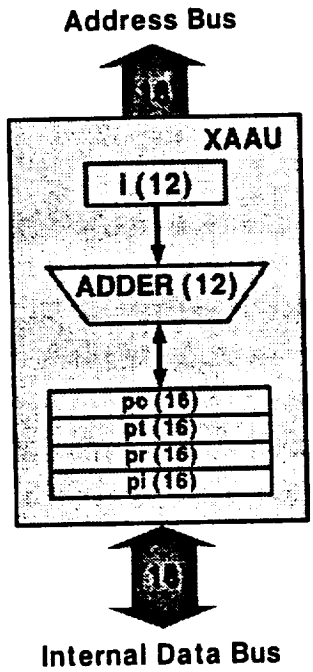
75922.45

© 1987 AT&T Technologies, Inc.

Notes

X ADDRESS ARITHMETIC UNIT (XAAU)

- Calculates addresses to ROM
- XAAU contains five registers:
  - Program count (pc)
  - Pointer for ROM look-up table (pt)
  - User defined post-increment register (i)
  - Return address for subroutines (pr)
  - Return address for interrupts (pi)
- Nested subroutine (store pr instruction)



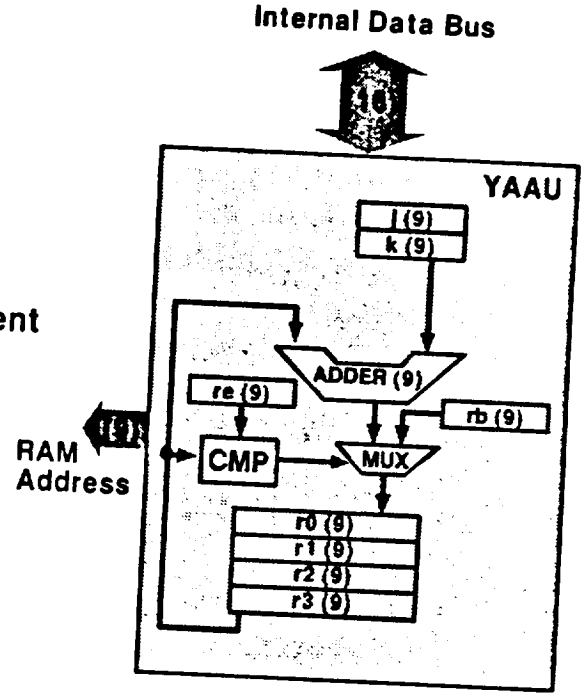
65169.69

© 1986 AT&T Technologies, Inc.

Notes

Y ADDRESS ARITHMETIC UNIT (YAAU)

- Calculates addresses to RAM
- Four address registers:
  - Used to read and write RAM without restriction
  - Registers (rb, re) to support modulo addressing
  - Two programmable two's complement post modify registers
  - Fixed post modifications of +1, -1, are available



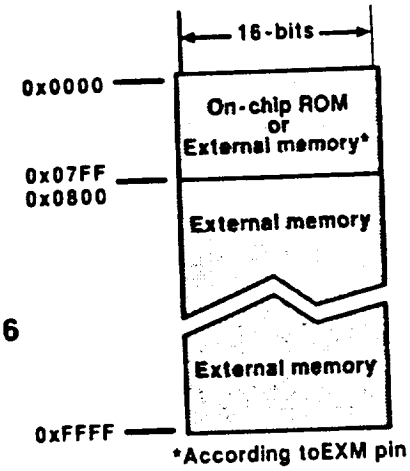
65169.70

© 1986 AT&T Technologies, Inc.

Notes

MEMORY

- On-chip ROM 2048x16
  - Instructions (16-bits)
  - Coefficients (16-bits)
  - Mask-programmable (on second level metal)
- On-chip RAM 512x16
  - Data storage
  - Static (no refreshing)
- Instruction cache (15x16)
- Program memory expandable off-chip to 64Kx16
- Full speed prototyping with 45ns memories



DSP16 Memory Map

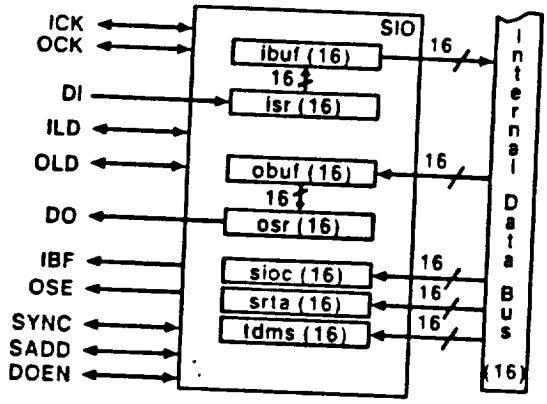
© 1986 AT&T Technologies, Inc.

65169.71

Notes

SERIAL I/O

- 10 Mbits/sec maximum data rate
- Full duplex
- Independent I/O clocks
- Double-buffered for back-to-back transfers
- Multiprocessor mode (zero chip interface up to 8 *WE*® DSP16s)
- I/O configurable through software
- Zero chip interface to:
  - Codecs
  - DSP devices (e.g. *WE* DSP16, *WE* DSP32)
  - Time Division Multiplexed channels (TDM)



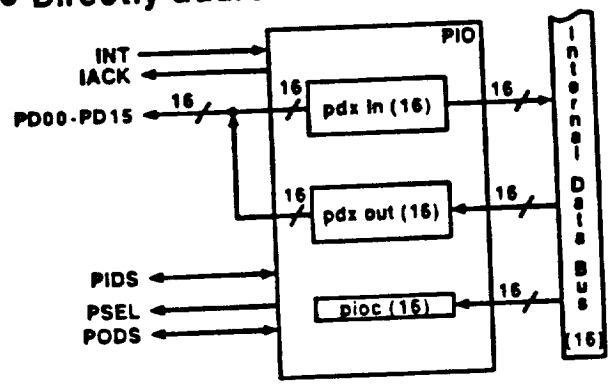
65169.73

© 1986 AT&T Technologies, Inc.

Notes

PARALLEL I/O

- 16-bit/8-bit interface to communicate with:
  - Microprocessors
  - Other DSP16's
  - Peripheral I/O devices
- Data rates up to 145 Mbits/sec
- Configure for internal and external modes
- Transmit and receive 8-bits simultaneously
- Input 16-bits and output 16-bits
- Directly address two devices



75922.47

© 1987 AT&T Technologies, Inc.

Notes

INTERFACING TO THE *WE*® DSP16

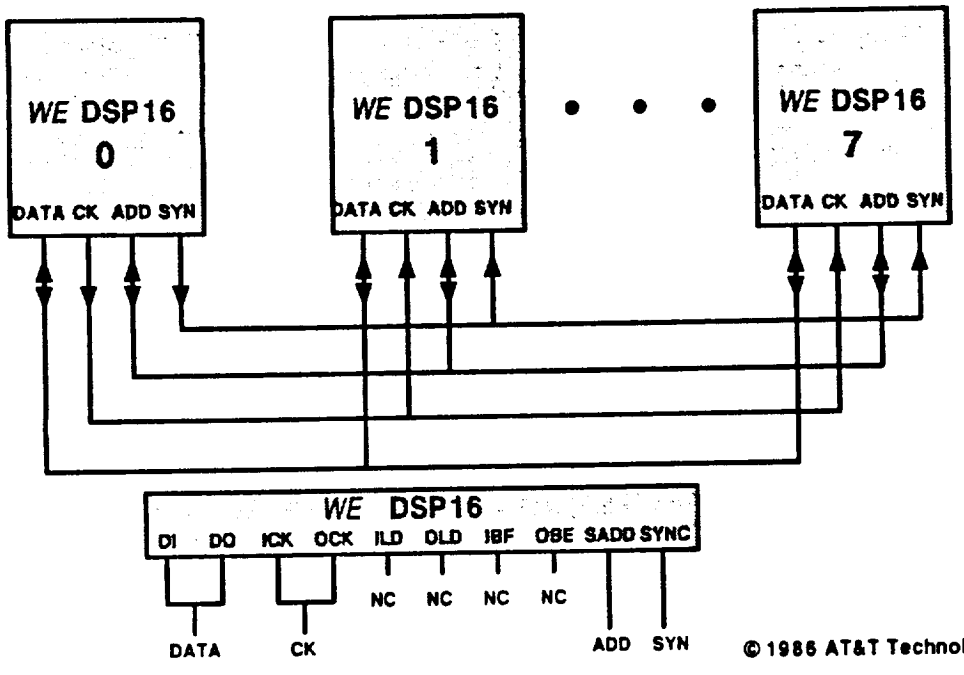
- Microcomputer
- External program memory
- External data memory
- Codec
- Multiprocessing

65169.78

© 1986 AT&T Technologies, Inc.

Notes

WE® SIO MULTIPROCESSOR CONFIGURATION



65169.82

© 1986 AT&T Technologies, Inc.

Notes

MULTIPROCESSOR I/O EXAMPLE

Registers:

Bit	9	8	7	6	5	4	3	2	1	0
Field	SYNC	MODE	Transmit Slot							

Time Division Multiplex Slot (TDMS)

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Receive Address								Transmit Address							

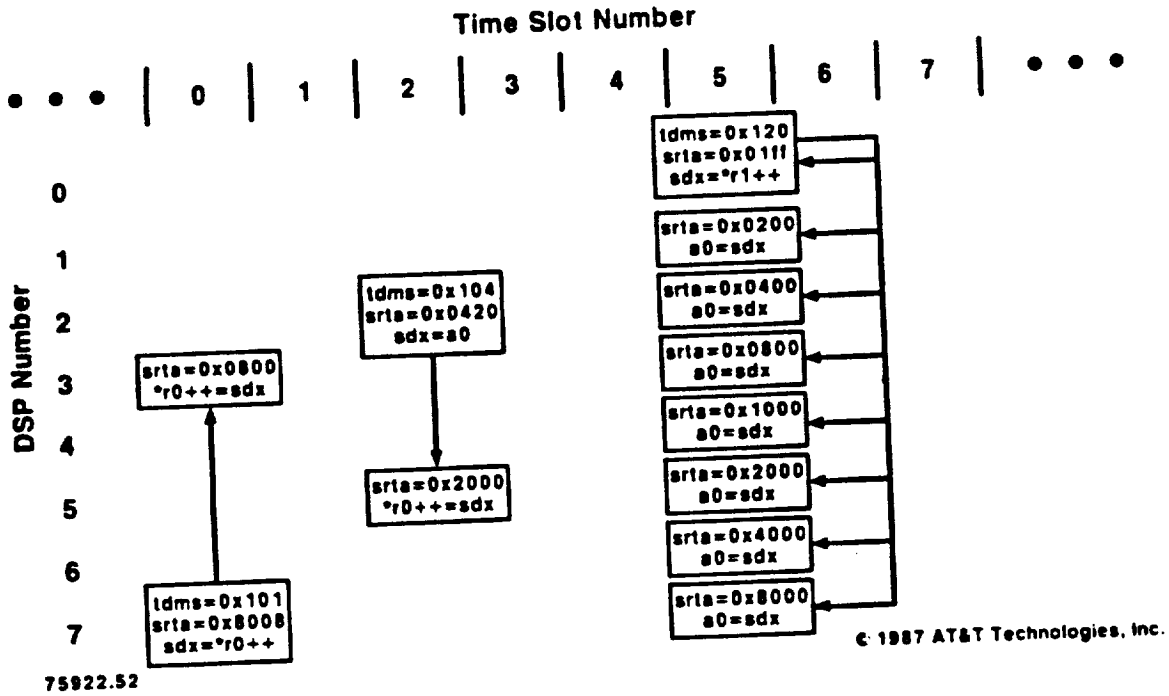
Serial Receive/Transmit Address (SRTA)

65169.83

© 1986 AT&T Technologies, Inc.

Notes

MULTIPROCESSOR I/O EXAMPLE



© 1987 AT&T Technologies, Inc.

Notes

WE® DSP16 MULTIPROCESSOR
I/O Programming

```
        goto start;
intrpt: *r1++ = sdx;
        goto pi;
start:  tdms = 0x101;
        •
        •
        •
        srta = 0x0102;
        sdx = *r0++;
        •
        •
        •
        srta = 0x0104;
        sdx = *r0++;
```

65169.85

© 1986 AT&T Technologies, Inc.

Notes

WE® DSP16 INSTRUCTION SET

- Multiply / Accumulate
- ALU
- Data move
- Control
- Special functions
- Cache

© 1986 AT&T Technologies, Inc.

65169.88

Notes

WE® DSP16 INSTRUCTIONS
Multiply / Accumulate

Function		Transfer	
Nothing		Nothing	
aD = p		y[I] = Y	
aD = aS + p		x = Y	
aD = aS - p		aT[I] = Y	
	p = x*y	Y	
aD = p	p = x*y	Y = aT[I]	
aD = aS + p	p = x*y	Y = y[I]	
aD = aS - p	p = x*y	y = aT	x = X
		y = Y	x = X
		Z : y	x = X
		Z : y[I]	
		Z : aT[I]	

Where:

aD,aS,aT - is one of two DAU accumulators
X - pointer to ROM (*pt++, *pt++I))
Y - pointer to RAM (*rM, *rM++, *rM--, *rM++j)
Z - read/write compound addressing (*rMzp, *rMpz, *rMm2, *rMjk)

65169.89

© 1986 AT&T Technologies, Inc.

Notes

WE® DSP16 INSTRUCTIONS
Multiply / Accumulate

Example:

aD = aS + p	p = x * y	y = Y	x = X
a0 = a0 + p	p = x * y	y = *r1++	x = *pt++
a0 = a0 + p	p = x * y		
a0 = a0 + p			

65169.90

© 1986 AT&T Technologies, Inc.

Notes

WE® DSP16 INSTRUCTIONS
ALU

Function	Transfer
Nothing	Nothing
aD = y	y[I] = Y
aD = aS + y	x = Y
aD = aS - y	aT[I] = Y
aS = aS & y	Y
aD = aS | y	Y = aT[I]
aD = aS ^ y	Y = y[I]
aS - y	y = aT x = X
aS & y	y = Y x = X
	Z : y x = X
	Z : y[I]
	Z : aT[I]

Where:

aD,aS,aT - is one of two DAU accumulators
X - pointer to ROM (*pt++, *pt++i)
Y - pointer to RAM (*rM, *rM++, *rM--, *rM++j)
Z - read/write compound addressing (*rMzp, *rMpz, *rMm2, *rMjk)

65169.91

© 1986 AT&T Technologies, Inc.

Notes

WE® DSP16 INSTRUCTIONS
Data Move

REG = IMMEDIATE
REG = Y
REG = aT
aT = REG
Y = REG
Z : REG

Where:

REG - DAU registers (x, y, auc, psw, c0-2)
XAAU registers (pt, pr, i)
YAAU registers (r0-r3, rb, re, j, k)
SIO registers (sioc, sdx, tdms, srta)
PIO registers (pioc, pdx0, pdx1)
aT - One of two accumulators
Y - RAM pointers (*rM, *rM++, *rM--, *rM++j)
Z - Compound addressing (*rMzp, *rMpz, *rMm2, *rMjk)

© 1986 AT&T Technologies, Inc.

65169.92

Notes

WE® DSP16 INSTRUCTIONS
Control

goto ADDRESS
goto pointer
call ADDRESS
call pointer
return
icall
ireturn

if COND goto ADDRESS
if COND goto pointer
if COND call ADDRESS
if COND call pointer
if COND return

Where:

COND - mi, pl, gt, le, lvs, mvs, mvc, c0ge, c0lt, c1ge, c1lt, heads, tails true, false

65169.93

© 1986 AT&T Technologies, Inc.

Notes

WE® DSP16 INSTRUCTIONS
Special Functions

```
aD=aS >> 1
aD=aS >> 4
aD=aS >> 8
aD=aS >> 16
aD=aS
aD=-aS
aD=rnd(aS)
aD=aS+1
aDh=aSh+1
aD=y
aD=p
aD=aS << 1
aD=aS << 4
aD= aS << 8
aD=aS << 16
```

The above instructions can be conditionally executed:

```
If COND aD = F(aS);
and with an event counter:
Ifc COND aD = F(aS)
which executes as:
If (COND) {c1++;aD=F(aS);c2=c1} else {c1++ }
```

65169.94

© 1986 AT&T Technologies, Inc.

Notes

WE® DSP16 INSTRUCTIONS
Cache Instructions

```
do K      {
          instr1
          instr2
            :
          instrN
}
redo K
```

Where:

N=1-15

K=1-127

65169.95

© 1986 AT&T Technologies, Inc.

Notes

ADDRESSING MODES

- Immediate addressing
- Indirect addressing
- Compound addressing
- Modulo addressing

65169.96

© 1986 AT&T Technologies, Inc.

Notes

ADDRESSING MODE
Immediate

- An operand is supplied by the instruction
- This mode applies to the data move group and control group

For Example:

REG=IMMEDIATE

x = 256

This instruction moves the immediate value 256 into the x register

65169.97

© 1986 AT&T Technologies, Inc.

Notes

ADDRESSING MODE
Indirect

- The instruction specifies the register that contains the address of the operand

For Example:

x = {*pt++, *pt++i}
y = {*rM, *rM++, *rM--, *rM++j}

Note: M=0-3

x = *pt++

In this instruction, the register pt holds the address of the operand to be moved into the x register.
The address is then post incremented by one.

65169.98

© 1986 AT&T Technologies, Inc.

Notes

ADDRESSING MODE
Compound

- This mode allows the contents of a RAM location and a register value to be exchanged

Z:R	Operation Step1	Step2	Step3
*rMzp:R	temp=R	R=*rM	*rM++=temp
*rMpz:R	temp=R	R=*rM++	*rM=temp
*rMm2:R	temp=R	R=*rM--	*rM++2=temp
*rMjk:R	temp=R	R=*rM++j	*rM++k=temp

65169.99

© 1986 AT&T Technologies, Inc.

Notes

INTERRUPTS

- Hardware Interrupt
  - When the interrupt request (INT) pin is asserted, the *WE*®DSP16 will branch to the interrupt service routine
  - The *WE*DSP16 will acknowledge the interrupt by asserting the interrupt acknowledge pin (IACK)
  - The IACK pin will be negated when the interrupt is complete (ireturn)
  - This interrupt can be masked by setting the pioc register (bit 5)

65169.100

© 1986 AT&T Technologies, Inc.

Notes

INTERRUPTS

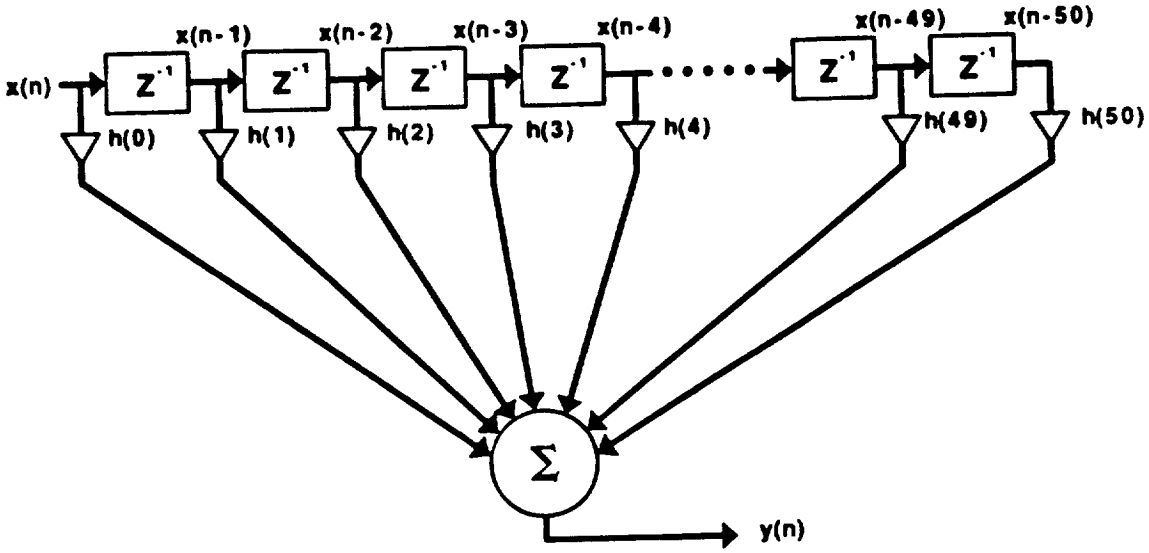
- Internal Hardware Interrupts
  - Interrupt when an external device writes to the parallel port (passive mode)
  - Interrupt when an external device reads from the parallel port (passive mode)
  - Interrupt when an external device reads from the serial port (passive mode)
  - Interrupt when an external device writes to the serial port (passive mode)
- These interrupts can be masked by setting pioc register (bits 6-9)

65169.101

© 1986 AT&T Technologies, Inc.

Notes

51 TAP FIR FILTER



75922.63

© 1987 AT&T Technologies, Inc.

Notes

WE® DSP16 FIR FILTER DESIGN EXAMPLE

```
.ram
X50:      49*int
X1:       int
Ibuf:     int
.endram

fir50:    auc=0x02
                pt=H50
                i=-50
                rb=X50
                re=X1
                r1=X50
                r0=ibuf

loop:           *r0=sdx
                a0=p                          y=*r1++          x=*pt++
          do 49 {
                a0=a0-p        p=x*y          y=*r1++          x=*pt++
                }
                a0=a0-p        p=x*y          y=*r0            x=*pt++i
                a0=a0-p        p=x*y          *r1++=y
                a0=a0-p
                sdx=a0
endl:     goto  loop

/*              Coefficients from h(n-50) to h(n)                    */

H50:      int   0.000545
          int   0.000000
          int   -0.000641
                .
                .
                .
                .
```

65169.103

© 1986 AT&T Technologies, Inc.

Notes

DSP16 RAM MEMORY MAP FOR 51 TAP FIR FILTER

Label	RAM location	Contents (16 bit values)
X50	00	Tap value
	01	Tap value
	02	Tap value
	03	Tap value
	•	
	•	
	•	
	48	Tap value
X1	49	Tap value
ibuf	50	New sample

© 1987 AT&T Technologies, Inc.

75922.65

Notes

DSP16 BENCHMARKS

	@55 ns	@75 ns	Units
FIR Filter	55	75	ns/tap
64 tap FIR Filter (with I/O)	233	171	KHZ (Sample Rate)
5-Multiply IIR Filter (SOS)	385	525	ns/Section
10th Order IIR (with I/O)	343	251	KHZ
Adaptive Filter (32 bit coefficients)	385	525	ns/tap
64 tap Complex FIR (with I/O)	52.7	38.6	KHZ

75922.66

© 1987 AT&T Technologies, Inc.

Notes

The *WE*® DSP16A Digital Signal Processor

75922.81

© 1988 AT&T Technologies, Inc.

Notes

DSP16A

- Source and object code compatible with DSP16
- Two speed versions
  - 25ns instruction cycle
  - 35ns instruction cycle
- 2K word on-chip RAM
- 4K word on-chip ROM
- .75μ double metal CMOS
- Samples July 1988

75922.82

© 1988 AT&T Technologies, Inc.

Notes

DSP16A BENCHMARKS

	@ 25 ns	@ 35 ns	Units
FIR filter	25	55	ns/tap
64 Tap FIR filter (with I/O)	512	366	kHz (Sample rate)
10th Order IIR filter (with I/O)	755	539	kHz
64 Tap complex FIR (with I/O)	115.9	82.8	kHz

75922.83

© 1988 AT&T Technologies, Inc.

Notes

DEVELOPMENT TOOLS FOR THE WE® DSP16

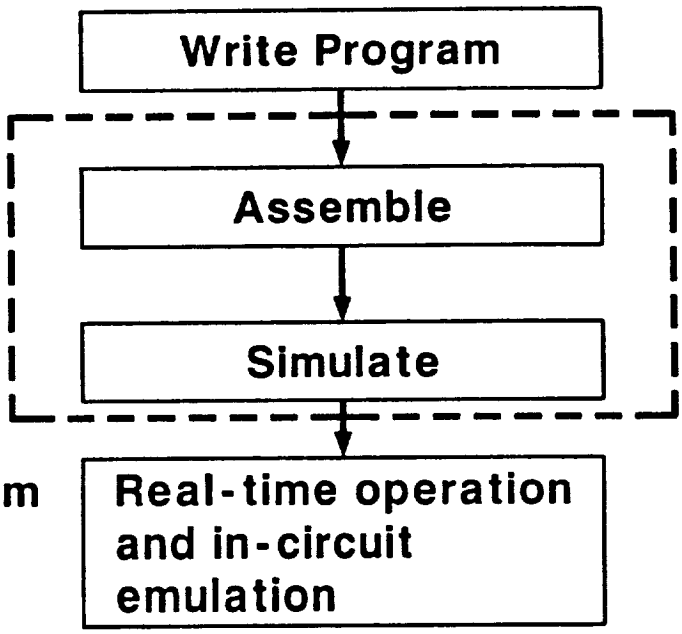
65169.106

© 1986 AT&T Technologies, Inc.

Notes

APPLICATIONS DEVELOPMENT SUPPORT

- Text editor
- Assembler
- Simulator
- Development system operating under simulator control

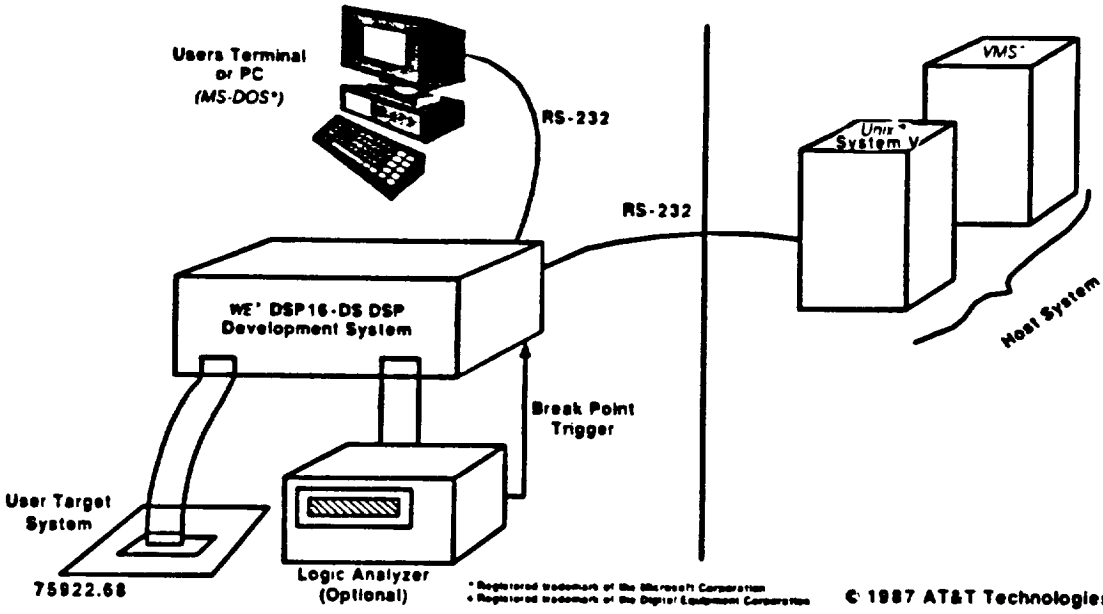


65169.107

© 1986 AT&T Technologies, Inc.

Notes

APPLICATION DEVELOPMENT ENVIRONMENT



* Registered trademark of the Microsoft Corporation
* Registered trademark of the Digital Equipment Corporation

© 1987 AT&T Technologies, Inc.

Notes

WE® DSP16-SL SUPPORT SOFTWARE LIBRARY

- *WE* DSP16 Assembler
- *WE* DSP16 Simulator
- *UNIX*,® *MS-DOS*,* or VMS+ Operating Systems

* Registered trademark of the Microsoft Corporation
+ Registered trademark of the Digital Equipment Corporation

© 1987 AT&T Technologies, Inc.

75922.69

Notes

WE® DSP16 ASSEMBLER

- Define labels
- Define tables
- Interactive assembly

65169.110

© 1986 AT&T Technologies, Inc.

Notes

WE® DSP16 SIMULATOR

- Set breakpoints
- Display registers and memory
- I/O files
- Comand files
- Change values in registers and memory
- User defined functions
- Single step
- Disassemble instructions

65169.111

© 1986 AT&T Technologies, Inc.

Notes

DSP16-DS DEVELOPMENT SYSTEM
Features

- Real-time development of application program
- Simulator provides user interface to development system
- In-circuit I/O emulation
- Hardware and software breakpoints
- Examine and change the contents of memory and registers
- 16K words of external program memory provided (expandable to 64K in 16K blocks)
- 16K words of external data memory provided (expandable to 64K in 16K blocks)
- Simulator accelerator
- Logic analyzer interface
- Single step function
- Multiple DSP16's supported (up to 16)

75922.72

© 1987 AT&T Technologies, Inc.

Notes

WE® DSP16-DS DEVELOPMENT SYSTEM
I/O Interfacing

- Analog I/O interface
  - mu-law codec
  - Linear codec
- Serial I/O
  - In-circuit emulation (target system)
  - Codecs
  - I/O files (simulator accelerator)
- Parallel I/O
  - In-circuit emulation (target system)
  - RAM on development system
- Serial RS-232 Interface
- High speed parallel interface, 100k bytes/sec (with a PC host system)

© 1986 AT&T Technologies, Inc.

65169.113

Notes

WE® DSP16-DS DEVELOPMENT SYSTEM

Simulator Accelerator

- Allows for algorithm development with hardware
- Real-time I/O from host system files
  - Input from files on host system
  - Capture and store real-time data
  - Output to files on host system
  - Analog output with stored input files

65169.114

© 1986 AT&T Technologies, Inc.

Notes

REVIEW

- Support
- Future directions

© 1987 AT&T Technologies, Inc.

75922.75

Notes

SUPPORT

- Documentation
- Application notes
- Application library
- Training
- Technical support
  - Regional technical consultants
  - Product application engineers
  - Bell Labs designers

65169.118

© 1986 AT&T Technologies, Inc.

Notes

SUPPORT

- 3rd party
  - ASPI
  - Burr Brown
  - CAC

75922.76

© 1987 AT&T Technologies, Inc.

Notes

FUTURE DIRECTIONS

- MIL-STD-883
  - DSP16 1988
  - DSP32 1989

75922.77

© 1987 AT&T Technologies, Inc.

Notes

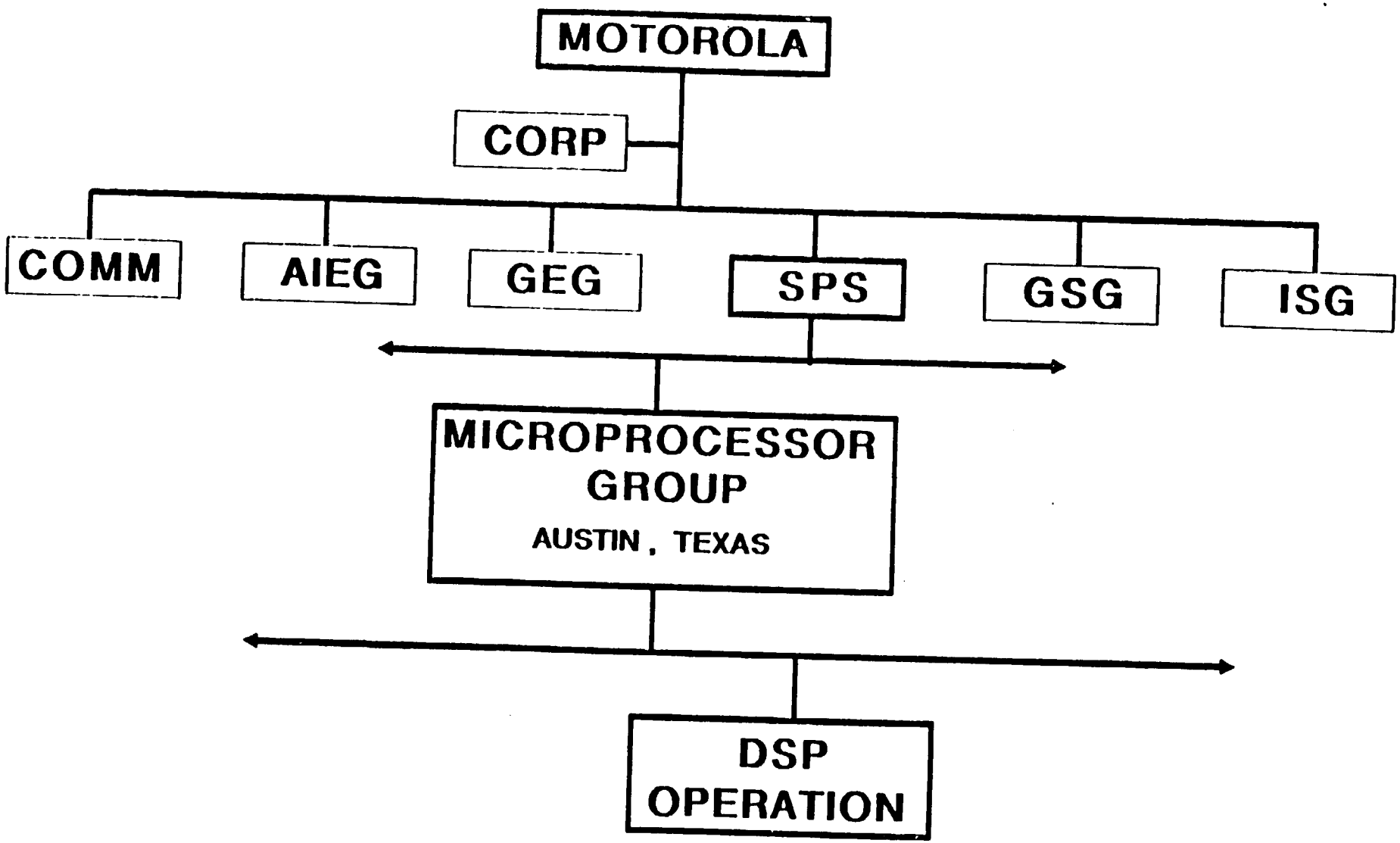
AT&T

The right choice.

65169.121

© 1986 AT&T Technologies, Inc.

Notes


MOTOROLA
CORP
COMM
AIEG
GEG
SPS
GSG
ISG
MICROPROCESSOR GROUP
AUSTIN, TEXAS
DSP OPERATION

DSP56000/1 INTRODUCTION

* PRODUCT OVERVIEW

* 4 th GENERATION Digital Signal Processor

* GENERAL PURPOSE [user programmable]
  - DSP56000 = ROM based version
  - DSP56001 = RAM based version

* SPEED:
  - 97.5 nsec Instruction Cycle Time (10.25 MIPS)

* PRECISION:
  - 24x24 + 56 - - > Two 56 - Bit Accumulators

* INTEGRATION:
  - Core Processor has Three Parallel Execution Units
  - Five(000) / Six(001) On - Chip Memories
  - Three MCU - Style Peripherals On - Chip
  - Full Speed Memory Expansion

* TECHNOLOGY:
  - 1.5 u, 2 Layer Metal, HCMOS III Process
  - 285K Transistors on a 445x 432 mil. Die

* POWER DISSIPATION:
  - 500 mW Typical @ 20.5 MHz.

* PACKAGING:
  - 88 Pin Pin Grid Array & Surface Mount

* SAMPLES:
  - Now

DSP56000/1 INSTRUCTION SET

* PARALLEL DATA MOVES

* SYNTAX:

Opcode	Operands	X Bus Data	Y Bus Data
MAC	X0,Y0,A	X:(R0)+,X0	Y:(R4)+,Y0

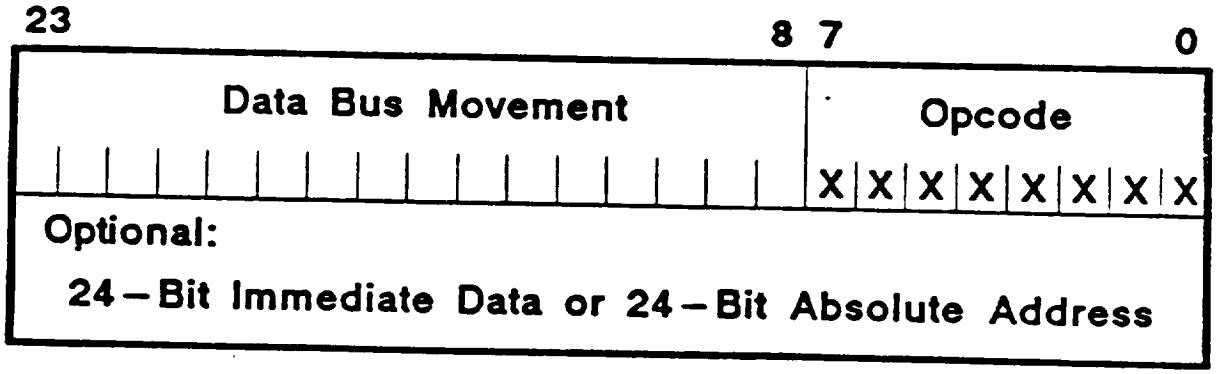
MAC → Specifies:

* Data ALU Operation
* Logical Operation

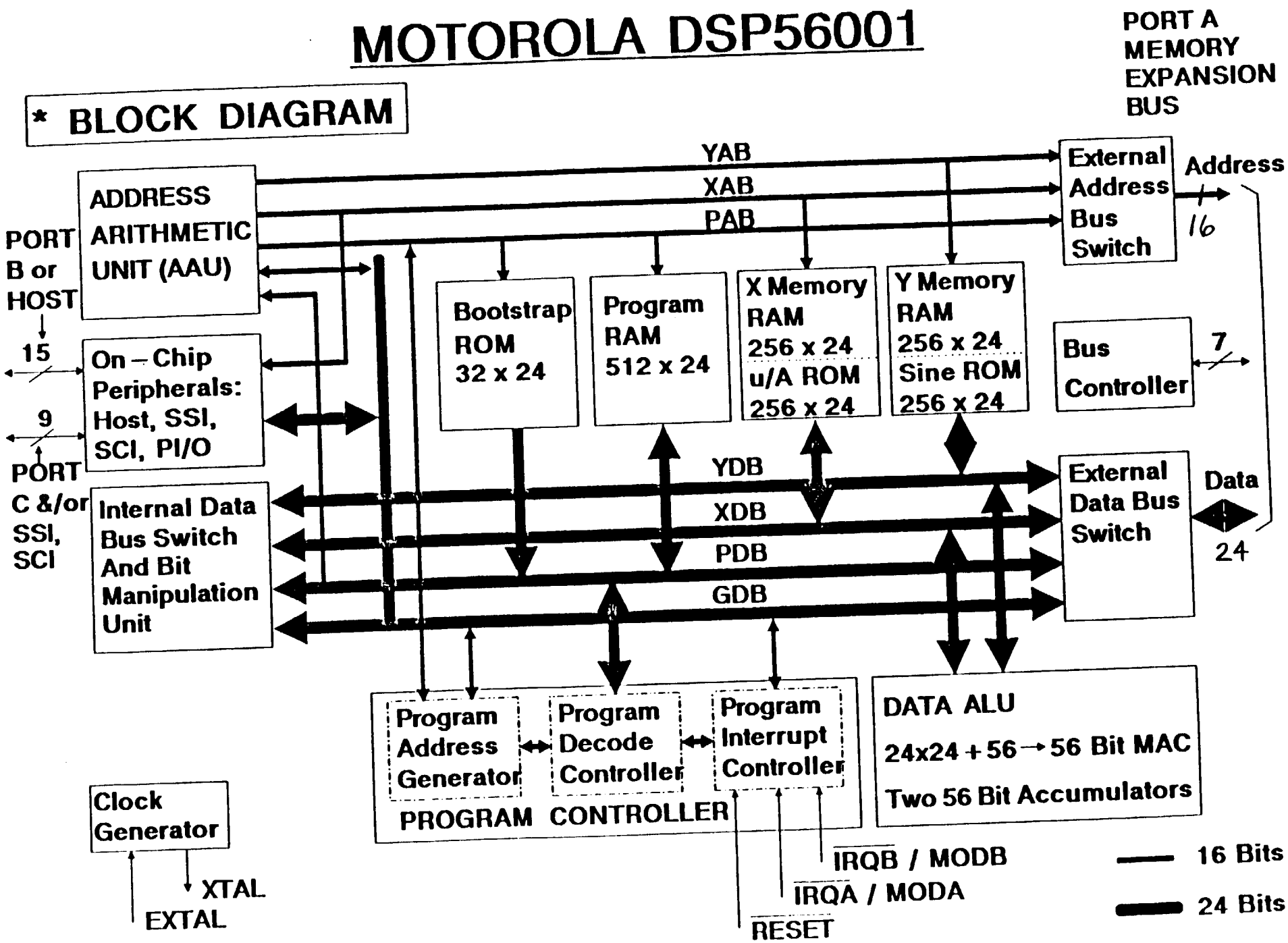
X:(R0)+,X0 / Y:(R4)+,Y0 → Specifies:

* Optional Data Transfers
* Addressing Modes
* Address Space Qualifiers [X:,Y:,L:]

* OBJECT CODE FORMAT:

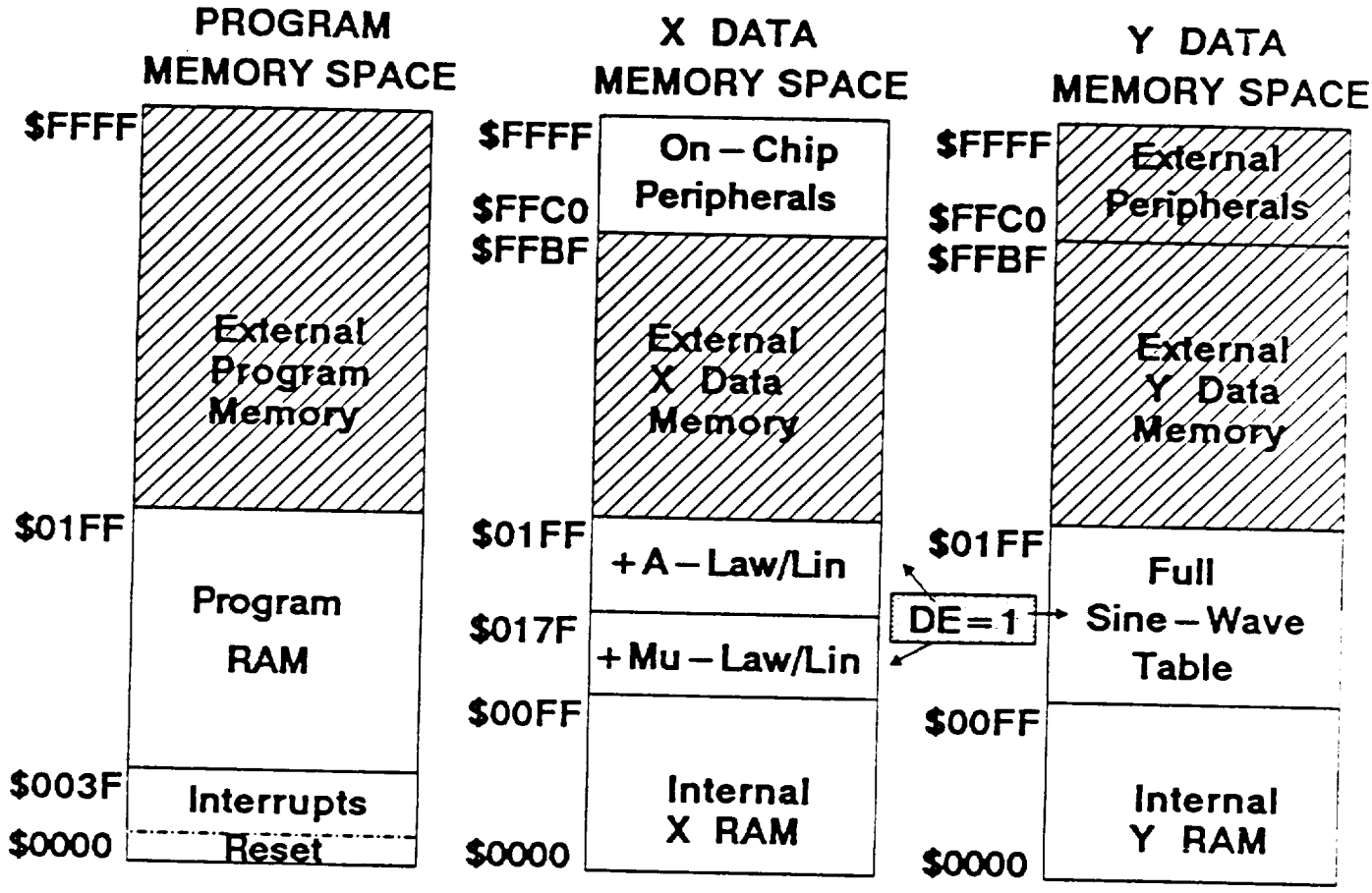


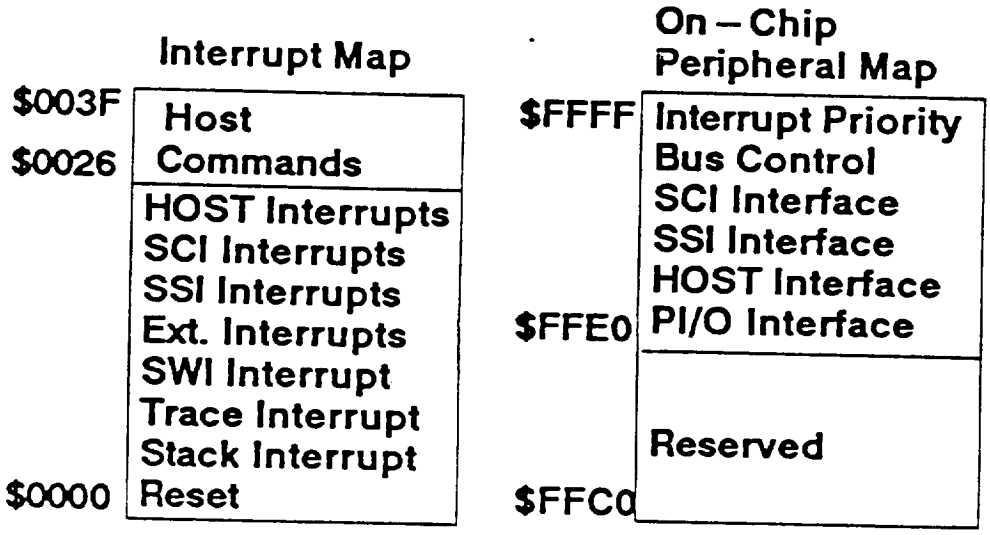
MOTOROLA DSP56001



DSP56001 MEMORY MAP

* MODE 0:NORMAL EXPANDED & INTERNAL RESET





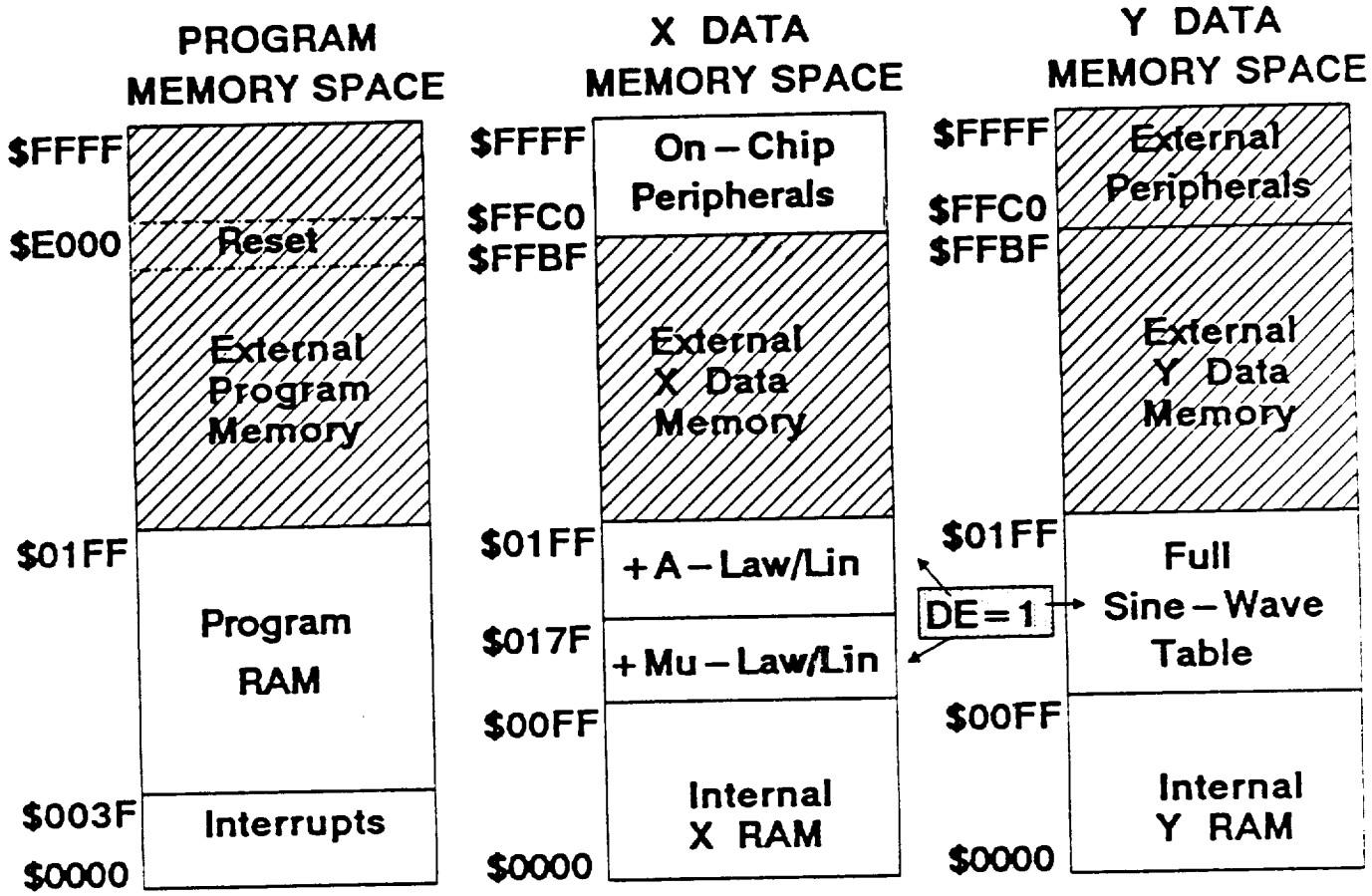
Note:

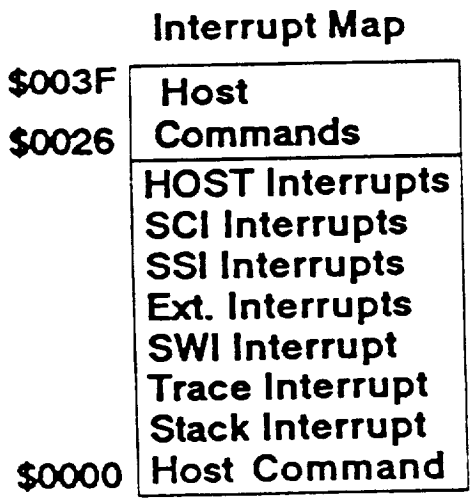
Addresses $FFC0 through $FFFF in X Data Memory are NOT available externally.

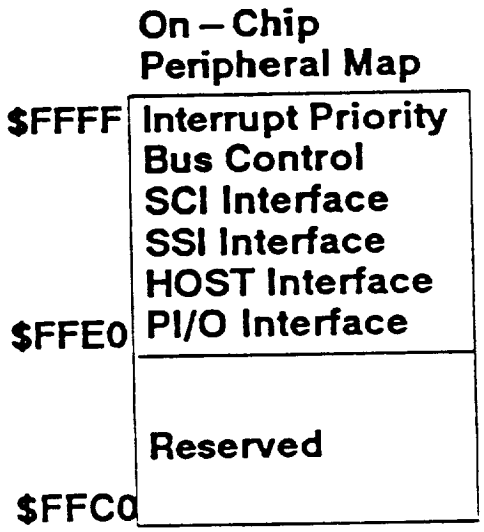
DE = Data ROM Enable (Operating Mode Register Bit 2)

DSP56001 MEMORY MAP

* MODE 2:NORMAL EXPANDED & EXTERNAL RESET







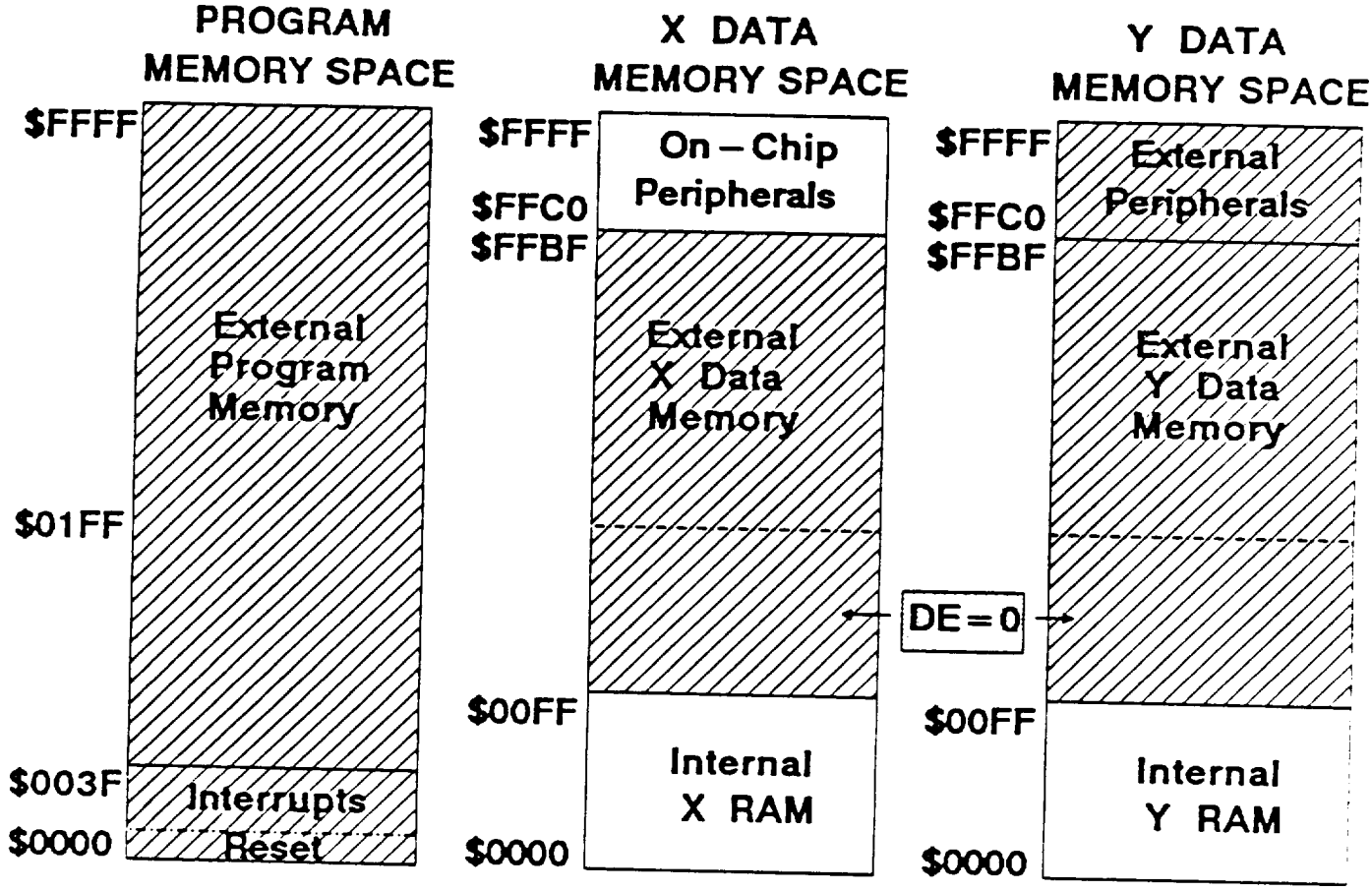
Note:

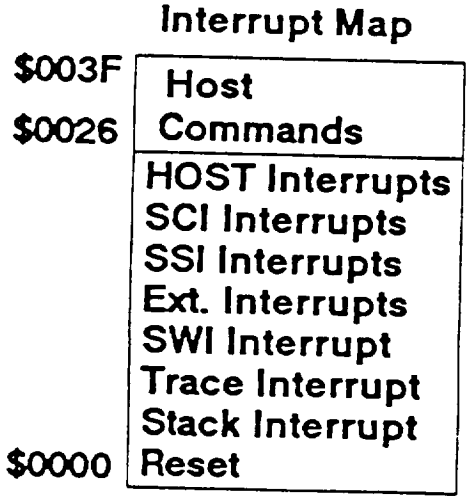
Addresses $FFC0 through $FFFF in X Data Memory are NOT available externally.

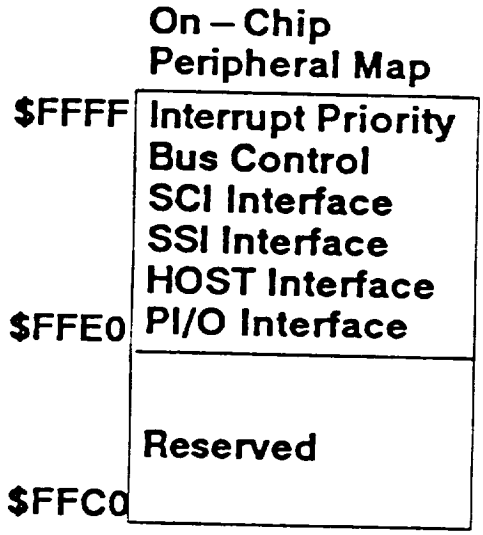
DE = Data ROM Enable (Operating Mode Register Bit 2)

DSP56001 MEMORY MAP

* MODE 3: EXTERNAL PROGRAM MEMORY







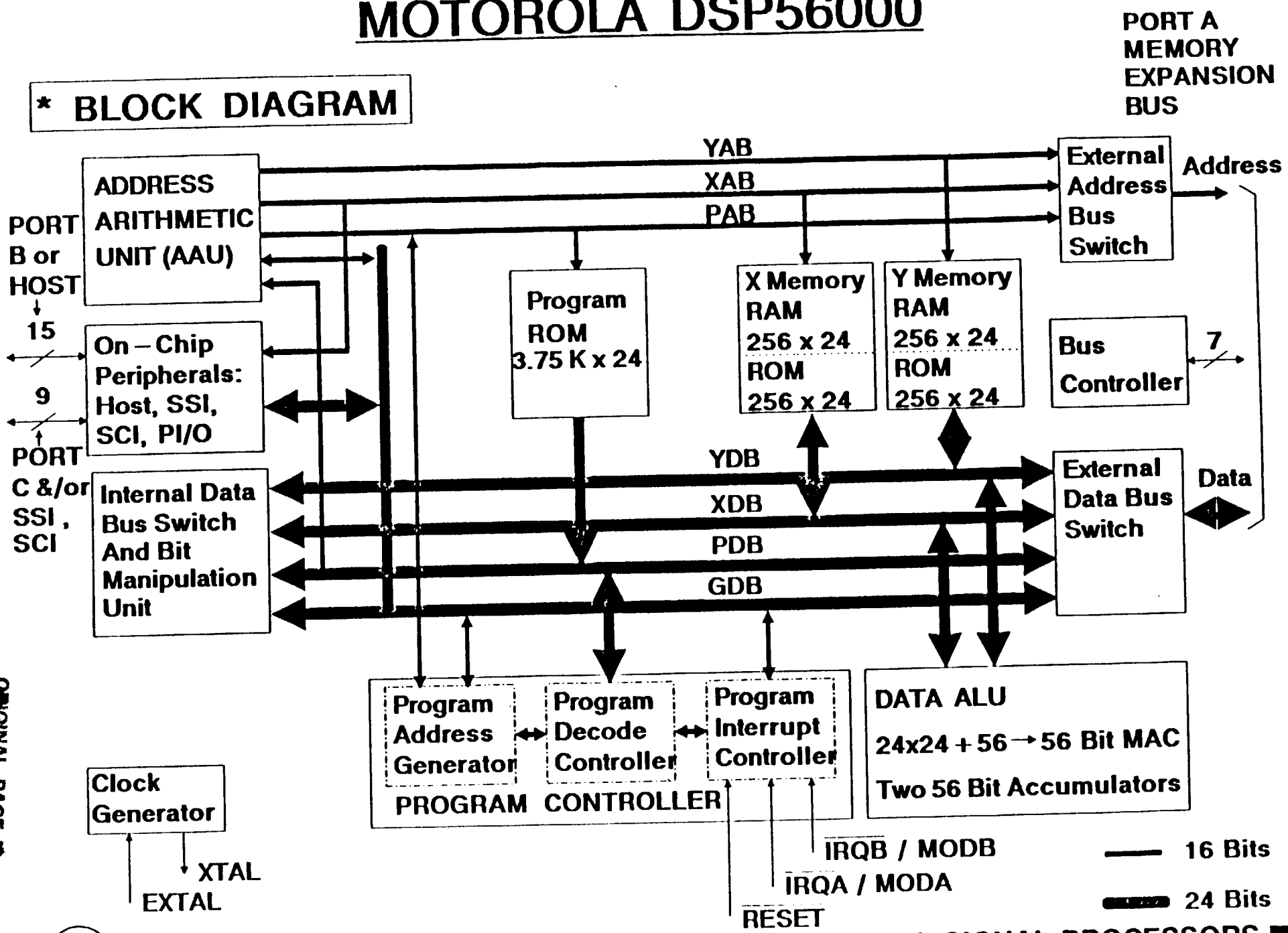
Note:

Addresses $FFC0 through $FFFF in X Data Memory are NOT available externally.

DE = Data ROM Enable (Operating Mode Register Bit 2).

MOTOROLA DSP56000

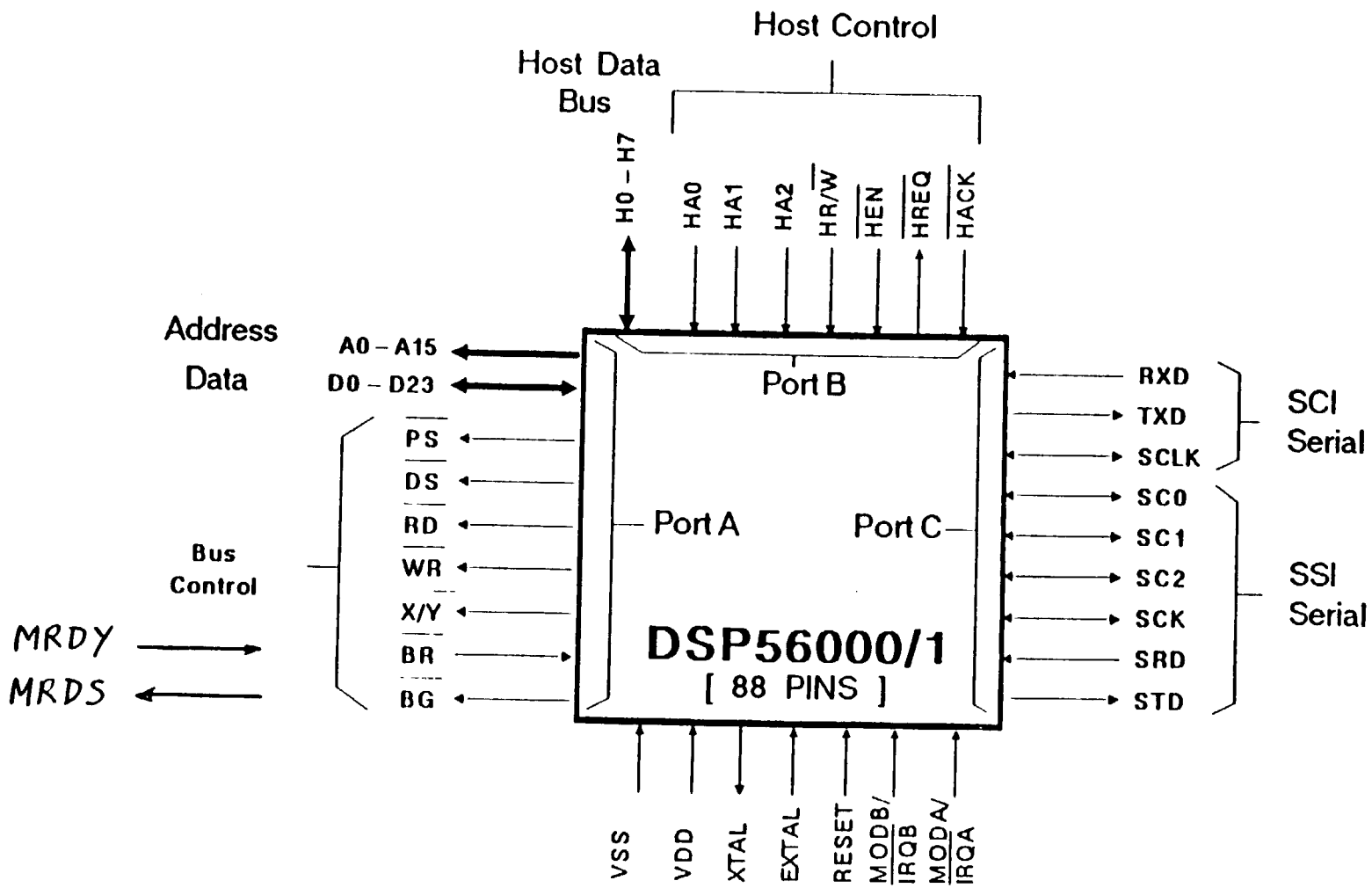
* BLOCK DIAGRAM



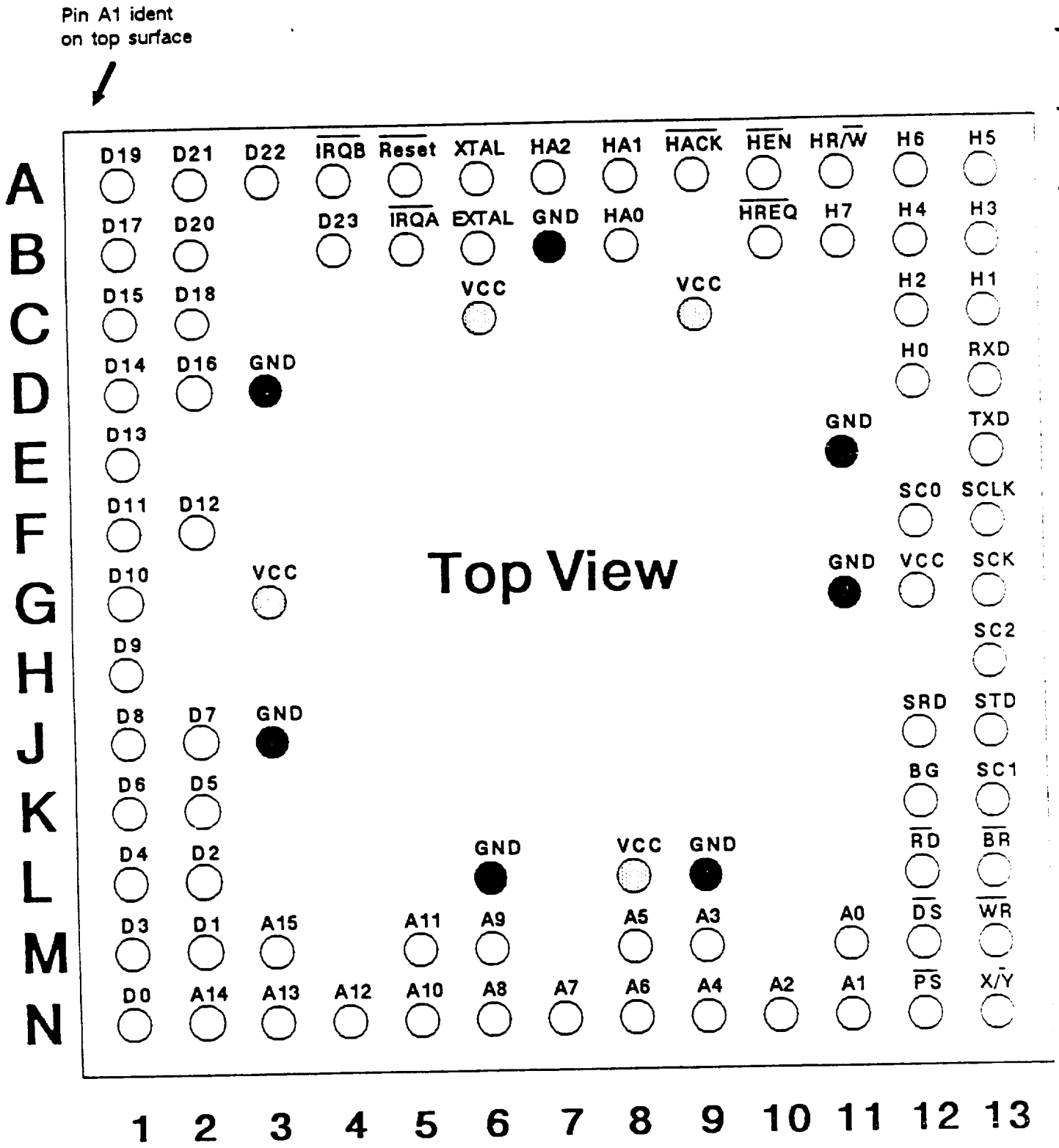
ORIGINAL PAGE IS OF POOR QUALITY

MOTOROLA DSP56000/1

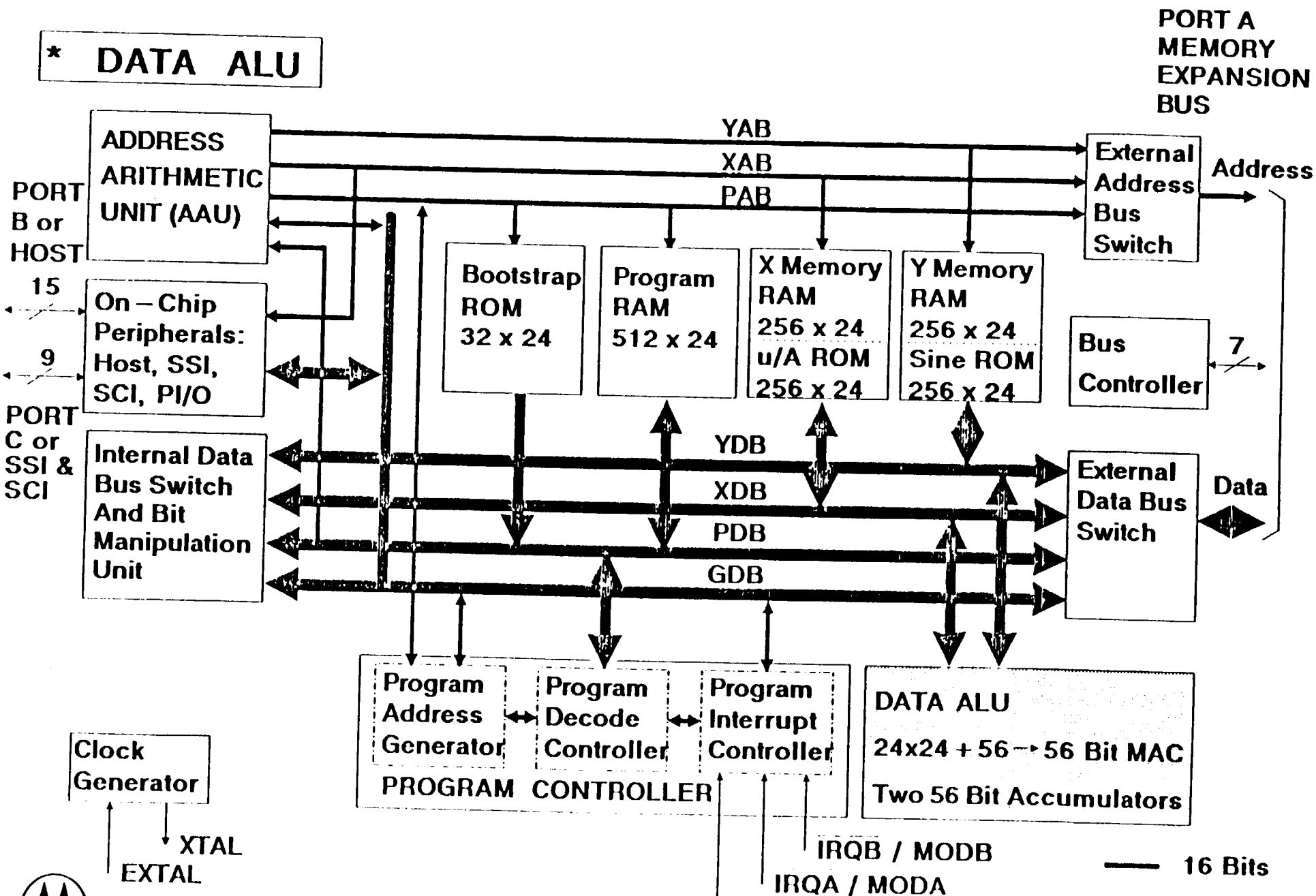
* FUNCTIONAL SIGNAL GROUPS



DSP56000/1 PIN GRID ARRAY



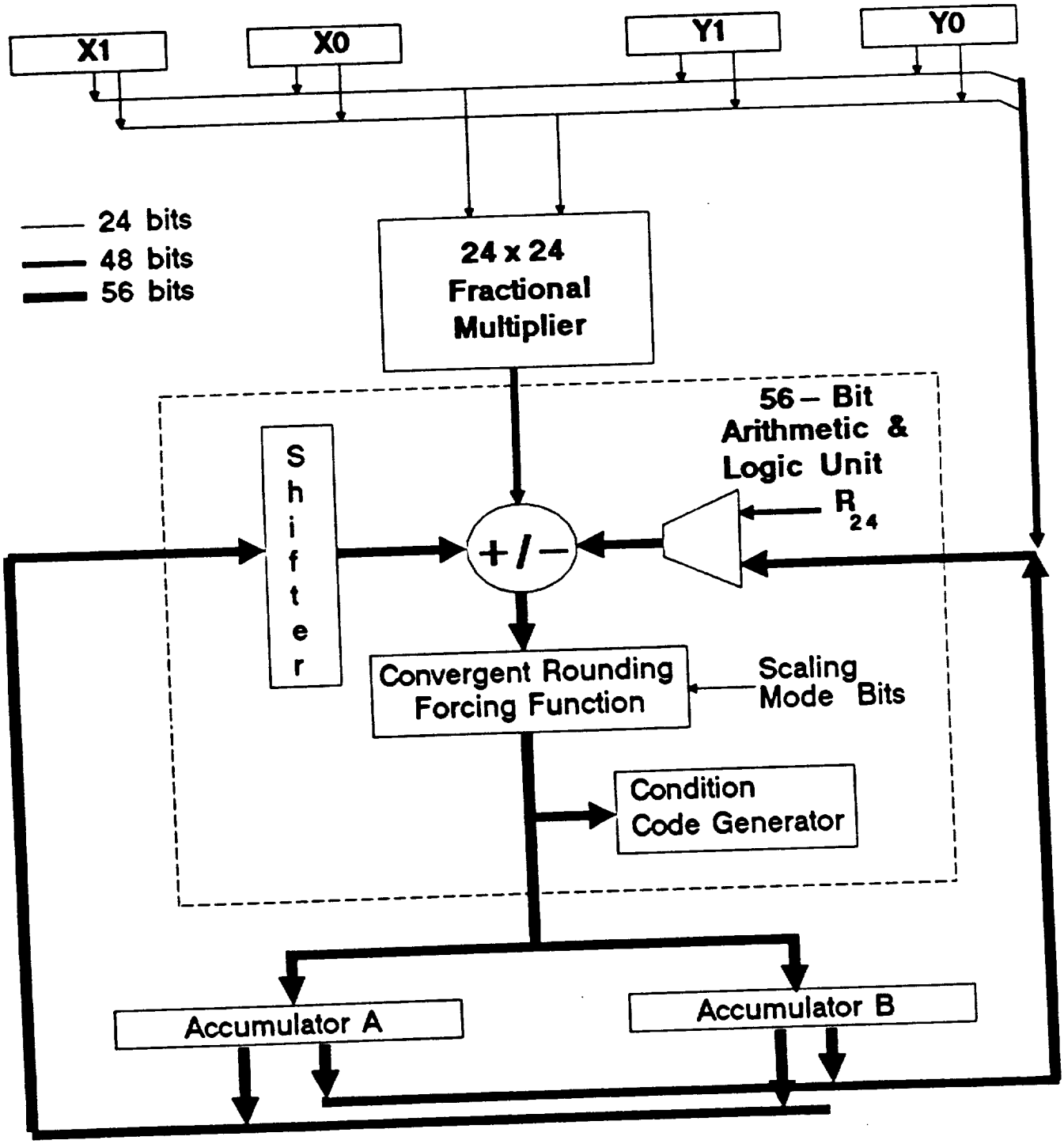
MOTOROLA DSP56001



DSP56000/1 DATA ALU

*** MULTIPLY – ARITHMETIC – LOGIC UNIT**

24 x 24 +/– 56 + R_{24} → 56; NOT Pipelined



May 13, 1987

Motorola, Inc.
300 Unicorn Park
Woburn, MA 01801
ph. (617) 932-9700

Dear

Thanks for keeping me involved in the conversations with the XXX engineers regarding using the DSP56001 in their speech application. Based on information obtained from the last several phone conversations, I can objectively say that the DSP56001 is a better choice than the Analog Devices ADSP-2100 in their application. Enclosed is the information requested on the FFT performance which the DSP56001 can achieve. Please pass this information to the engineers.

In sending this information, an important question came to mind regarding the XXX simulations comparing 24 bit arithmetic (DSP56001) to 16 bit arithmetic (ADSP-2100). Is using the Motorola software simulator package (SIM56000) or are they writing their own simulation in C or Fortran to evaluate the DSP56001? I strongly recommend that they use Motorola's SIM56000 simulator software since the large 48 bit product with 56 bit accumulation is accurately simulated by this package. It is important to ensure that the DSP56001 arithmetic is being accurately simulated for a fair comparison.

A simple test program was written to indicate the signal to quantization noise ratio (SQNR) performance of the DSP56001 for a typical FFT application. A block diagram of this test program is shown in Figure 1, where DSP56001 code implements the spectral analysis algorithm above the dashed line. A sinewave generator macro (SINEGEN) was used to generate a 16 bit complex test signal consisting of a real 16 bit sinewave with a frequency of 0.16666 fs and an imaginary 16 bit sinewave with a frequency of 0.235 fs, where fs is the sampling frequency. This test signal will allow us to see any <u>intermodulation</u> effects caused by <u>arithmetic errors</u>. The 16 bit sinewaves are placed in the 16 least significant bits of the 24 bit data word to simulate a <u>16 bit A/D</u> input to the FFT system. A Blackman-Harris 4 term minimum sidelobe window generator macro (W_BH4M) was used to window the data before the FFT. This window has one of the best sidelobe attenuations (-92 dB) available and its spectrum will hopefully remain below the noise spectrum. Otherwise we would only see the sidelobe spectrum of the window and not the spectrum of the

noise. After windowing, the FFT input is still in the 16 least significant bits of the 24 bit data word. A 1024 point complex FFT macro (FFT_R2E) is performed with the DSP56001's native 24 bit fixed point arithmetic using no data scaling. The 24 bit twiddle factors are provided by a lookup table generator macro (SINCOS). After the FFT, the 24 bit magnitude of the spectrum is calculated by the DSP56001 using a magnitude squared macro (MAGSQR) and a square root macro (SQRT3). Finally, the bit reversed spectral magnitude is read out in normal order to a disk file for plotting. Below the dashed line in Figure 1, the VAX plotting program takes the logarithm of the disk file data and plots the results as the frequency spectrum of the test input.

Observation of the plotted spectrum shows several advantages of the DSP56001. First, over 100 dB SQNR has been achieved using 24 bit data storage and 56 bit arithmetic. The noise level seen is essentially the quantization noise of the simulated 16 bit A/D input. Typical SQNR for 16 bit machines is about 55 dB SQNR for fixed point with scaling by 1/2 after each stage and about 90 dB for full block floating point. Note that 16 bit machines will need some form of scaling for overflow control with CODEC inputs and 200 point transforms. The DSP56001 will not limit SQNR performance even if 16 bit A/D converters are used. The sidelobes around the signal peaks are the sidelobes of the window rising above the input signal quantization noise. Second, no intermodulation effects are seen. This is due to the large 56 bit accumulators and use of rounding. Third, no overflow problems occur for large transform sizes (1024 points) with 16 bit input data because the 24 bit word size allows for 8 bits of word growth. Finally, no scaling, block floating point, etc. needs to be performed to achieve this level of performance. System performance, execution speed, program size and algorithm development time are all improved over 16 bit machines.

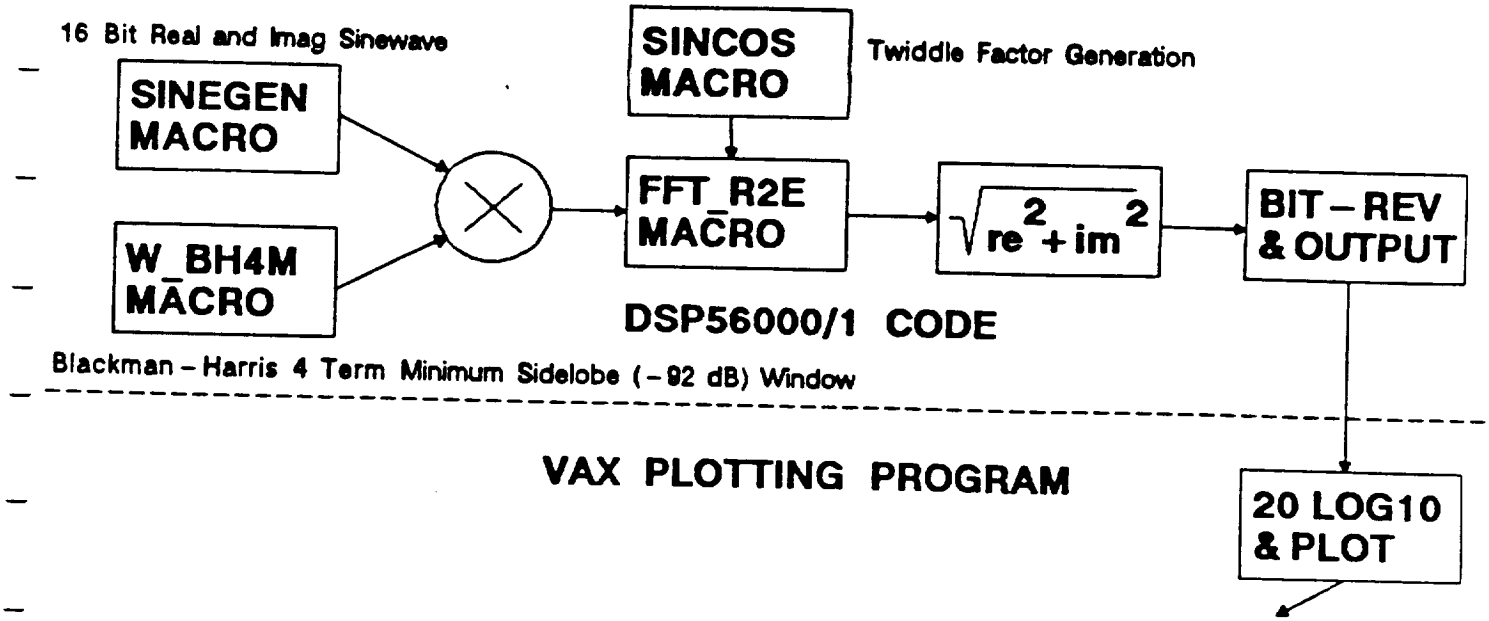
Regards,

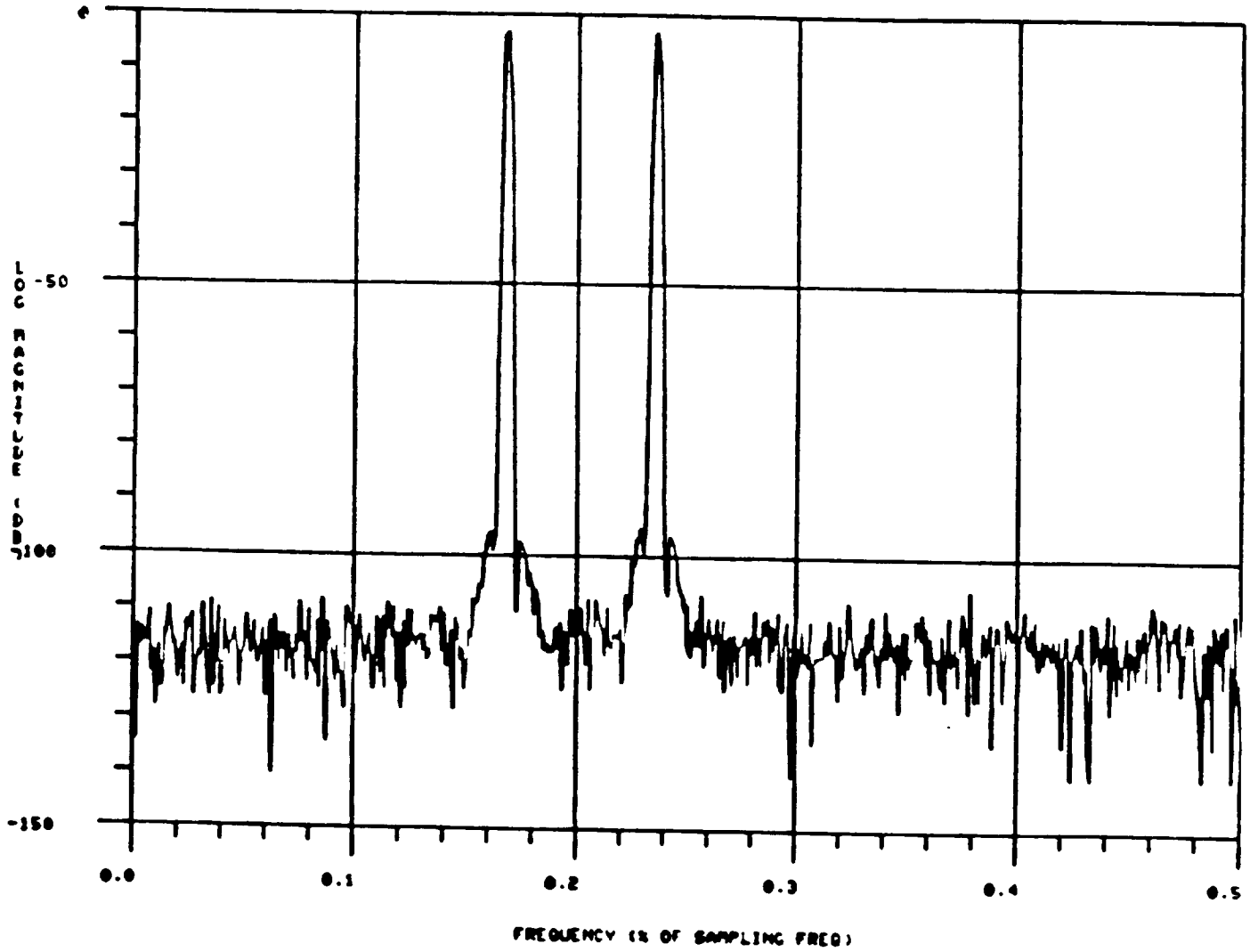
Kevin L. Kloker

Kevin L. Kloker
Principal Staff Engineer
Corporate Research Labs

cc: Jane Bates

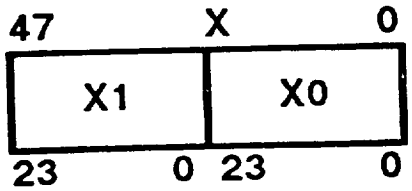
DSP56000/1 1024 PT FFT TEST

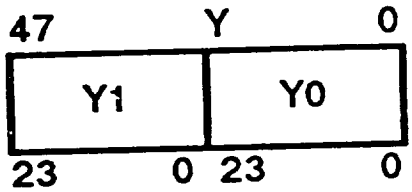




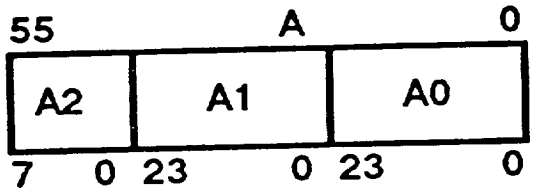
DSP56000/1 DATA ALU

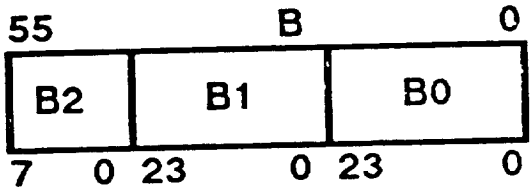
* PROGRAMMING MODEL





Data ALU Input Registers





Data ALU Accumulators

DSP56000/1 DATA ALU

* DATA REPRESENTATION = FRACTIONAL

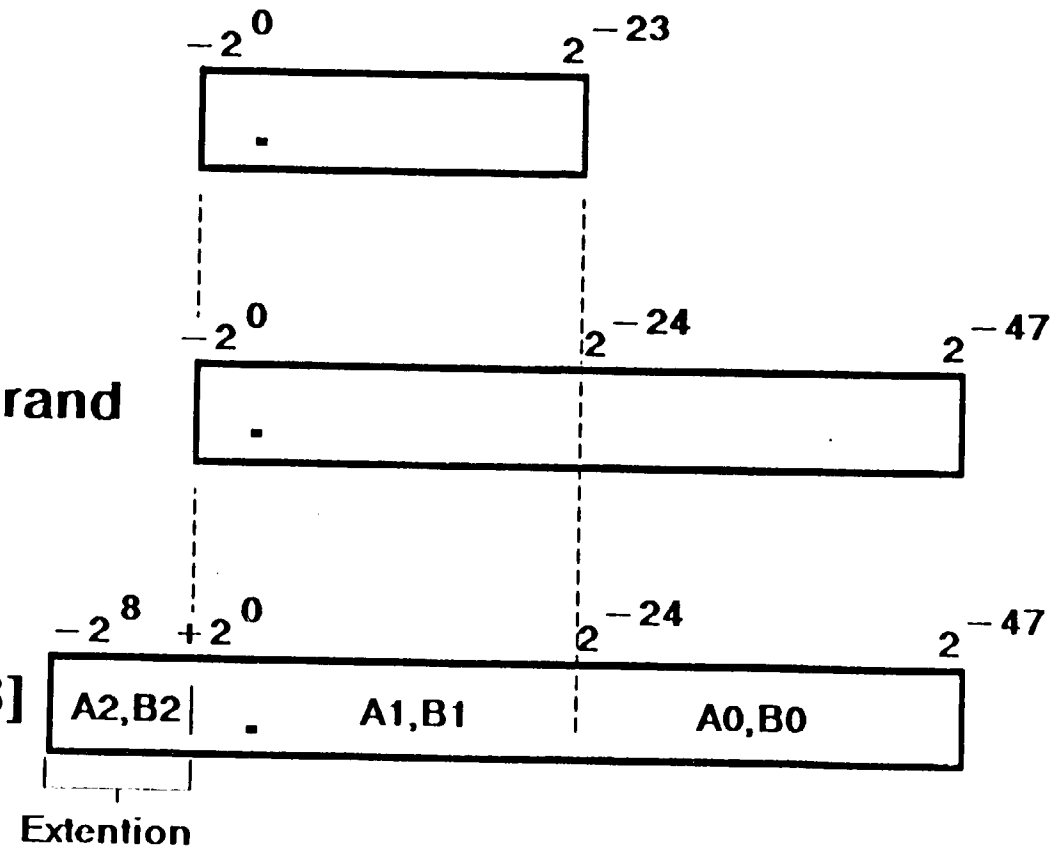
Word Operand

X1,X0
Y1,Y0
A1,A0
B1,B0

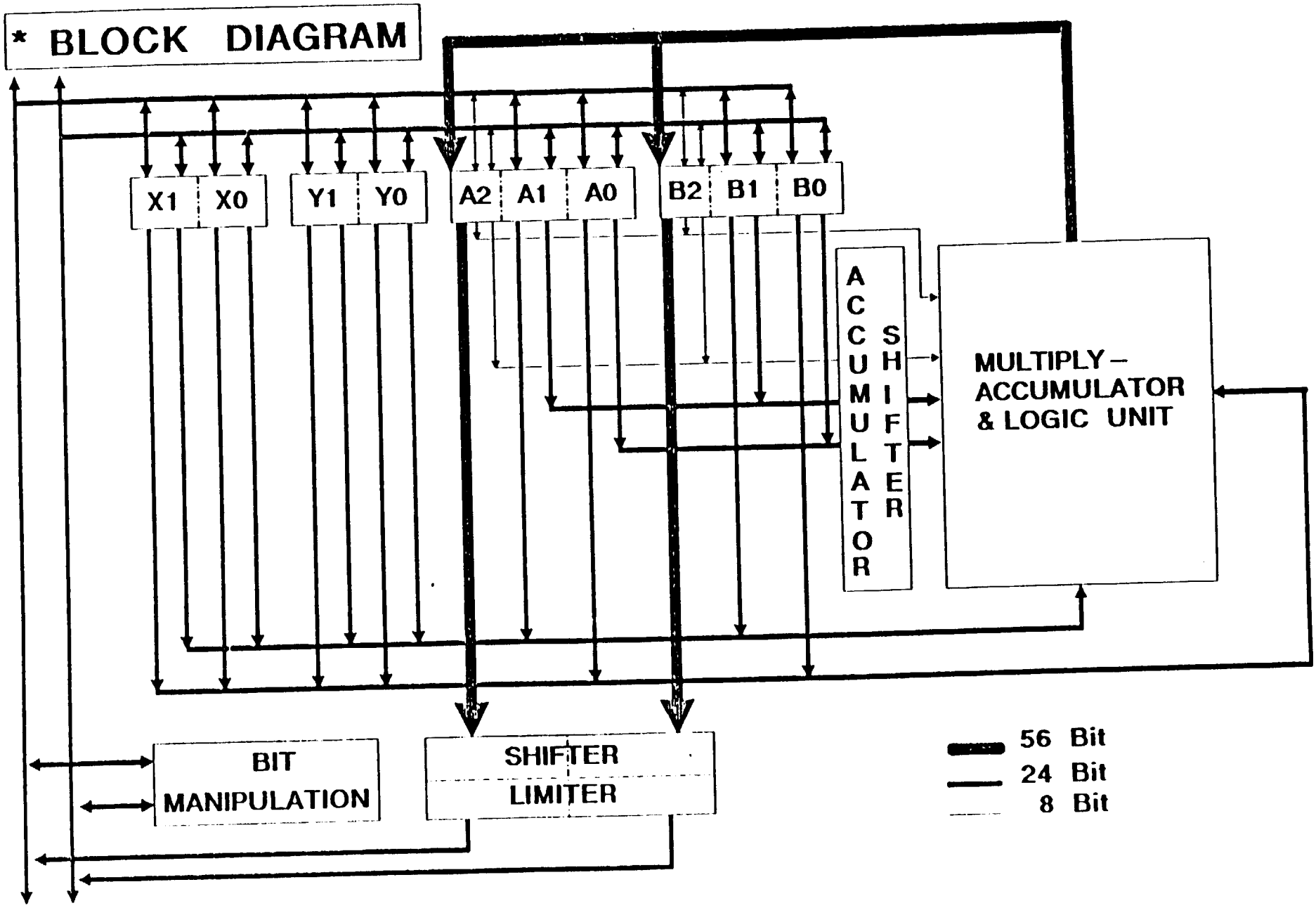
Long Word Operand

X1:X0 = X
Y1:Y0 = Y
A1:A0 = A10
B1:B0 = B10

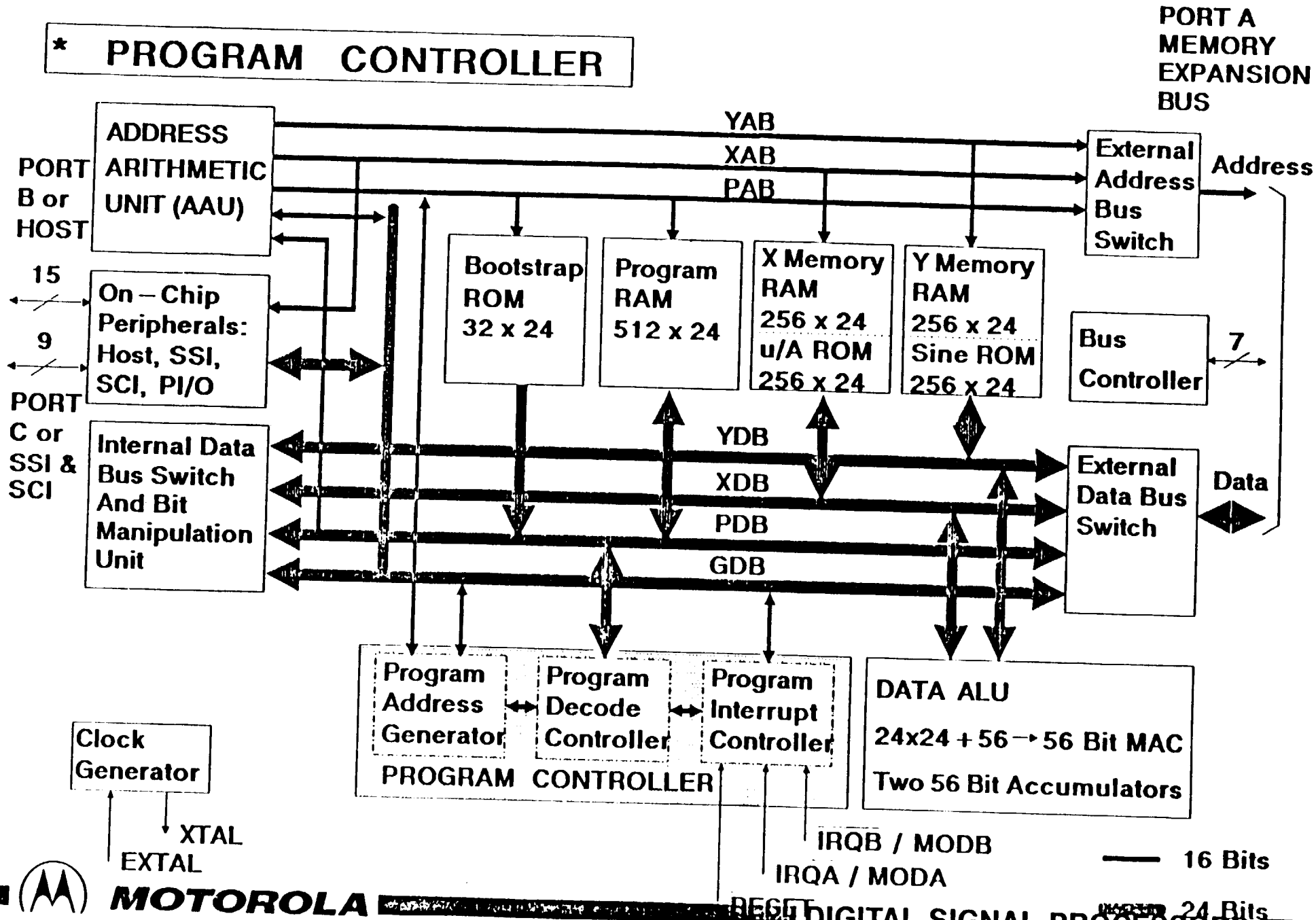
Accumulator [A or B]



DSP56000/1 DATA ALU



MOTOROLA DSP56001



DSP56000/1 INTERRUPT STRUCTURE

* INTERRUPT SOURCES

	VECTOR ADDRESS	INTERRUPT SOURCE
P:	$0000	Hardware $\overline{\text{RESET}}$
	$0002	Stack Error
	$0004	Trace
	$0006	$\overline{\text{SWI}}$
	$0008	$\overline{\text{IRQA}}$
	$000A	$\overline{\text{IRQB}}$
	$000C	SSI Receive Data
	$000E	SSI Receive Data with Exception Status
	$0010	SSI Transmit Data
	$0012	SSI Transmit Data with Exception Status
*	$0014	SCI Receive Data
*	$0016	SCI Receive Data with Exception Status
	$0018	SCI Transmit Data
	$001A	SCI Idle Line
	$001C	SCI Timer
	$001E	Reserved for H/W Development
	$0020	Host Receive Data
	$0022	Host Transmit Data
	$0024	Host Command (default)
	$0026	Available for Host Command (AHC)
	$0028	AHC
	$002A	AHC
	⋮	⋮
	$003E	AHC

* INTERRUPT LOGIC LEVELS

- Internal interrupts are LEVEL sensitive
- $\overline{\text{IRQA}}$ and $\overline{\text{IRQB}}$ are programmable :

  LEVEL sensitive or negative EDGE triggered

DSP56000/1 INTERRUPT STRUCTURE

* 4 PRIORITY LEVELS

* LEVEL 3 (NON–MASKABLE)

- Highest Level
- Priority within Level 3 is Fixed as:

highest	Hardware $\overline{\text{RESET}}$
¦	Hardware Development
¦	Stack Error
¦	Trace
¦	SWI

* LEVELS 0, 1, 2 (MASKABLE)

- Priority Level is Programmable for $\overline{\text{IRQA}}$, $\overline{\text{IRQB}}$, HOST, SSI and SCI.
- Prioritized within each assigned level as:

highest	$\overline{\text{IRQA}}$ (external interrupt)
¦	$\overline{\text{IRQB}}$ (external interrupt)
¦	HOST Command Interrupt
¦	HOST Receive Data Interrupt
¦	HOST Transmit Data Interrupt
¦	SSI Rx Data with Exception Interrupt
¦	SSI Rx Data Interrupt
¦	SSI Tx Data with Exception Interrupt
¦	SSI Tx Data Interrupt
¦	SCI Rx Data with Exception Interrupt
¦	SCI Rx Data Interrupt
¦	SCI Tx Data Interrupt
¦	SCI Idle Line Interrupt
¦	SCI Timer Interrupt

DSP56000/1 INTERRUPT STRUCTURE

* FEATURES OF LONG AND FAST INTERRUPTS

1] LONG INTERRUPTS:

- A JSR to the starting address of the interrupt service routine is located at one of the two interrupt vector addresses.
- PC, SR are pushed onto the stack.
- The interrupt service routine can be any length and is terminated by a RTI.
- The interrupt service routine can be interrupted i.e., NESTED INTERRUPTS are supported.

2] FAST INTERRUPTS:

- A JSR to the starting address of the interrupt service routine is NOT located at one of the two interrupt vector addresses.
- Processor status is NOT saved.
- Interrupt routine is TWO instruction words long (one 2 – word or two 1 – word instructions) and auto – returns without a RTI.
- A fast interrupt is NOT INTERRUPTIBLE.
- PRIMARY APPLICATION is to move data between memory and I/O devices.

DSP56000/1 INTERRUPT STRUCTURE

NORMAL INSTRUCTION TIMING

INSTRUCTION CYCLE	1	2	3	4	5	6	7	8	9	10
FETCH	n3	n4	n5	n6	n7	n8	n9	n10	n11	n12
DECODE	n2	n3	n4	n5	n6	n7	n8	n9	n10	n11
EXECUTE	n1	n2	n3	n4	n5	n6	n7	n8	n9	n10

where n = normal instruction word

FAST INTERRUPT INSTRUCTION TIMING

↓ (Interrupt synchronized and recognized as pending)

INSTRUCTION CYCLE	1	2	3	4	5	6	7	8	9	10
FETCH	n3	n4	iv1	iv2	n5	n6	n7	n8	n9	n10
DECODE	n2	n3	n4	iv1	iv2	n5	n6	n7	n8	n9
EXECUTE	n1	n2	n3	n4	iv1	iv2	n5	n6	n7	n8

where n = normal instruction word
iv = interrupt vector instruction word (no change – of – flow)

LONG INTERRUPT INSTRUCTION TIMING

INSTRUCTION CYCLE	1	2	3	4	5	6	7	8	9	10
FETCH	n3	n4	iv1	iv2	i1	i2	i3	n5	n6	n7
DECODE	n2	n3	n4	jsr	–	i1	rti	–	n5	n6
EXECUTE	n1	n2	n3	n4	jsr	–	i1	rti	–	n5

where n = normal instruction word
iv = interrupt vector instruction word (change – of – flow)
jsr = jump to interrupt service routine instruction
i = interrupt service routine instruction word
rti = return from interrupt instruction

DSP56000/1 HARDWARE DO LOOP

* OPERATION

Start of Loop:

1] SP+1->SP; LA->SSH; LC->SSL; #xxx->LC;

2] SP+1->SP; PC->SSH; SR->SSL; Expr-1->LA;

3] 1->LF.

End of Loop:

SSL(LF)->SR; SP-1->SP; SSH->LA; SSL->LC; SP-1->SP.

* EXAMPLE of NESTED DO LOOPS

```
        DO    #n1,END1
              .
        DO    #n2,END2
              .
              .
              .
        MOVE            A,X:(R0)+
END2    ADD   A,B       X:(R1)+,X0
END1
```

DSP56000/1 HARDWARE DO LOOP

* FEATURES

- No Overhead Cycles (Looped code runs as fast as Straight–Line Code).
- Looped Code Saves Program Memory.
- Supports Nested Loops.
- Supports Dynamic Loop Counts.

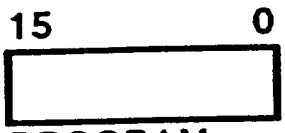
* SYNTAX

```
DO    #xxx,Expr          ;(Static)
DO    X:<ea>,Expr        ;(Dynamic)
```

Expr(ession) = assembler expression or absolute address

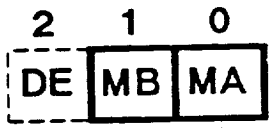
DSP56000/1 PROGRAM CONTROLLER

* PROGRAMMING MODEL

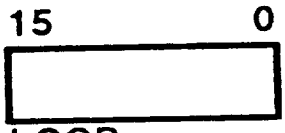


PROGRAM COUNTER (PC)

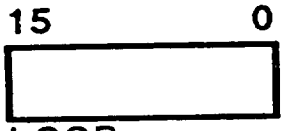
STATUS REGISTER (SR)



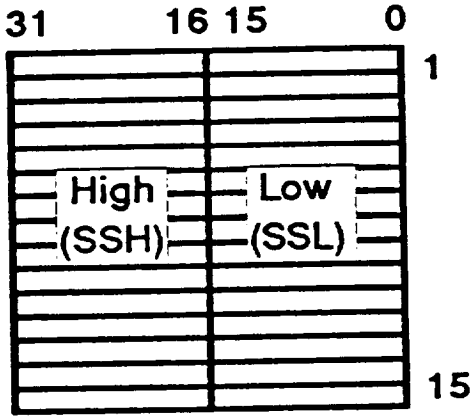
OPERATING MODE REGISTER (OMR)



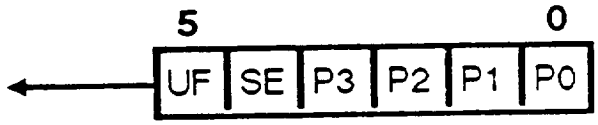
LOOP ADDRESS (LA)



LOOP COUNTER (LC)



SYSTEM STACK (SS)

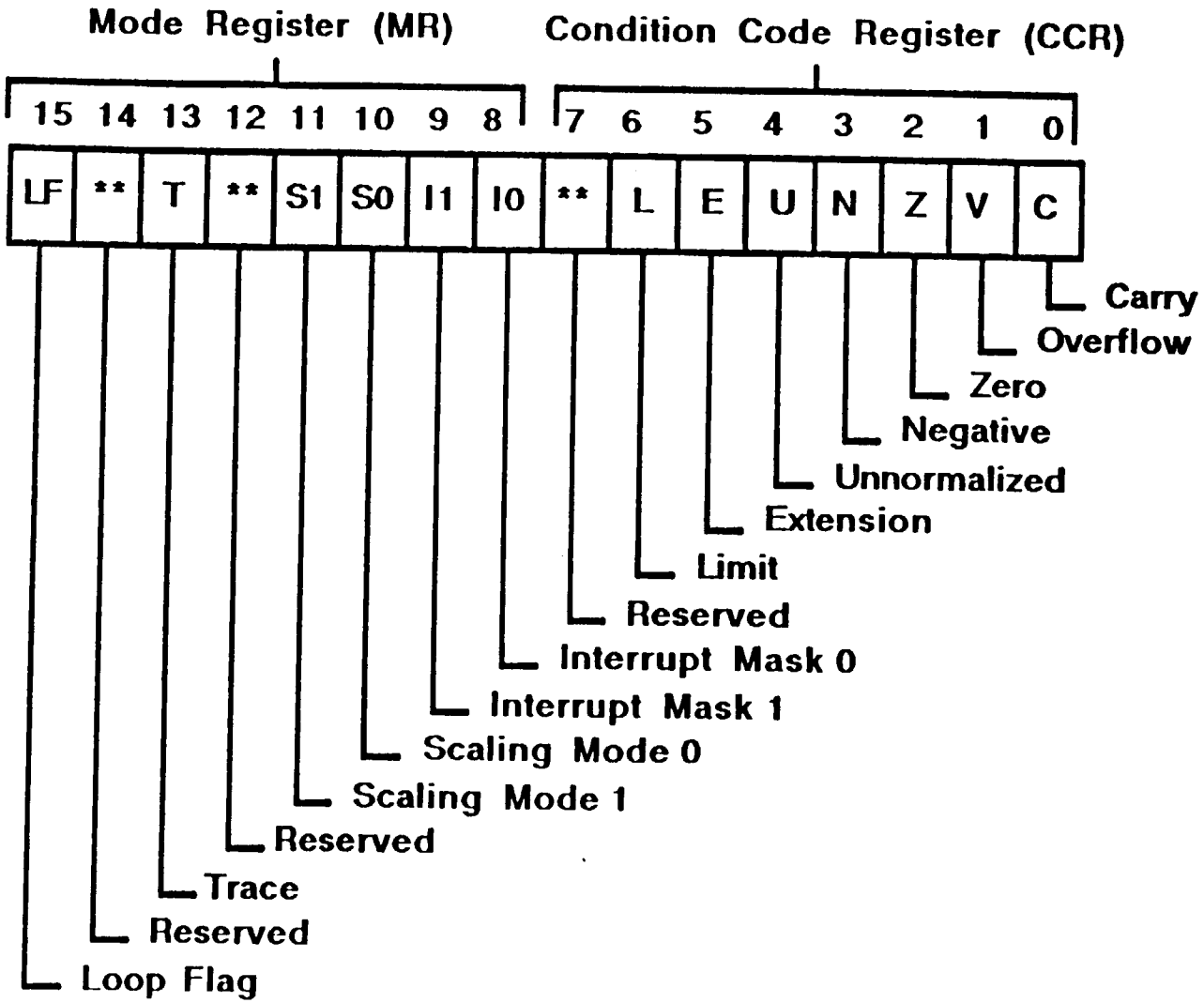


STACK POINTER (SP)

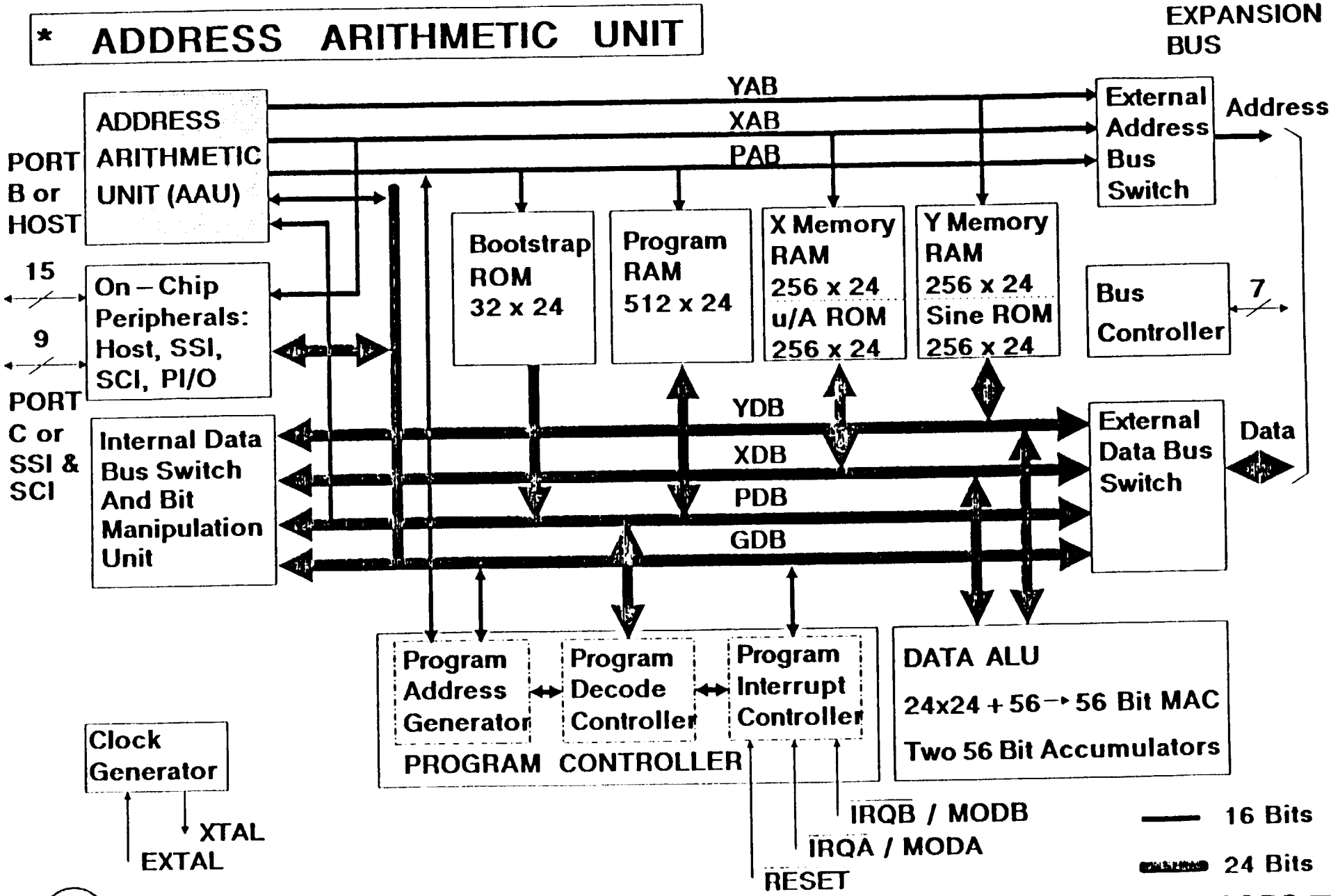
DSP56001 ONLY

DSP56000/1 PROGRAM CONTROLLER

* STATUS REGISTER

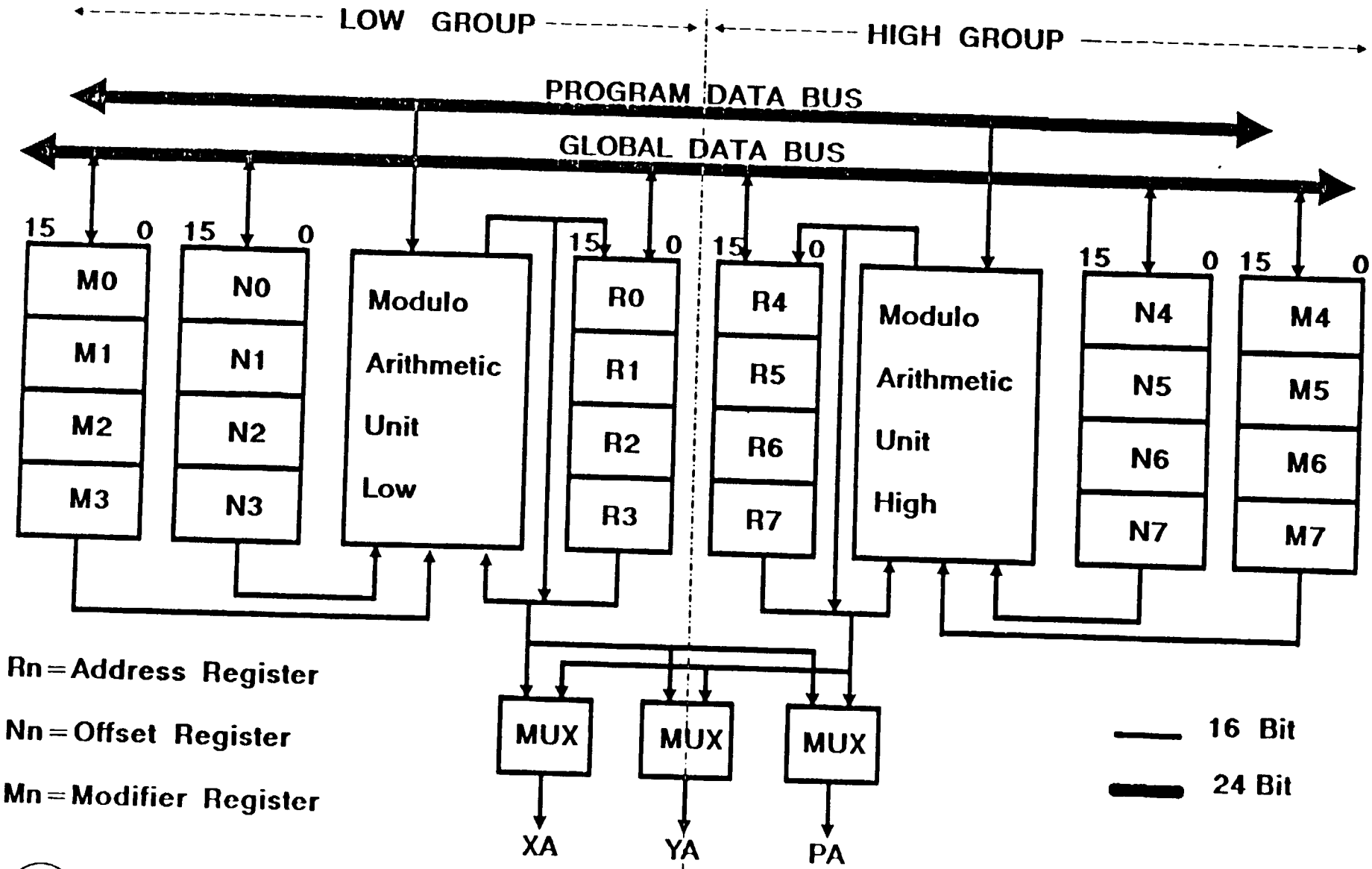


MOTOROLA DSP56001



DSP56000/1 ADDRESS ARITHMETIC UNIT

* BLOCK DIAGRAM



DSP56000/1
ADDRESS ARITHMETIC UNIT

* ARITHMETIC CAPABILITY

* Each Unit Can Perform:

1] LINEAR Arithmetic

R +/− N

2] MODULO Arithmetic

(R +/− N) mod M

Example: (3+4) mod 5 = 2

7÷5=1 remainder 2

3] REVERSE CARRY (BIT−REVERSED) Arithmetic

R +/− N
rev

carry

Example: 0 1 0 1 (5)

+
rev 0 1 0 0 (4)

0 0 1 1 (3)

DSP56000/1 ADDR. REG. INDIRECT

* ADDRESS MODIFIERS – LINEAR

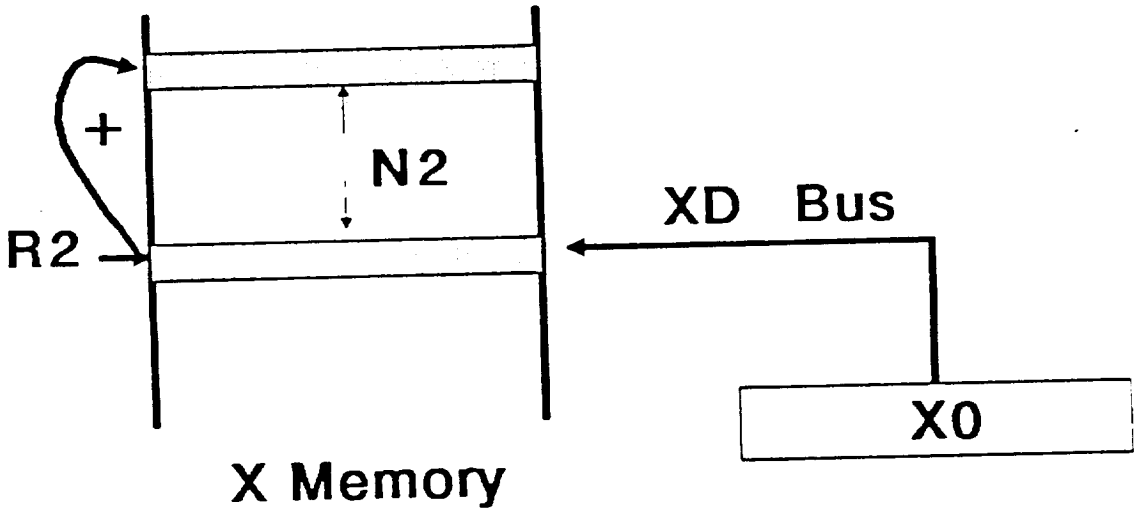
MODIFIER Mn VALUE	ADDRESSING MODE ARITHMETIC
0	Reverse Carry (Bit Reversed)
1	Modulo 2
2	Modulo 3
:	:
:	Modulo (Mn + 1)
:	:
32766	Modulo 32767
32767	Modulo 32768
:	Reserved
65535	Linear (Modulo 65536)

DSP56000/1 ADDR. REG. INDIRECT

* Linear Post Increment by Offset: (Rn)+Nn

* Example: MOVE X0,X:(R2)+N2

M2 must be 11.....111



DSP56000/1 ADDRESSING MODES

* SUMMARY

ADDRESSING MODE	ADDRESS MODIFIER	ASSEMBLER SYNTAX
REGISTER DIRECT	No	any reg. name
ADDRESS REGISTER INDIRECT		
No Update	Yes	(Rn)
Postincrement by 1	Yes	(Rn) +
Postdecrement by 1	Yes	(Rn) –
Postincrement by Offset Nn	Yes	(Rn) + Nn
Postdecrement by Offset Nn	Yes	(Rn) – Nn
Predecrement by 1	Yes	– (Rn)
Indexed by Offset Nn (Rn and Nn unchanged)	Yes	(Rn + Nn)
SPECIAL		
Immediate Data	No	#expr(ession) (24)
Absolute Address	No	expr (16)
Immediate Short Data	No	#expr (8,12)
Short Jump Address	No	expr (12)
Absolute Short Address	No	expr (6)
I/O Short Address	No	expr (6)

DSP56000/1 ADDR. REG. INDIRECT

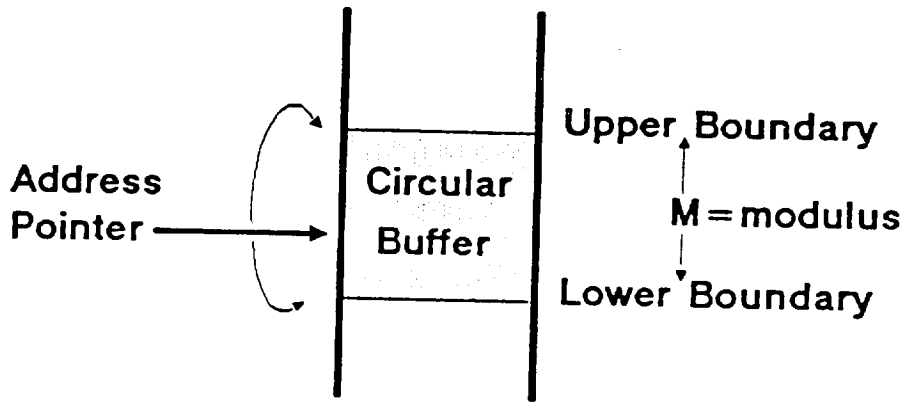
* ADDRESS MODIFIERS – MODULO

MODIFIER Mn VALUE	ADDRESSING MODE ARITHMETIC
0	Reverse Carry (Bit Reversed)
1	Modulo 2
2	Modulo 3
:	:
:	Modulo (Mn + 1)
:	:
32766	Modulo 32767
32767	Modulo 32768
:	Reserved
65535	Linear (Modulo 65536)

DSP56000/1 ADDR. REG. INDIRECT

* MODULO ADDRESSING

* What is it?



* Why use it?

Implements:
- FIFOs (queues)
- Delay Lines
- Sample Buffers

* Example: M^{th} Order FIR Filter

$$Y(n) = \sum_{k=0}^{M-1} C(k)\, X(n-k) \quad , \qquad n = 0,1,2, \ldots$$

A Technique to Simulate a Shift Register of Length M in RAM by simply UPDATING a Pointer instead of Moving Data.

DSP56000/1 ADDR. REG. INDIRECT

* MODULO ADDRESSING – SET UP

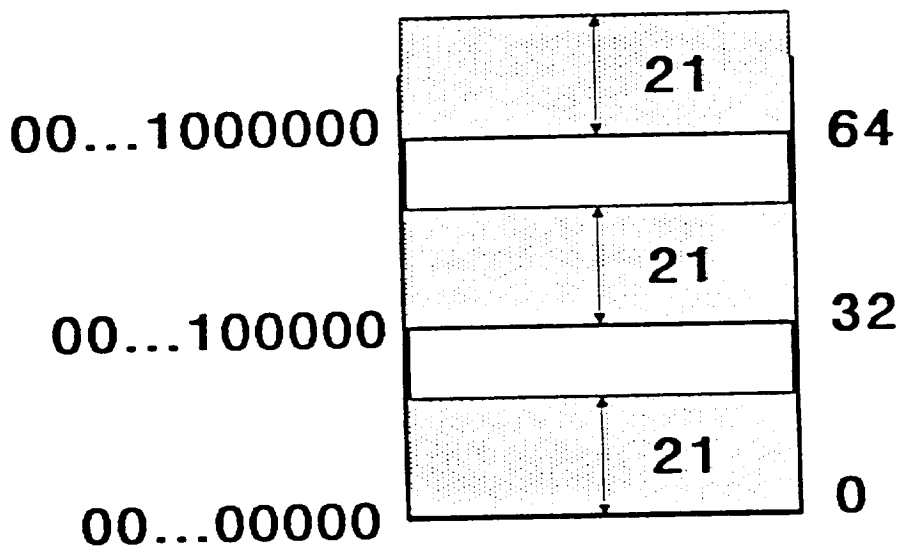
1] Load Mn with M−1

2] Initiallize Rn to any value between

$$xx..xx\underbrace{00..00}_{k} \quad \text{and} \quad xx..xx\underbrace{00..00}_{k}+M-1$$

where $2^k \geq M$

Example: If M=21, then $2^5 > 21$ so $k \geq 5$.



The lower boundary can be any multiple of a power of 2 with five or more least significant zeros

3] Use Any Address Register Indirect Mode

Note: |Nn| must be $\leq$ M

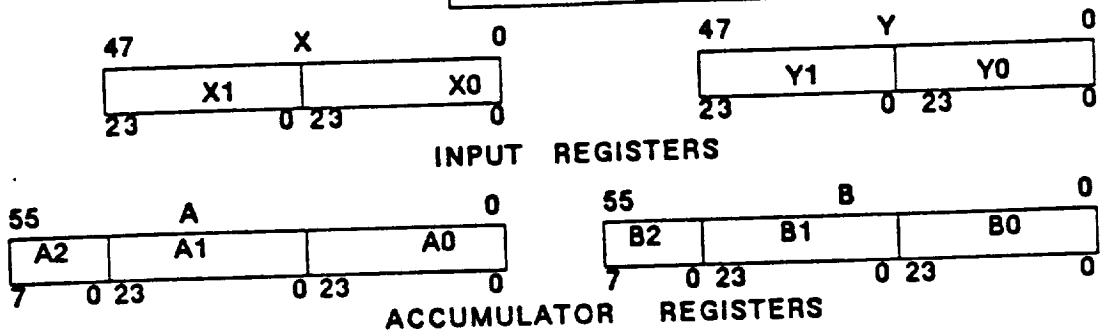
DSP56000/1 ADDR. REG. INDIRECT

* ADDRESS MODIFIERS – REVERSE CARRY

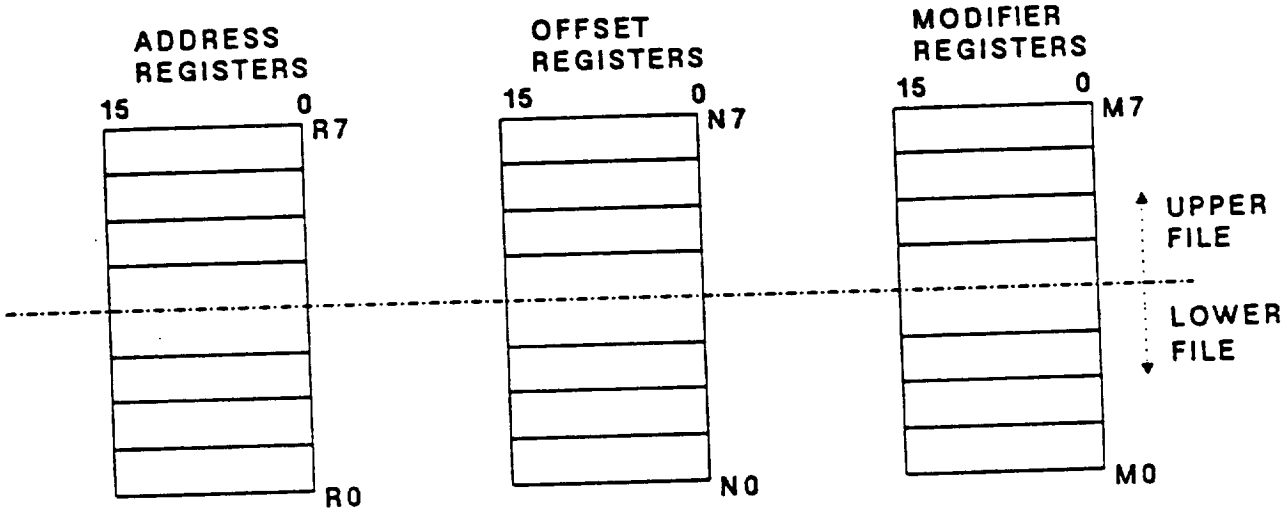
MODIFIER Mn VALUE	ADDRESSING MODE ARITHMETIC
0	Reverse Carry (Bit Reversed)
1	Modulo 2
2	Modulo 3
⋮	⋮
⋮	Modulo (Mn + 1)
⋮	⋮
32766	Modulo 32767
32767	Modulo 32768
⋮	Reserved
65535	Linear (Modulo 65536)

DSP56000/1 PROGRAMMING MODEL

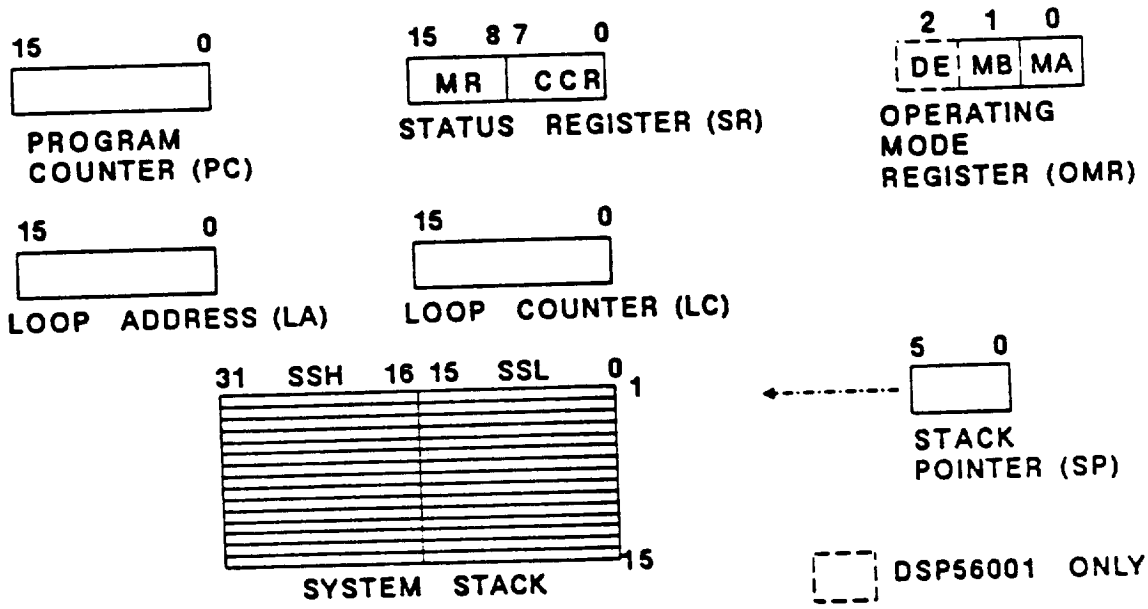
DATA ALU



ADDRESS ARITHMETIC UNIT



PROGRAM CONTROLLER



DSP56000/1 INSTRUCTION SET

* INTRODUCTION

* MPU – Like
* Makes Pipeline Invisible
* 62 Instruction Opcodes
* 6 Groups
  - Arithmetic Instructions □
  - Logical Instructions □
  - Bit Manipulation Instructions
  - Program Control Instructions
  - Loop Instructions
  - Move Instructions

□ Most of these instructions allow up to two move operations (called parallel data moves) to be executed concurrently with opcode execution.

DSP56000/1 INSTRUCTION SET

* ARITHMETIC INSTRUCTIONS(Dst=56-bit A,B)

ABS	Absolute Value
ADC	Add Long with Carry
ADD	Add
ADDL	Shift Left and Add Accumulators
ADDR	Shift Right and Add Accumulators
ASL	Arithmetic Shift Accumulator Left
ASR	Arithmetic Shift Accumulator Right
CLR	Clear Accumulator
CMP	Compare
CMPM	Compare Magnitude
DIV	Divide Iteration △
MAC	Signed Multiply-Accumulate
MACR	Signed Multiply-Accumulate & Round
MPY	Signed Multiply
MPYR	Signed Multiply and Round
NEG	Negate
NORM	Normalize Accumulator Iteration △
RND	Round Accumulator
SBC	Subtract Long with Carry
SUB	Subtract
SUBL	Shift Left and Subtract Accumulators
SUBR	Shift Right and Subtract Accumulators
Tcc	Transfer Conditionally △
TFR	Transfer Data ALU Register
TST	Test Accumulator

△ These instructions do NOT allow parallel data moves

DSP56000/1 INSTRUCTION SET

* LOGICAL INSTRUCTIONS (Dst=24-Bit A1,B1)†

AND	Logical AND
ANDI	AND Immediate with Control Register △
EOR	Logical Exclusive OR
LSL	Logical Shift Accumulator Left
LSR	Logical Shift Accumulator Right
NOT	Logical Complement
OR	Logical Inclusive OR
ORI	OR Immediate with Control Register △
ROL	Rotate Left
ROR	Rotate Right

△ These instructions do NOT allow parallel data moves

† Except ANDI and ORI

DSP56000/1 INSTRUCTION SET

* PARALLEL DATA MOVES

* SYNTAX:

Opcode	Operands	X Bus Data	Y Bus Data
MAC	X0,Y0,A	X:(R0)+,X0	Y:(R4)+,Y0

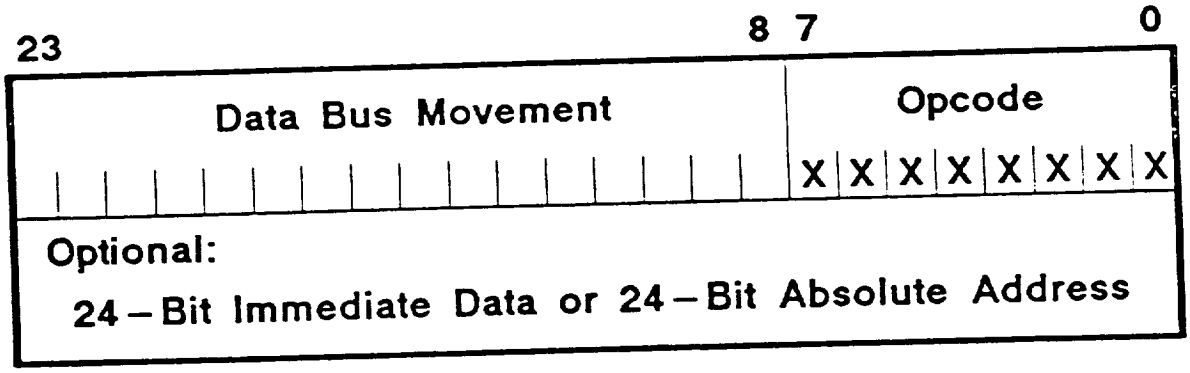
MAC → Specifies:

* Data ALU Operation
* Logical Operation

X:(R0)+,X0 / Y:(R4)+,Y0 → Specifies:

* Optional Data Transfers
* Addressing Modes
* Address Space Qualifiers [X:,Y:,L:]

* OBJECT CODE FORMAT:



DSP56000/1 INSTRUCTION SET

* BIT MANIPULATION INSTRUCTIONS (test mem)

BCLR	Bit Test and Clear
BSET	Bit Test and Set
BCHG	Bit Test and Change
BTST	Bit Test on Memory

Syntax: BCHG #n,X:<ea>
BCHG #n,Y:<ea>

JCLR	Jump if Bit Clear
JSET	Jump if Bit Set
JSCLR	Jump to Subroutine if Bit Clear
JSSET	Jump to Subroutine if Bit Set

Syntax: JSET #n,X:<ea>,xxxx
JSET #n,Y:<ea>,xxxx

DSP56000/1 INSTRUCTION SET

* PROGRAM CONTROL INSTRUCTIONS

Jcc	Jump Conditionally
JMP	Jump
JScc	Jump to Subroutine Conditionally
JSR	Jump to Subroutine
NOP	No Operation
REP	Repeat Next instruction
RESET	Reset Peripheral Devices
RTI	Return from Interrupt
RTS	Return from Subroutine
STOP	Stop Instruction Processing ▽
SWI	Software Interrupt
WAIT	Wait for Interrupt ▽

▽ Low power standby mode.

DSP56000/1 INSTRUCTION SET

* LOOP INSTRUCTIONS

DO	Start Hardware Loop
ENDDO	End Current DO Loop

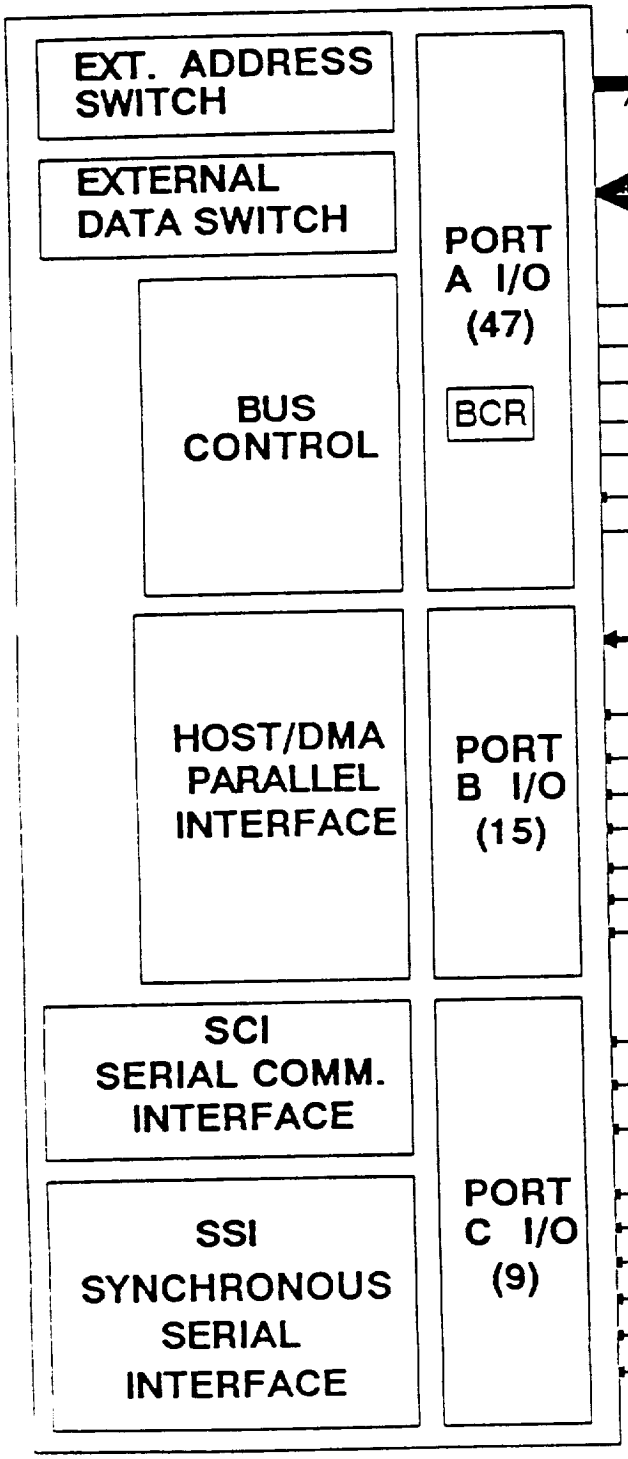
* MOVE INSTRUCTIONS

LUA	Load Updated Address
MOVE	Move Data Registers ◁
MOVEC	Move Control Register
MOVEM	Move Program Memory
MOVEP	Move Peripheral Data

◁ Equivalent to Data ALU NOP with parallel data moves

DSP56000/1 INPUT / OUTPUT

* BLOCK DIAGRAM

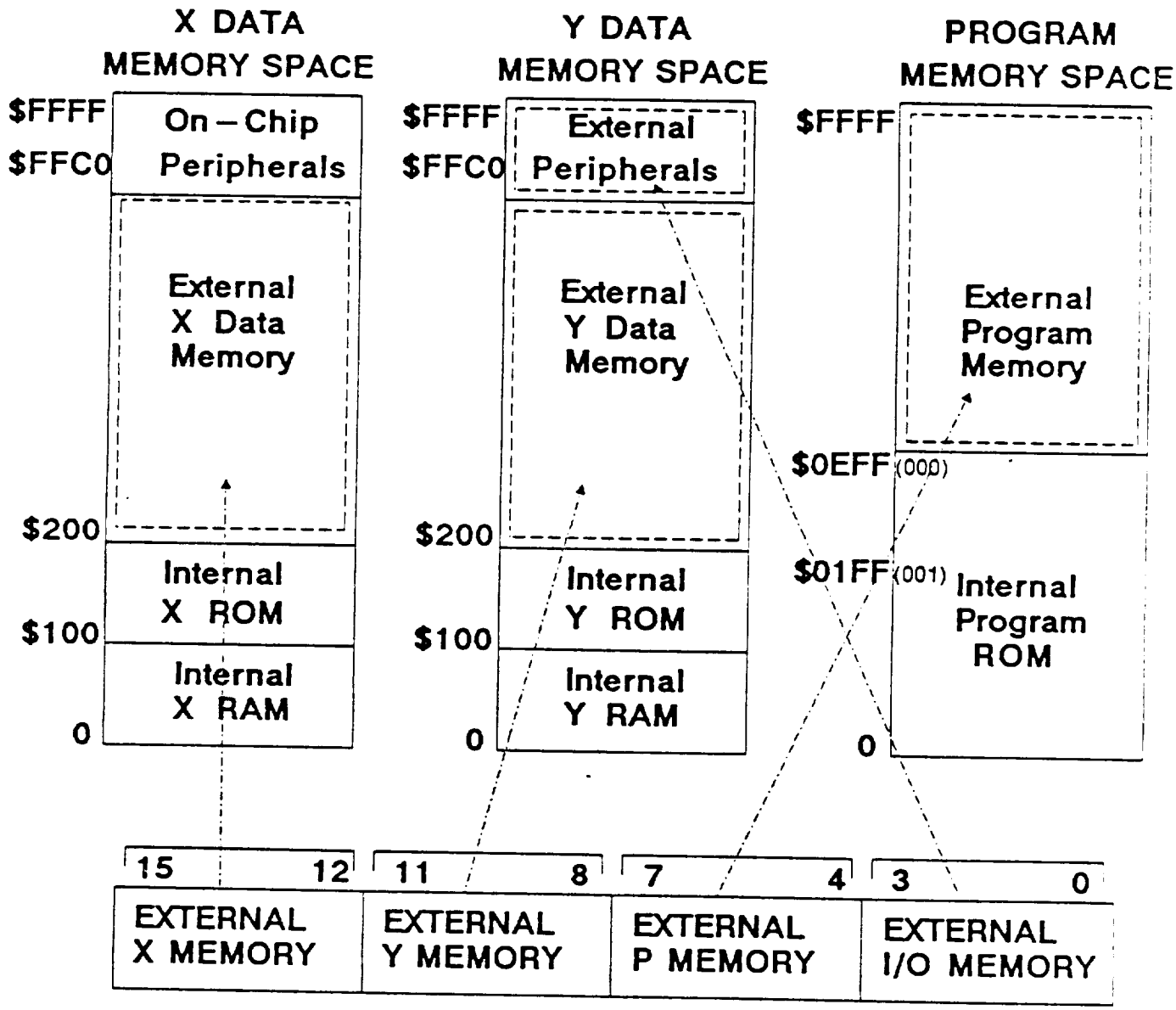


Default Function		Alternate Function
A0 – A15		– –
D0 – D23		– –
$\overline{PS}$		– –
$\overline{DS}$		– –
X/$\overline{Y}$		– –
$\overline{RD}$		– –
$\overline{WR}$		– –
$\overline{BR}$		– –
$\overline{BG}$		– –
PB0 – PB7	↔	H0 – H7
PB8	←	HA0
PB9	←	HA1
PB10	←	HA2
PB11	←	HR/$\overline{W}$
PB12	←	$\overline{HEN}$
PB13	→	$\overline{HREQ}$
PB14	←	$\overline{HACK}$
PC0	←	RXD
PC1	→	TXD
PC2	↔	SCLK
PC3	↔	SC0
PC4	↔	SC1
PC5	↔	SC2
PC6	↔	SCK
PC7	←	SRD
PC8	→	STD

DSP56000/1 PORT A

* BUS CONTROL REGISTER (BCR)

DSP56000/1 NORMAL EXPANDED MODE

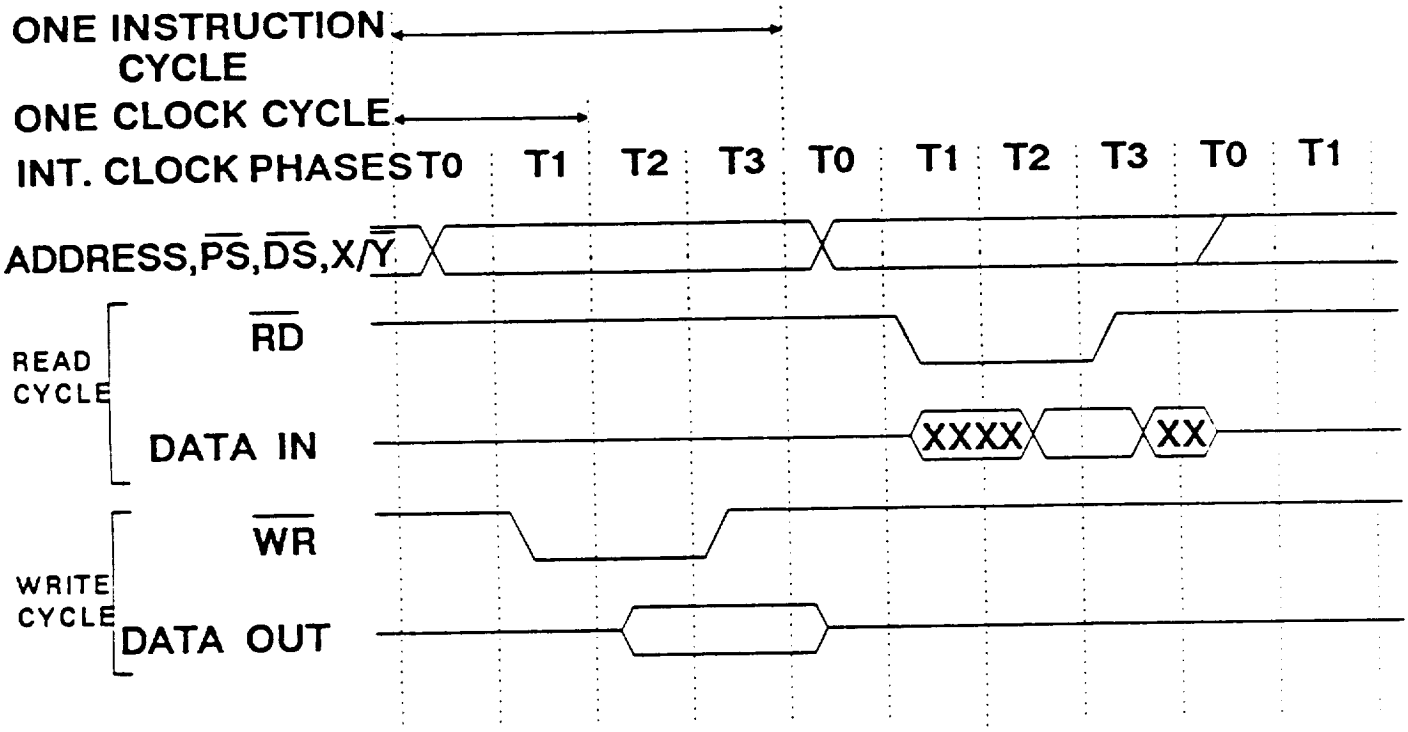


PORT A – BUS CONTROL REGISTER (BCR)

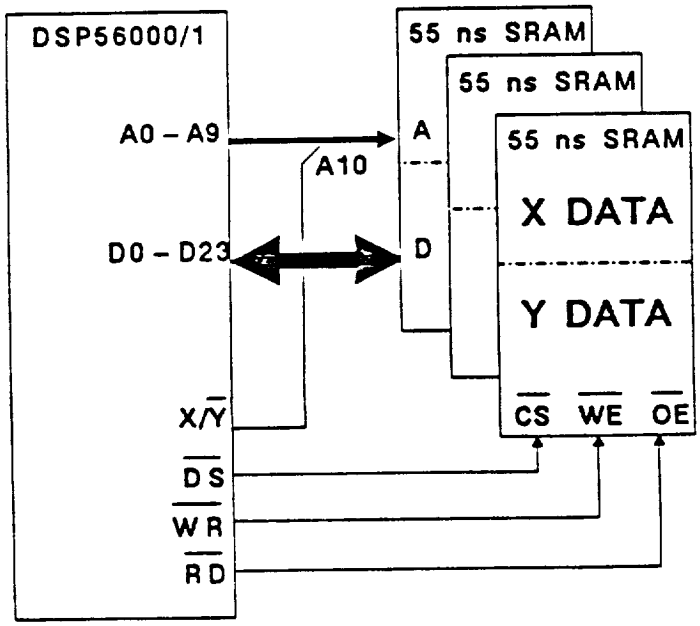
* 0 – 15 WAIT STATES Can Be Inserted Into Each External Memory Access Category

DSP56000/1 PORT A

*** NO WAIT STATES (BCR=$0000)**



* Full Speed 10.25 MHz Instruction Execution with 55 nsec Access Time 2016 And 2018 RAMs



EXTERNAL DATA RAM EXPANSION

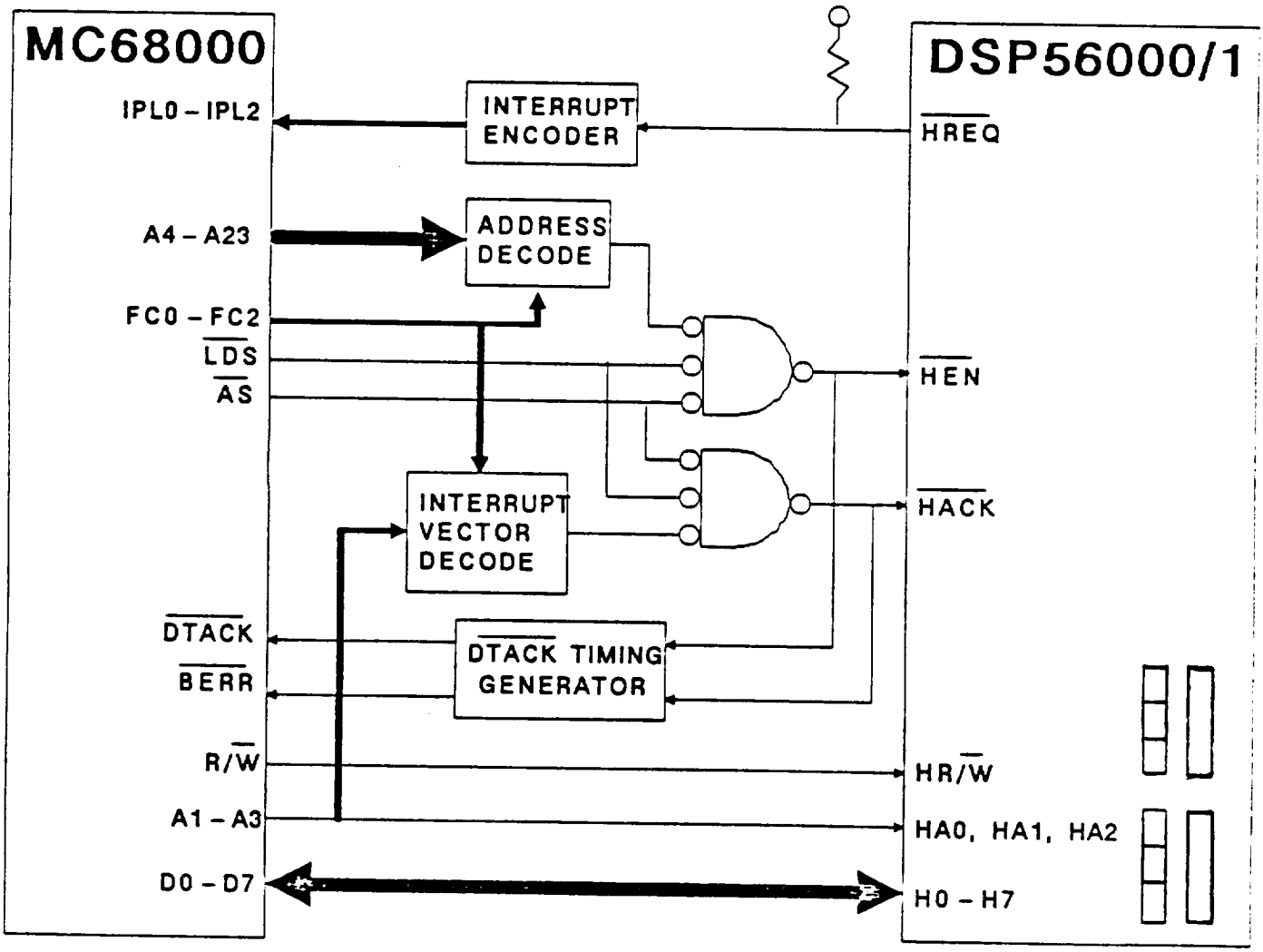
3 – 2K x 8 RAMs
ORGANIZED AS
1K x 24 X DATA
1K x 24 Y DATA

DSP56000/1 HOST INTERFACE

* 68000 TO DSP Example

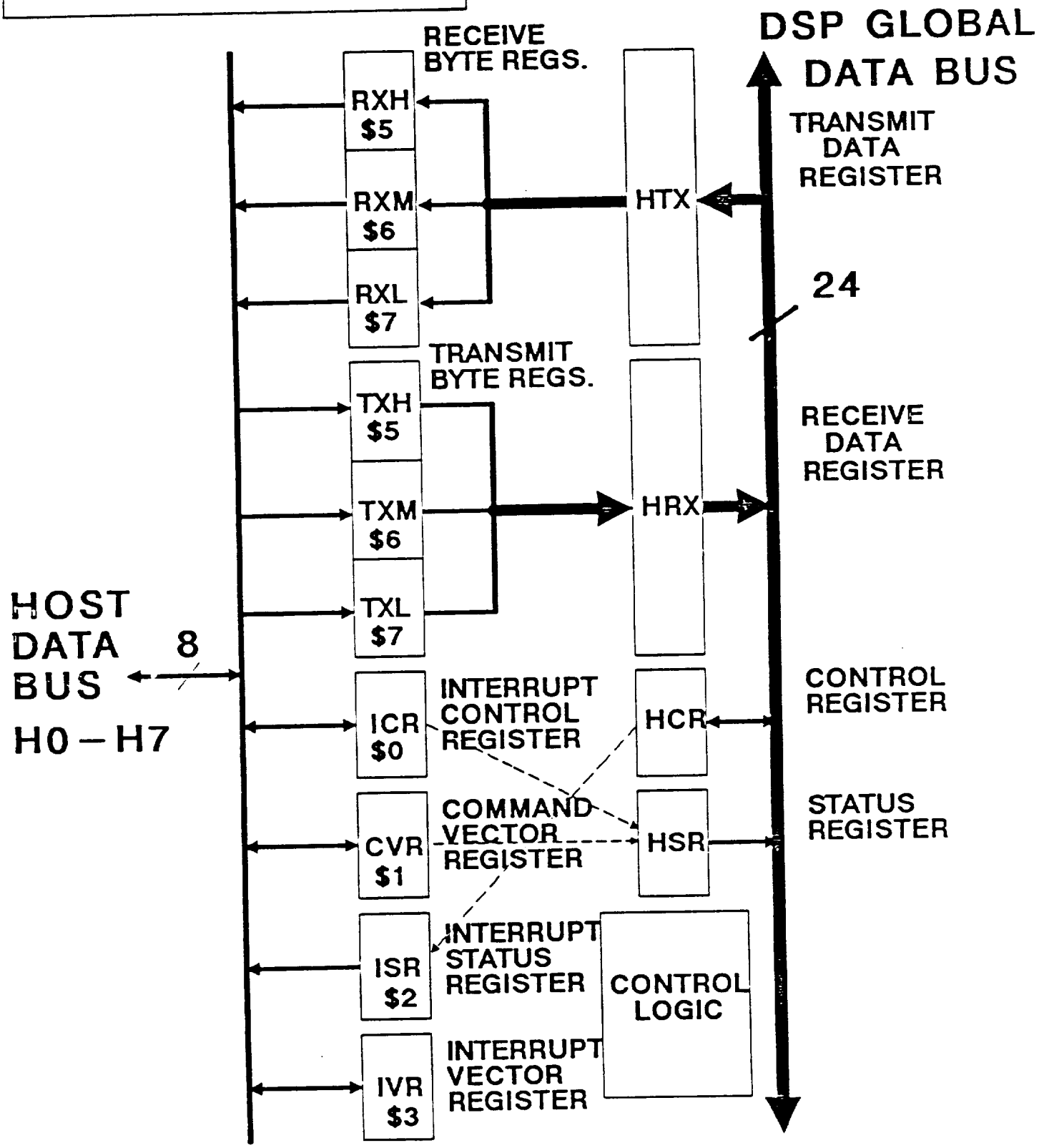
* Use MOVEP Instruction with WORD and LONG WORD Data Size to Transfer MULTIPLE Bytes

* 68020 can use DYNAMIC BUS SIZING to Transfer Multiple Bytes with Any Instruction



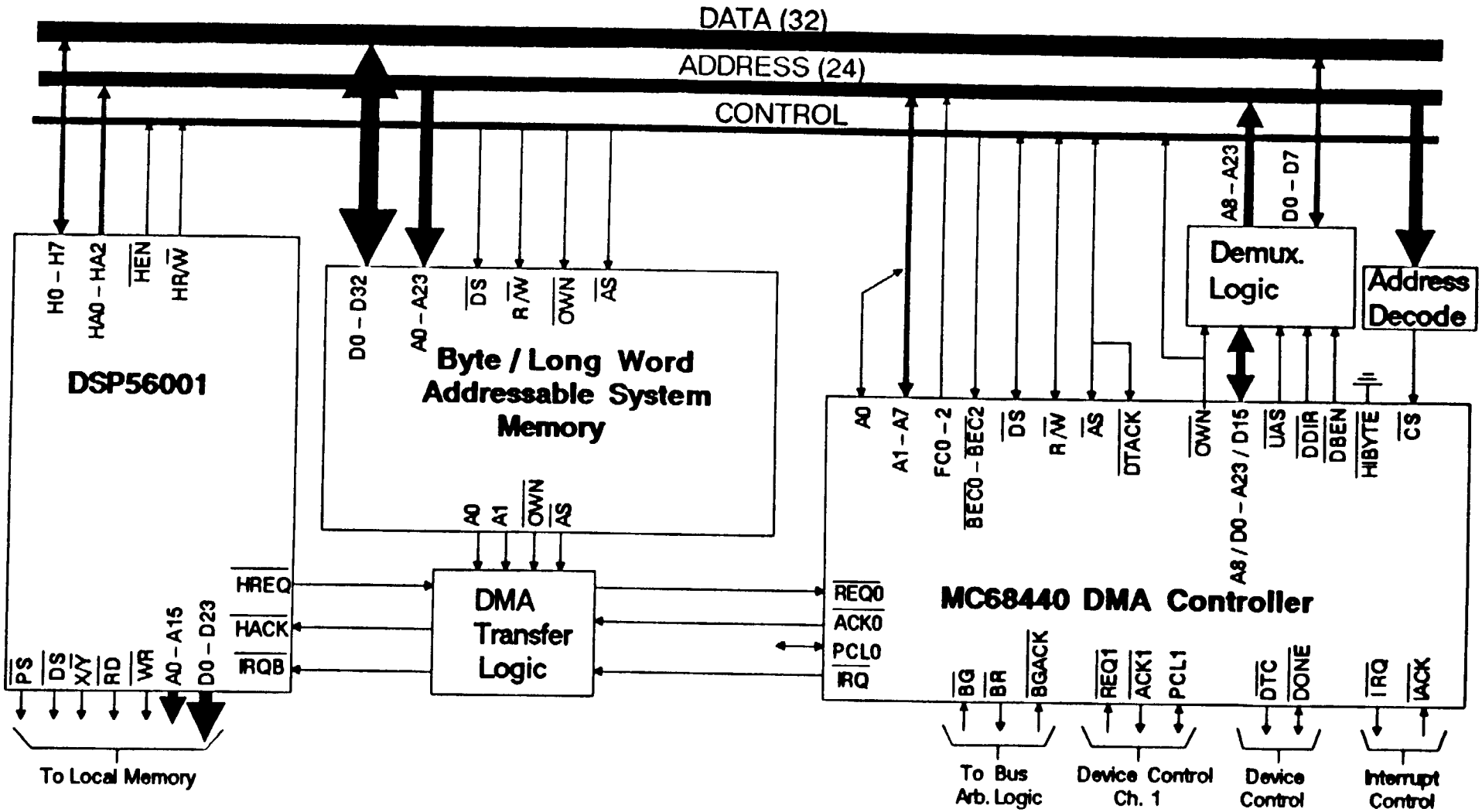
DSP56000/1 HOST INTERFACE

* BLOCK DIAGRAM



DSP56001 HOST INTERFACE DMA

* SYSTEM BLOCK DIAGRAM



DMA APPLICATION

* SUMMARY :

Type : Device / Memory Transfers

Device : 8 – Bit DSP56001 Host Interface

Memory : Byte / Long Word Addressable System Memory

DMA Controller: MC68440

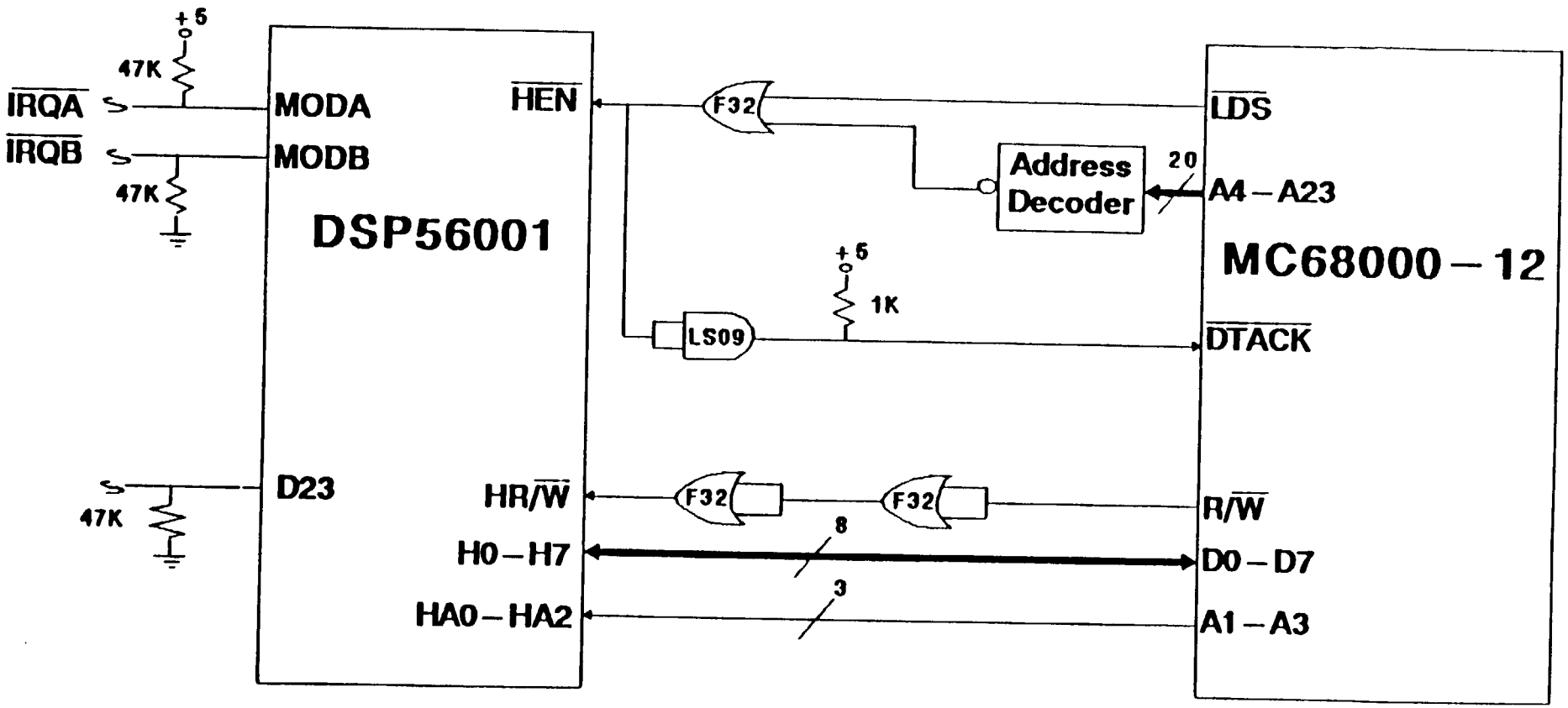
Modes:

MC68440 = Burst
DSP56001 = Cycle Steel
Transfer = Single Address
Block
Bytes

Performance: 24 – Bit DSP56001 word memory to memory every 1.6 us

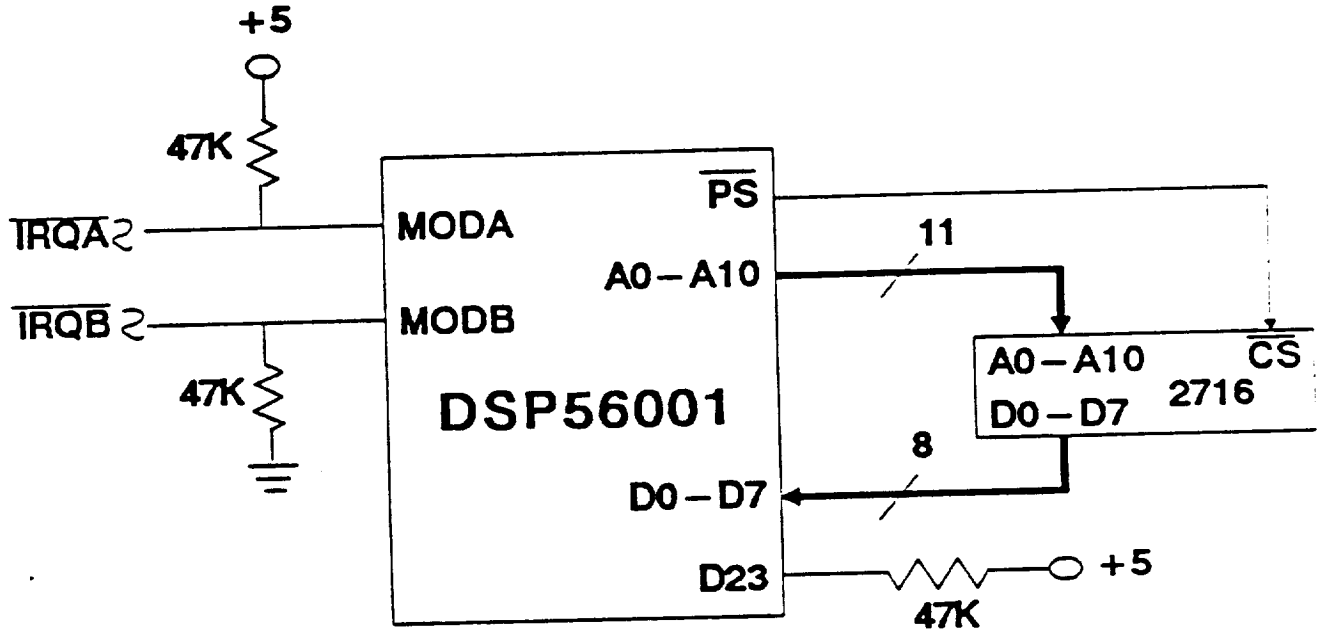
DSP56001 BOOTSTRAP HARDWARE

*** HOST MPU – MC68000–12**



DSP56001 BOOTSTRAP HARDWARE

* BYTE WIDE EPROM @ $C000 – $C5FF



* No Glue Logic
* t_{acc} < 780 ns @ 20.5 MHz

DSP56000/1 INPUT – OUTPUT

* SERIAL COMMUNICATIONS INTERFACE [SCI]

* A 320 KBit Per Second, ASYNCHRONOUS Serial Communications Interface to a Wide Variety of Industry–Standard Devices, including:
  - MCUs with Asynchronous Serial Ports
  - ACIAs and UARTs
  - RS232 Lines and Modems
  - Remote Terminals and Computers

* Super Set of MC6801 and MC68HC11 SCI

* MULTIDROP mode for Multi–Processor Systems
  - TWO Receiver WAKE–UP Modes
    1) Wake–up on IDLE LINE
    2) Wake–up on ADDRESS BIT
  - 'Wire OR' Mode

* On–Chip BAUD RATE GENERATOR / INTERRUPT TIMER

* 2.5 MBit per Second SYNCHRONOUS Serial Mode
  - Compatible with 8051/96 Shift Register Mode
  - Serial to Parallel / Parallel to Serial Function
  - Useful for Bit Stream I/O

DSP56000/1 INPUT / OUTPUT

* SYNCHRONOUS SERIAL INTERFACE [SSI]

* A 5 MBit Per Second Serial Interface to a Wide Variety of Serial Devices, Including:
  - Industry – Standard CODECS
  - Time Division Multiplexed(TDM) NETWORKS
  - Serial Peripherals (A/D, D/A)
  - DSP to DSP Networks
  - Motorola SPI Peripherals, Processors (HC11)
  - Shift Registers, S to P, P to S

* 6 Pins –

STD	SSI Transmit Data	
SRD	SSI Receive Data	
SCK	SSI Serial Clock	
SC0	Serial Control 0	Depends on SSI Mode
SC1	Serial Control 1	
SC2	Serial Control 2	

* On – Chip Programmable Functions Include:

1] Clock – Continuous, Gated, Internal, External

2] Bit Length, Word Length Sync. Signals

3] TX / RX Timing – Synchronous, Asynchronous

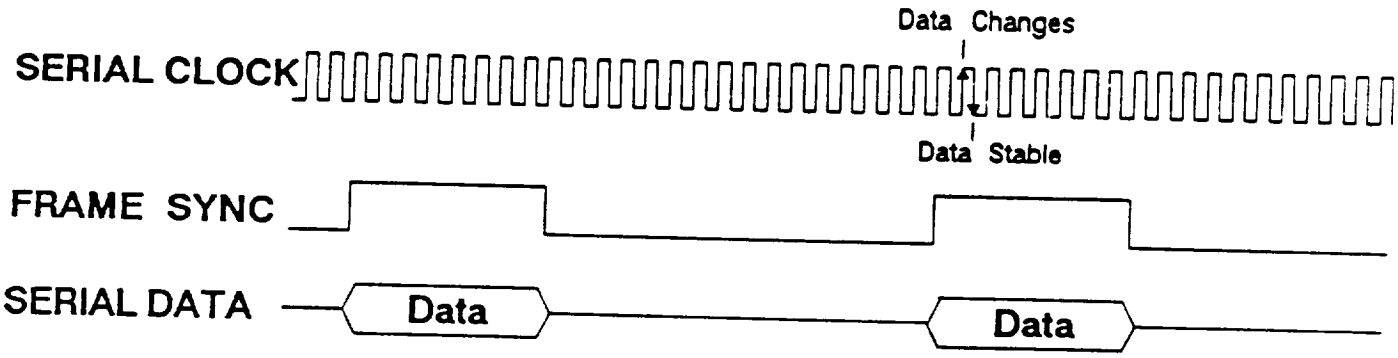
4] Protocol – Normal, Network, On – Demand

5] Word Length – 8, 12, 16, 24 Bits

DSP56000/1 SSI

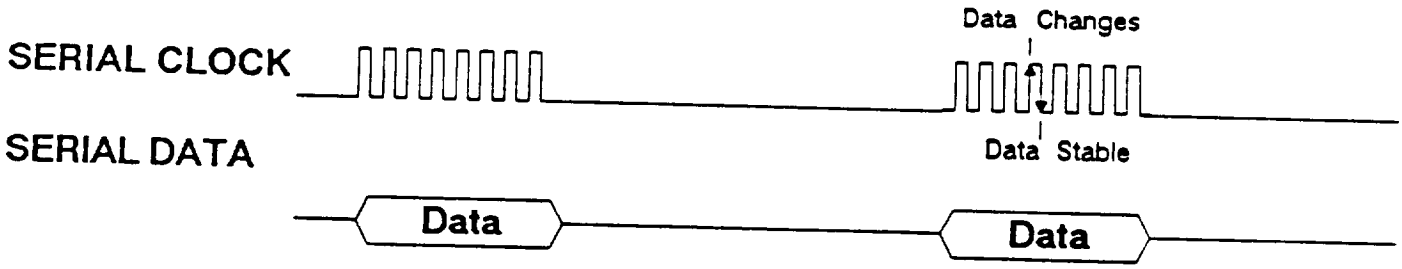
* CLOCK MODES

CONTINUOUS CLOCK



* Separate Frame Sync Signal Indicates First Time Slot in the Frame
* Continuous Clock Output is Required for CODECS and can be used throughout the System

GATED CLOCK



* Word Synchronization is Inherent in the Clock Signal
* Protocol must provide Frame Synchronization
* Uses Only 2 Interface Pins per Channel

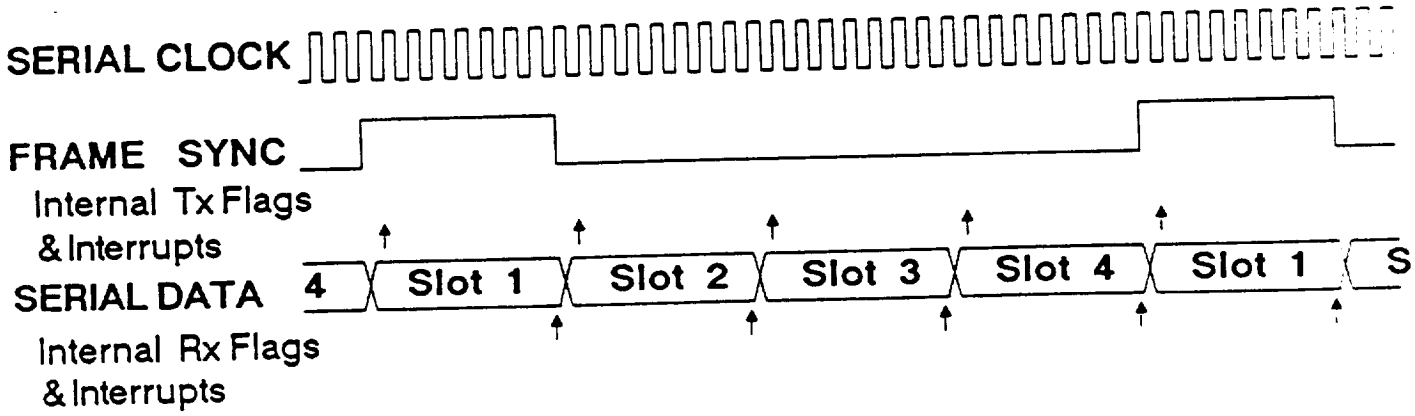
DSP56000/1 SSI

* NETWORK OPERATING MODE

FEATURES

* 2 to 32 TIME SLOTS Per Serial Frame for TIME DIVISION MULTIPLEXED (TDM) I/O
* Data may be Transferred EVERY TIME SLOT
* COMPATIBLE with BELL and CCITT PCM Protocols
  - 8 Bits per Word
  - Bit or Word Frame Sync
  - MSB First
  - 24 or 32 Words per Frame
  - Signaling Frame Sync Generation and Detection using Flags
* TIME SLOT ASSIGNMENT (TSA) is Totally under SOFTWARE CONTROL
  - Devices can Transmit on Multiple Time Slots
  - Devices can Receive Multiple Time Slots
  - TSA can be Changed Dynamically
* OVERRUN and UNDERRUN ERROR DETECTION

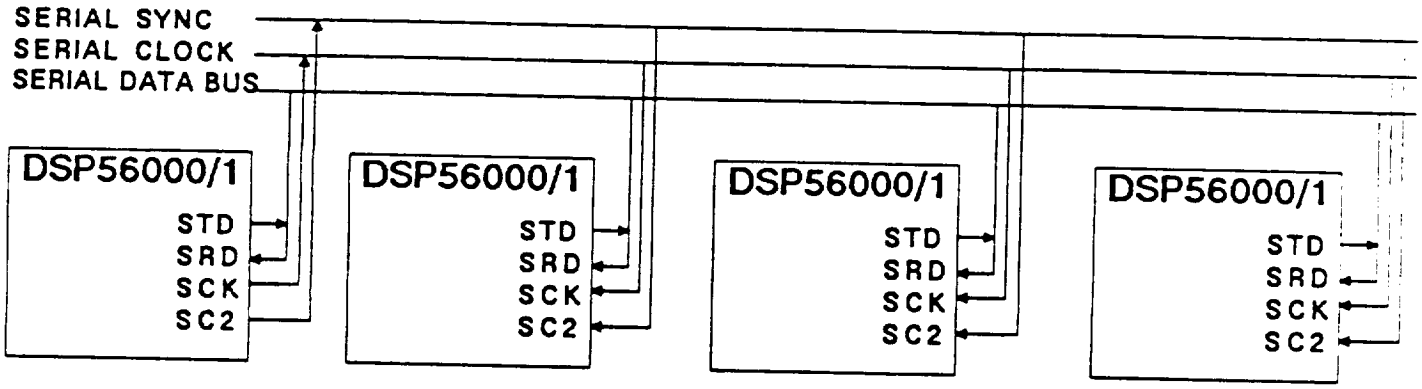
FOUR TIME SLOT NETWORK EXAMPLE



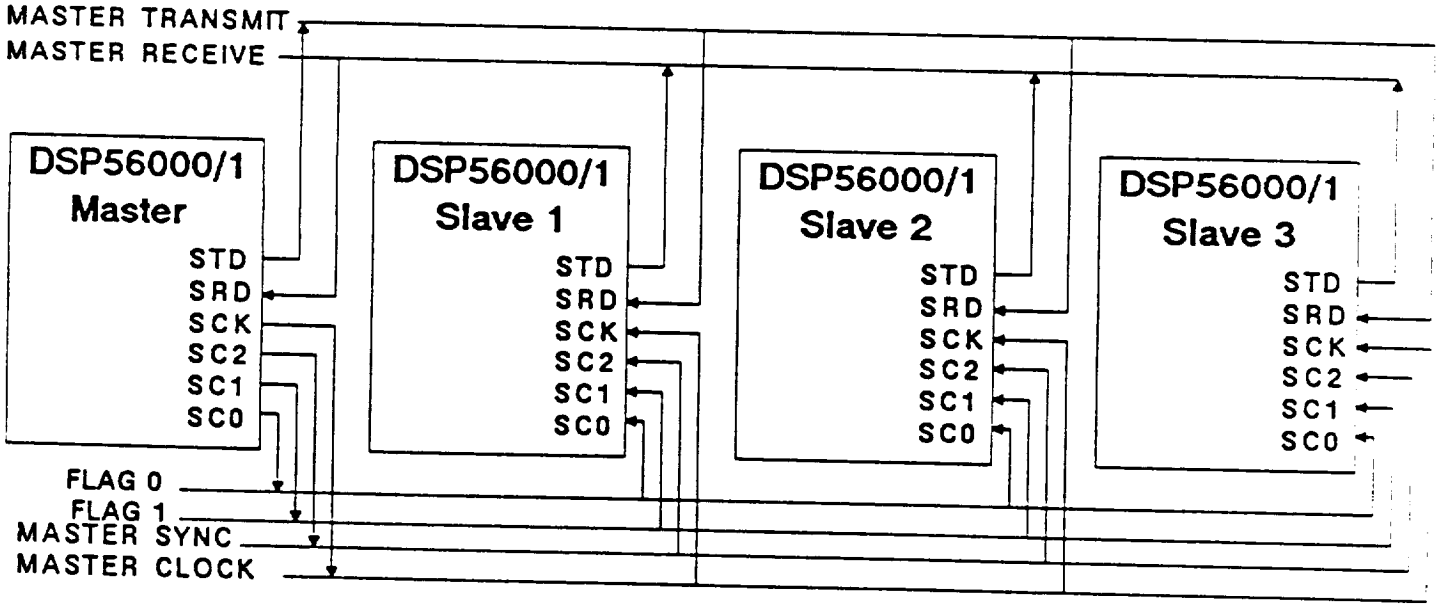
DSP56000/1 SSI

* DSP TO DSP SERIAL NETWORKS

SSI TDM BUS DSP NETWORK



SSI TDM MASTER–SLAVE DSP NETWORK



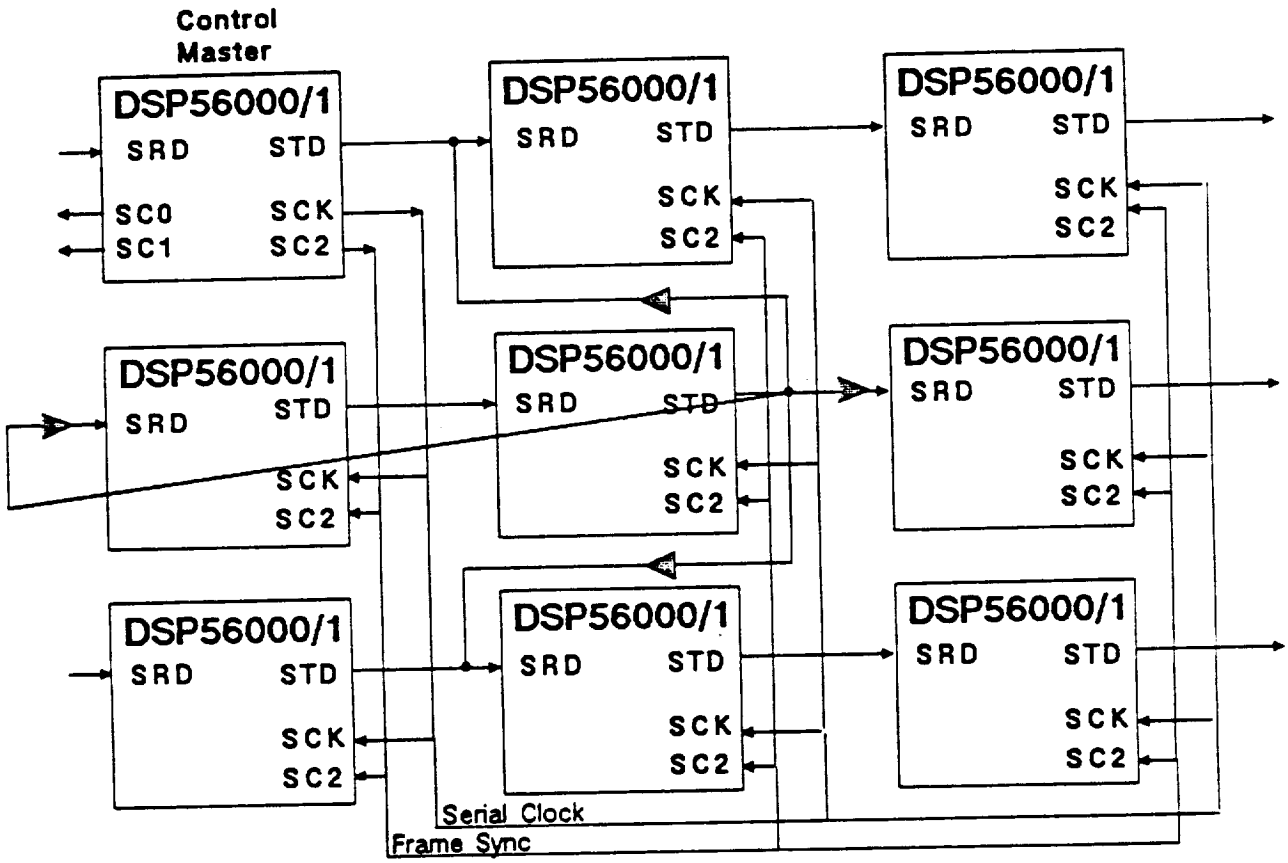
* Flags can specify data types – control, address, data

NETWORK CONFIGURATIONS ARE UNLIMITED !!

DSP56000/1 SSI

* DSP to DSP SERIAL NETWORKS

SSI TDM PARALLEL PROCESSING NEAREST NEIGHBOR ARRAY



Note: Only one node connection is shown

* Flags (SC0, SC1) on Control Master control the Three State Buffers (◁)
* Synchronous Network Mode with Four Time Slots (N,S,E,W) per Frame

Dr. BuB

DSP ELECTRONIC BULLETIN BOARD

Dr. BuB is the name of the electronic bulletin board dedicated to Digital Signal Processing (DSP) at Motorola. This bulletin board offers the DSP community information on Motorola's DSP products, including:

- Current Documentation on
  - — New Products
  - — Improvements to Existing Products
- Application Notes
  - — New
  - — Updates to Existing Notes
- DSP Contest Information
- Question and Answer Forum
- Confidential Mail Service

HOW TO MAKE AN APPOINTMENT

A user can access Dr. BuB from anywhere in the world. To make an appointment with Dr. BuB, you need the following equipment:

1. 1200-baud modem (Bell 212A or V.22)
2. Terminal or personal computer (PC)
3. Telephone line

With the above *minimum* configuration, you will be able to read Motorola DSP information, post questions and comments, and participate in contests. However, if you acquire a file transfer program such as XMODEM, YMODEM, or KERMIT, you will be able to **download** documentation, contest information, etc.* (The XMODEM programs are generally bundled together with the purchase of PC modems.)

* Some programs designed for PC-to-PC file transfer may not fulfill the interface requirements of Dr. BuB.

After obtaining the hardware and ensuring that it is operating properly, use the following procedure to log on:

1. Dial (512)440-DSP1 (440-3771) to access Bell 212A modems, or (512)440-DSP2 (440-3772) to access V.22 modems. BE SURE TO SET the character format to **7 data, even parity, 1 stop bit**.

2. Now the tricky part. . .
   Once the connection has been established, the computer will respond with "Dr. BuB login:". You need to type **guest** followed by a carriage return.

3. DO NOT BE CONFUSED if "password:" appears on the screen. Simply hit a carriage return and wait for "Dr. BuB login:" to reappear.

4. Finally, you will be asked to identify your "terminal type." **Terminal Type** is one of two things:

   dumb, if you have NO terminal emulation software,

   or

   the type of terminal you are using or emulating in software on your PC. A brief list of the most common terminals emulated is displayed to assist you in wording your entry.

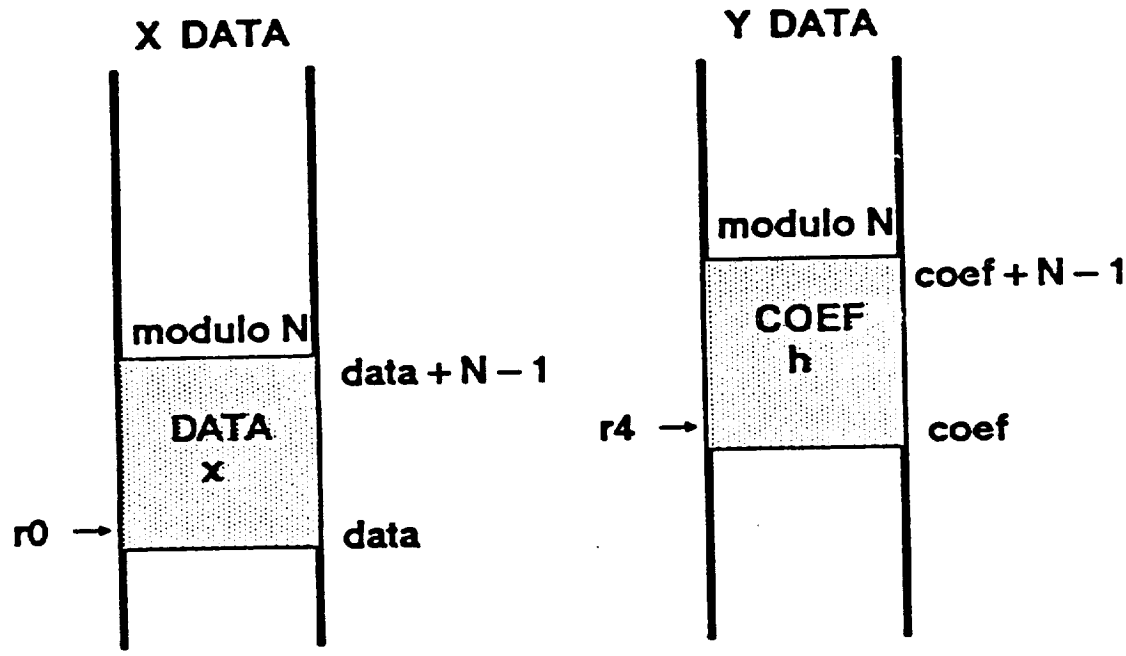
5. Make the appropriate menu selections.

The first two screens the user will encounter are presented on the back of this sheet.

©MOTOROLA INC. 1986

DSP56000/1 FIR FILTER PROGRAM

DATA MEMORY ORGANIZATION




```
start   move    #data,r0
        move    #coef,r4
        move    #N-1,m0
        move    m0,m4

fir     movep                   x:input,x:(r0)
        clr     a               x:(r0)-,x0      y:(r4)+,y0
        rep     m0
        mac     x0,y0,a         x:(r0)-,x0      y:(r4)+,y0
        macr    x0,y0,a         (r0)+
        movep                   a,x:output
        jmp     fir
```

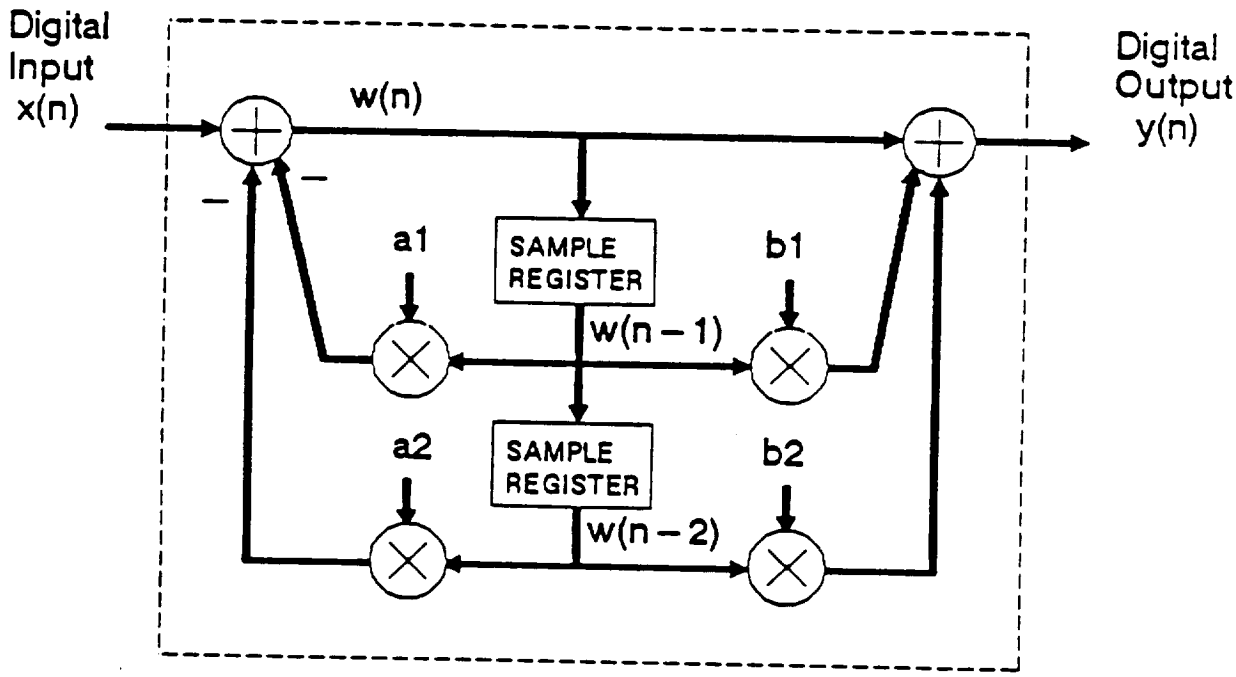
EXECUTION TIME: For the "fir" loop, N + 9 instruction cycles.
SAMPLING RATE fs: Assuming N = 32 and a 20.5 MHz clock, fs = 250 KHz.
PROGRAM MEMORY: 11 words.

DSP56000/1 IIR DIGITAL FILTER

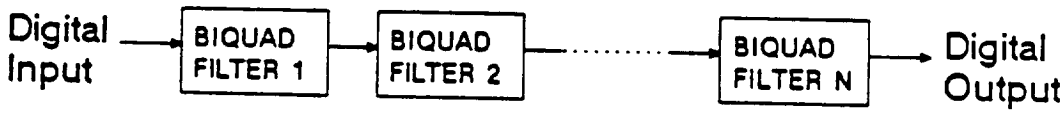
Second–Order Biquad Z Transform

$$Y(z) = \left[\frac{1 + b1\, z^{-1} + b2\, z^{-2}}{1 - a1\, z^{-1} - a2\, z^{-2}} \right] X(z)$$

Second–Order Biquad Block Diagram

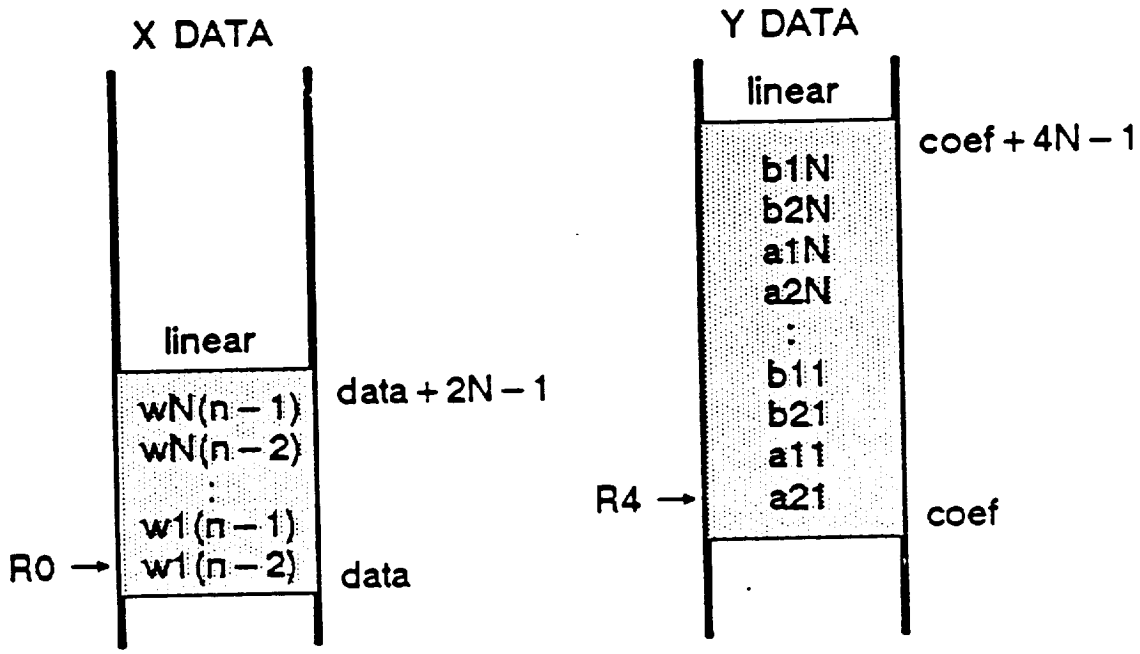


N Cascaded Biquad IIR Digital Filters



DSP56000/1 IIR FILTER PROGRAM

IIR Filter Data Memory Organization




```
start   move    #$ffff,m0
        move    m0,m4

iir     move    #data,r0
        move    #coef,r4
        movep                   x:input,a
        move                    x:(r0)+,x0      y:(r4)+,y0

        do      #N,end_iir
        mac     -x0,y0,a        x:(r0)-,x1      y:(r4)+,y0
        macr    -x1,y0,a        x1,x:(r0)+      y:(r4)+,y0
        mac     x0,y0,a         a,x:(r0)+       y:(r4)+,y0
        mac     x1,y0,a         x:(r0)+,x0      y:(r4)+,y0
end_iir
        rnd     a
        movep                   a,x:output
        jmp     iir
```

EXECUTION TIME: For the "iir" loop, 4N + 13 instruction cycles.
THROUGHPUT RATE: Assuming a sampling rate fs = 8 KHz and a 20.5 MHz clock, N < 317 cascaded biquads. Alternatively, assuming N = 5 (10th order filter) and a 20.5 MHz clock, fs = 310 KHz.
PROGRAM MEMORY: 16 words.

DSP56000/1 FFT MACRO – MINIMUM CODE


```
        Name: FFTR2A.ASM
        Type: Assembler Macro
     Version: 1.1
 Last Change:  2-Oct-86

 Description: Radix 2, In-place, Decimation-in-time Complex FFT Macro
```

This macro performs a complete Fast Fourier Transform (FFT) on complex data. The basic algorithm is the Decimation-in-time (DIT), Radix 2 FFT algorithm using 24 bit fixed-point arithmetic. The algorithm uses a sine-cosine lookup table for the FFT coefficients (twiddle factors). The macro can be called to perform any FFT from 2-32768 points. Simply call it with the arguments of number of FFT points, location of the data array and location of the sine-cosine table. All register initialization is performed by this macro. However, the macro assumes that registers which should not be altered by the FFT have already been saved by the main program. This allows the user to fit the FFT macro into his application and thus control the context switching overhead. No data scaling is performed and no overflow detection is done. Modifications to this routine could allow it to be used with the scaling modes and thus allow dynamic scaling for each FFT pass.

All data and coefficients are complex, with the real part in X Data memory and the imaginary part in Y Data memory. For an N point FFT, the data buffer requires N X Data and N Y Data memory locations. The algorithm is performed "in-place", meaning that only one data buffer is required for both input and output data. The input data is assumed to be in normal (time-sequential) order and the output is in bit-reversed order. By using the reverse-carry address modifier and a separate output data buffer, the output data may be easily unscrambled. Other methods also exist to unscramble the output data without a separate output data buffer. The FFTR2A macro uses "twiddle factors" (-cosine and -sine tables) stored in data memory. For maximum speed, the FFT macro performs a lookup table operation to get new sine and cosine values for each group of butterflies. A SINCOS macro is available to generate these tables. For an N point FFT, N/2 X Data and N/2 Y Data locations are required. Sine and cosine values could be calculated in real-time to save data memory at the expense of execution time.

The FFTR2A macro requires about 40 words of program memory for any size FFT from 2-32768 points. This is the most compact FFT routine available for the DSP56000. Additional details are included in the source file; however, more algorithm description would be required for complete understanding by typical users. A test program FFTR2AT is included in the library to demonstrate the FFTR2A calling procedure.

DSP56000/1 FFT MACRO — MINIMUM CODE

```
fi .2a.asm          Tue Mar 17 16:29:57 1987
            move    #points/2,n0        ;initialize butterflies per group
            move    #1,n2               ;initialize groups per pass
            move    #points/4,n6        ;initialize C pointer offset
            move    #-1,m0              ;initialize A and B address modifiers
            move    m0,m1               ;for linear addressing
            move    m0,m4
            move    m0,m5
            move    #0,m6               ;initialize C address modifier for
                                        ;reverse carry (bit-reversed) addressing
;
; Perform all FFT passes with triple nested DO loop
;
            do      #@cvi(@log(points)/@log(2)+0.5),_end_pass
            move    #data,r0            ;initialize A input pointer
            move    r0,r4               ;initialize A output pointer
            lua     (r0)+n0,r1          ;initialize B input pointer
            move    #coef,r6            ;initialize C input pointer
            lua     (r1)-,r5            ;initialize B output pointer
            move    n0,n1               ;initialize pointer offsets
            move    n0,n4
            move    n0,n5

            do      n2,_end_grp
            move    x:(r1),x1           y:(r6),y0       ;lookup -sine and
                                                        ; -cosine values
            move    x:(r5),a            y:(r0),b        ;preload data
            move    x:(r6)+n6,x0                ;update C pointer


            do      n0,_end_bfy
            mac     x1,y0,b                             y:(r1)+,y1  ;Radix 2 DIT
                                                                    ;butterfly kernel
            macr    -x0,y1,b            a,x:(r5)+       y:(r0),a
            subl    b,a                 x:(r0),b        b,y:(r4)
            mac     -x1,x0,b            x:(r0)+,a       a,y:(r5)
            macr    -y1,y0,b            x:(r1),x1
            subl    b,a                 b,x:(r4)+       y:(r0),b
_end_bfy
            move    a,x:(r5)+n5         y:(r1)+n1,y1    ;update A and B pointers
            move    x:(r0)+n0,x1        y:(r4)+n4,y1
_end_grp
            move    n0,b1
            lsr     b       n2,a1       ;divide butterflies per group by two
            lsl     a       b1,n0       ;multiply groups per pass by two
            move    a1,n2
_end_pass
            endm
```

EXECUTION TIME: For the "butterfly" loop, 6 instruction cycles.

THROUGHPUT RATE: For a 256 point complex FFT with internal program and data and a 20.5 MHz clock, 814 usec. For a 1024 point complex FFT with internal program and external data and a 20.5 MHz clock, 6.41 msec.

PROGRAM MEMORY: 42 words.

DSP56000/1 FFT MACRO – FAST 1024 Pt.


```
        Name: FFTR2E.ASM
        Type: Assembler Macro
     Version: 1.0
 Last Change:  4-Feb-87

 Description: 1024 Point, Non-In-place, Complex FFT Macro
```

This FFT macro performs a 1024 point complex FFT on external data using the Radix 2, Decimation in Time, Cooley-Tukey FFT algorithm [1][2]. The Cooley-Tukey algorithm is perhaps the simplest and most widely used form of FFT. Here we demonstrate several programming techniques and memory mapping strategies which take advantage of the DSP56000/1 architecture to perform very fast FFT's. It does not use straight line code to speed up FFT's but rather exploits the use of the dual internal data buses and memories and the ability of the external bus controller to perform one external (off-chip) access without adding any wait states.

FFTR2E starts by performing two Radix 2 passes on external data to partition the 1024 complex points into four sets of 256 complex points which can be processed independently. We take advantage of the fact that half of the FFT coefficients (twiddle factors) are zero for the first two passes to reduce the number of operations by one half. The remaining coefficients are +1 and -1, allowing add and subtract operations to replace coefficient lookup table accesses. Because of these facts, the output of the first pass can be kept internal to the Data ALU and only the output of the second pass is written back to memory. At this point all data is still external. Note that the coding of the first two passes is written with only one external access per instruction. Assuming the FFTR2E macro is in internal program memory, this guarantees that external data can be accessed at the same speed as internal data.

Each set of 256 complex points is then moved into the DSP56000/1 internal data memories and a Radix 2 FFT is performed on-chip. Thus the calculation is non-in-place since the internal data memories are used as a temporary workspace. The first pass of each 256 point FFT reads the external data input and writes the intermediate data into the internal memories. All intermediate passes work on internal data. However, the output data from the last pass of each 256 point FFT is written back to the external data memories. Thus the output of the 1024 point complex FFT overwrites the input data in external data memory.

DSP56000/1 FFT MACRO — FAST 1024 Pt.

The implementation uses 24 bit fixed point data storage and 24 bit fixed point FFT coefficients (sine and cosine lookup tables). The Data ALU maintains 56 bit accumulator precision whenever possible. All data is complex, with the real part in X Data memory and the imaginary part in Y Data memory. The external data buffer requires 1024 X Data and 1024 Y Data memory locations. The algorithm is performed "non-in-place", since the internal DSP56000/1 Data RAM's are used as additional workspace storage. The input data is assumed to be in normal (time-sequential) order and the output is in bit-reversed order. By using the reverse-carry address modifier and a separate output data buffer, the output data may be easily unscrambled. Other methods also exist to unscramble the output data without a separate output data buffer. For maximum speed, the FFT macro performs a lookup table operation to get new FFT coefficients (called twiddle factors) for each group of butterflies. The FFTR2E macro uses "twiddle factor" lookup tables (-cosine in X memory and -sine in Y memory) stored in external data memory. A SINCOS macro is available to generate these tables. The -sine table requires 512 X Data locations and the -cosine table requires 512 Y Data locations.

All register initialization is performed by this macro. However, the macro assumes that registers which should not be altered by the FFT have already been saved by the main program. This allows the user to fit the FFT macro into his application and thus control the context switching overhead. No data scaling is performed and no overflow detection is done. Modifications to this routine could allow it to be used with the scaling modes and thus allow dynamic scaling for each FFT pass.

The FFTR2E macro requires 104 words of program memory. Using a 20.5 MHz DSP56000/1, the FFTR2E macro can perform a 1024 point complex FFT in 3.39 milliseconds. Additional algorithm details are included in the source file; however, more algorithm description would be required for complete understanding by typical users. The test program FFTR2ET demonstrates the calling procedure for this macro.

References

[1] Cooley, J. W., and Tukey, J. W., "An algorithm for the machine calculation of complex Fourier series," Math. Comput., vol. 19, pp. 297-301, April 1965.

[2] Bergland, G. D., "A guided tour of the fast Fourier transform," IEEE Spectrum, vol. 6, pp. 41-52, July 1969, also in Digital Signal Processing, edited by L. R. Rabiner and C. M. Rader, IEEE Press, pp. 228-239, 1972.

DSP56000/1 FFT MACRO – FAST 1024 Pt.


```
;
; This program originally available on the Motorola DSP bulletin board.
; It is provided under a DISCLAMER OF WARRANTY available from
; Motorola DSP Operation, 6501 Wm. Cannon Drive W., Austin, Tx., 78735.
;
; 1024-Point, 3.39ms Non-In-Place FFT.
;
; Last Update 04 Feb 87   Version 1.0
;
fftr2e  macro   data,coef
fftr2e  ident   1,0
;
; 1024 Point Complex Fast Fourier Transform Routine
;
; This routine performs a 1024 point complex FFT on external data
; using the Radix 2, Decimation in Time, Cooley-Tukey FFT algorithm.
;
;    Complex input and output data
;        Real data in X memory
;        Imaginary data in Y memory
;    Normally ordered input data
;    Bit reversed output data
;       Coefficient lookup table
;        -Cosine values in X memory
;        -Sine values in Y memory
;
; Macro Call - fftr2e   data,coef
;
;       data        start of external data buffer
;       coef        start of sine/cosine table
;
; Radix 2, Decimation In Time Cooley-Tukey FFT algorithm
;
;                 ____________
;                |            |
; ar,ai ---->|  Radix 2   |----> ar',ai'
; br,bi ---->| Butterfly  |----> br',bi'
;                |____________|
;                     ^
;                     |
;                   wr,wi
;
;       ar' = ar + wr*br - wi*bi
;       ai' = ai + wi*br + wr*bi
;       br' = ar - wr*br + wi*bi = 2*ar - ar'
;       bi' = ai - wi*br - wr*bi = 2*ai - ai'
```

DSP56000/1 FFT MACRO – FAST 1024 Pt.

```
;
; Address pointers are organized as follows:
;
;  r0 = ar,ai input pointer      n0 = group offset          m0 = modulo (points)
;  r1 = br,bi input pointer      n1 = group offset          m1 = modulo (points)
;  r2 = ext. data base address   n2 = groups per pass       m2 = 256 pt fft counter
;  r3 = coef. offset each pass   n3 = coefficient base addr.     m3 = linear
;  r4 = ar',ai' output pointer   n4 = group offset          m4 = modulo (points)
;  r5 = br',bi' output pointer   n5 = group offset          m5 = modulo (points)
;  r6 = wr,wi input pointer      n6 = coef. offset          m6 = bit reversed
;  r7 = not used (*)             n7 = not used (*)          m7 = not used (*)
;
;       * - r7, n7 and m7 are typically reserved for a user stack pointer.
;
; Alters Data ALU Registers
;       x1      x0      y1      y0
;       a2      a1      a0      a
;       b2      b1      b0      b
;
; Alters Address Registers

;       r0      n0      m0
;       r1      n1      m1
;       r2      n2      m2
;       r3      n3      m3
;       r4      n4      m4
;       r5      n5      m5
;       r6      n6      m6
;
; Alters Program Control Registers
;       pc      sr
;
; Uses 8 locations on System Stack
;
; Latest Revision - 4-Feb-87
;
_points equ     1024            ;number of FFT points
_intdata        equ     0       ;address of internal data workspace
        move    #data,r2        ;initialize input pointers
        move    r2,r0
        move    #_points/4,n0   ;initialize butterflies per group
        move    n0,n4           ;initialize pointer offsets
        move    n0,n6           ;initialize coefficient offset
        move    #coef,n3        ;initialize coefficient base address
        move    #_points-1,m0   ;initialize address modifiers
        move    m0,m1           ;for modulo(points) addressing
        move    m0,m4
        move    m0,m5
        move    #-1,m3          ;linear addressing for coefficient base offset
        move    #0,m2           ;initialize 256 point fft counter
        move    m2,m6           ;initialize coefficient address modifier
                                ;for reverse carry (bit reversed) addressing
```

DSP56000/1 FFT MACRO — FAST 1024 Pt.

```
;
; Do first and second Radix 2 FFT passes
;
        move                x:(r0)+n0,x0
        tfr     x0,a        x:(r0)+n0,y1

        do      n0,_twopass
        tfr     y1,b        x:(r0)+n0,y0
        add     y0,a        x:(r0),x1                       ;ar+cr
        add     x1,b        r0,r4                           ;br+dr
        add     a,b         (r0)+n0                         ;ar'=(ar+cr)+(br+dr)
        subl    b,a         b,x:(r0)+n0                     ;br'=(ar+cr)-(br+dr)
        tfr     x0,a        a,x0            y:(r0),b
        sub     y0,a                        y:(r4)+n4,y0    ;ar-cr
        sub     y0,b        x0,x:(r0)                       ;bi-di
        add     a,b                         y:(r0)+n0,x0    ;cr'=(ar-cr)+(bi-di)
        subl    b,a         b,x:(r0)                        ;dr'=(ar-cr)-(bi-di)
        tfr     x0,a        a,x0            y:(r4),b
        add     y0,a                        y:(r0)+n0,y0    ;bi+di
        add     y0,b        x0,x:(r0)+n0                    ;ai+ci
        add     b,a                         y:(r0)+,x0      ;ai'=(ai+ci)+(bi+di)
        subl    a,b                         a,y:(r4)+n4     ;bi'=(ai+ci)-(bi+di)
        tfr     x0,a                        b,y:(r4)+n4
        sub     y0,a        x1,b                            ;ai-ci
        sub     y1,b        x:(r0)+n0,x0                    ;dr-br
        add     a,b         x:(r0)+n0,y1                    ;ci'=(ai-ci)+(dr-br)
        subl    b,a                         b,y:(r4)+n4     ;di'=(ai-ci)-(dr-br)
        tfr     x0,a                        a,y:(r4)+
_twopass
;
; Do remaining 8 FFT passes as four 256 point Radix 2 FFT's using internal data
; and external coefficients.
;
; Each 256 point Radix 2 FFT consists of 8 passes.  The first pass uses external
; input data and internal output data to move the data on-chip.  Intermediate
; passes use internal input data and internal output data to keep the data
; on-chip.  The last pass uses internal input data and external output data
; to move the data off-chip.
;
        do      #4,_end_fft         ;do 256 point fft four times
        move    r2,r0               ;get external data input address for first pass
        move    m2,r3               ;update coefficient offset
        move    #256/2,n0           ;initialize butterflies per group
        move    #1,n2               ;initialize groups per pass

        do      #7,_end_pass        ;do first 7 passes out of 8
        move    #_intdata,r4        ;initialize A output pointer
        move    n0,n1               ;initialize pointer offsets
        move    n0,n4
        move    n0,n5
        lua     (r0)+n0,r1          ;initialize B input pointer
        lua     (r4)+n4,r5          ;initialize B output pointer
        lua     (r3)+n3,r6          ;initialize W input pointer
        move    (r5)-

        do      n2,_end_grp
        move    x:(r1),x1           y:(r6),y0       ;lookup -sine value
        move    x:(r5),a            y:(r0),b
        move    x:(r6)+n6,x0                ;lookup -cosine value
```

DSP56000/1 FFT MACRO — FAST 1024 Pt.

```
        do      n0,_end_bfy     ;Radix 2 DIT butterfly kernel with constant
                                                y:(r1)+,y1   ;twiddle factor
        mac     x1,y0,b
        macr    -x0,y1,b        a,x:(r5)+       y:(r0),a
        subl    b,a             x:(r0),b        b,y:(r4)
        mac     -x1,x0,b        x:(r0)+,a       a,y:(r5)
        macr    -y1,y0,b        x:(r1),x1
        subl    b,a             b,x:(r4)+       y:(r0),b
_end_bfy
        move    a,x:(r5)+n5     y:(r1)+n1,y1    ;dummy load of x1 and y1
        move    x:(r0)+n0,x1    y:(r4)+n4,y1
_end_grp
        move    n0,b1
        lsr     b       n2,a1   ;divide butterflies per group by two
        lsl     a       b1,n0   ;multiply groups per pass by two
        move            r3,b1
        lsr     b       a1,n2   ;divide coefficient offset by two
        move            b1,r3
        move            #_intdata,r0    ;intermediate passes use internal input dat
_end_pass
;
; Do last FFT pass and move output data off-chip to external data memory.
;
        move    n1,n0           ;correct pointer offset for last pass
        move    r2,r4           ;initialize A output pointer
        lua     (r0)+,r1        ;initialize B input pointer
        lua     (r4)-n4,r5      ;initialize B output pointer
        lua     (r3)+n3,r6      ;initialize W input pointer
        move    (r5)+
        move    x:(r1),x1       y:(r6),y0
        move    x:(r5),a        y:(r0),b

        do      n2,_lastpass    ;Radix 2 butterfly kernel with one butterfly
                                ;per group and changing twiddle factor
        mac     x1,y0,b         x:(r6)+n6,x0    y:(r1)+n1,y1
        macr    -x0,y1,b        a,x:(r5)+n5     y:(r0),a
        subl    b,a             x:(r0),b        b,y:(r4)
        mac     -x1,x0,b        x:(r0)+n0,a     a,y:(r5)
        macr    -y1,y0,b        x:(r1),x1       y:(r6),y0
        subl    b,a             b,x:(r4)+n4     y:(r0),b
_lastpass
        move    a,x:(r5)+n5
        move    m2,r6           ;get fft counter
        move    n6,n2           ;get fft data input offset
        move    m3,m2           ;external data pointer uses linear arithmetic
        move    (r6)+n6         ;increment fft counter (bit reversed)
        move    (r2)+n2         ;point to next 256 point fft input data
        move    r6,m2           ;save fft counter
_end_fft
        endm
```

ORIGINAL PAGE IS OF POOR QUALITY

DSP56000/1 FLOATING POINT LIBRARY (FPLIB)

* PERFORMANCE (assumes 20.5 MHz.)

Function	Best Case		Worst Case		Comment
	(I_{cyc})	(us)	(I_{cyc})	(us)	
R = A + X	12	1.17	75	7.32	
R = Y + X	13	1.27	76	7.41	
R = A − X	11	1.07	71	6.93	
R = Y − X	12	1.17	72	7.02	
A − X	9	0.88	11	1.07	Compare; used to set condition codes: =, >, ≥, <, ≤
Y − X	10	0.98	12	1.17	
R = A * X	18	1.76	63	6.15	
R = Y * X	19	1.85	64	6.24	
R = A + X * Y	39	3.80	147	14.34	
R = A − X * Y	38	3.71	143	13.95	
R = A / X	12	1.17	109	10.63	
R = Y / X	13	1.27	110	10.73	
$R = A * 2^{eX}$	12	1.17	15	1.46	Scale A by the exponent of X
$R = Y * 2^{eX}$	13	1.27	16	1.56	Scale Y by the exponent of X
$R = \sqrt{A}$	7	0.68	158	15.41	
$R = \sqrt{X}$	8	0,78	159	15.51	
R = − A	16	1.56	61	5.95	
R = − X	17	1.66	62	6.05	
R = | A |	5	0.49	64	6.24	
R = | X |	6	0.59	65	6.34	
Fix A −> Float A	5	0.49	27	2.63	
Fix X −> Float X	6	0.59	28	2.73	
Float A −> Fix A	11	1.07	83	8.10	
Float X −> Fix X	12	1.17	84	8.20	

DSP56000/1 COMPETITIVE ANALYSIS

* ON – CHIP RESOURCES – TI

Resource	TI 32010	TI 32020	TI 320C25	Motorola 56000/1
INSTRUCTION RATE (NSEC)	200	200	100	97.5
DATA SIZE (BITS)	16	16	16	24
PROGRAM SIZE (BITS)	16	16	16	24
DATA RAM SIZE	144	544(256 can be Prog.)	544	2 x 256
DATA ROM SIZE	none	none	none	2 x 256
PROGRAM MEMORY SIZE	1536 ROM	none	4096 ROM	3840 ROM or 512 RAM
PROGRAM STACK DEPTH	4	4	8	2 x 15
MULTIPLIER SIZE	16 x 16	16 x 16	16 x 16	24 x 24
MULTIPLIER INPUT REGS.	1	1	1	4
ALU SIZE (BITS)	32	32	32	56
ACCUMULATOR REGS.	1 x 32	1 x 32	1 x 32	2 x 56
ADDRESS REGS.	2	5	8	24
MEMORY EXPANSION	4 K	128 K	128 K	192 K
INTERRUPT SOURCES	1	7	7	18
ON – CHIP PERIPHERALS	none	1S	1S	2S,1P
PERIPHERAL REGISTERS	none	6	6	32
I/O FLAGS	1	1	1	24
PINS	40	68	68	88
TECHNOLOGY	2.7u NMOS	2.4u NMOS	1.8u CMOS	1.5u CMOS
MAX. POWER DISSIP.	1.4 W	1.8 W	1 W	0.5 W typical @ 20.5 MHz.

DSP56000/1 COMPETITIVE ANALYSIS

*** BENCHMARKS – TI**

NOTE: (S) = STRAIGHT – LINE CODE
(L) = LOOPED CODE

Benchmark	TI 32010	TI 32020	TI 320C25	Motorola 56000/1
Real FIR Filter with Data Shift	0.4 usec(S) 0.8 usec(L) per tap(1)	0.2 usec(L) per tap (4)	0.1 usec(L) per tap (Int. Data only)(7)	0.1 usec(L) per tap
IIR Biquad Filter	2.0 usec(S) (2)	2.0 usec(S) (4)	1.0 usec(S) (7)	0.4 usec(L)
[1x3][3x3] Matrix Multiplication	5.4 usec(S) (1)	5.4 usec(S) (1)	2.7 usec(S) (7)	1.7 usec(L)
256 pt Complex FFT	4100 usec(S) (3)	2440 usec(S) (5)	1500 usec (S) 3500 usec (L) (7)	713 usec (L)
1024 pt Complex FFT	31000 usec(S) (3)	14180 usec(S) (5)	7100 usec(S)	3391 usec(L)
LMS Adaptive Real FIR Filter	?	1.2 usec(S) per tap (6),(8)	0.4 usec(S) 0.6 usec(L) per tap(6),(8)	0.3 usec(L) per tap

References

(1) Digital Signal Processing Applications with the TMS320 Family, 1986
(2) TMS320(10) Product Description, TI SPRV001
(3) Details on Signal Processing, Issue 5, June 1985, TI SPRN005
(4) TMS32020 Product Description, TI SPRV002
(5) Details on Signal Processing, Issue 6, November 1985, TI SPRN006
(6) Electronics Magazine article, Feb. 3, 1986, pg. 23
(7) TMS320C25 Product Description, TI SPRV006
(8) TI FYI Newsletter, Vol. 3, #2, Feb. '86.

DSP56000/1

DEVELOPMENT TOOLS

* Macro Cross Assembler
* Software Simulator
* Applications Development System

DSP56000/1 MACRO ASSEMBLER

* Two Pass Absolute Assembler
* Produces Object Code Compatible with DSP56000/1 Software Simulator
* Runs Under:MS–DOS, UNIX[4.2BSD], VMS[V4.2]
* Offers 6 Command Line Options: MAC II
  –D, –L, –M, –B, –I, –O
* Supports Nested Macros
* Provides Modular Programming Constructs:
  Logical Code Sections
  Structured Control Statements
* Provides 23 Powerful Utility Functions For:
  Data Conversions from Floating Point to Integer and Valid Fractions
  String Comparison
  Common Transcendental Math Operations
* Provides 41 Directives Including:
  Define Modulo Buffer (Base Address, Range Check)
  Define Bit–Reverse Buffer (Power of 2)
* Supports Conditional Assembly
* Supports Include Files
* Provides Extensive Error Checking and Error Messages:
  Syntax and Source Field Consistency Checks
  Instruction Sequence and DO Loop Restriction Checks
  Range Checking on Buffer Allocation and Floating Point to Fractional Conversions

DSP56000/1 SIMULATOR

* Simulates Operation of the DSP56000/1
* Simulates on a CLOCK CYCLE Basis
  - Provides Clock and Instruction Cycle Counts
* Runs Under MS-DOS, UNIX[4.2BSD], VMS[V4.2]
* Simulates I/O Including ON-CHIP Peripherals
* Conditional or Unconditional Breakpoints
* Single Stepping Through Object Code
* Program Patching Using Single-Line Assembler / Disassembler

  Macro Command Definition and Execution
* Loading and Saving of Simulator Files
* Session and/or Command Logging
* USER FRIENDLY:
  - Will display all registers in the programmers model format
  - Will highlight those registers which have changed as a result of executing the last instruction
  - Will optionally display only the registers and memory which changed
  - Provides HELP LINE and HELP FILE
  - Will display the last 100 lines of display so the programmer can scroll back through previous breakpoint displays and view the results of a sequence of instructions. Limiting the number of displayed registers allows more instruction traces to be displayed within the 100 line buffer.

* MULTIPLE DSP'S

DSP56000/1 SIMULATOR (1.11)

* COMMAND MENUs

1] asm break change copy disassemble display <space> = more

2] evaluate go help history input load log output path

3] quit radix reset save step system trace wait

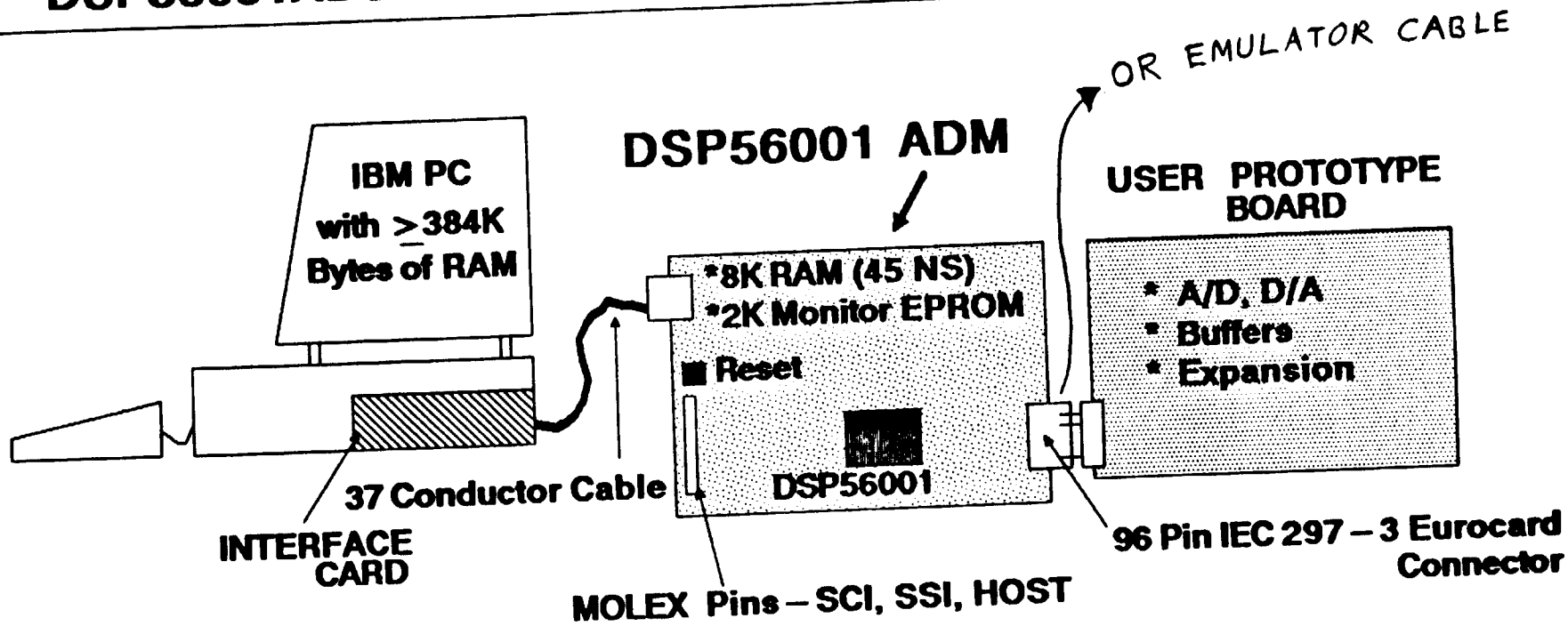
4] ^U = ln-up ^N = ln-dn ^T = pg-up ^V = pg-dn ^R = right ^L = left

5] ^H = del-lft ^K = del-rt ^O = overwrite/insert ESC = clear command line

6] ^I = tab-rt ^E = eol ^Z = del-eol CR = execute ? = more help

DSP56001 APPLICATION DEVELOPMENT SYSTEM

* DSP56001ADS HOOKUP TO USER PROTOTYPE BOARD



COMMANDS:

Memory/Register Modification

Assemble
Change
Copy
Display
Disassemble

File I/O

Input
Output
Load
Log
Save

Execution Control

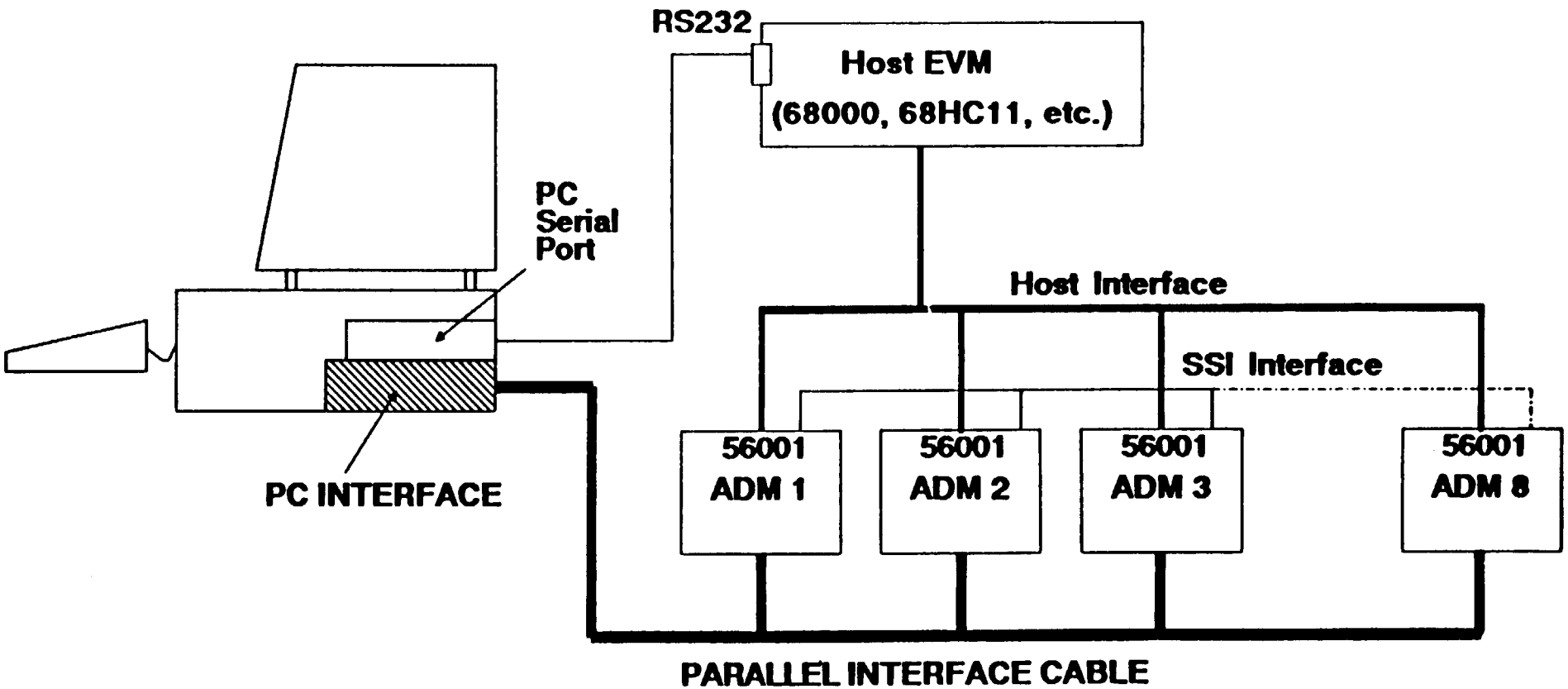
Break
Go
Force
Step
Trace

Miscellaneous

Evaluate
Device
Quit
Help
Radix
System
Path
PC

DSP56001 APPLICATION DEVELOPMENT SYSTEM

* MULTIPLE ADM HOOKUP



MOTOROLA
SEMICONDUCTOR
TECHNICAL DATA

DSP56KCCx

LOANER

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.

This document contains information on a new product. Specifications and information herein are subject to change without notice.

MOTOROLA

©MOTOROLA INC., 1987

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.

This document contains information on a new product. Specifications and information herein are subject to change without notice.

© MOTOROLA INC., 1987

MOTOROLA

DSP56KPAK/D

DIGITAL SIGNAL PROCESSOR 56000 FAMILY INFORMATION PACKAGE

DSP56000 Digital Signal Processor User's Manual	DSP56000UM/AD
Motorola's Sizzling New Signal Processor	AR237/D
DSP56001 Advance Information	DSP56001/D
DSP Competitive Analysis	
DSP56000CLASx Software Summary	BR526/D
DSP56KCCx Software Summary	BR541/D
DSP56000ADS	BR517/D
DSP320to56001 Software Summary	BR522/D
Digital Sine-Wave Synthesis Using the DSP56001	APR1/D
* Motorola/Tektronix Flyer	
* Momentum Flyer	
DSP56200 Advance Information	DSP56200/D
DSP56200 Technical Summary	BR283/D
Dr. Bub	BR297/D

FLOATING POINT MILESTONE:
AMD, TI, AND WEITEK LAUNCH FIRST
SINGLE-CHIP, IEEE-STANDARD PROCESSORS/77

Electronics®

MARCH 17, 1988

THE EMBEDDED PROCESSOR BREAKS OUT OF ITS NICHE

PAGE 61

ORIGINAL PAGE IS
OF POOR QUALITY

**********5-DIGIT 01801
JUL88 S01
MOTOROLA INC
300 INICORN PARK DR
WOBURN MA 01801

MOTOROLA'S NEW DSP ENGINE COMES WITH FLOATING POINT/64
NATIONAL ALTERS ITS CISC CHIP TO DO EMBEDDED CONTROL/66

MOTOROLA'S NEW DSP ENGINE COMES WITH FLOATING POINT

The company hangs on to the record for fastest single-chip DSPs—its latest chip runs at 13.3 mips

In the drive to push embedded processors beyond their traditional applications, Motorola Inc. is launching two versions of a powerful new chip. The DSP96001 and DSP96002 promise to smash the speed records set by the Schaumberg, Ill., company's own DSP56000 single-chip digital signal processor two years ago. With that kind of performance, the new device can compete for the same high-end embedded control and processing applications that makers of specialized reduced-instruction-set computers and complex-instruction-set chips are targeting.

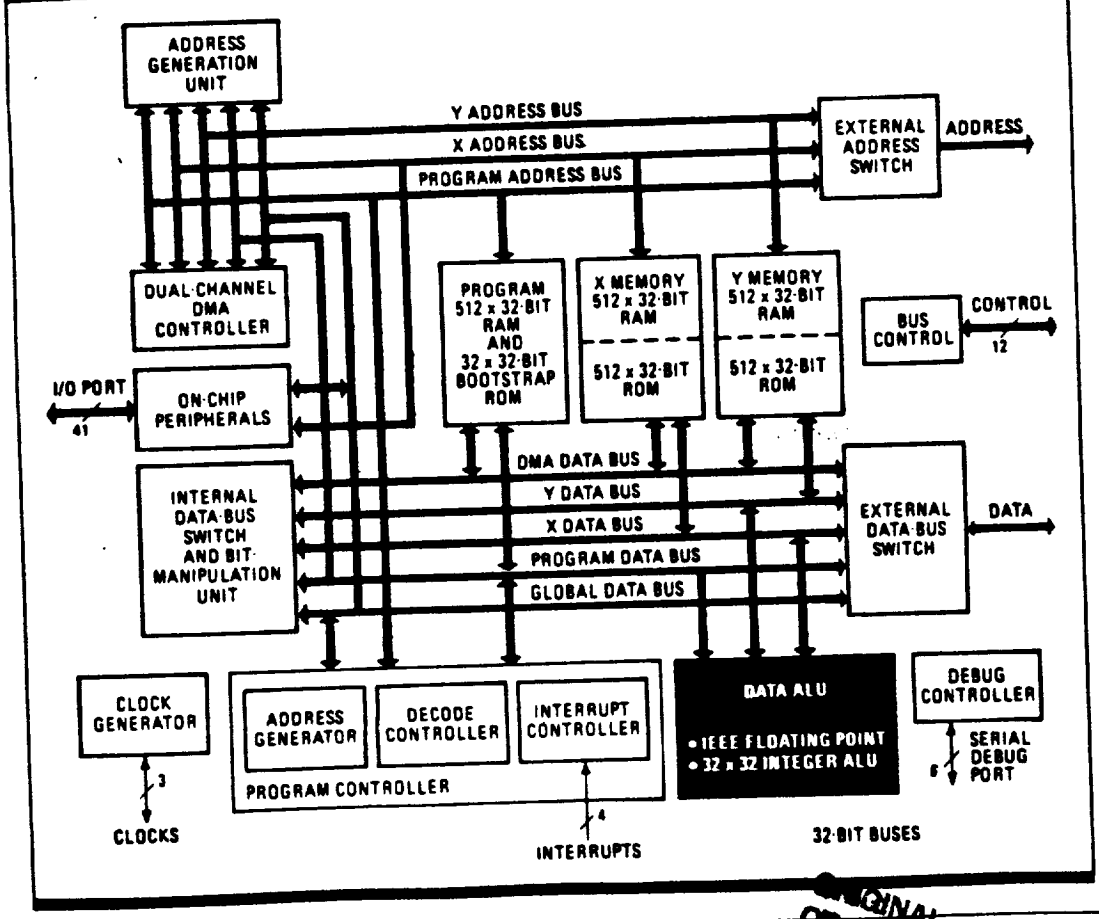
The DSP96001/2 keeps the speed record for single-chip DSPs set by its predecessor, Motorola claims, running at 13.3 million instructions/s compared with 10.25 mips for the DSP56000 [*Electronics*, March 10, 1986, p. 30]. Sample quantities of the DSP96001/2 are due by the fourth quarter.

If initial performance specifications are any indication, the new DSP chip, which Motorola is calling an "attached processor," will give competitors more than a run for their money. At 13.3 mips, the 161-pin device can execute a 1,034-point complex fast Fourier transform in less than two milliseconds, a full 30% to 35% faster than its predecessor. Featuring a peak performance of 40 million floating-point operations/s with an instruction execution cycle time of only 75 ns, the DSP96001 can execute 66.66 million addresses/s external, with an internal peak performance of 256.66 Mbytes/s. On-chip functions include a host central processing unit and serial and synchronous-serial interfaces. On the DSP96002, the on-chip peripherals are removed and a bus controller and a second extended address switch added.

The performance improvement comes from revamping the architecture (see fig. 1)—up to a full 32-bit implementation—expanding the data arithmetic logic unit's register set from a 56- to a 96-bit-wide word configuration. On-chip random-access and read-only memories have also been expanded. An improved host interface and a dual-channel, direct-memory-access controller to support the higher bandwidth have been added. In addition, an on-chip barrel shifter and a 32-by-32 multiplier/adder are now incorporated on-chip. And the basic 56000 instruction set has been enhanced to implement fully the IEEE 754-1985 standard for binary single-precision and extended single-precision floating-point arithmetic.

The DSP96001 outdoes the DSP56000 when performing many DSP benchmarks, says Bryant Wilder, manager of digital signal processing operations in Austin, Texas. For example, he says, a finite impulse response filter with data shift is per-

FLOATING POINT ON AN EMBEDDED PROCESSOR



1. The DSP96001/2 hits a performance rate of 40 megaflops by increasing the number of data buses, adding a dual-channel DMA controller, and incorporating a multiplier and barrel shifter into its data ALU.

PRECEDING PAGE BLANK NOT FILMED

ORIGINAL PAGE IS OF POOR QUALITY

formed in 0.075 μs per tap, 25% faster than the DSP56000 and 4 to 10 times faster than competitive single-chip DSP circuits.

The most significant modification to the architecture has been in the register structure of the data ALU portion, says Kevin Kloker, principal staff engineer at the systems research laboratory in Schaumberg. In the earlier unit, the data ALU contained four 24-bit general purpose input registers which could be treated as four independent 24-bit registers or as two 48-bit registers, and six accumulator registers, four of which were 24 bits wide and two that were 8 bits wide.

To implement the IEEE 754-1985 standard for single-precision and extended single-precision binary floating-point arithmetic, the data ALU in the 96001/2 is organized as one file of ten 96-bit general-purpose registers that can also be addressed as thirty 32-bit registers. Also added into the ALU for the same purpose is a 32-bit barrel shifter, 32-bit adder, and 32-bit parallel multiplier. The width of the register file means the data ALU can do any mix of multiplication, addition, subtraction, format conversion, shifting, and logical operations in one instruction cycle. Multiplication with positive or negative accumulation takes no more than two cycles.

To boost overall performance over the DSP56000—by up to 35%—to 13 to 14 mips, Motorola designers completely revamped the internal architecture on the DSP96001/2. All of the buses, internal and external, have been expanded to 32 bits—the DSP56000 had only three internal and one external address buses, which were 16 bits wide, and internal and external data buses, which were 24 bits wide. The total number of internal buses has been upped as well, from four on the DSP56000 to five on the DSP96001/2. In addition to the bidirectional x and y data buses—the program and global data buses—the chip also has an additional bidirectional bus to accommodate the on-chip dual channel DMA controller. To give system designers even more flexibility, the DSP96001/2 bus structure is designed so that under certain conditions and with certain instructions, the two 32-bit x and y data buses can function as a single 64-bit-wide bidirectional bus.

Also, says Wilder, the highly parallel architecture of the DSP56000 has been enhanced in the DSP96001/2. The data ALU, address arithmetic units, and program controller operate in parallel within the CPU so that an instruction prefetch, up to three floating-point operations, two data moves, and two address pointer updates can be executed in a single instruction cycle. These functions may use any of three types of arithmetic—linear, modular, or reverse carry. "I think you will find it hard to dig up any RISC controller, much less any CISC machine, that can do anything comparable," says Kloker.

A dual-channel DMA controller on the chip operates in parallel with the CPU. The DMA controller performs all the address storage and effective address calculations that are necessary to address the DMA source and destination-memory locations on two channels.

To support its use in embedded control applications, the DSP96001/2 also incorporates on-chip RAM and ROM. Both the x- and y-data RAMs contain 32 by 512 bits each, versus 24 by 256 bits on the DSP56000. On-chip program RAM space has also been expanded, from 512 by 24 bits to 512 by 32 bits. In addition, the on-board x and y data bootstrap ROM space has been expanded, from 32 by 24 bits to 32 by 32 bits each.

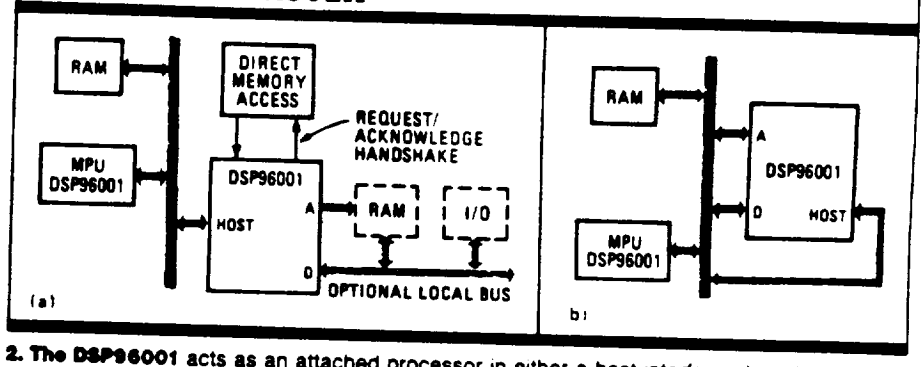
Just as Motorola offered the 56-bit-word-wide DSP56000 as a single-chip alternative to traditional 8- and 16-bit microcontrollers such as Intel Corp.'s 8051 and 8096 in real-time event-control applications requiring digital signal processing, a similar strategy is being taken with the DSP96001/2 in the high-end embedded processor market. It is aimed at applications such as graphics and image processing and robotics as well as the more traditional DSP applications—array processing, digital audio, instrumentation, medical equipment, and speech processing. Indeed, says Wilder, the DSP96001/2 should be a real contender on two counts. "Unlike most of the other offerings, we have not given up on the single-chip solution [but instead offer] a single device which incorporates not only the CPU and some RAM and ROM, but program control and floating-point processing as well."

A second benefit is architectural flexibility. If, for example, the system designer wants to embed the additional processing functions within the host system itself, the DSP96001 would be used, configured as a host interface slave (see fig. 2a) using its own internal RAM resources or additional RAM and input/output resources via an optional local bus. It could also be used as a bus interface master where it accesses the data and programs in the system RAM (see fig. 2b). Where the system designer wants to embed the additional processing in a peripheral, such as a printer, rather than the host, the DSP96002 would be used with additional RAM and ROM linking to the host CPU via some external bus or network connection.

–Bernard C. Cole

For more information, circle 480 on the reader service card.

SLAVE AND MASTER



2. The DSP96001 acts as an attached processor in either a host interface slave (a) or a bus interface master (b) configuration, sharing the system memory.

TECHNOLOGY TO WATCH is a regular feature of *Electronics* that provides readers with exclusive, in-depth reports on important technical innovations from companies around the world. It covers significant technology, processes, and developments incorporated in major new products.

ORIGINAL PAGE IS OF POOR QUALITY

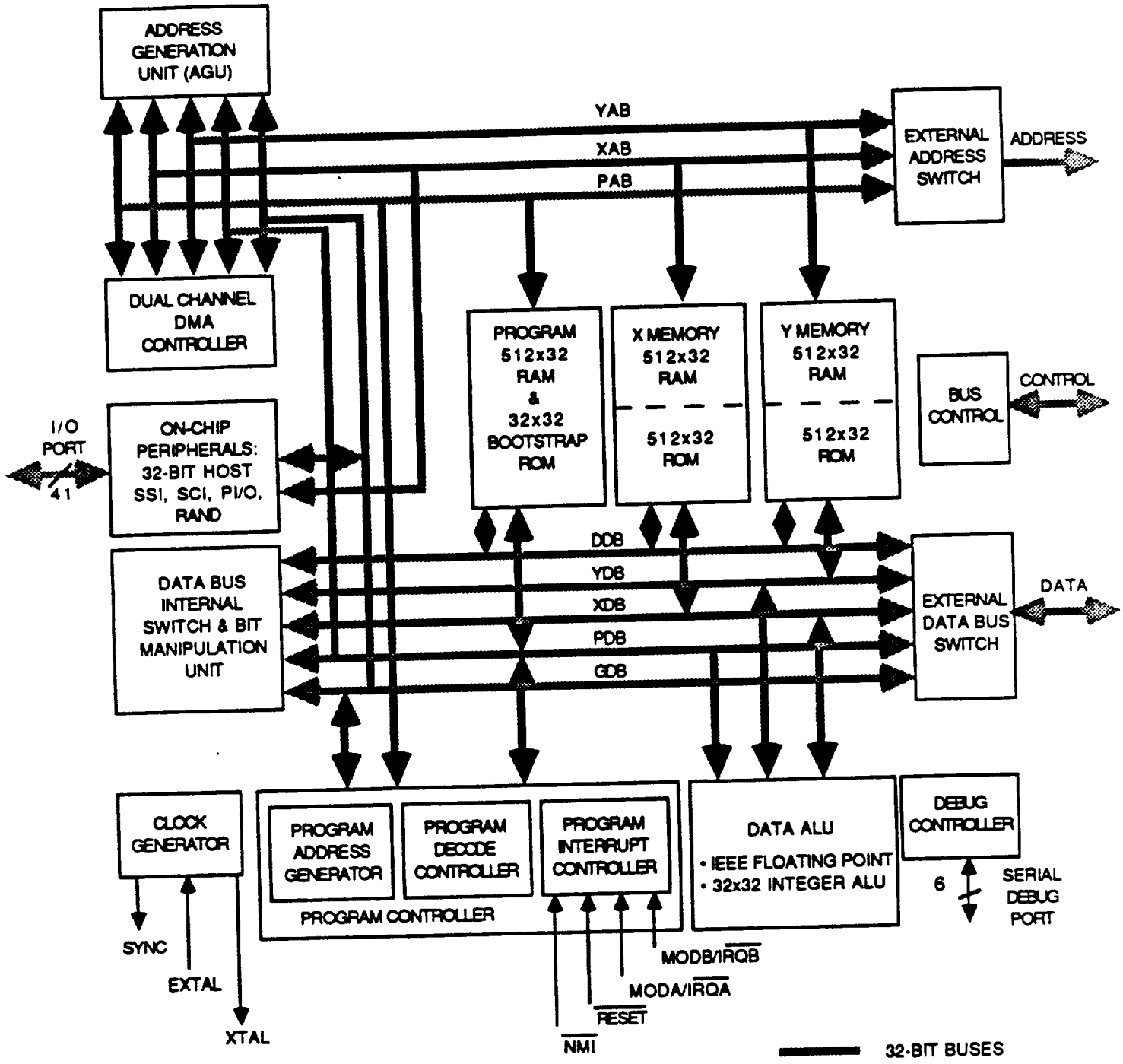


Figure 2. DSP96001 Block Diagram

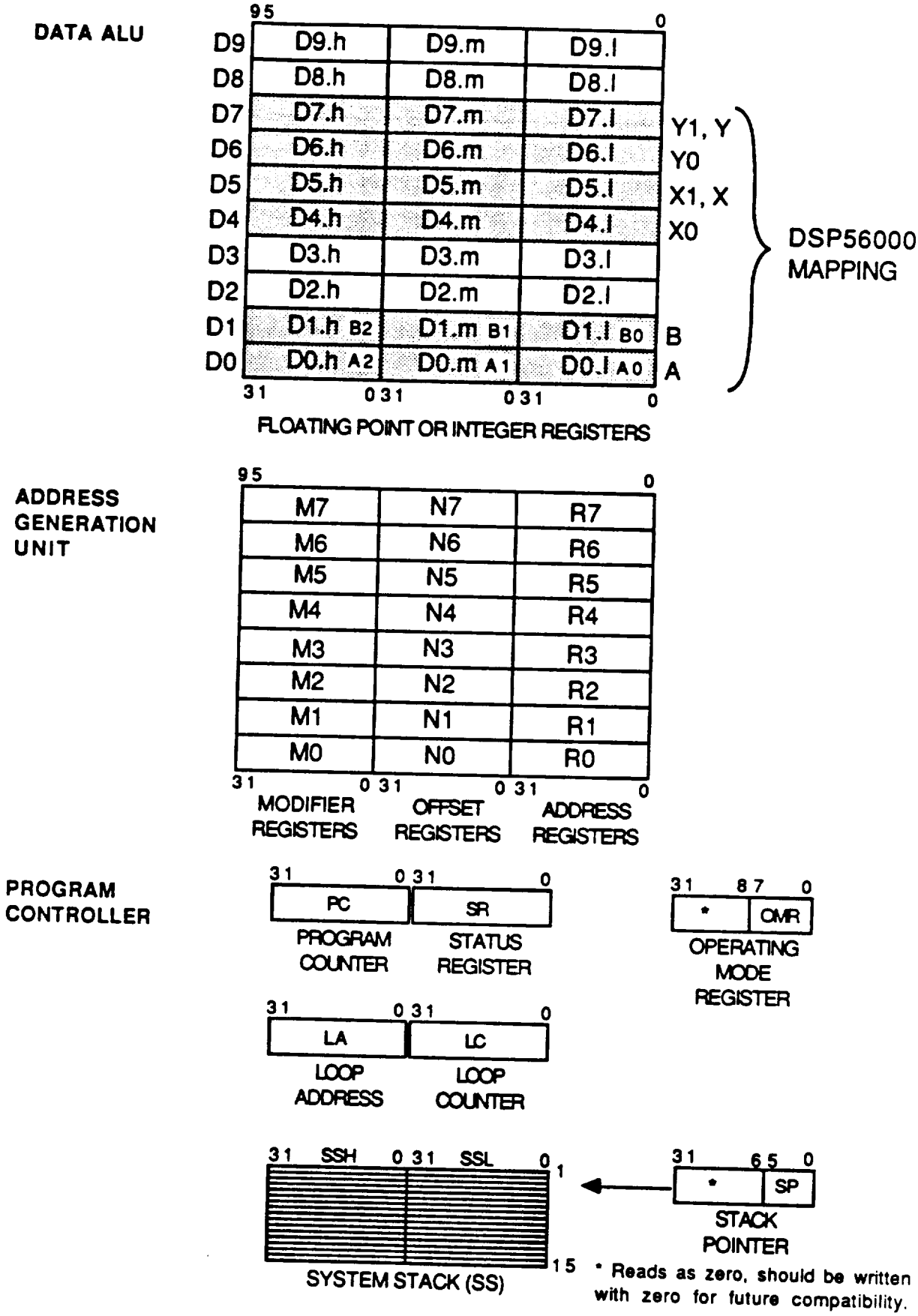


Figure 5. User Programming Model

Table 5. Address Modes Summary

Addressing Mode	Modifier MMMM	Memory/Registers Referenced								
		S	C	D	A	P	X	Y	L	XY
Register Direct										
Data or Control Register	No	x	x	x						
Address Register	No				x					
Address Modifier Register	No				x					
Address Offset Register	No				x					
Address Register Indirect										
No Update	Yes					x	x	x	x	x
Postincrement by 1	Yes					x	x	x	x	x
Postdecrement by 1	Yes					x	x	x	x	x
Postincrement by Offset Nn	Yes					x	x	x	x	x
Postdecrement by Offset Nn	Yes					x	x	x	x	
Indexed by Offset Nn	Yes					x	x	x	x	
Indexed by Displacement	Yes						x	x	x	
Predecrement by 1	Yes					x	x	x	x	
Special										
Immediate Data	No					x				
Absolute Address	No					x	x	x	x	
Immediate Short Data	No					x				
Short Jump Address	No					x				
PC Relative	No					x				
I/O Short Address	No						x	x		
Implicit	No	x	x			x				

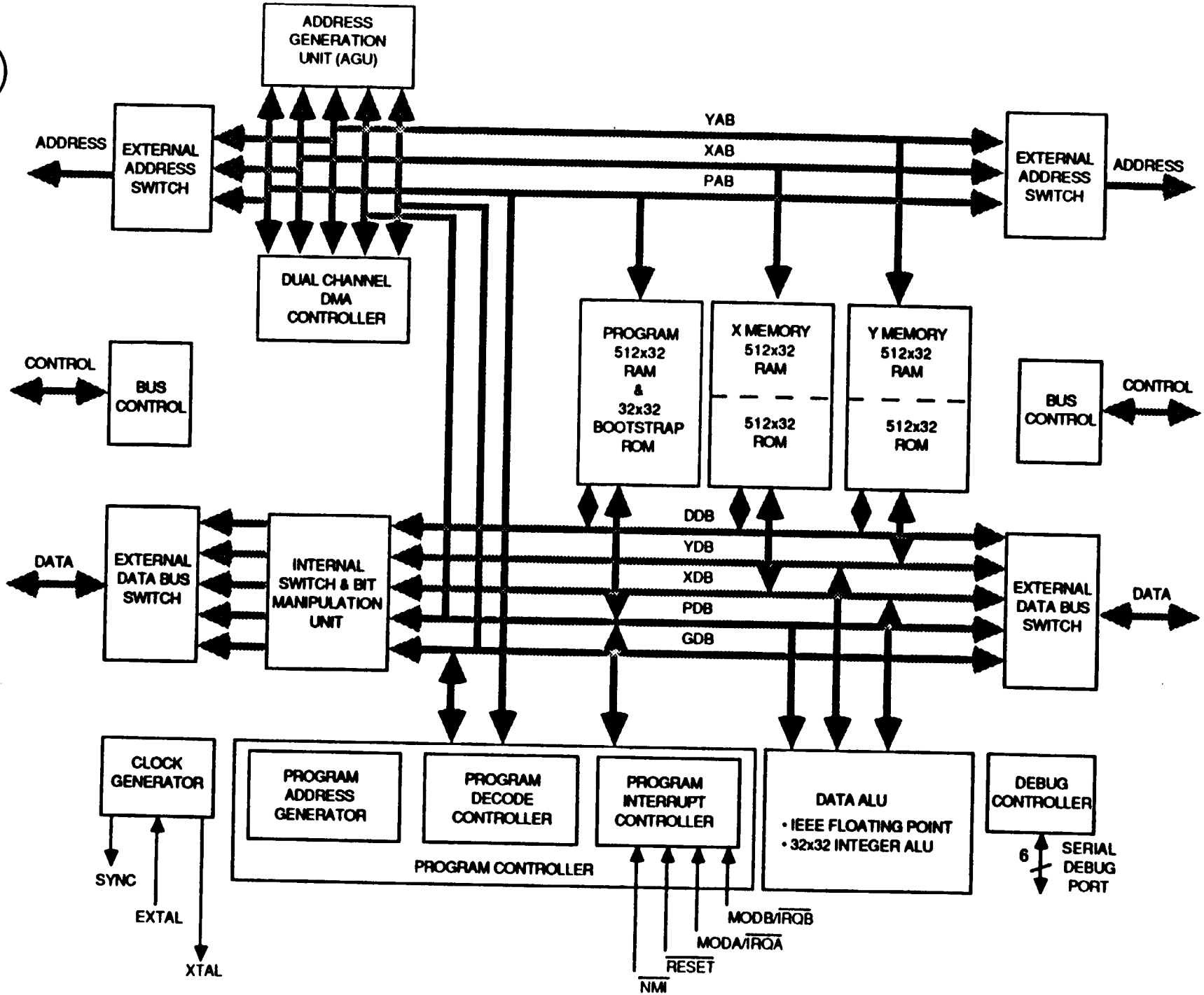


Figure 2. DSP96002 Block Diagram

MOTOROLA
SEMICONDUCTOR
TECHNICAL DATA

DSP56200

Technical Summary

Cascadable Adaptive Finite Impulse Response Digital Filter (CAFIR)

This document provides a technical summary of the DSP56200, a Cascadable Adaptive Finite Impulse Response (CAFIR) digital filter chip. For users not familiar with digital filtering, an appendix on this topic is also included.

The DSP56200 is an algorithm specific, digital signal processing (DSP) peripheral designed to perform computationally-intensive tasks associated with digital filtering. Two principal functions are performed by the DSP56200 — FIR filtering and adaptive FIR filtering using the Least-Mean-Square (LMS) algorithm. A flexible chip-cascading scheme enables the user to easily 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. Key features of the DSP56200 are:

- Three Modes of Operation
  - Single FIR Filter
  - Dual FIR Filter (Two Independent FIR Filters)
  - Single Adaptive FIR Filter
- High Performance Hardware
  - 24 x 16 Bit Multiplication with 40-Bit Accumulation
  - 10.25 MHz Internal Operation
  - Single-Cycle Multiply-Accumulate
  - Single-Cycle Update of a Coefficient
  - Ultra-Low-Power Standby Mode
  - 256 x 24-bit Coefficient RAM
  - 256 x 16-bit Data RAM
  - Unused RAM Available for System Storage
  - 28-Pin Dual-in-Line Package
- Architecture Optimized for Digital Filtering
  - Three Execution Units Operate in Parallel
  - Multiple Internal Buses
- Digital Filtering Features
  - 16-Bit Rounding Option on Filtered Output
  - dc Tap Option
  - Programmable Filter Tap Length (4 to 256 Taps)
  - Programmable Loop Gain[1]
  - Programmable Coefficient Leakage Term[1]
  - Adaptation Disable Capability[1]
  - LMS Adaptation Algorithm[1]
- Unlimited Cascadability Offers[2]
  - Longer-Tap Filters
  - Higher Sampling Rates
- High Throughput Rates — Examples of Possible Configurations
  - 227 kHz FIR Filter (32 Taps, 1 DSP56200)
  - 37 kHz FIR Filter (256 Taps, 1 DSP56200)
  - 115 kHz Adaptive Filter (256 Taps, 8 Cascaded DSP56200s)
  - 19 kHz Adaptive Filter (256 Taps, 1 DSP56200)
  - Many Other Configurations Possible
- Simple Interface to Popular Hosts
  - Microprocessors
  - Microcomputers
  - General Purpose Digital Signal Processors

[1]Options unique to adaptive filtering mode
[2]The DSP56200 implements a true cascade. A true cascade means that the error term in the adaptive filtering mode is calculated from the partial sums of ***ALL*** chips in the cascade.

This document contains information on a new product. Specifications and information herein are subject to change without notice.

MOTOROLA

©MOTOROLA INC., 1986

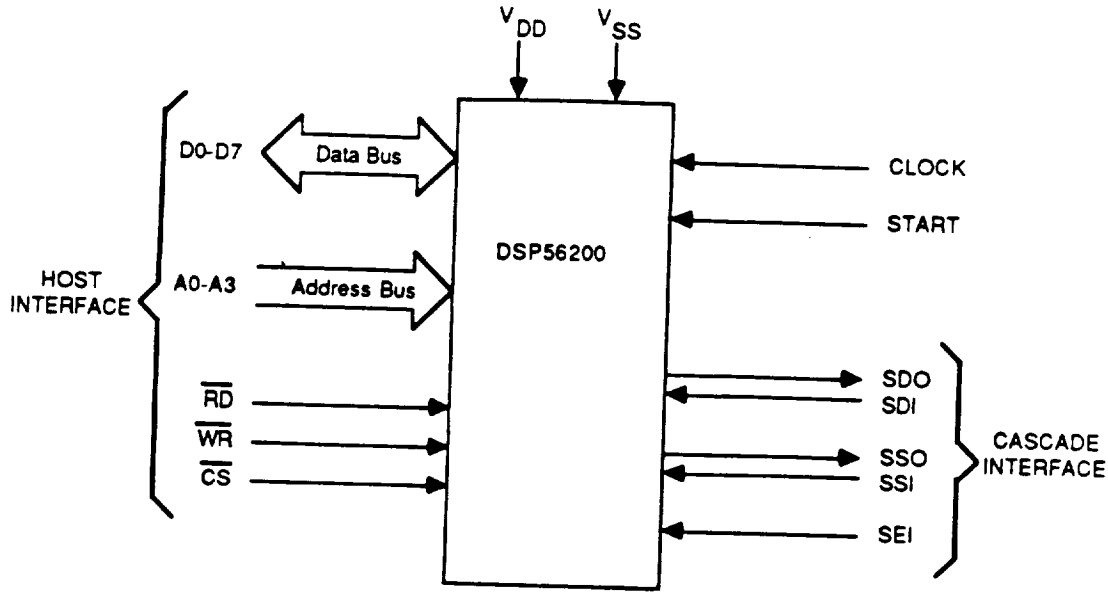


Figure 1. Functional Signal Groups

ARCHITECTURE DESCRIPTION

The DSP56200 offers two advantages over general purpose DSPs — higher performance and minimal development time. The high performance of the DSP56200 results from an algorithm specific architecture featuring three execution units — the Asynchronous Parallel Interface, the Cascade Interface, and the Computation Unit — which execute in parallel (Figure 2). Development time is minimized because the DSP56200 is an off-the-shelf chip which requires no software development. The algorithms are implemented in hardware.

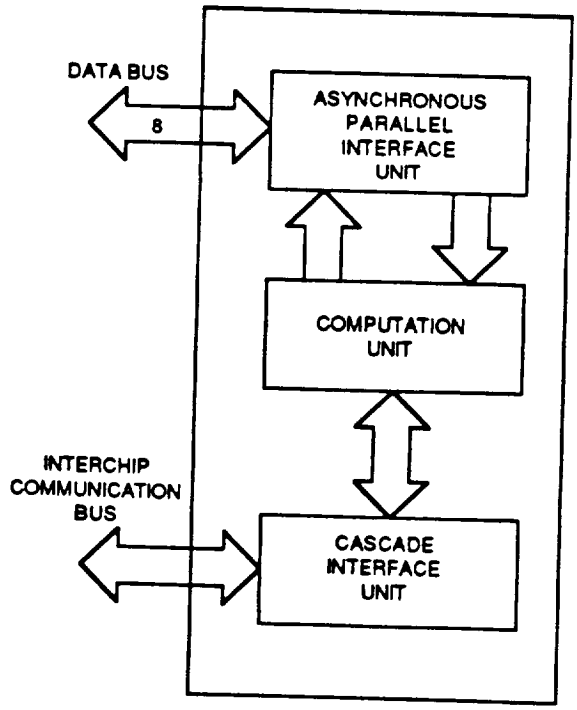


Figure 2. Parallelism Within the DSP56200

Careful design of the interface units has virtually eliminated the need for "glue" logic when interfacing to a host processor or cascading DSP56200 chips. The Parallel Interface resembles that of a fast static RAM, and allows interfacing to fast, general purpose DSPs and MPUs having tight timing requirements. The Cascade Interface performs all the functions associated with cascading, thereby simplifying the design of multi-chip systems.

The Computation Unit performs the arithmetic necessary in FIR and adaptive FIR filtering (Figure 3). It contains the hardware necessary for implementing a 256-tap FIR filter with adaptation capability, including a 256 x 24-bit Coefficient RAM, a 256 × 16-bit Data RAM, and an Arithmetic Unit. The Data RAM is configured as a variable length circular queue, allowing it to function as a virtual shift register. The Arithmetic Unit configures itself for both the LMS adaptation algorithm and the FIR filter calculation, implements the available digital filtering options, performs 100 ns 24 × 16-bit multiply-accumulates, and performs 100 ns coefficient updates. It is further described in the **INTERNAL ARITHMETIC DESCRIPTION**.

MODES OF OPERATION

The DSP56200 is designed to operate in one of three modes — Single FIR Filter, Dual FIR Filter, or Single Adaptive FIR Filter. Functional diagrams of these three configurations are shown in Figure 4. Users unfamiliar with digital filtering should refer to Appendix 1.

FIR filtering is a simple way to perform digital filtering. The Single FIR Filter mode is used to implement one FIR filter, either on a single chip or on several DSP56200s in a cascade (with up to 256 taps per chip). Using software design tools, filter coefficients can be calculated to obtain the desired frequency response for performing functions such as differentiating, bandpass filtering, and lowpass filtering. In this mode, the DSP56200 is configured to implement the FIR filtering equation (see Appendix 1).

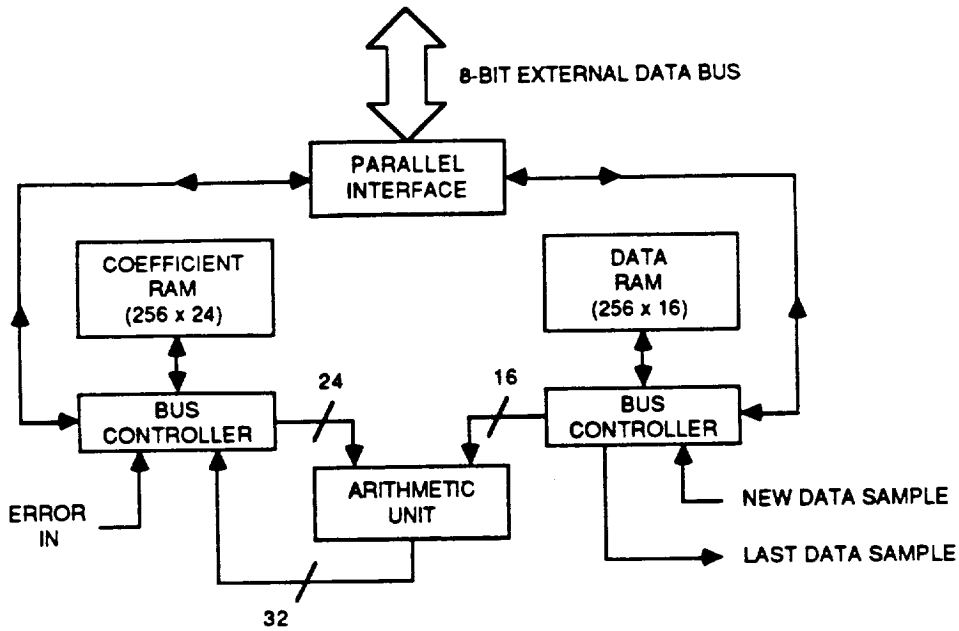


Figure 3. Computation Unit Block Diagram

The Dual FIR Filter mode is an extension of the Single FIR Filter mode. It allows two independent inputs, $x_1(n)$ and $x_2(n)$, to be FIR filtered using only one DSP56200. Note that both filters must use the same number of taps up to a maximum of 128. The DSP56200 is not adaptive or cascadable in the Dual FIR Filter mode.

The Single Adaptive FIR Filter mode provides a unique solution to problems such as echo cancelling and adaptive equalization of telephone lines. This mode is used to implement one adaptive filter, either on a single chip or on several DSP56200s in cascade. Adaptive filters must perform the FIR filter multiply-accumulate operation, followed by an adaptation operation to modify the coefficients. The DSP56200 updates every filter coefficient once during each sample period using the Least-Mean-Squares (LMS) adaptation algorithm.

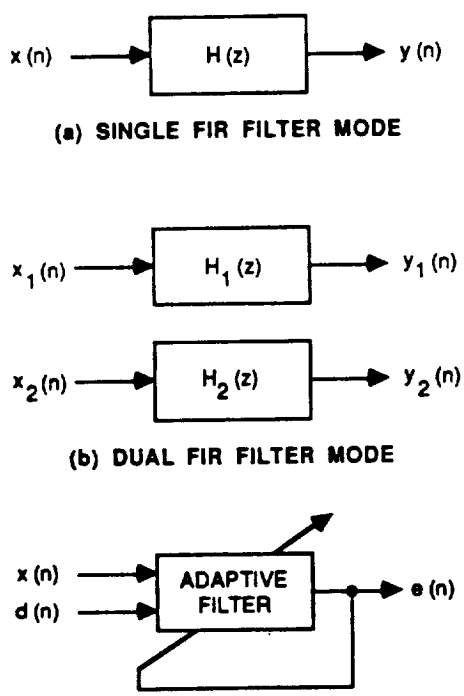


Figure 4. DSP56200 Configurations

INTERNAL ARITHMETIC DESCRIPTION

The key to the accuracy of the DSP56200 is its Arithmetic Unit. Accuracy is affected by the number of bits in the coefficient and the errors due to rounding. In FIR filter applications, the actual frequency response will deviate from the user's desired response if not enough bits are used to represent the coefficients. Wider coefficient word widths also enable adaptive filters to more closely approximate a desired impulse response, resulting in smaller error terms. Roundoff errors decrease as the size of the accumulator and coefficients increases. The DSP56200 uses a 24-bit coefficient, and will accept data words having up to 16 bits. The results of the multiply-accumulate operation are stored in a 40-bit accumulator. Both the 16-bit data samples and 24-bit coefficients are represented as signed fractional numbers.

In FIR filtering applications, the filtering process is described by the following equation:

$$y(n) = \sum_{i=0}^{N-1} h(i)\, x(n-i) \qquad (1)$$

The "ith" coefficient is represented by h(i), and x(n-i) represents one of the previous data samples. In this mode, the Arithmetic Unit is configured as a multiplier-accumulator as shown in Figure 5.

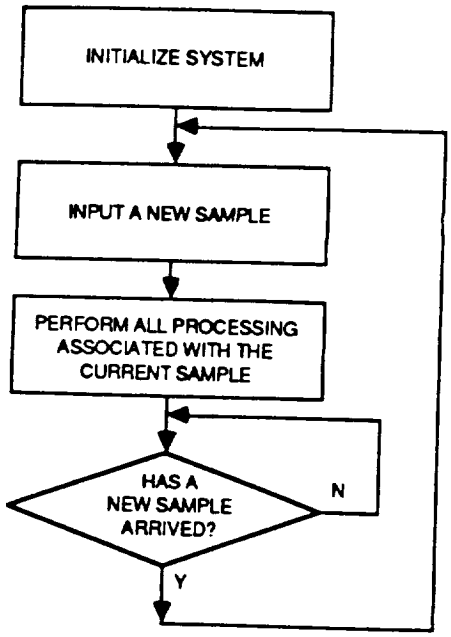


Figure 7. Flowchart for Real-Time Systems

The formulas below are used to calculate the DSP56200's maximum sampling frequency for a given system. In many cases, this maximum rate can be increased by cascading more DSP56200 chips together and using fewer taps on each chip.

$$\text{Maximum } f_s \leq f_{ck}/\#\text{cycles}$$

Where:

f_{ck} = DSP56200 input clock frequency

$$\#\text{ cycles} = \begin{cases} 12+N+q & \text{: Single FIR Filter Mode} \\ 18+2N+q & \text{: Dual FIR Filter Mode} \\ 17+2N+r & \text{: Single Adaptive Filter Mode} \end{cases}$$

$$q = \begin{cases} 29+n\text{-}N & \text{: } (29+n\text{-}N) > 0 \\ 0 & \text{: otherwise} \end{cases}$$

$$r = \begin{cases} 30+n\text{-}N & \text{: } (30+n\text{-}N) > 0 \\ 0 & \text{: otherwise} \end{cases}$$

n = Number of chips cascaded together

N = Number of taps used on each chip

The intermediate terms, q and r, can never be negative. Also note that in Dual FIR Filter mode, N is the number of taps available for each of the two independent filters, and therefore $N \leq 128$. Table 1 shows some performance figures for the DSP56200 in different configurations assuming a 10.0 MHz external clock.

Table 1. DSP56200 Performance Figures

Single Chip

Maximum Sampling Frequency (kHz)				
Mode	Number of Taps			
	32	64	128	256
FIR Filter	227	132	71	37
Adaptive Filter	123	69	37	19
Dual FIR Filters	122	68	36	—

Four Chips in Cascade

Maximum Sampling Frequency (kHz)				
Mode	Total Number of Taps			
	128	256	512	1024
FIR Filter	222	132	71	37
Adaptive Filter	120	69	37	19

Eight Chips in Cascade

Maximum Sampling Frequency (kHz)				
Mode	Total Number of Taps			
	256	512	1024	2048
FIR Filter	204	132	71	37
Adaptive Filter	115	69	37	19

Sixteen Chips in Cascade

Maximum Sampling Frequency (kHz)				
Mode	Total Number of Taps			
	512	1024	2048	4096
FIR Filter	175	132	71	37
Adaptive Filter	105	69	37	19

PRECEDING PAGE BLANK NOT FILMED

NEC

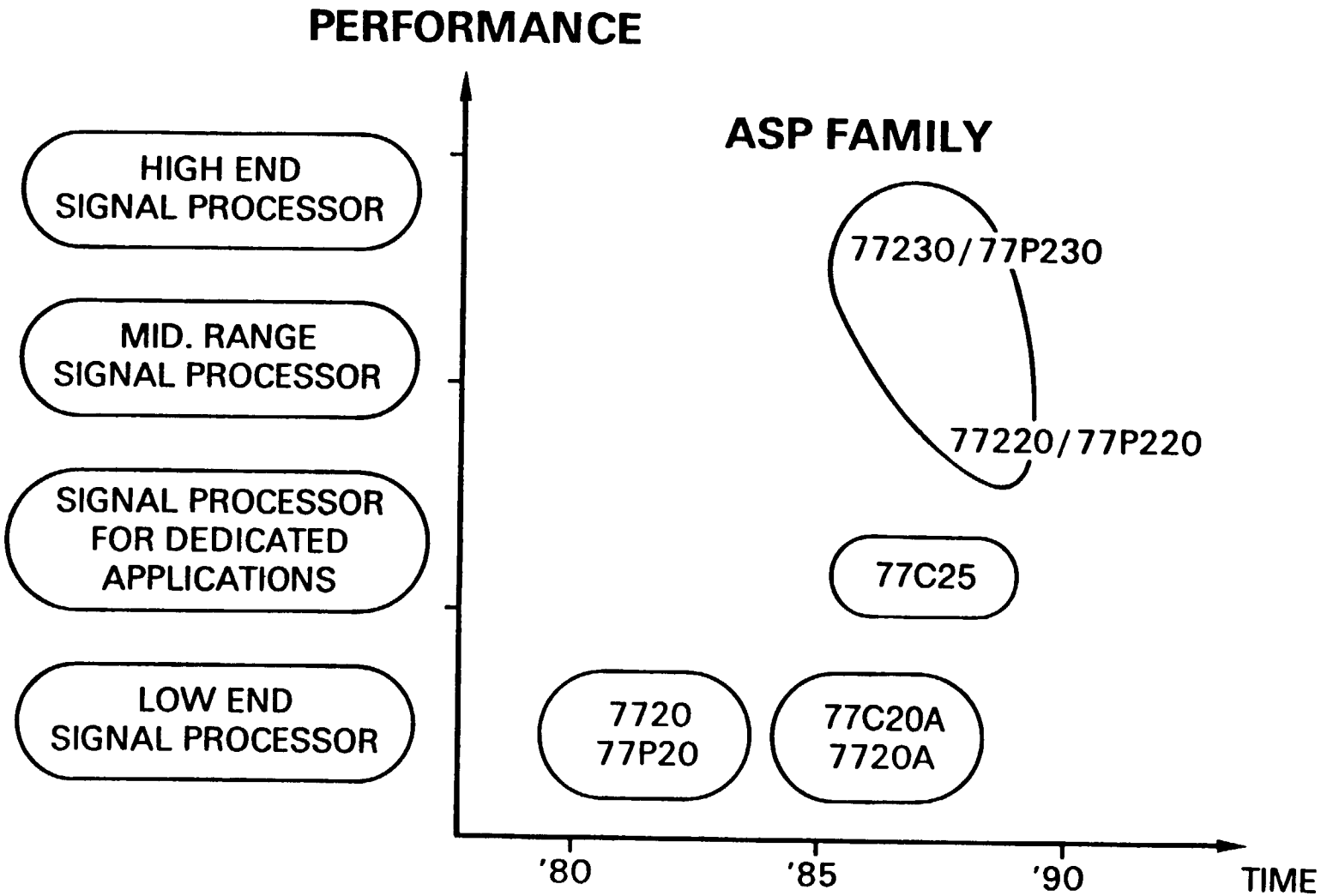
uPD772x & uPD772xx

Digital Signal Processors

Presentation

PRECEDING PAGE BLANK NOT FILMED

PRECEDING PAGE BLANK NOT FILMED



General Purpose/Telecom DSP Devices

Design Philosophy

1) Minimize System Cost:

   a) Provide large amount of on-chip, high-speed memory, (including masked instruction ROM).

   b) On-chip serial interface.

   c) Stand-alone or use as peripheral to existing microprocessor.

   d) Provide different performance level devices; you only pay for the level you need.

2) Code Efficiency:

   >> Wide "microcode-like" instructions perform many parallel operations. More functions packed into each instruction.

3) Built for real time operation:

   a) Predictable execution time -- one cycle per instruction: always. (Including multiply/accumulate.)

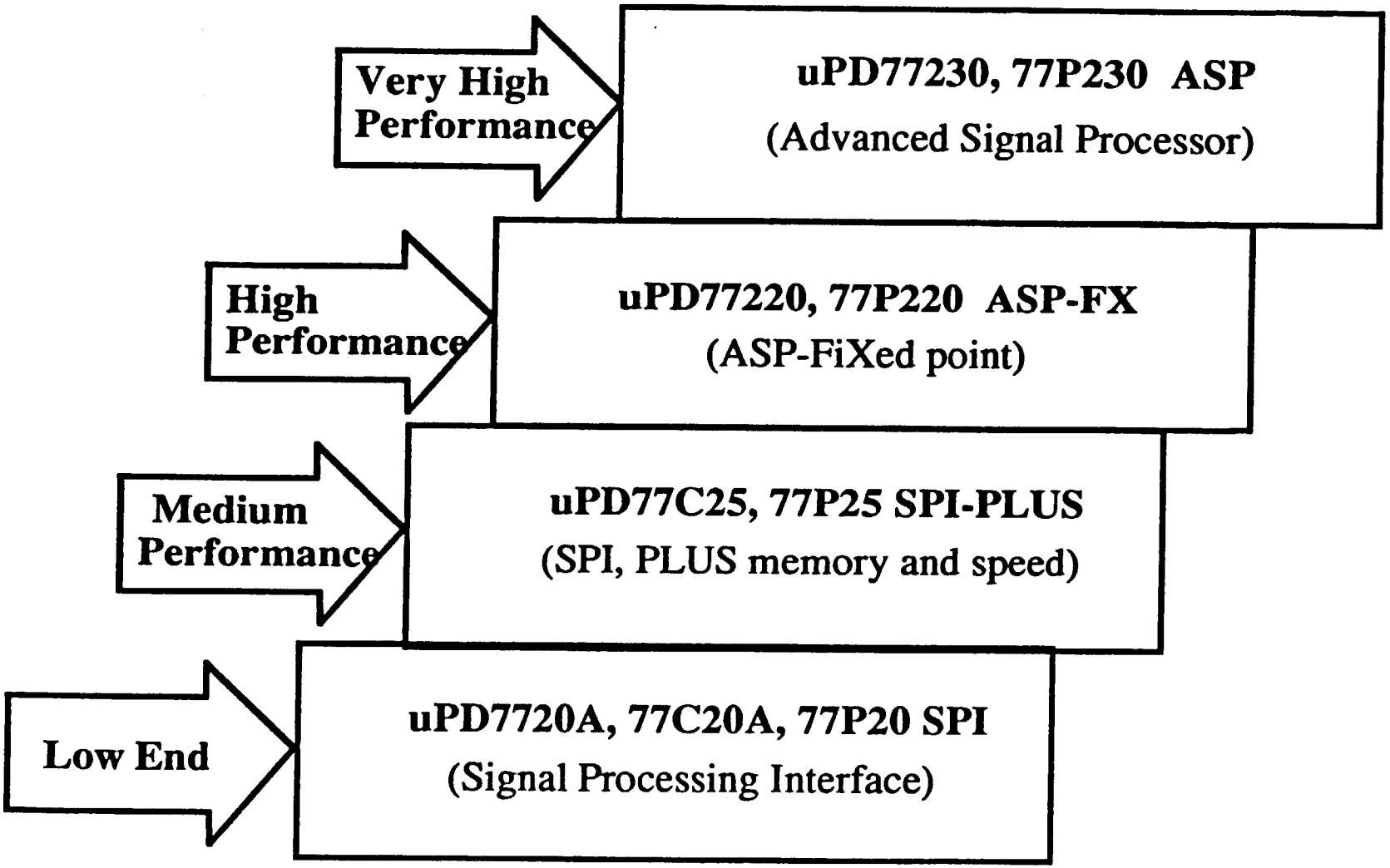
   b) Can be interrupt-driven.

   c) Multiple functions/instruction means more functions/time.

NEC

General Purpose/Telecom DSP Devices

Relative Performance



uPD 77C20A/20A/P20

FEATURES

ALL INSTRUCTIONS: 250 NS (8 MHZ)

INTERNAL PARALLEL MULTIPLIER (250 NS)

16 BITS x 16 BITS -> 31 BITS

INSTRUCTION ROM - 512 WORDS x 23 BITS

DATA ROM - 512 WORDS x 13 BITS

DATA RAM - 128 WORDS x 16 BITS

DUAL 16-BIT ACCUMULATORS

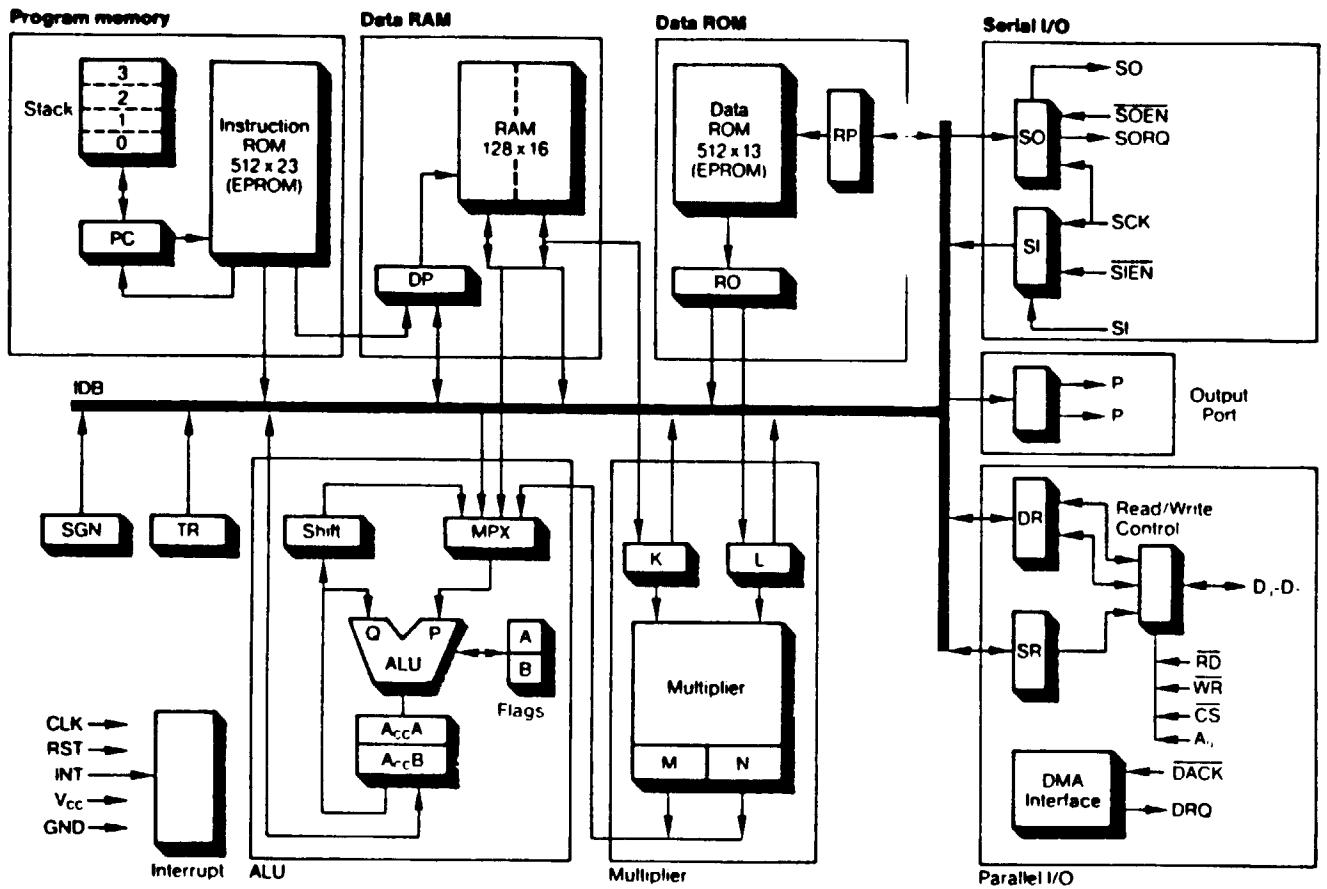
MULTIPLE I/O CAPABILITIES

-SERIAL: 8 OR 16 BIT

-PARALLEL: 8 OR 16 BIT

COMPATIBLE WITH MOST uPS, INCLUDING 8080/8085/8086/Z80

7720


Program memory
Stack
3
2
1
0
PC
Instruction ROM 512 x 23 (EPROM)
Data RAM
RAM 128 x 16
DP
Data ROM
Data ROM 512 x 13 (EPROM)
RP
RO
Serial I/O
SO
SOEN
SORQ
SCK
SIEN
SI
Output Port
P
P
IDB
SGN
TR
Shift
MPX
Q
P
ALU
A
B
Flags
AccA
AccB
K
L
Multiplier
M
N
DR
SR
Read/Write Control
D7-D0
RD
WR
CS
A0
DMA Interface
DACK
DRQ
CLK
RST
INT
Vcc
GND
Interrupt
ALU
Multiplier
Parallel I/O

NEC's uPD77C25

CMOS

DIGITAL SIGNAL PROCESSOR

"SPI PLUS"

uPD77C25/77P25

MAIN FEATURES

16-BIT CMOS SIGNAL PROCESSOR
CMOS EPROM VERSION

COMPATIBILITY WITH 7720
>> PIN-FOR-PIN COMPATIBLE
>> SOFTWARE COMPATIBLE (SEE NOTE 1)

HIGHER PROCESSING SPEED
122 NSEC INSTRUCTION CYCLE (8.192 MHZ CLOCK)

LARGER MEMORY SIZE
>> INSTRUCTION ROM: 2048 x 24B
>> DATA ROM: 1024 x 16B
>> DATA RAM: 256 x 16B

CMOS 1.6 um TECHNOLOGY

APPLICATIONS

MODEMS
SPEECH COMPRESSION
ROBOTICS CONTROL
AUDIO

POSITIONING OF 77C25 IN NEC'S DSP FAMILY

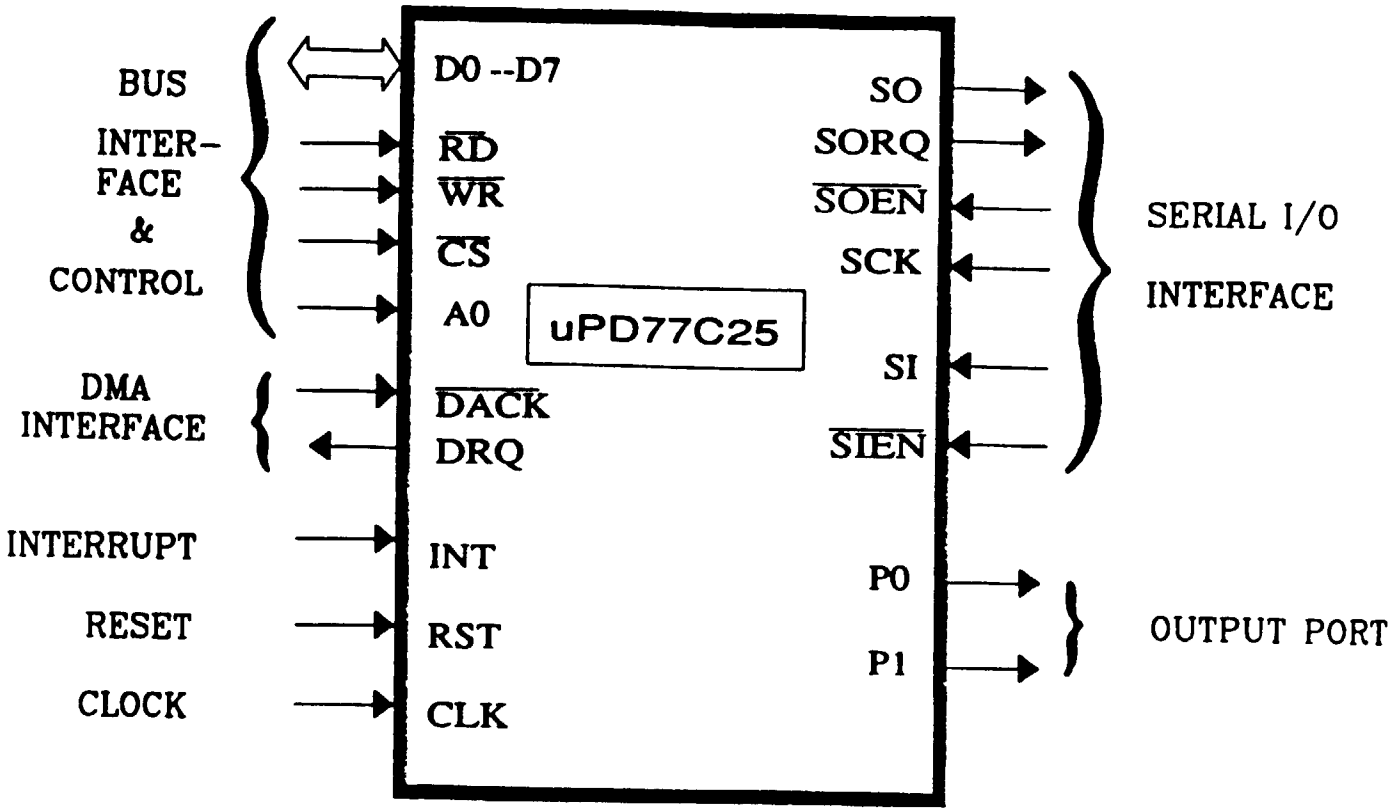
IMPROVED VERSION OF 7720 FAMILY

NOTE 1. 7720 SOURCE PROGRAM IS APPLICABLE FOR 77C25 ASSEMBLER

uPD77C25/77P25 DETAILED COMPARISON WITH 77C20A

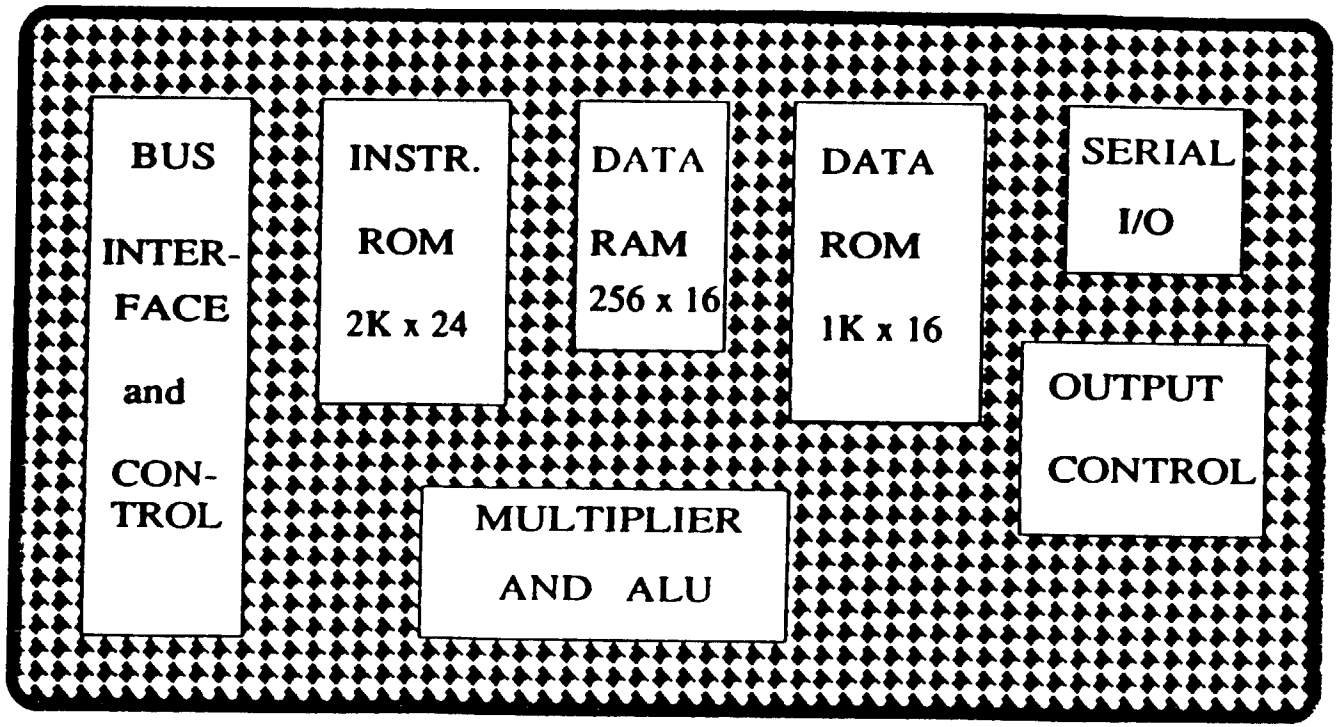
	77C20A/77P20	77C25/77P25
TECHNOLOGY	CMOS/NMOS	CMOS/CMOS
INSTRUCTION CYCLE	244 ns	122 ns
INSTRUCTION ROM	512 x 23b	2048 x 24b
DATA ROM	510 x 13b	1024 x 16b
DATA RAM	128 x 16b	256 x 16b
FIXED POINT MULTIPLIER	16b x 16b -> 31b	16b x 16b -> 31b
ALU	16b FIX. PT.	16b FIX. PT.
ACCUMULATOR	2 X 16b	2 x 16b
HOST CPU INTERFACE	8 BIT BUS	8 BIT BUS
SERIAL INTERFACE	INPUT/OUTPUT 1 CHANNEL EACH 2 MHz	INPUT/OUTPUT 1 CHANNEL EACH 4 MHz
TEMPORARY REGISTER	ONE	TWO
ADDITIONAL INSTRUCTIONS	– –	JDPLN0 JDPLNF modification of RAM column data pointer M8 - MF
DMA MODE	–	BIDIRECTIONAL
PACKAGE	28 PIN DIP 44 PIN PLCC	28 PIN DIP 44 PIN PLCC
POWER SUPPLY	5 V	5 V
POWER CONSUMPTION	40 mA (MAX) 8.192 MHz	40 mA (MAX) 8.192 MHz

uPD77C25/77P25 PIN ASSIGNMENTS

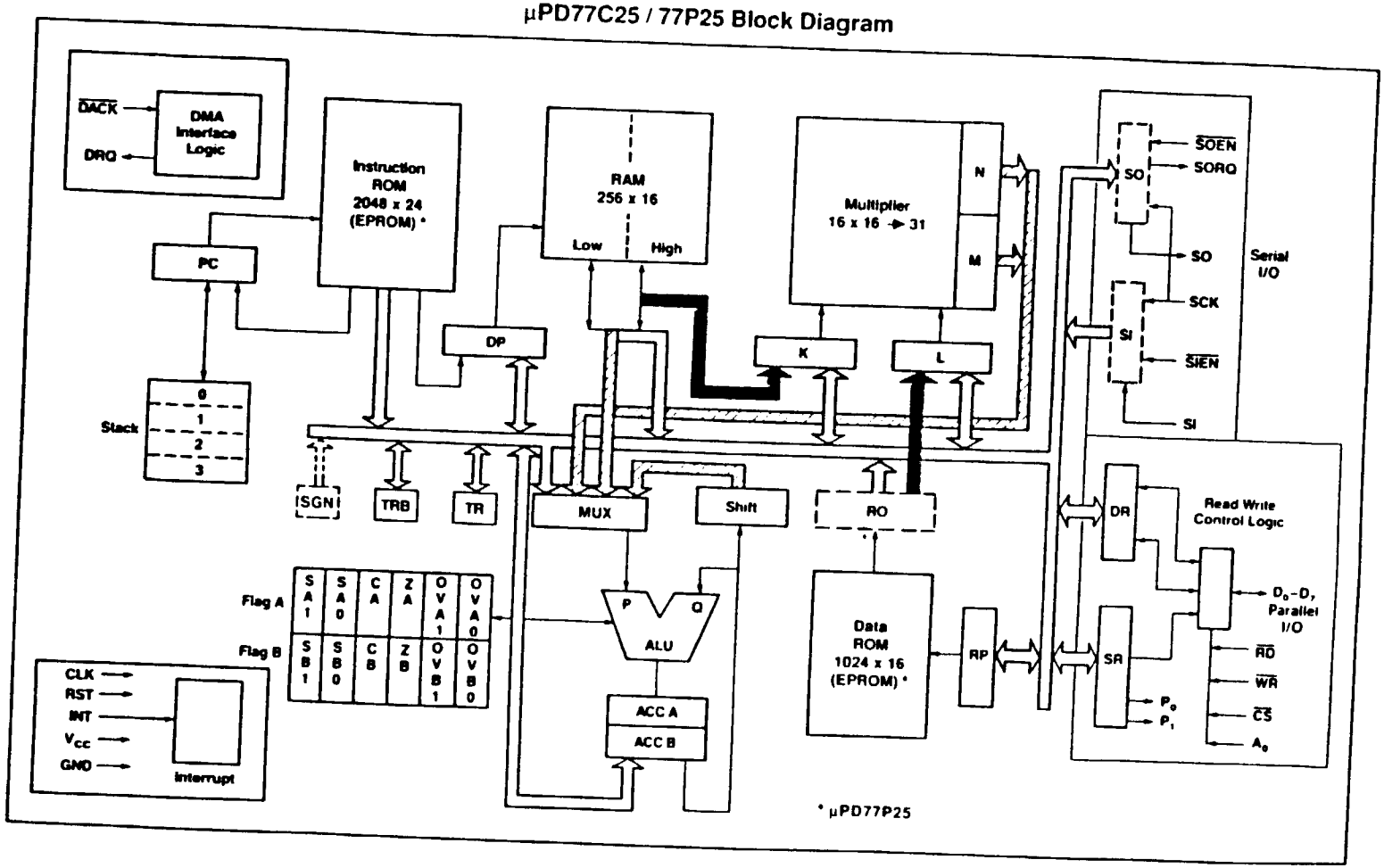


uPD77C25/77P25 IS PIN-FOR-PIN COMPATIBLE WITH uPD77C20

uPD77C25 Functional Blocks

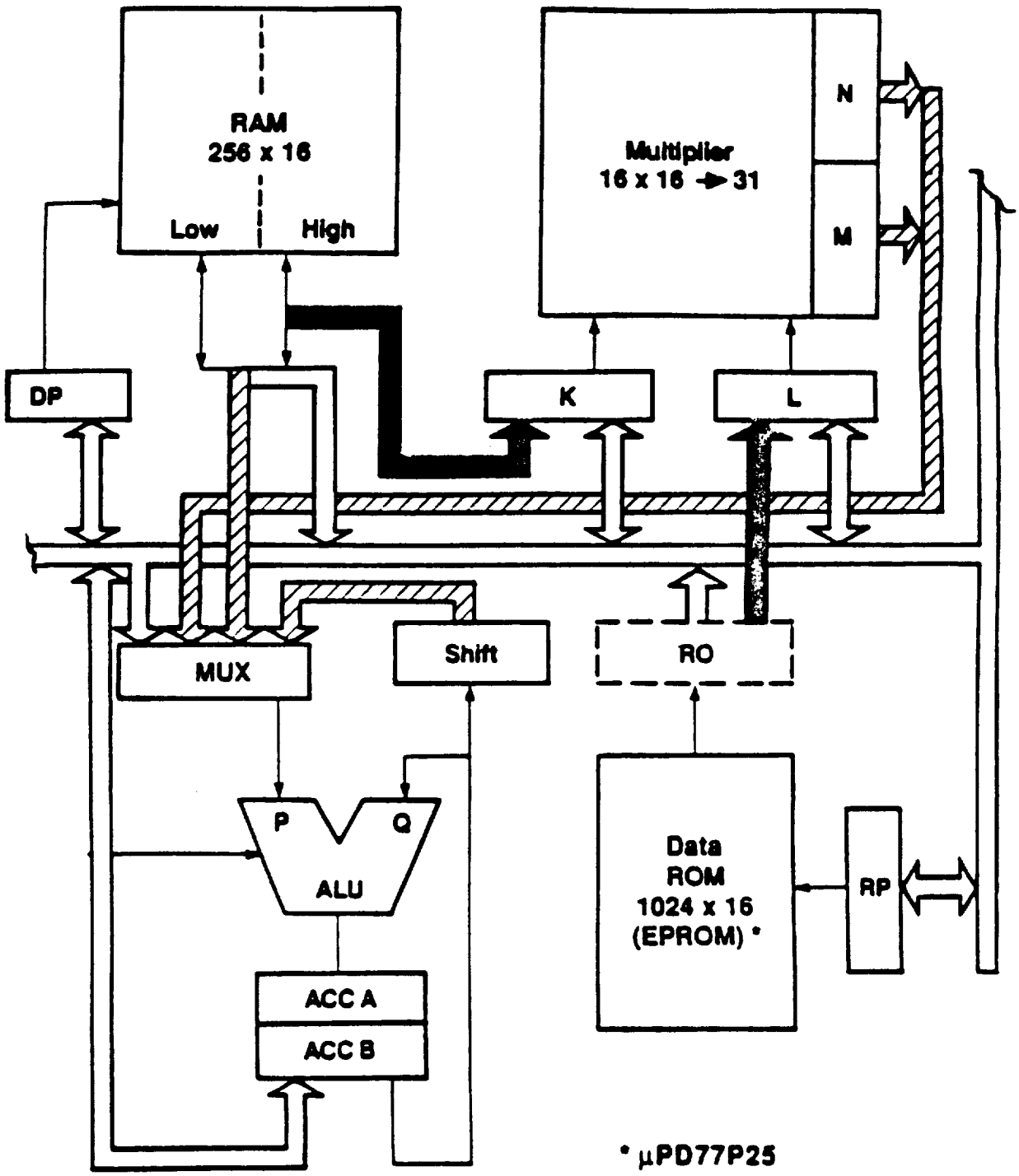


μPD77C25 / 77P25 Block Diagram



DSP Digital Signal Processing

uPD77C2X HARVARD ARCHITECTURE



* μPD77P25

NEC Electronics, Inc.

uPD77C2X MULTIPLIER INSTRUCTIONS

One instruction that does it all!

MOV @KLR,MEM ADD ACCA,M DPINC RPDEC

Meaning:
Load one multiplier input from RAM; load the other from Data ROM;
Add product to Accumulator;
Move RAM column pointer. Move Data ROM pointer.

Another single instruction where all data is in RAM

MOV @KLM,MEM ADD ACCA,M DPINC M1

Meaning:

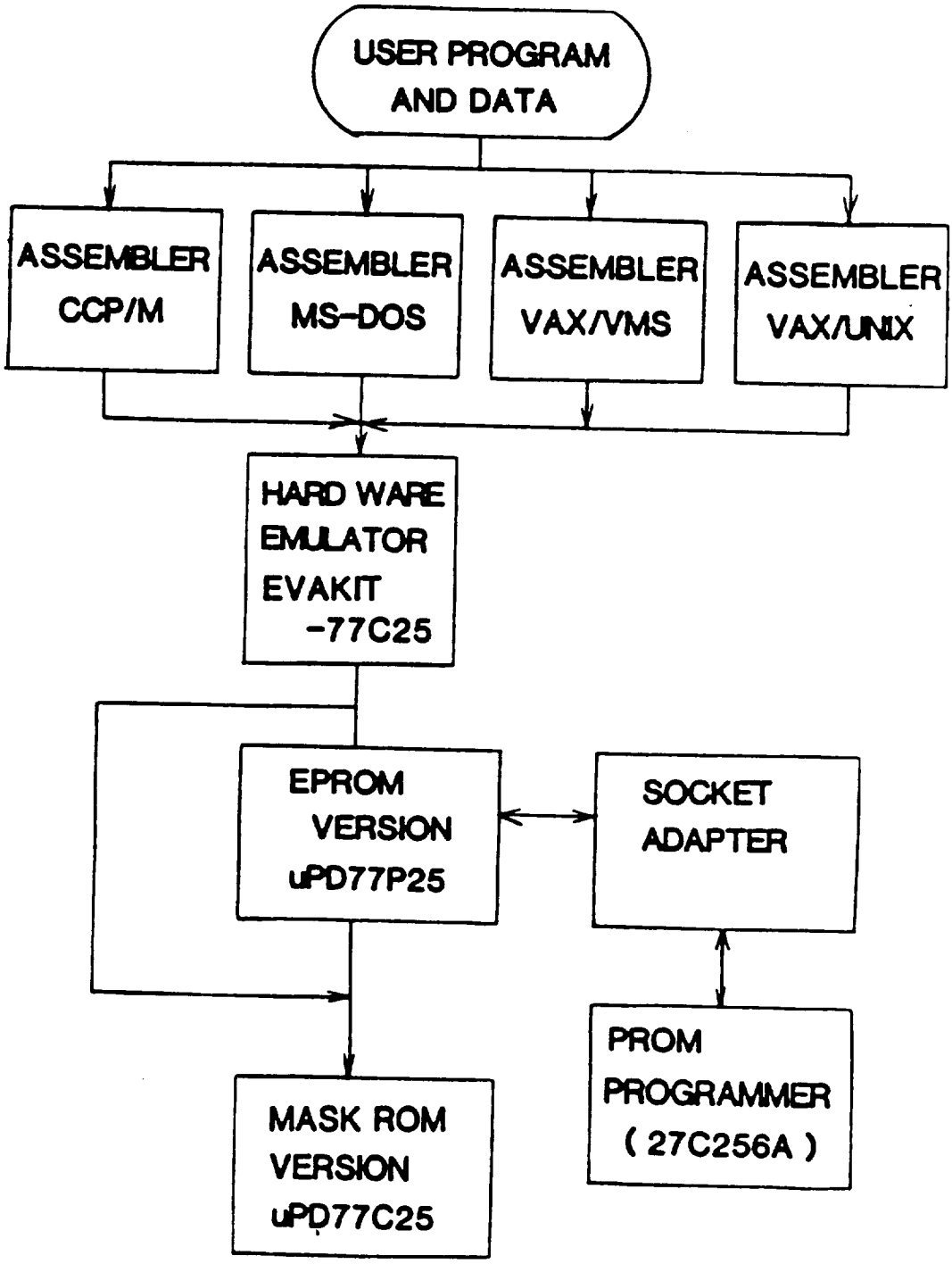
Load one multiplier input from High RAM; load the other from Low RAM;
Add product to Accumulator;
Move RAM column pointer. Move RAM row pointer.

SPI/SPI+ COMBINATION INSTRUCTION

1. Load K Register
2. Load L Register
3. Multiply K & L register contents (automatic)
4. Load product in M & N registers (automatic)
5. Add product to Accumulator
6. Move RAM column pointer
7. Move RAM row pointer
8. Move Data ROM pointer
9. Return from subroutine

Any part, any combination, or all of the above may be part of one instruction which executes in <u>one instruction cycle.</u>

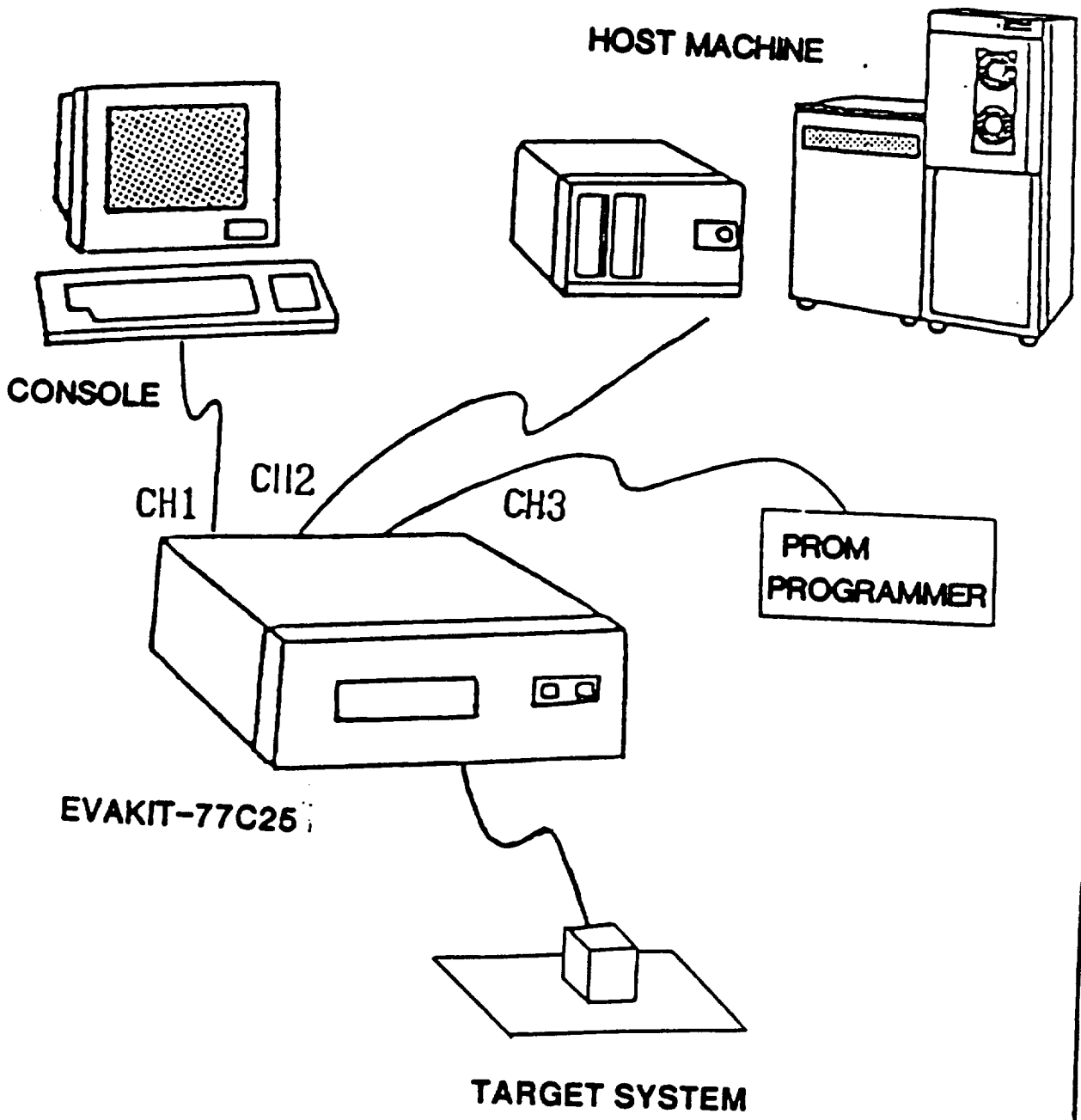
uPD77C25 / 77P25 DEVELOPMENT ENVIRONMENT



NEC

PRECEDING PAGE BLANK NOT FILMED

DEVELOPMENT SYSTEM CONFIGURATION



NEC

EXISTING 7720 CODE CAN BE ASSEMBLED AS IS BY RA77C25 RELOCATABLE ASSEMBLER.

MINOR ASSEMBLER DIRECTIVE CHANGES REQUIRED

77C25/77P25 IS DROP-IN COMPATIBLE WITH 7720/77C20/77P20

uPD 7720/77C25 - BENCHMARKS

NEC

	7720	77C25
Biquad Filter (2nd order)	2.21us	1.1us
FIR tap	244ns	122ns
Sin/Cos	5.16 us	2.5us
u/A law to linear	0.49 us	0.25 us
FFT (including windowing bit reversal & I/O)		
32-pt	1212us	553us
1024-pt	77ms	35.75ms

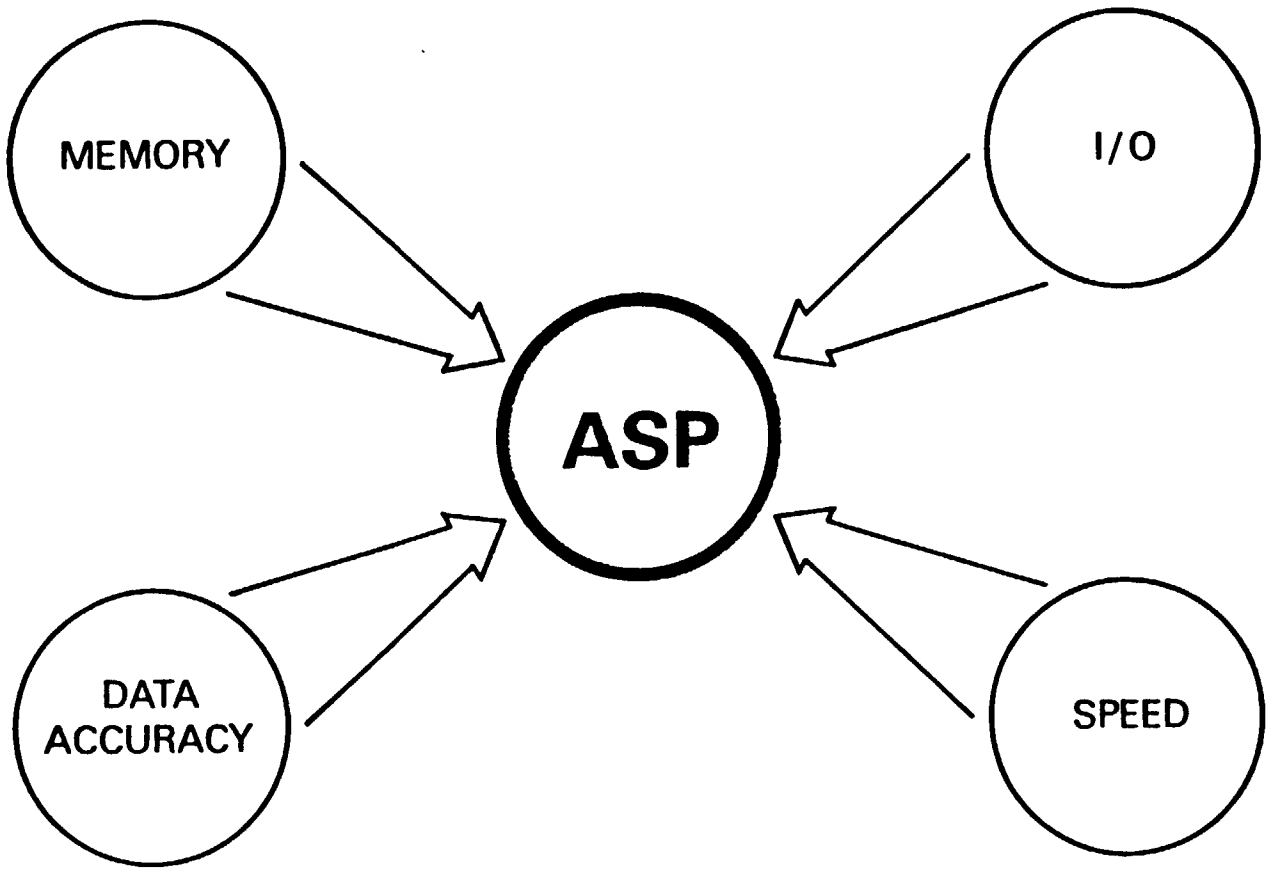
NEC

Advanced Signal Processor

(ASP)

MCP-000715

ASP FAMILY IMPROVEMENTS



WHY USE FLOATING POINT?

NEC

Fixed Point - Signals Peak Limited [-1,+1]

- Inputs are prescaled (precision loss)
- Prescaling introduces roundoff & truncation error (white noise)
- Alters transfer function (instabilities)
- Signals over/underflow because of limited wordlength
- Programmer must create overflow checking routines to saturate (clamp)
- Overflow causes nonlinear errors and additional program overhead

Floating-point

- Higher precision
- Wide dynamic range 1.7×10^{38} to 3.5×10^{-46}
- Auto scaling

uPD77230 Features

NEC

- 32-Bit Floating Point Arithmetic Operation
  - -8-Bit Exponent, 24-Bit Mantissa
- Also 24-Bit Fixed Point Arithmetic Operation
- External Memory Capability
  - -Ext. Instruction 4K X 32 Bits
  - -Ext. Data 8K X 32 Bits

MCP-000716

uPD77230 Features (cont)

NEC

- Large Capacity On-Chip Memories
  - -Data RAM 1K X 32 Bits
  - -Data ROM 1K X 32 Bits
  - -Instruction ROM 2K X 32 Bits
- Two Interface Modes
  - -Master
  - -Slave
- Fast Instruction Execution:
  - 150 ns/Instruction for ALL Instructions

uPD77230 Features (cont)

NEC

- Single +5 V Power Supply (1 W typ)
- CMOS Technology (1.5 um)
- 68-Pin Grid Array

Feature Comparison

NEC

Feature	7720A	77230
On-Chip Memory		
RAM	128 X 16	1K X 32
ROM Data	510 X 13	1K X 32
Program	512 X 23	2K X 32
External Memory		
Data Memory	None	8K X 32
Prog Memory	None	4K X 32
Multiplier	16 X 16 -› 31	Mantissa 24 X 24 -› 47 Exponent 8B Adder Exponent & Mantissa -› 55
Barrel Shifter	None	47B
Working Regs	2 X 16	8 X 55
Aux. Regs	1 X 16	1 X 32
Internal Bus	16 B	32 B
ALU	16 B	55 B

MCP-000719

Feature Comparison (cont)

NEC

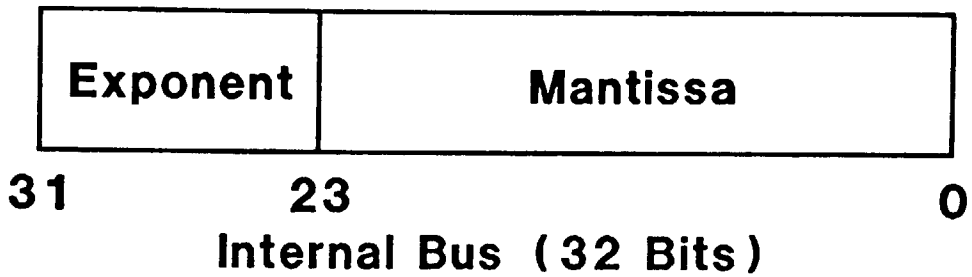
Feature	7720A	77230
Parallel I/O	8 B	32 B Master
Serial I/O	2 MHz	5 MHz
Interrupts	1 Maskable	1 Maskable
		1 Non-Maskable
Interface	Slave	Master/Slave
Performance		
Cycle Time (ns)	250	150
Multiply/Accum	250	150
ADRS Compute	Special But Limited	Special
Technology Process	NMOS	1.5 um CMOS
Supply Volts	5 V	5 V
Power Disp Typical	0.6 W	1 W
Package	28-Pin DIP	68-Pin Grid Array

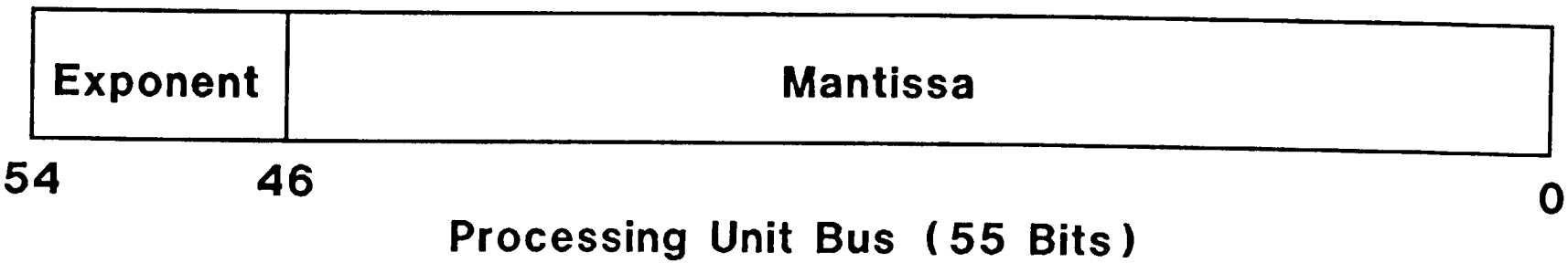
MCP-000720

ASP Floating Point Data Format

NEC

2's Complement Data

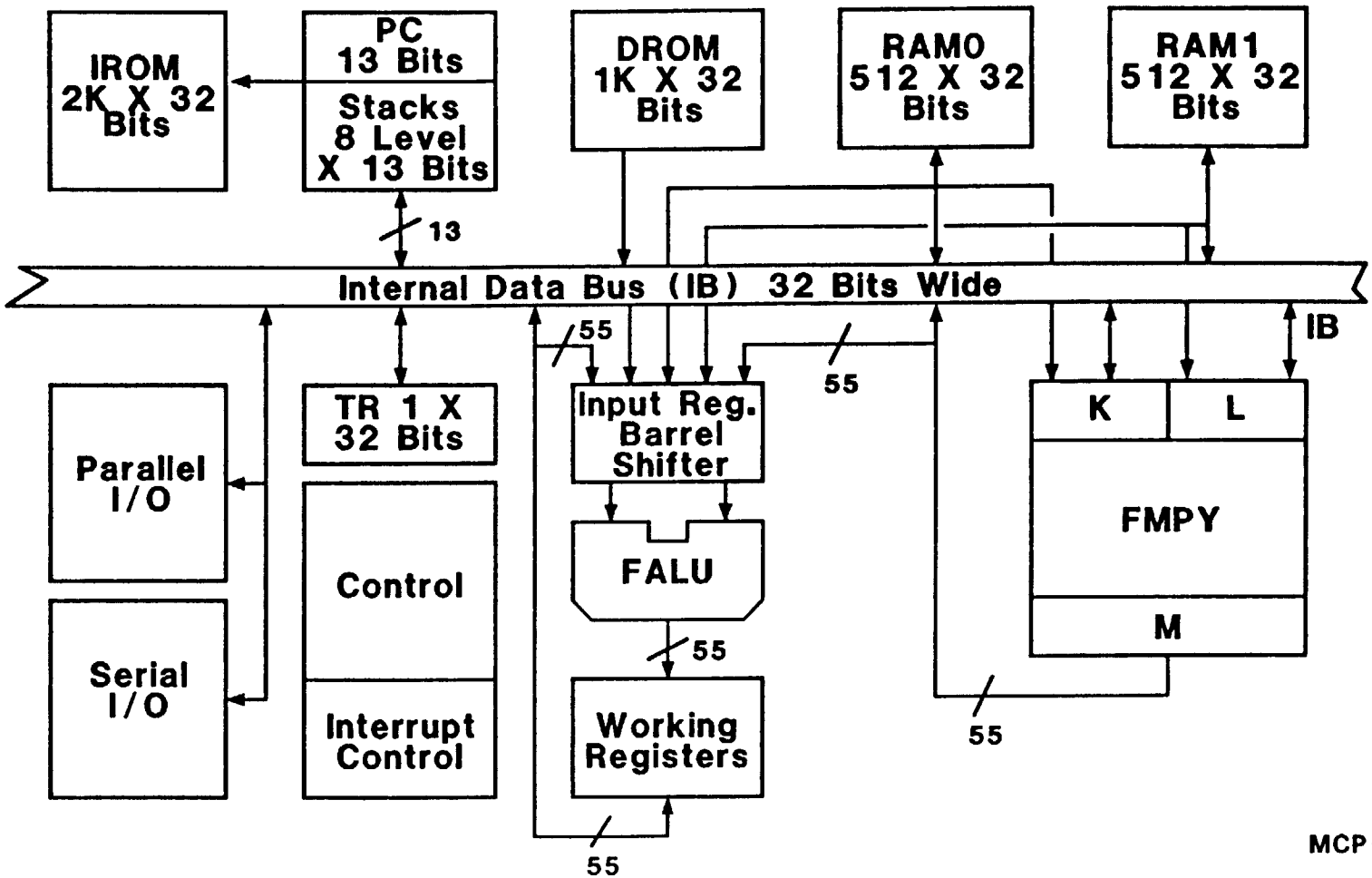




MCP-000721

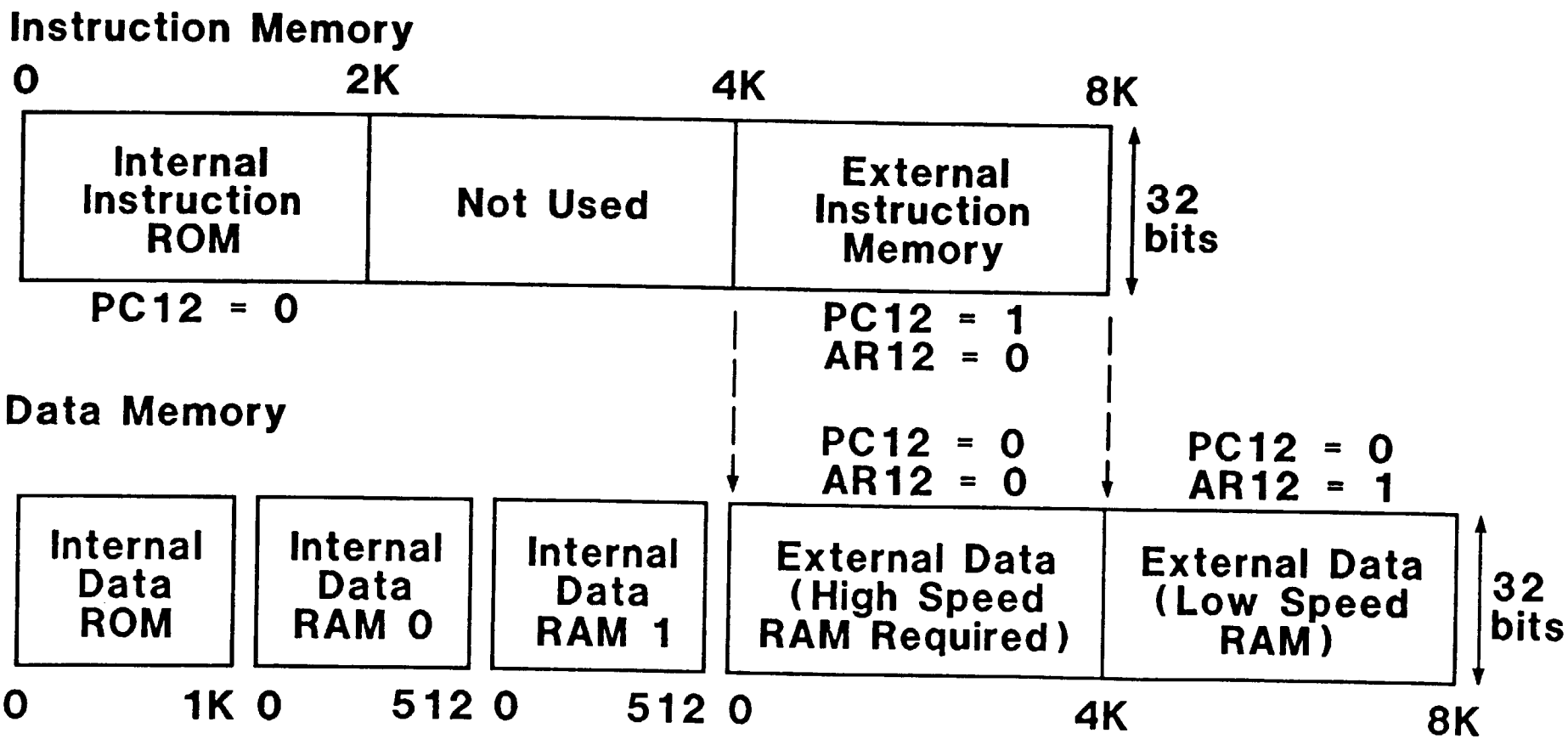
uPD77230 ASP

NEC



MCP-000722

Internal and External Memory Map



MCP-000723

uPD77230 ASP External Memory

NEC

- 8K External Memory Must be Shared by Instructions and Data
  - -If 4K External Instruction Memory is Used, then 4K Maximum Data Memory is Available
- External Instruction Memory Available ONLY in Master Mode

MCP-000724

uPD77230
ASP External Memory (cont)

NEC

- o High Speed RAM Area (Lower 4K)
  - May Contain Data and/or Instructions
  - Single Cycle Load of External Instructions
    - 35 ns RAMs
  - Two Cycle Load of External Data
    - 150 ns RAMs

- o Low speed RAM Area (Upper 4K)
  - For Data Only
  - Four Cylce/1 Instruction Data Load
    - 450 ns RAMs

PU (Processing Unit)

NEC

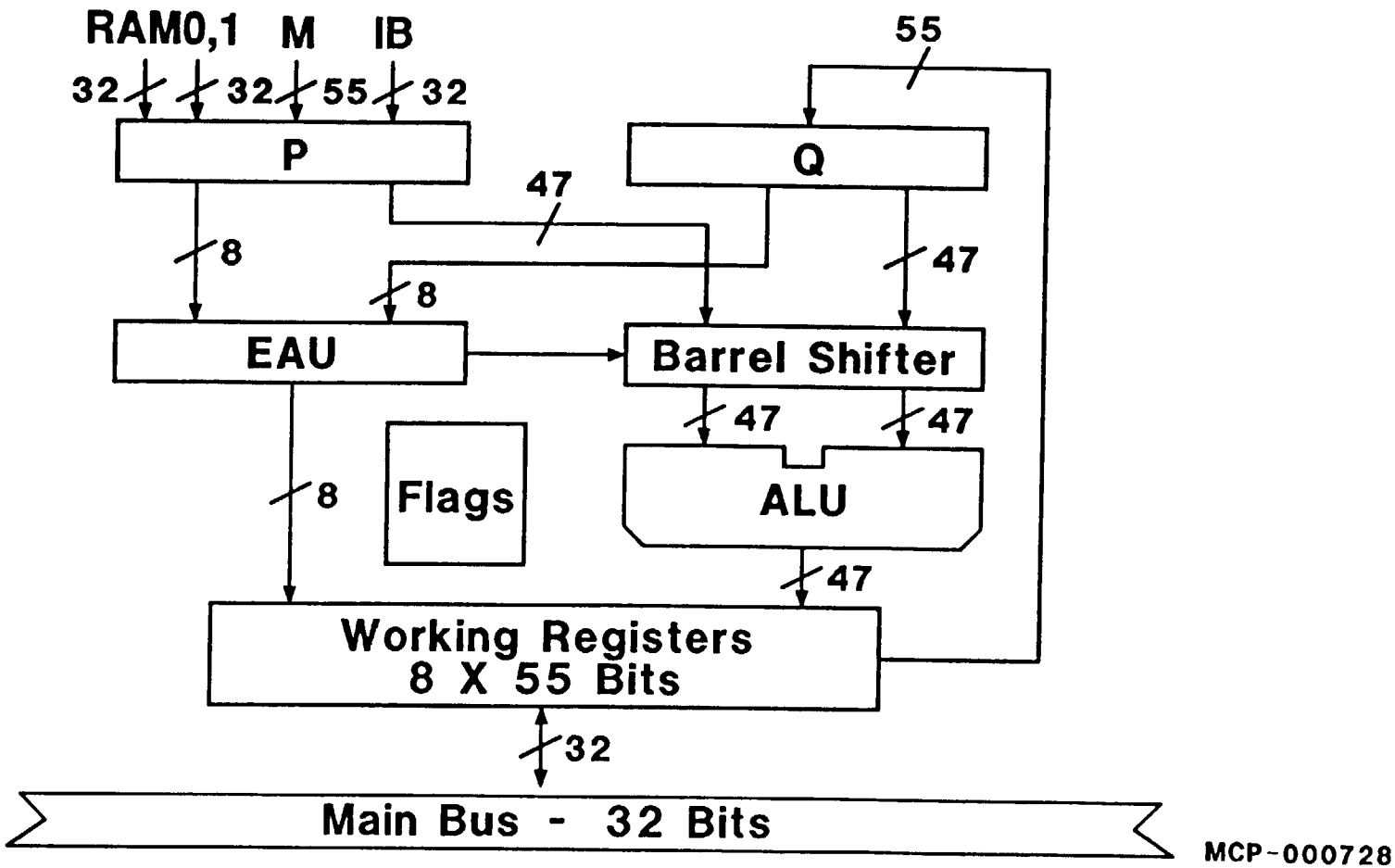
- 55-Bit Floating Point Operation
- 47-Bit Fixed Point Operation
- 47-Bit Logical Operation
- 47-Bit Barrel Shifter
- 8 55-Bit Working Registers
- Continuous Accumulation Capability
- Specific Control of Transfer Between 55-Bit Working Registers and 32-Bit Bus
  - -Bit Reverse, Exchange, etc.

MCP-000727

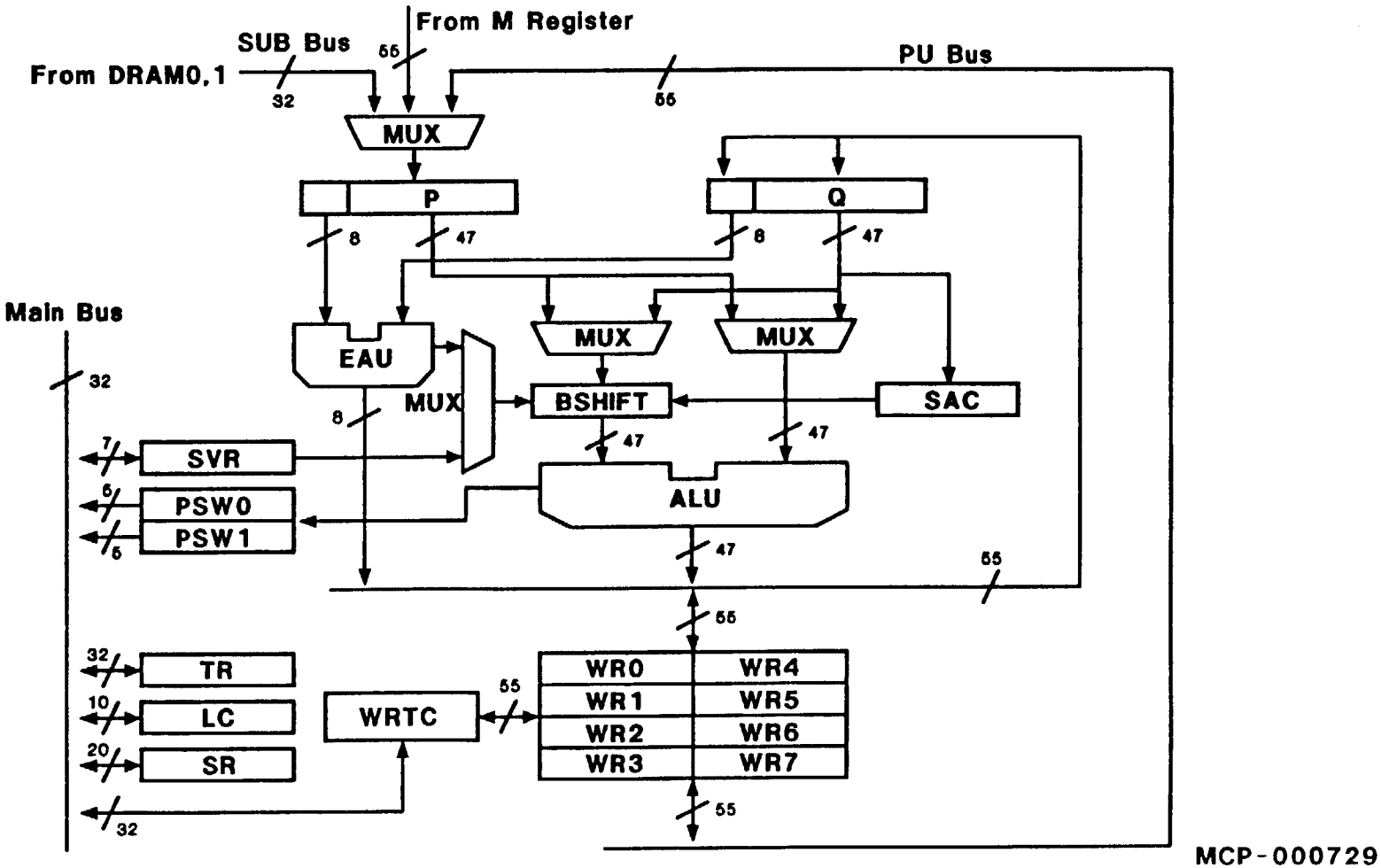
PRECEDING PAGE BLANK NOT FILMED

PU Block

NEC


RAM0,1
M
IB
32
32
55
32
P
Q
55
47
8
8
47
EAU
Barrel Shifter
47
47
8
Flags
ALU
47
Working Registers
8 X 55 Bits
32
Main Bus - 32 Bits
MCP-000728

PU Block


From M Register
SUB Bus
From DRAM0,1
55
32
PU Bus
55
MUX
P
Q
8
47
8
47
Main Bus
EAU
MUX
MUX
MUX
32
8
BSHIFT
SAC
47
47
7
SVR
6
PSW0
PSW1
6
ALU
47
55
55
32
TR
10
LC
20
SR
32
WRTC
55
WR0
WR1
WR2
WR3
WR4
WR5
WR6
WR7
55
MCP-000729

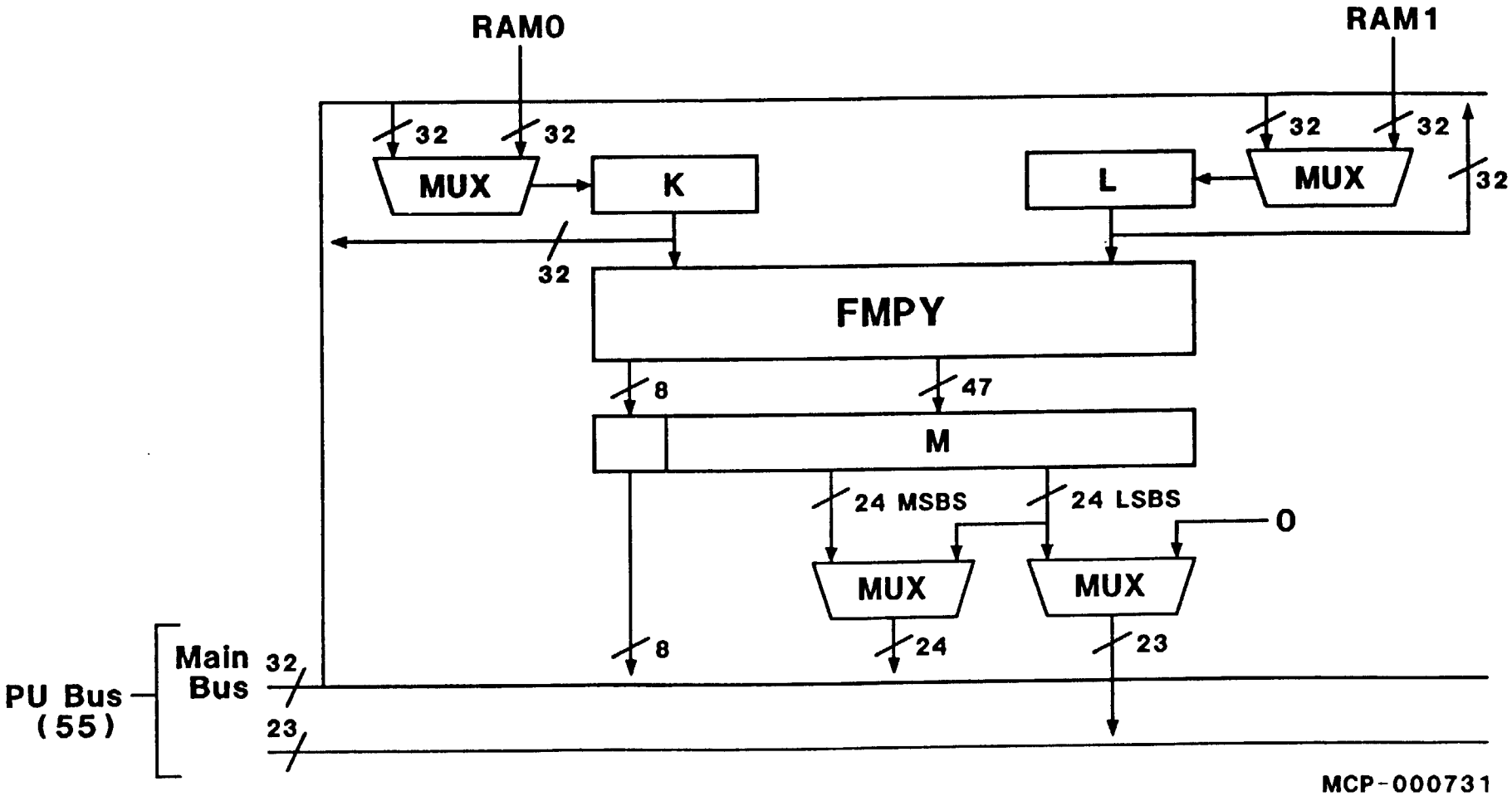
Floating Point Multiplier

NEC

- 32-Bit Inputs
- 8-Bit Exponent Adder
- 24-Bit Mantissa Multiplier
- 55-Bit Result
  - -8-Bit Exponent, 47-Bit Mantissa

Multiplier Block

NEC



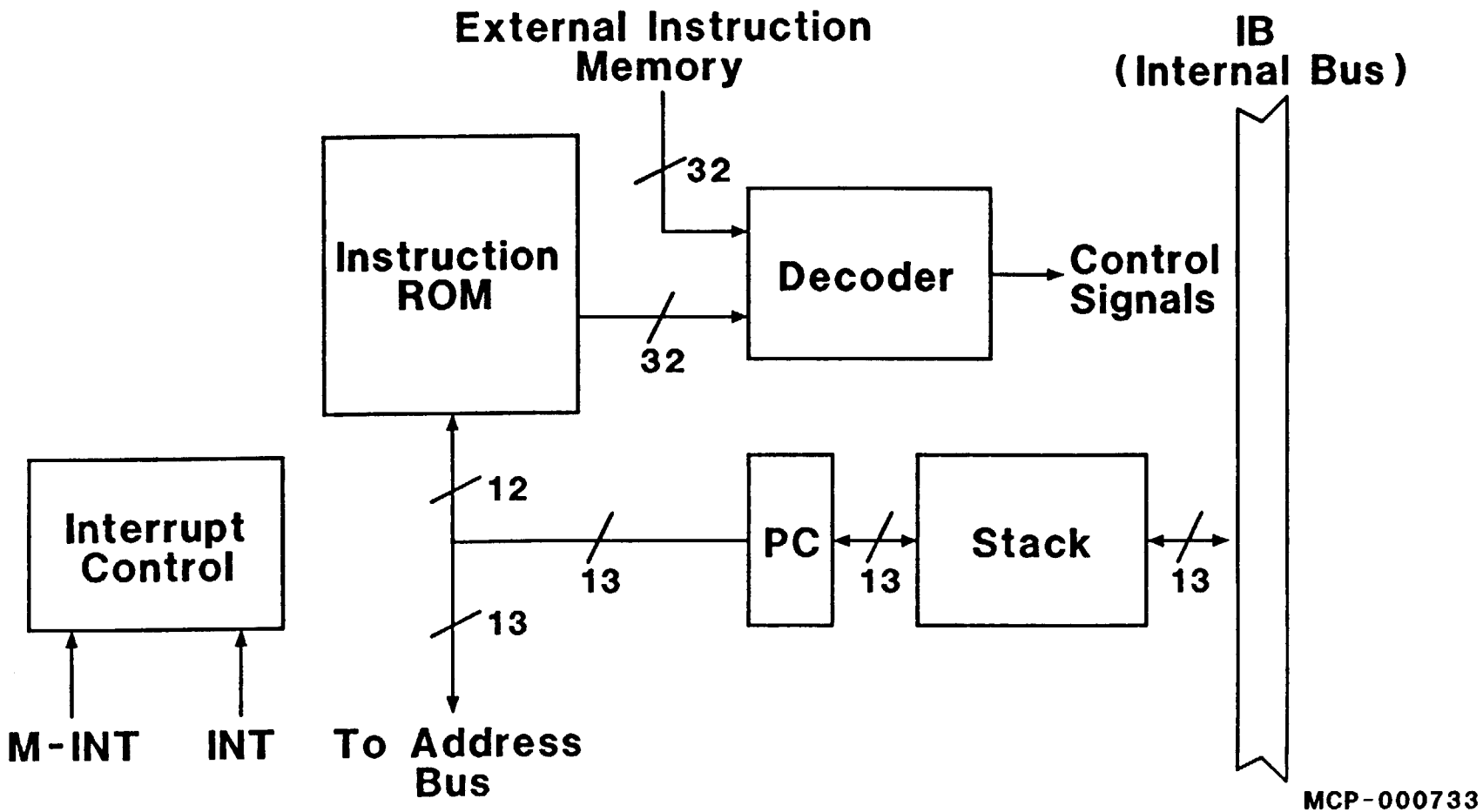
Control Block

NEC

- System Clock Output
- Internal Serial Clock Generation
- Non-Maskable Interrupt
- Maskable Interrupt Which can be Memorized
- 20-Bit Status Register
- R/W Control for Access to/from External Memory

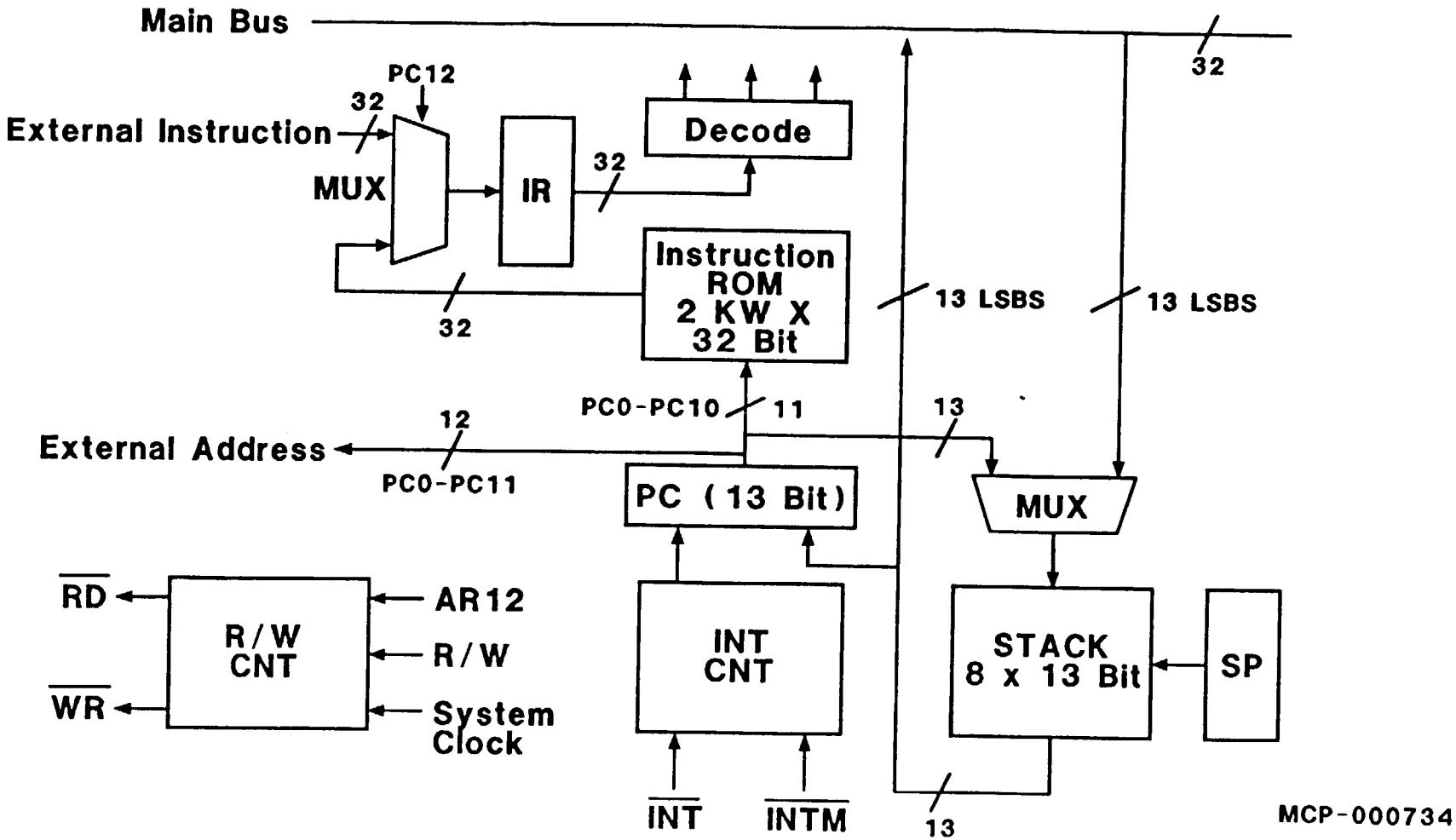
MCP-000732

Control Block



Control Block

NEC



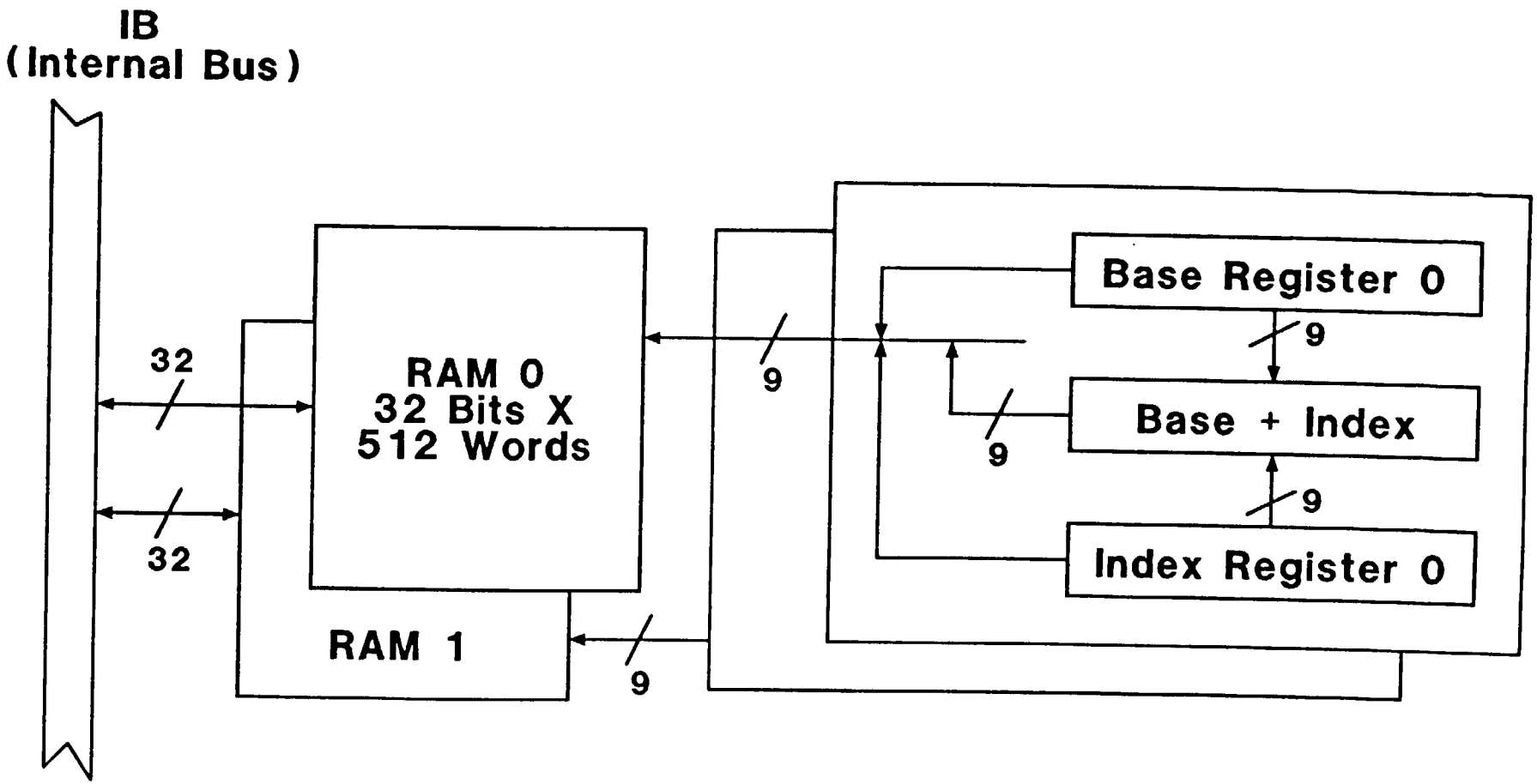
Internal Data RAM

NEC

- Two Separately Addressable 512 Word Blocks
- Multiple Addressing Modes
  - -Base Register
  - -Index Register
  - -Base + Index Register
  - -Base Register Modulo Count
    (2^1 - 2^7 Wrap Around)
- Separate Bus to Multiplier and ALU

Internal Data RAM Block

NEC



MCP-000736

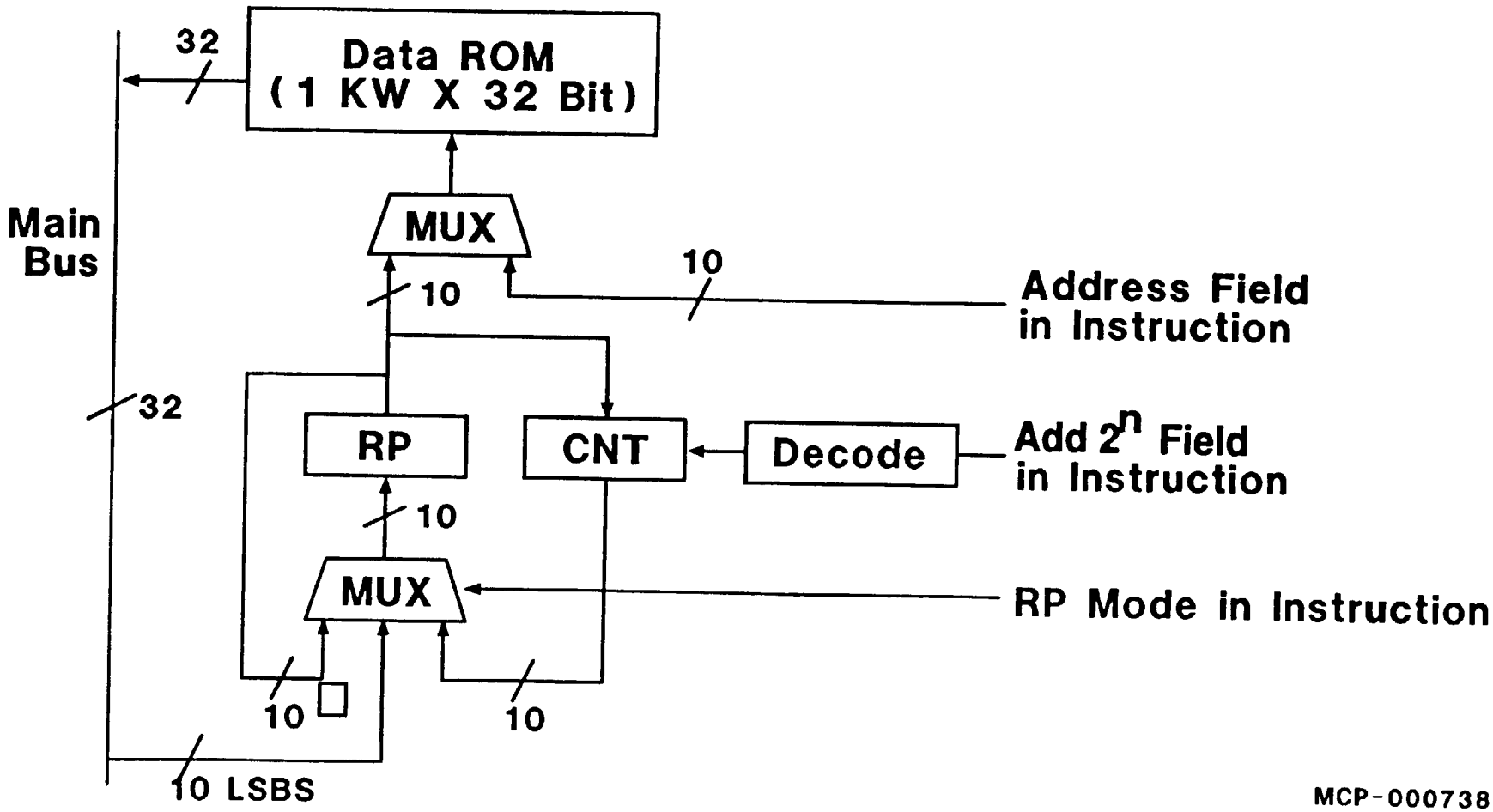
Internal Data ROM:

NEC

- Two Addressing Methods:
  - -Addressing by ROM Pointer
  - -Direct Addressing by Instruction Field
- Special Modification Function:
  - -ROM Pointer 2^n Add Operation
- ROM Pointer Increment and Decrement

Internal Data ROM

NEC



MCP-000738

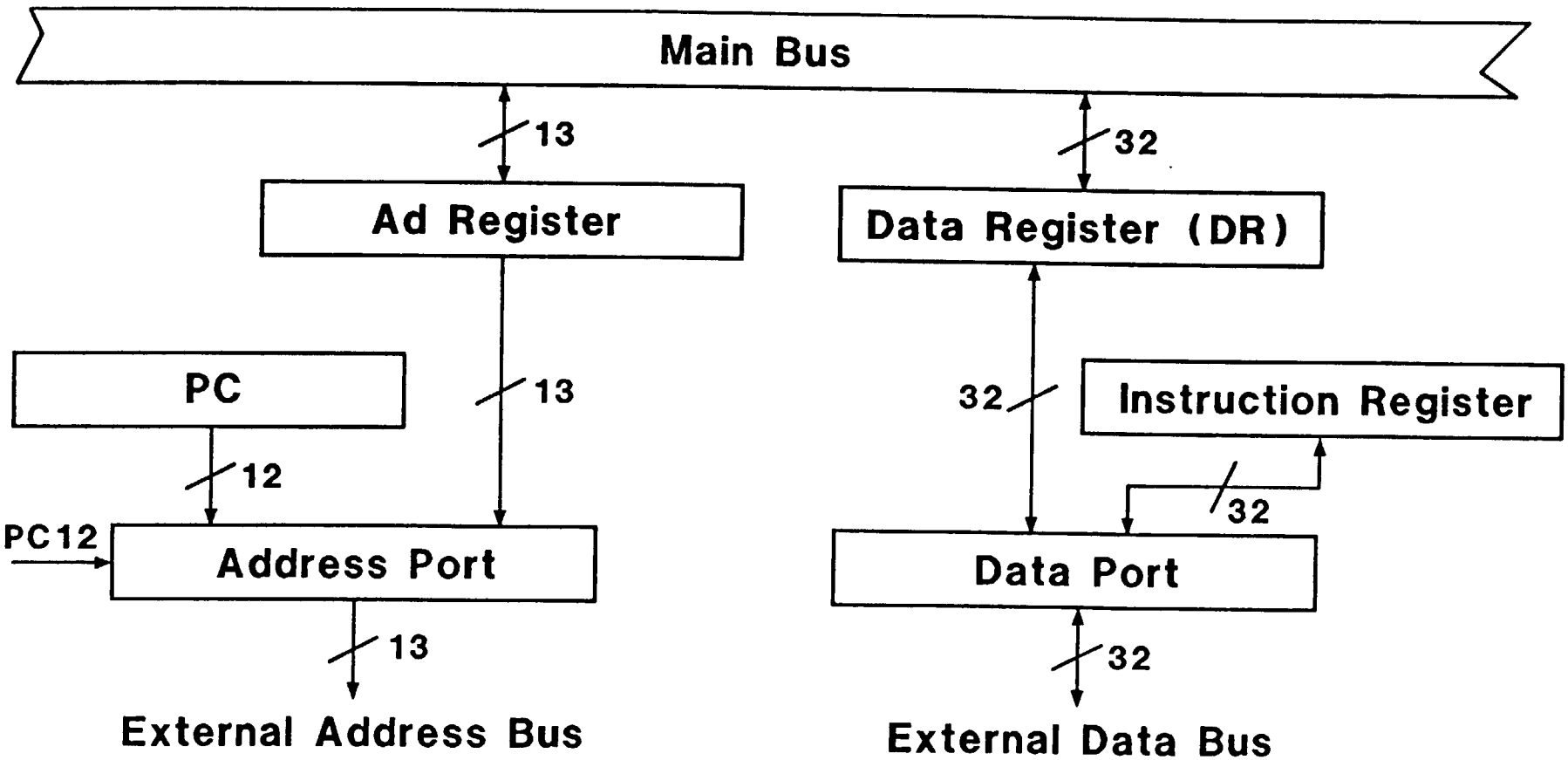
Parallel I/O

NEC

- Master Mode:
  - 32-Bit Data Bus Port
  - Under ASP Control

MCP-000739

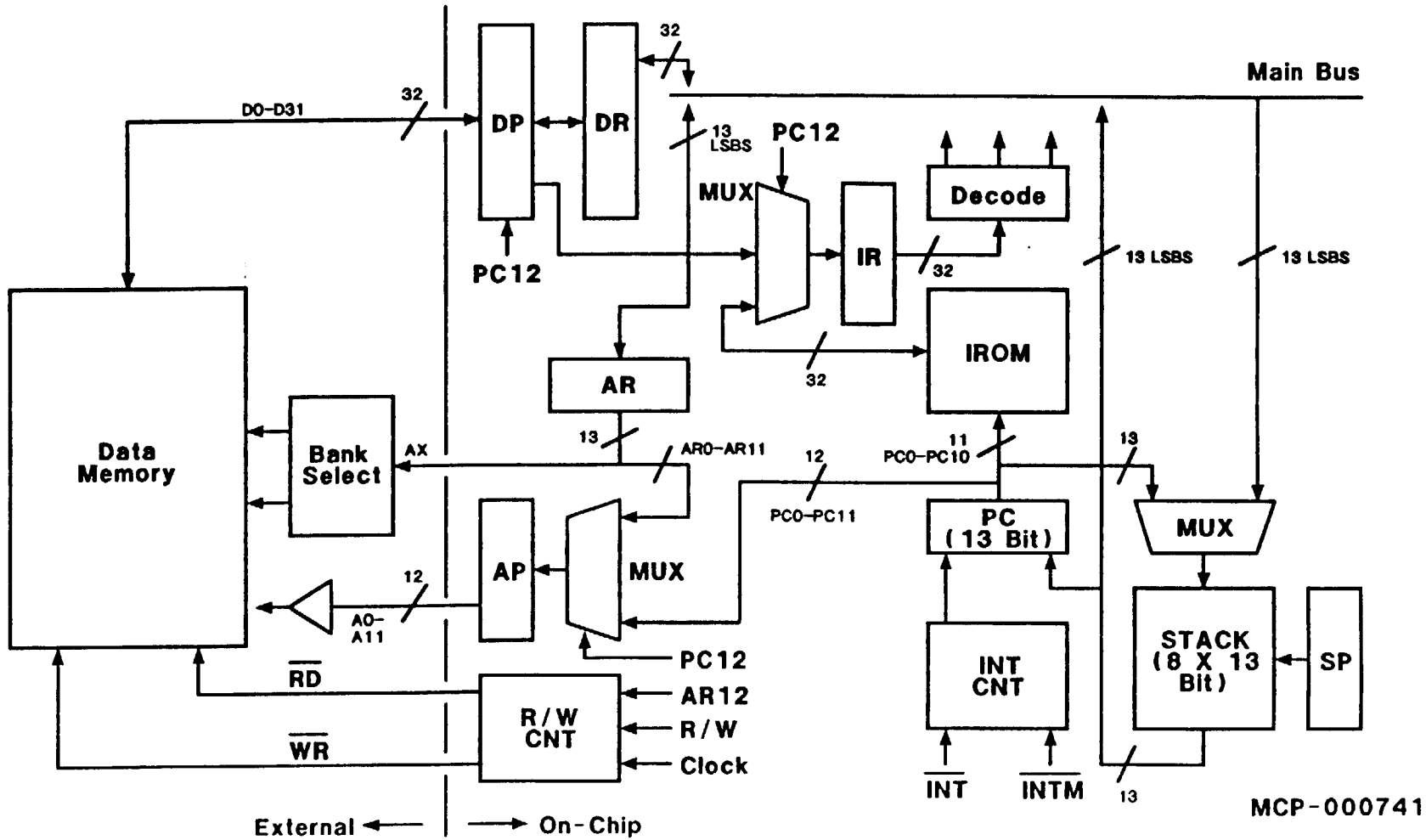
Master Mode Parallel Interface



MCP-000740

Master Mode Parallel Interface

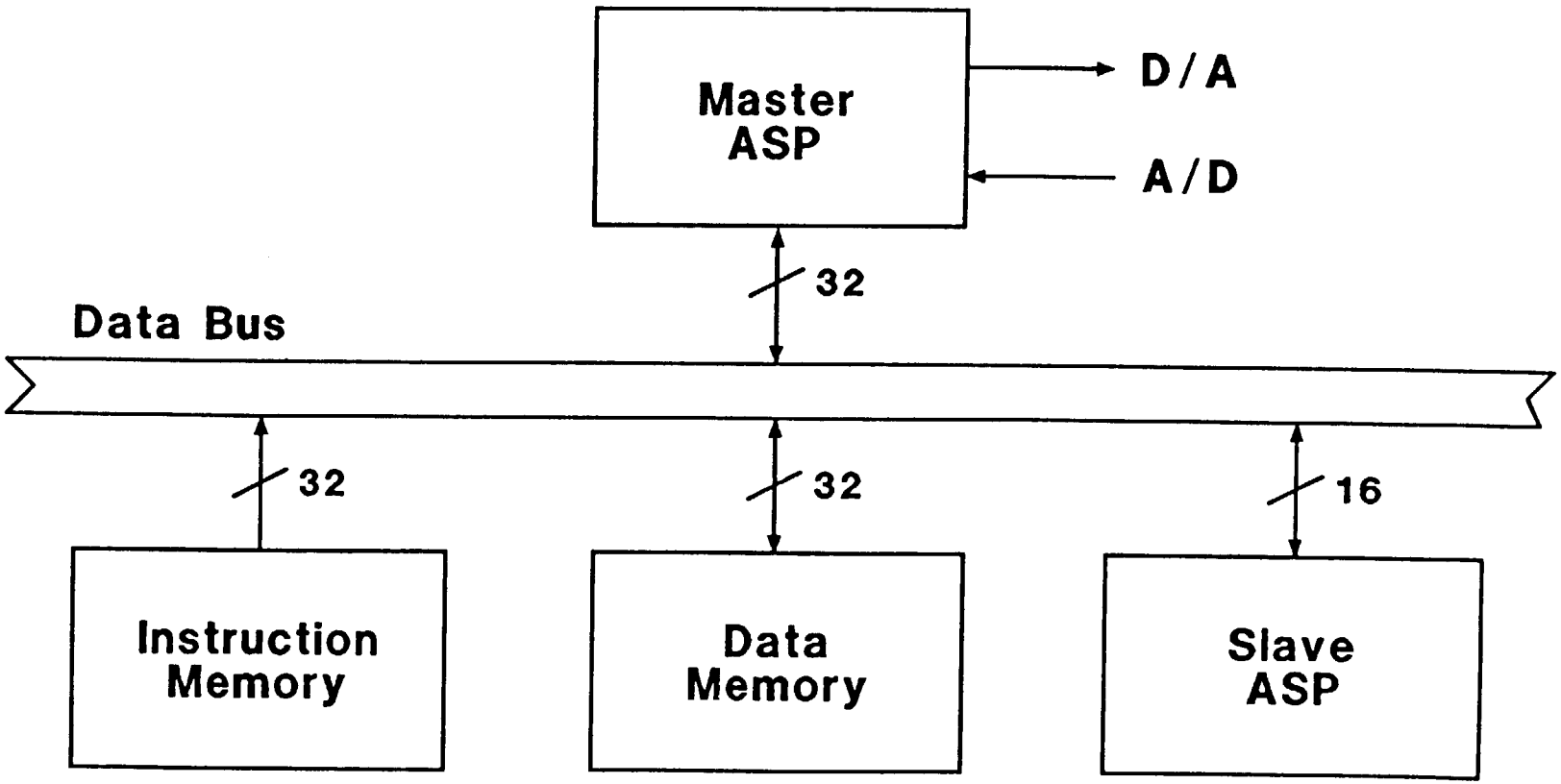
NEC



C-2

Stand Alone System Example

NEC



MCP-000742

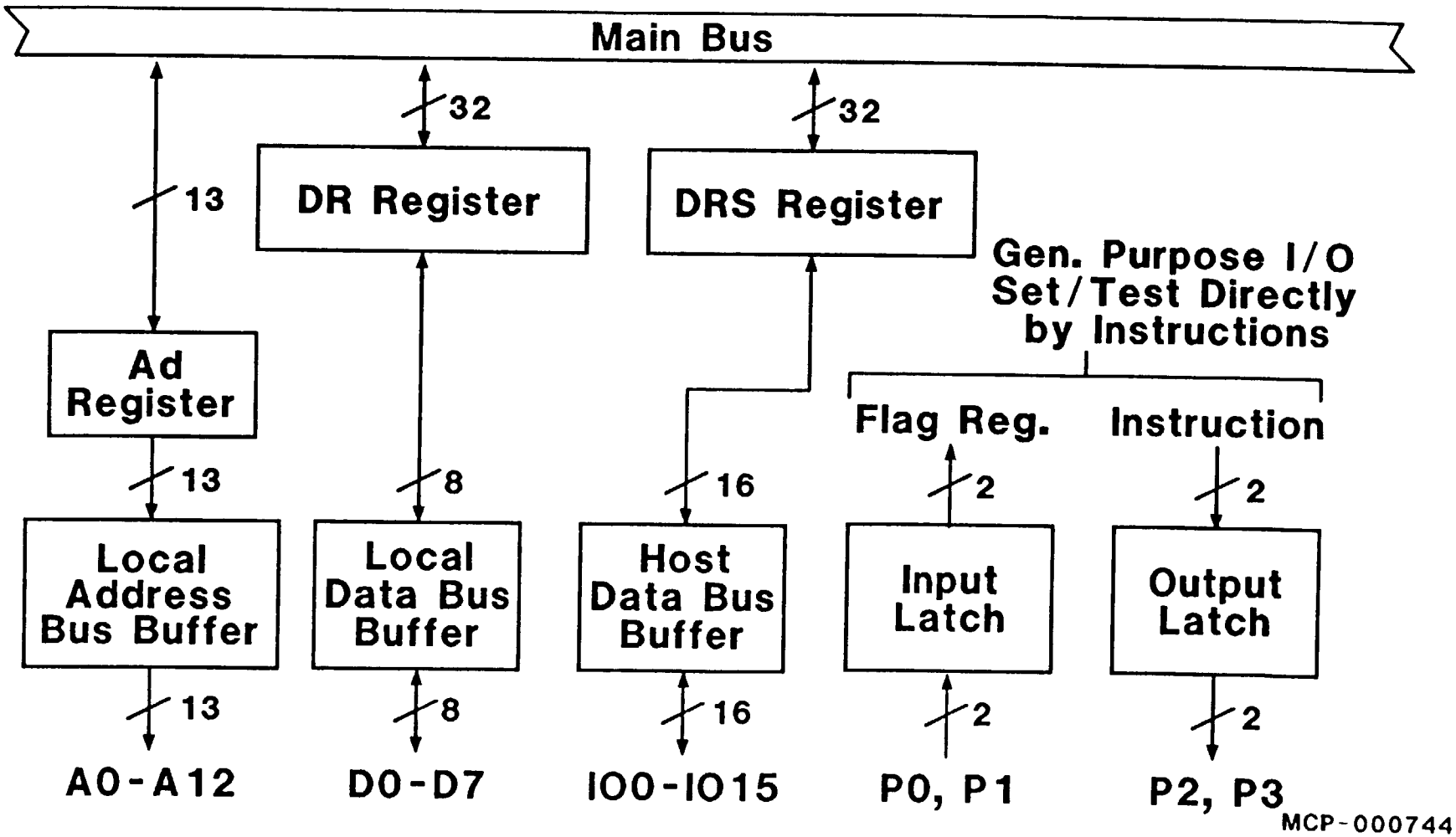
Parallel I/O - Slave Mode

NEC

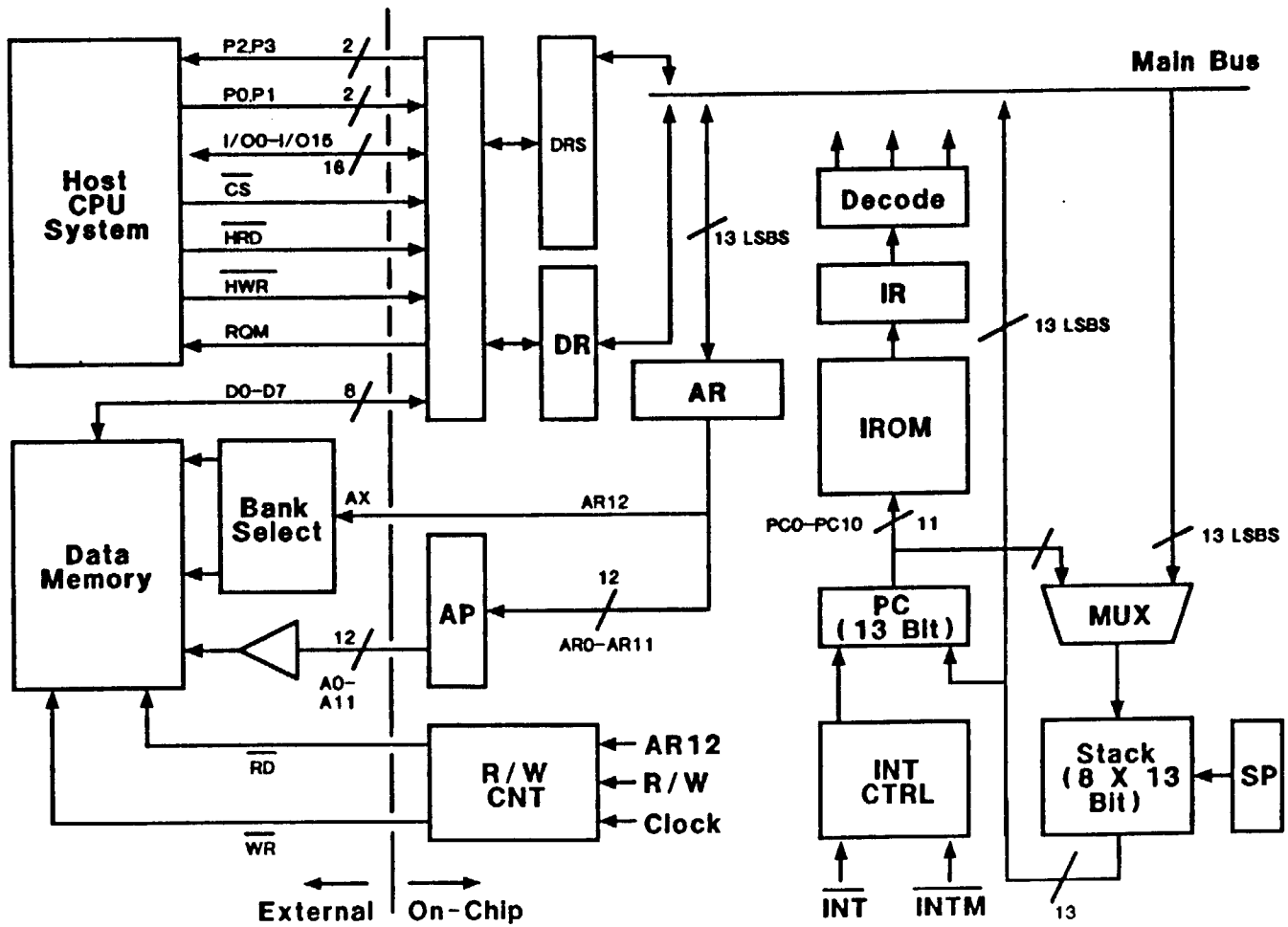
- 8-Bit Local Data Bus (Controlled by ASP)
  - -Programmable Data Length:
    8, 16, 24, or 32 Bits
- 16-Bit Host I/O Data Bus (Controlled by Host)
  - -Programmable Data Length:
    16 or 32 Bits
- Two General Purpose I/O Ports

Slave Mode Parallel Interface

NEC

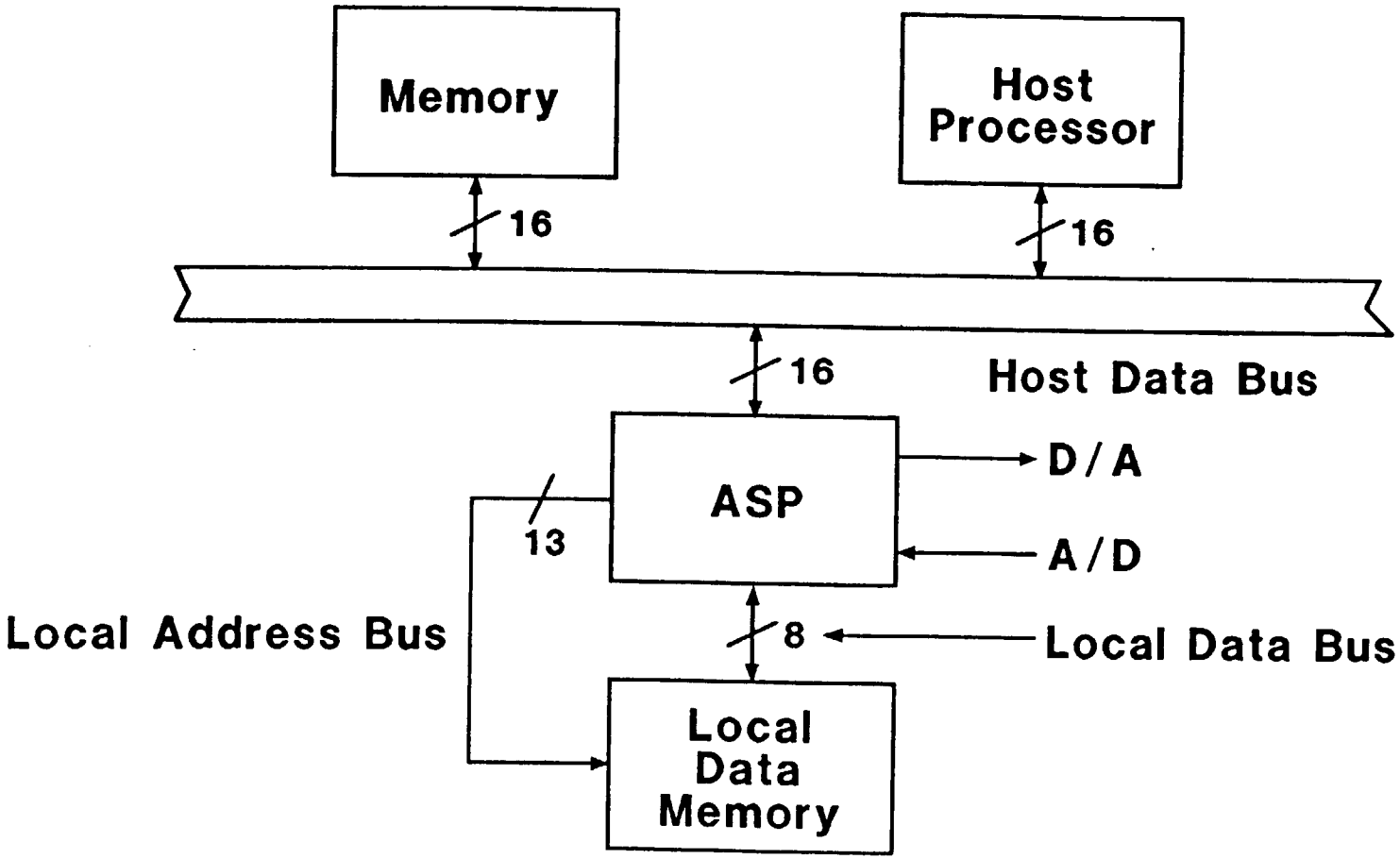


Slave Mode Parallel Interface



MCP-000745

Slave System Example



MCP-000746

Data Port Structure (DP0 - DP31)

NEC

Master	Slave
D 0	D 0
D 1	D 1
D 2	D 2
D 3	D 3
D 4	D 4
D 5	D 5
D 6	D 6
D 7	D 7
D 8	I/O 0
D 9	I/O 1
D 10	I/O 2

Master	Slave
D 11	I/O 3
D 12	I/O 4
D 13	I/O 5
D 14	I/O 6
D 15	I/O 7
D 16	I/O 8
D 17	I/O 9
D 18	I/O 10
D 19	I/O 11
D 20	I/O 12
D 21	I/O 13

Master	Slave
D 22	I/O 14
D 23	I/O 15
D 24	$\overline{\text{HRD}}$
D 25	$\overline{\text{HWR}}$
D 26	$\overline{\text{CS}}$
D 27	REQ
D 28	PO 0
D 29	PO 1
D 30	PO 2
D 31	PO 3

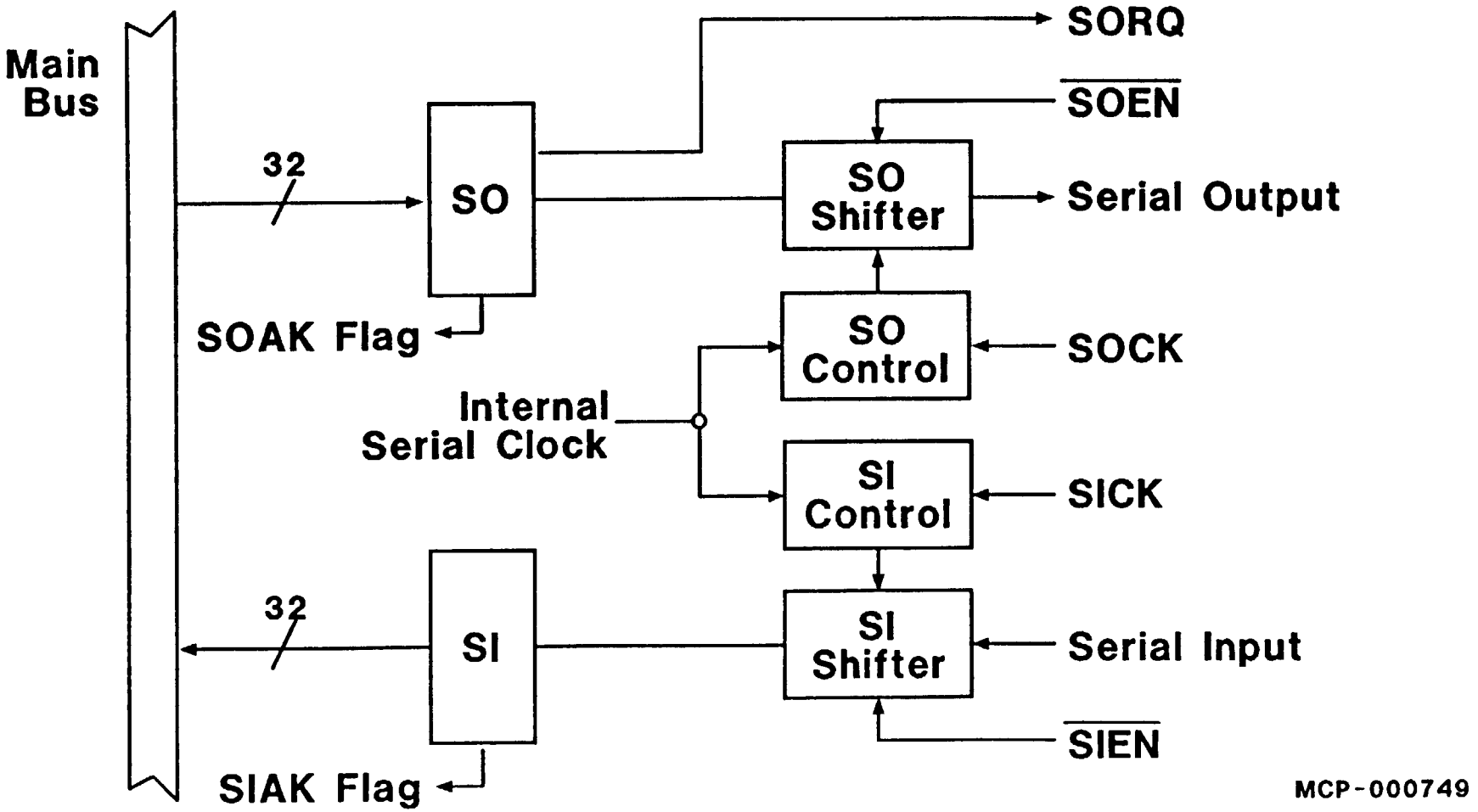
MCP-000747

Serial I/O

NEC

- Selectable Serial Clocks
  - Internal or External
- Separate Operation of SI & SO
- Programmable Transfer Data Length
  - 8/16/24/32 Bits
- **4.13 MHz Serial Clock Rate**

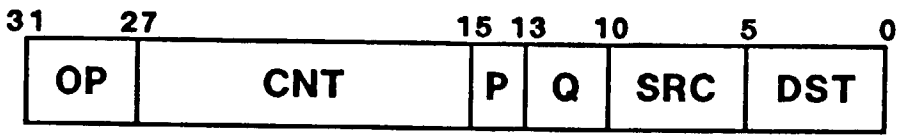
Serial I/O Interface



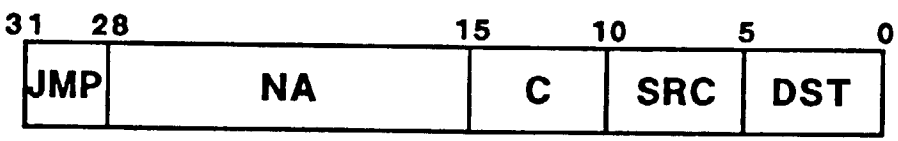
Instruction Types

NEC

Operation Instruction



Branch Instruction



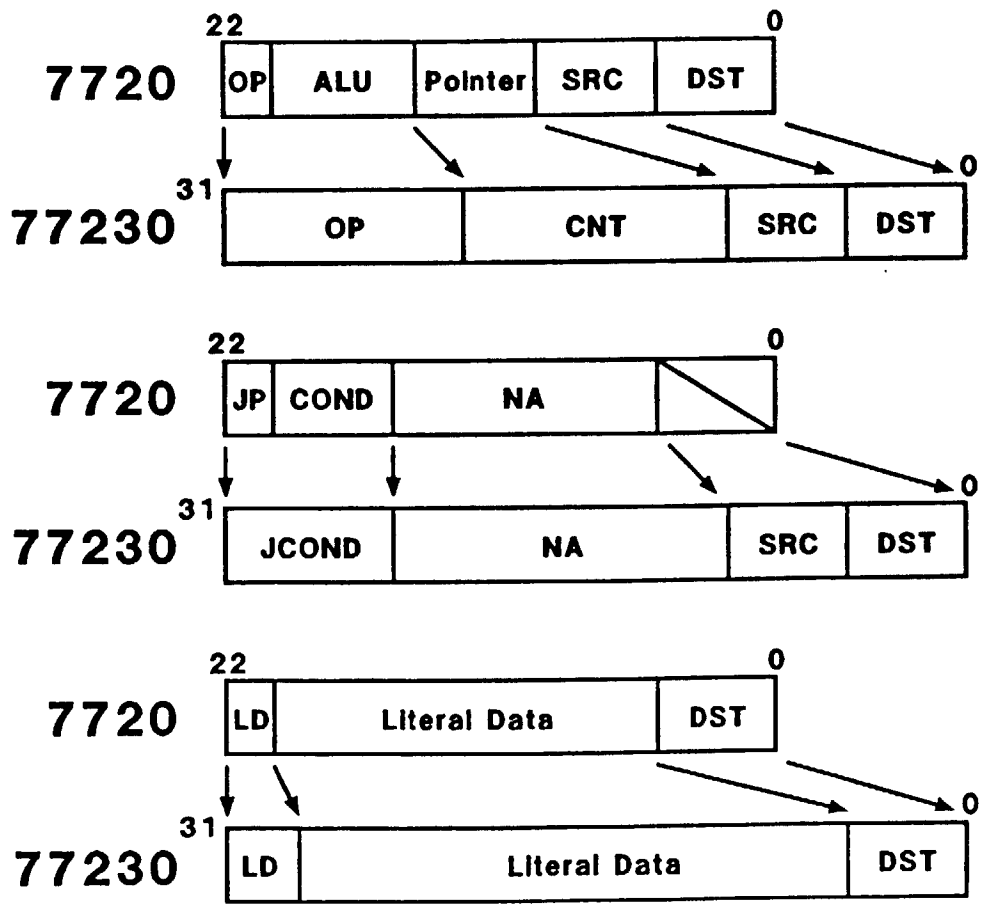
Load Instruction

OP - PU Operation
CNT - Control Field
P - P Reg. Input Source
Q - Working Reg. Select
NA - Next Address
C - Condition for Jump

MCP-000750

Instruction Comparison

NEC



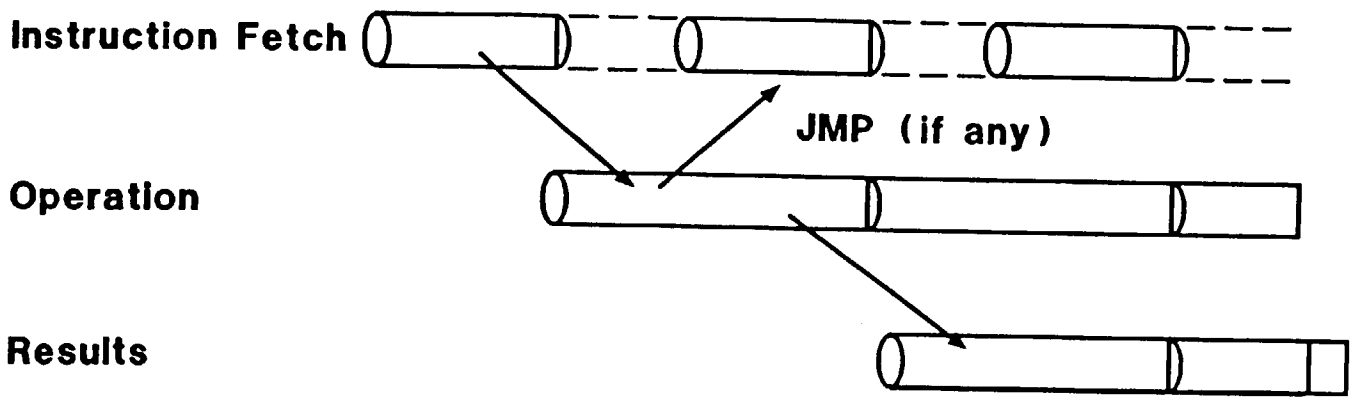
MCP-000752

PRECEDING PAGE BLANK NOT FILMED

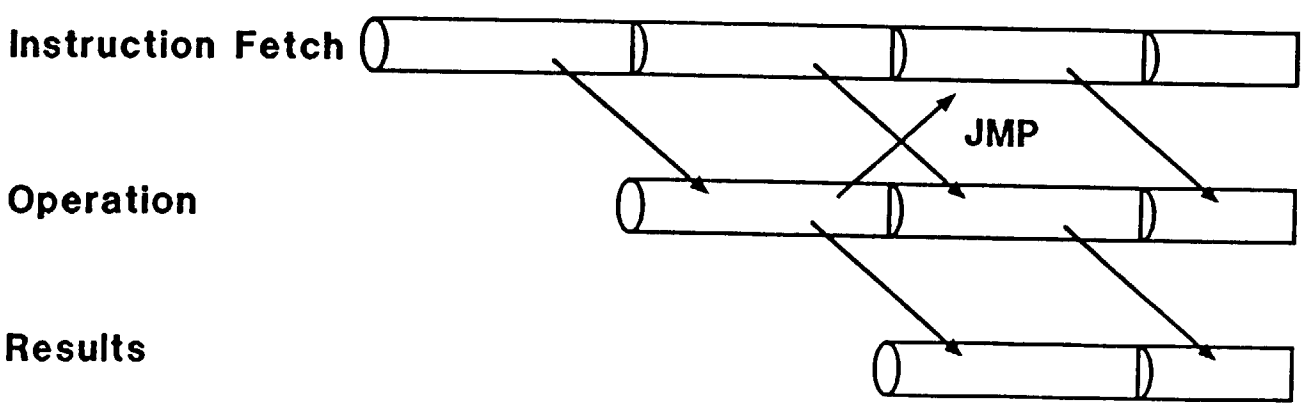
Instruction Timing Comparison

NEC

7720



77230



MCP-000753

Various PU Operations

NEC

- Floating Point ADD, SUB
- Fixed Point ADD, SUB
- Logical Operations
- N Value Shift or Rotate
- Working Register Clear

Special PU Operations

NEC

- Data Normalization Instructions
- Rounding and Normalization Instruction
- Data Type Conversion
  - Floating → Fixed
  - ASP ↔ IEEE Format
- Loop Counter Decrement

Branch Instructions

NEC

- Jump
- Conditional Branch
- Subroutine Call
- Return
- Local Branch
  - -OP Instr. Field: 9-Bit Address

Literal Load Instructions

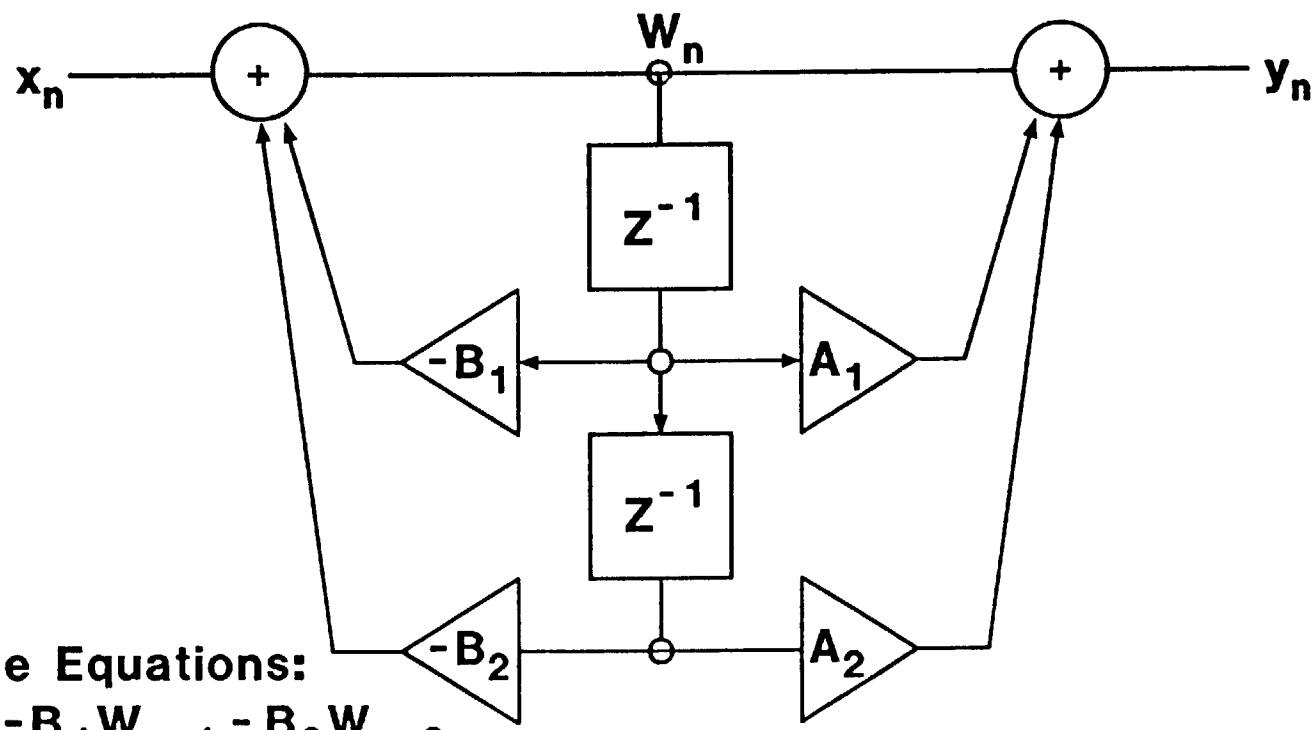
NEC

- Direct Data Load Capability:
  24-Bit Word
- 32-Bit Floating Point Data Load
  Capability Using TR Register

Instruction Examples

NEC

Biquad Filter Section



Difference Equations:

$$W_n = x_n - B_1 W_{n-1} - B_2 W_{n-2}$$

$$y_n = W_n + A_1 W_{n-1} + A_2 W_{n-2}$$

MCP-000758

Instruction Examples (cont)

NEC

Biquad Filter Section

Memory Map

Data RAM0

00	W_{n-1}
01	W_{n-2}
⋮	⋮

Data ROM

00	$-B_1$
01	$-B_2$
02	A_2
03	A_1
⋮	⋮

MCP-000759

Instruction Examples (cont)

NEC

Biquad Filter Section

Initial Conditions: WR0 = x_n, BP0 = 0H, RP = 0H

Program:

Step	Mnemonics		Comments
1.	MOV LKR0, ROM		K← W_{n-1} ; L← $-B_1$
	CLR WR1		clear WR1
	INCBP0		BP0 = 01H
	INCRP	;	RP = 01H
2.	MOV LKR0, ROM		K← W_{n-2} ; L← $-B_2$
	ADDF WR0, M		WR0 ← $x_n - B_1 W_{n-1}$
	INCRP	;	RP = 02H

MCP-000760

Instruction Examples (cont)

NEC

Biquad Filter Section

Program (cont)

Step	Mnemonics	Comments
3.	MOV LKR0, ROM	$K \leftarrow W_{n-2}$; $L \leftarrow A_2$
	ADDF WR0, M	$WR0 \leftarrow x_n - B_1 W_{n-1} - B_2 W_{n-2} = W_n$
	DECBP0	BP0 = 00H
	INCRP ;	RP = 03H
4.	MOV LKR0, ROM	$K \leftarrow W_{n-1}$; $L \leftarrow A_1$
	ADDF WR1, M ;	$WR1 \leftarrow A_2 W_{n-2}$

MCP-000761

Instruction Examples (cont)

NEC

Biquad Filter Section
Program (cont)

Step	Mnemonics		Comments
5.	MOV	RAM0,WR0	RAM0(00) ← W_n = new W_{n-1}
	ADDF	WR1,IB	WR1 ← $W_n = A_2 W_{n-2}$
	INCBP0		BP0 = 01H
	INCRP	;	RP = 04H (for next stage)
6.	MOV	RAM0,K	RAM0(01) ← W_{n-1} = new W_{n-2}
	ADDF	WR1,M	WR1 ← y (for next stage)
	INCBP0	;	BP0 = 02H (for next stage)

MCP-000762

uPD77230 Benchmarks

NEC

Operation	Execution Time
Add	0.15 us
Subtract	0.15 us
Multiplication	0.15 us
Division	4.8 us
Reciprocal	4.5 us
Root	6 us
SIN	10.8 us
COS	10.8 us
ATAN	40 us
Biquad Filter (1 Stage)	0.9 us
FIR Filter (32 Taps)	5.25 us
Complex FFT	
(32 Points)	150 us
(512 Points)	4.7 ms
(1024 Points)	12.6 ms (use external memory)

PRECEDING PAGE BLANK NOT FILMED

uPD77230 Support Tools

NEC

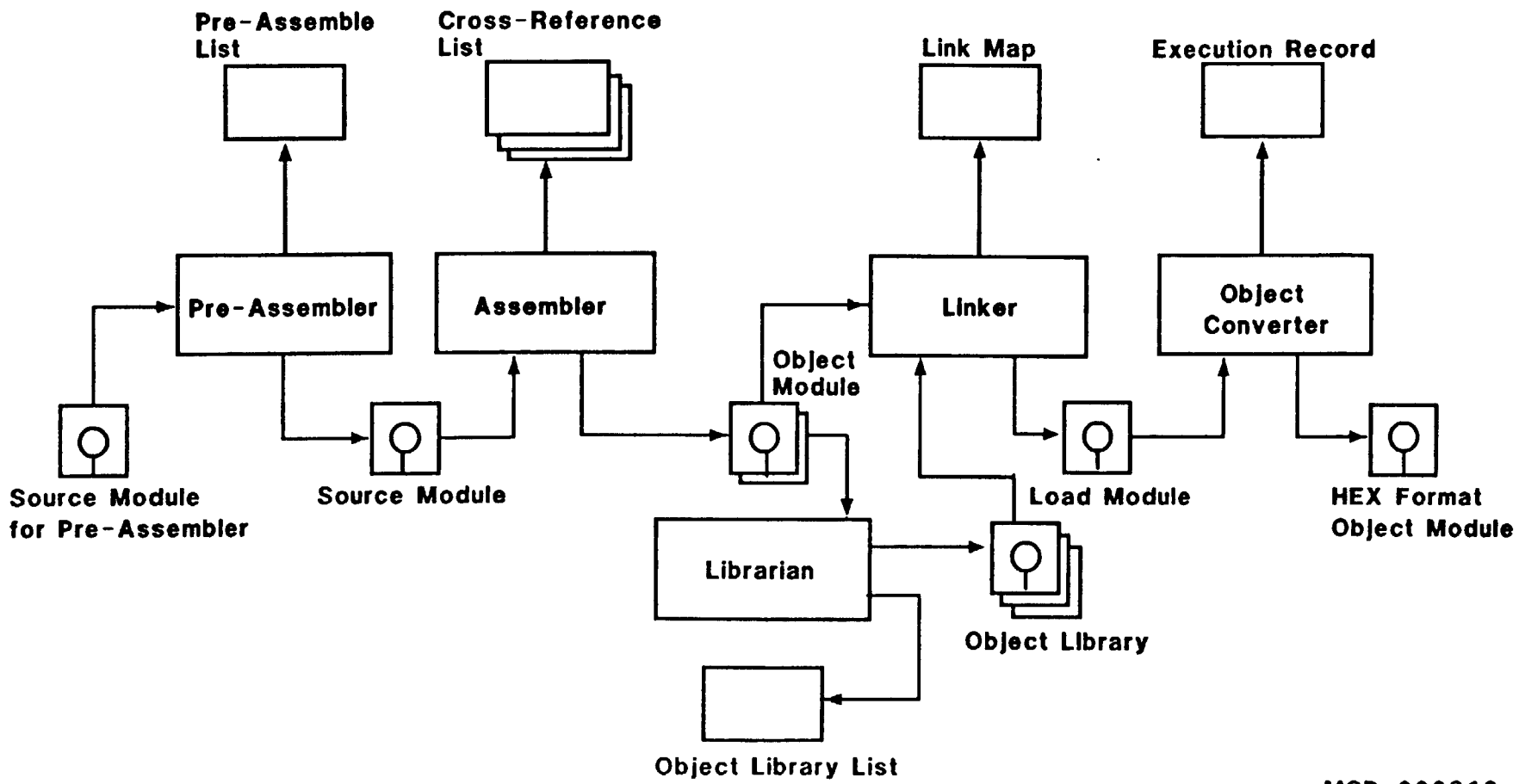
- o Relocatable Assembler Package
- o Software Simulator
- o In-circuit Emulator (Evakit)
- o Program Library
- o Evaluation Board

uPD77230 Assembler Package

NEC

- o Relocatable Code Modules
- o Macro Support
- o Librarian
- o Pre-assembler
- o Operating Systems
  - MS-DOS
  - VAX VMS, UNIX
  - CP/M-86, MP/M-86

ASP Assembler



PRECEDING PAGE BLANK NOT FILMED

uPD77230 Software Simulator

NEC

- o Supports ANY System Configuration
  - Master and Slave, Serial and Parallel I/O
- o Interactive and Batch Modes, Macro Commands
- o Symbolic Debugging
- o Log and Resource Files
- o In-line Assembly and Disassembly
- o Break, Trace, and Single Step Modes
- o VAX VMS or UNIX

uPD77230 Software Simulator

NEC

```
[uPD77230] <master mode><fast=non><slow=non><clock=12.50><step=    C11> NMI INT
LD: SIMDEMO.LNK      SI:                    PI:                    LG: TEMP.LOG
CD:                  SO: TEMP.SOD           PO: TEMP.POD           RS: TEMP.RES
<Register>                                                         IB:  FFFFFFFF000000
IR: 0007  JNFSI   SINAIT
PC: 0008     PSWS: PSW0    OE  C  Z  S  OM      SR: 00280   SP:  0   STK0: 1FFF
                   PSW0:    0  0  1  0  0       WI: BWRORD  SVR: 00  STK1: 0007
                   PSW1:    0  0  0  0  0       WT: WRBORD           STK2: 1FFF
WR0:  00000000000000                SI:  00000000                    STK3: 1FFF
WR1:  80000000000000                SO:  00000000                    STK4: 1FFF
WR2:  00000000000000                TR:  03400000                    STK5: 1FFF
WR3:  03400000000000                K:   00000000                    STK6: 1FFF
WR4:  00000000000000                L:   00000000                    STK7: 1FFF
WR5:  00000000000000                M:   00000000000000      DR:  00000080
WR6:  00000000000000                AR:  0FFF   LC:    3FF
WR7:  80000000000000                EM:  DI BM: NO_BOOKING   NF: TRNORM  FD: SPIE
M0:   BPO  -> 000 (80000000)        BPO: 000    BASE0: 3     IX0: 000
M1:   IX1  -> 000 (80000000)        BP1: 000    BASE1: 0     IX1: 000
ROM:  RP   -> 000 (00000000)        RP:  000    RPC:   0     RPS: 000
*break
*register
*go break 3
Bit break #3.
*
```

uPD77230 Evakit

NEC

- o Full speed In-circuit Emulation
- o Real-time Trace and Debug Features
- o Line Assembler and Disassembler
- o Internal/External Memory Mapping
- o Host Independent
- o Optional PROM Programmer

uPD77230 Program Library

NEC

- o Pre-programmed uPD77230
- o Master or Slave Mode Operation
- o Data I/O and Utility Routines
- o Floating Point Math Functions
  - Trig, Root, Log, etc.
- o Signal Processing Functions
  - FIR, IIR, FFT, etc.

uPD77230 Evaluation Board

NEC

- o IBM PC Plug-in Card with
  Pre-programmed Master Mode ASP
- o Full External Memory
  - 4K x 32 High-speed memory
    expandable to 16K x 32
  - 4K x 32 Low-speed memory
    expandable to 32K x 32
- o Serial I/O
  - On-board Codec/External Input
- o Simple Debug Features
  - Breakpoint and Register/RAM Dump

uPD 77220 - 24 BIT FIXED POINT DSP

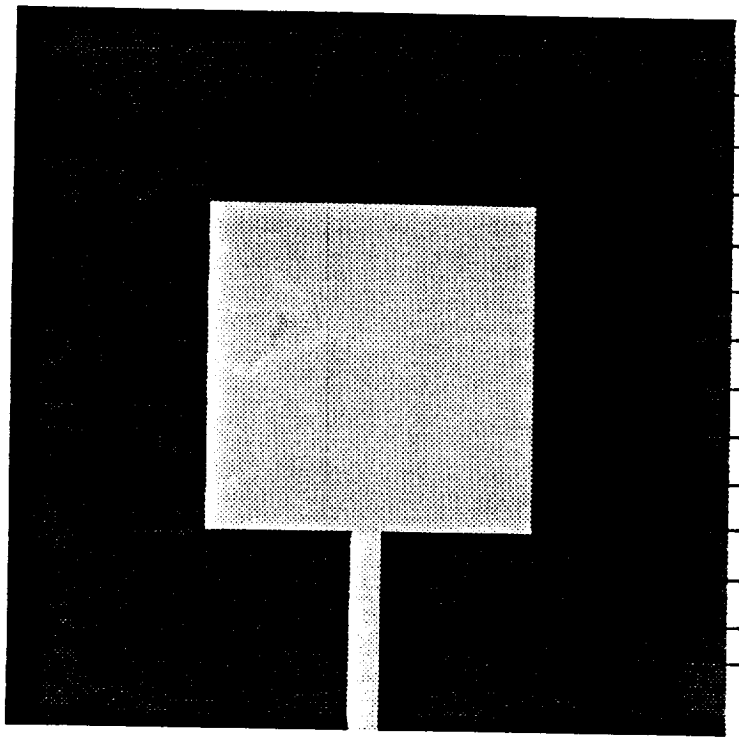
NEC

FEATURES

- PIN COMPATIBLE TO 77230, BUT NO FLOATING-POINT CAPABILITY IN HARDWARE
- ARITHMETIC 24-BIT FIXED-POINT ONLY
- SUB-SET INSTRUCTIONS OF 77230 (NO FLOATING-POINT INSTRUCTIONS)
- FAST INSTRUCTION EXECUTION (150/100 NS)
- REDUCED ON-CHIP MEMORY (2 x 256 x 24 BIT)
- EXTERNAL MEMORY SAME AS 77230

ANALOG DEVICES

DIGITAL SIGNAL PROCESSING DIVISION



ADSP-2100

Digital Signal Processor

General Description

- The ADSP-2100A a single-chip microprocessor optimized for digital signal processing applications.

- Seven major functional units provide capabilities to rival bit-slice/building block processors.

- Direct memory access to 16K of instruction codes and 32K of data at full speed.

- The architecture allows full parallelism in operations by performing the following all in one cycle:

  -- compute next program address
  -- fetch next instruction
  -- perform one or two data transfers
  -- update one or two address pointers
  -- arithmetic operation

- Powerful instruction set: all instructions can be executed in a single cycle.

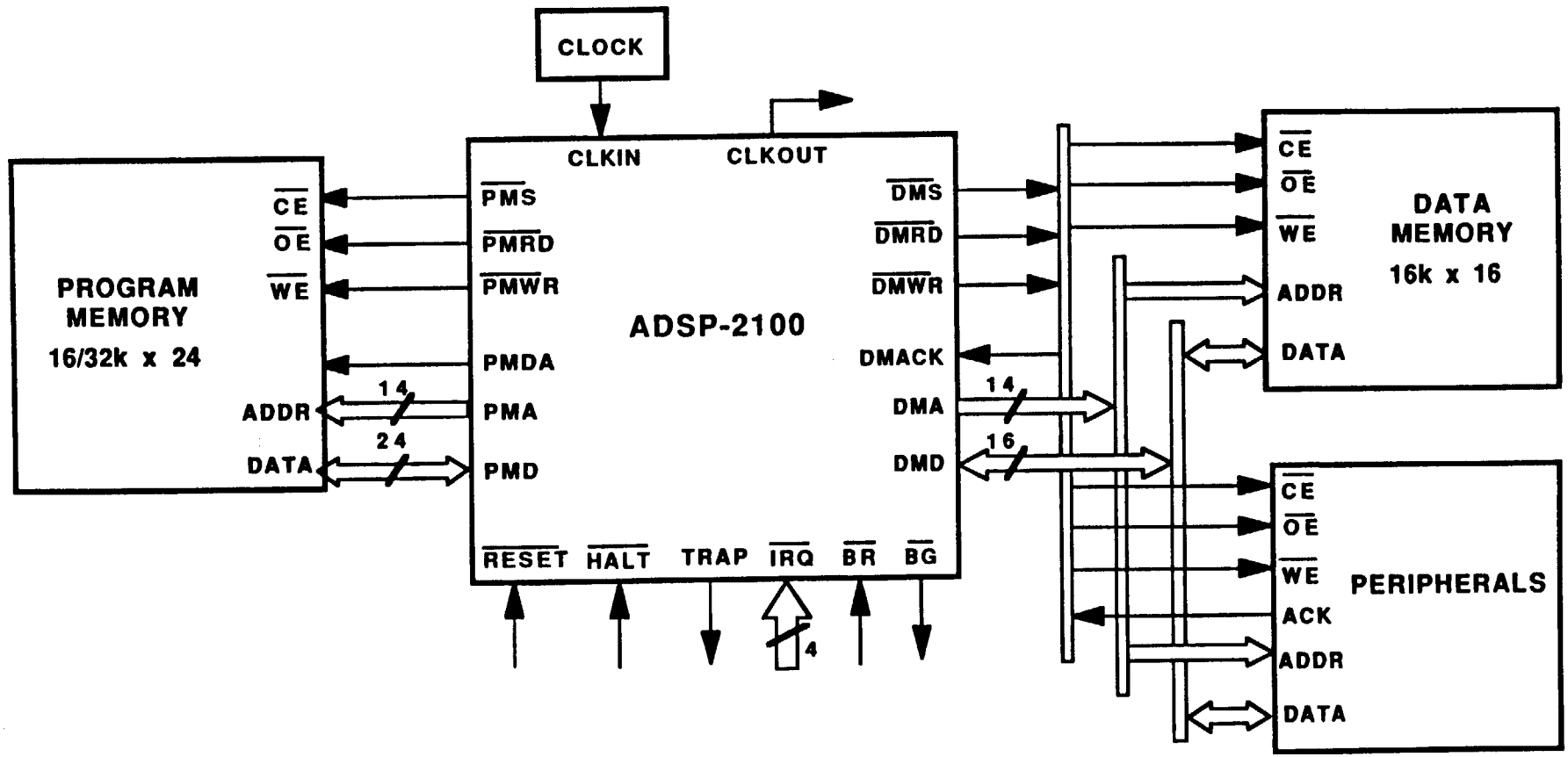
- A complete set of software and hardware development tools enables a short design cycle.

- The ADSP-2100 is fabricated in a high speed 1.0 μm double-layer metal CMOS process and housed in a 100-pin grid array package.

- 80 nsec cycle time.

PRECEDING PAGE BLANK NOT FILMED

ADSP-2100 Basic System Configuration



System Concepts

- Separate Program and Data buses

  -- 14-bit Program Memory Address (PMA)
  -- 24-bit Program Memory Data (PMD)

  -- 14-bit Data Memory Address (DMA)
  -- 16-bit Data Memory Data (DMD)

- Dual-Purpose Program Memory

  -- Program Memory can be used to store data as well as instructions (e.g. coefficients)

  -- Program Memory Write (PMWR) line enables both read/write to Program Memory

  -- Program Memory Data Access (PMDA) line distinguishes between instruction and data fetches, and can be used to expand Program Memory

- Memory-Mapped Peripheral Interface

  -- Wait state generation capability with Data Memory Acknowledge (DMACK) signal

- Control Signals

  -- External devices can gain control of program and data buses with bus request/grant signals

  -- Four external interrupt pins: internally prioritized and maskable; independently programmable as either edge-or level-sensitive

  -- Additional external control using RESET, HALT, and TRAP

- Clock Signals

  -- CLKIN at four times the processor cycle rate

  -- CLKOUT at the processor cycle rate and synchronized to the internal states

Internal Architecture

- Seven Major Functional Units:

  -- Program Sequencer
  -- Cache Memory
  -- Two Data Address Generators
  -- Arithmetic-Logic Unit (ALU)
  -- Mutliplier-Accumulator (MACC)
  -- Barrel Shifter

- Separate Program and Data Buses, Extended Off-Chip

  -- Data Memory Data (DMD) bus is the primary bus for routing data internally and to/from data memory

  -- Program Memory Data (PMD) bus provides instructions to the instruction register and a secondary path for routing data to/from program memory

  -- Direct connection of ALU/MACC input registers to both DMD and PMD buses allow dual operand fetches in a single cycle

- Program Sequencer allows single-cycled conditional branch and zero-overhead looping

- The internal instruction cache memory acts as an alternate source to the instruction register and can free the Program Memory for data accesses

- Two independent address generators provide data address pointers for both program memory and data memory

- Three independent computational units:

  -- side-by-side arrangement permits flexible operation sequencing

  -- input/output registers isolate computational operations from data fetches, allowing generous memory access times

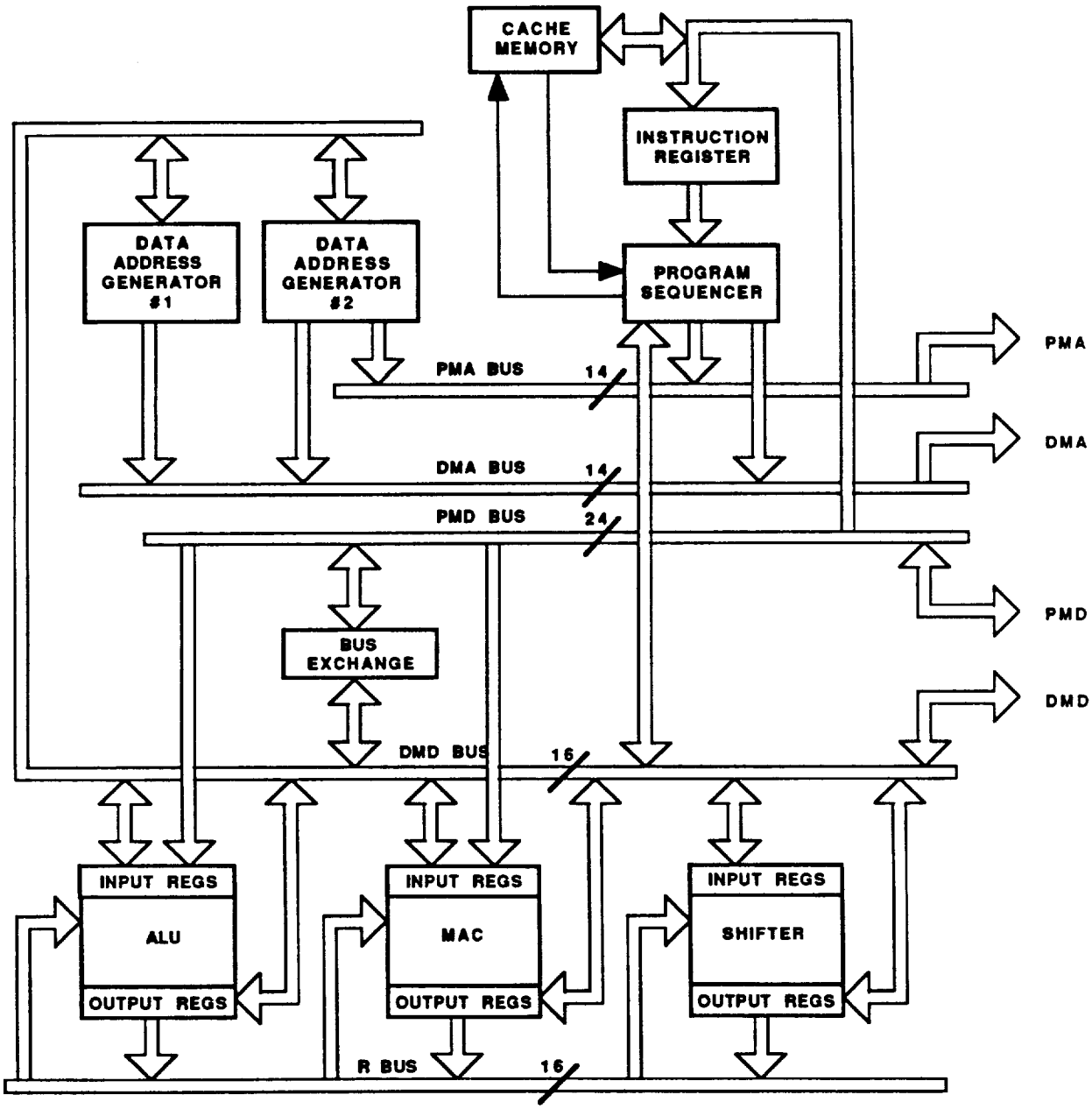
  -- two complete sets of registers for each computational unit allow fast context switching

  -- the R-bus provides a flexible interconnect path between computational units and eliminates unnecessary pipelining of operands

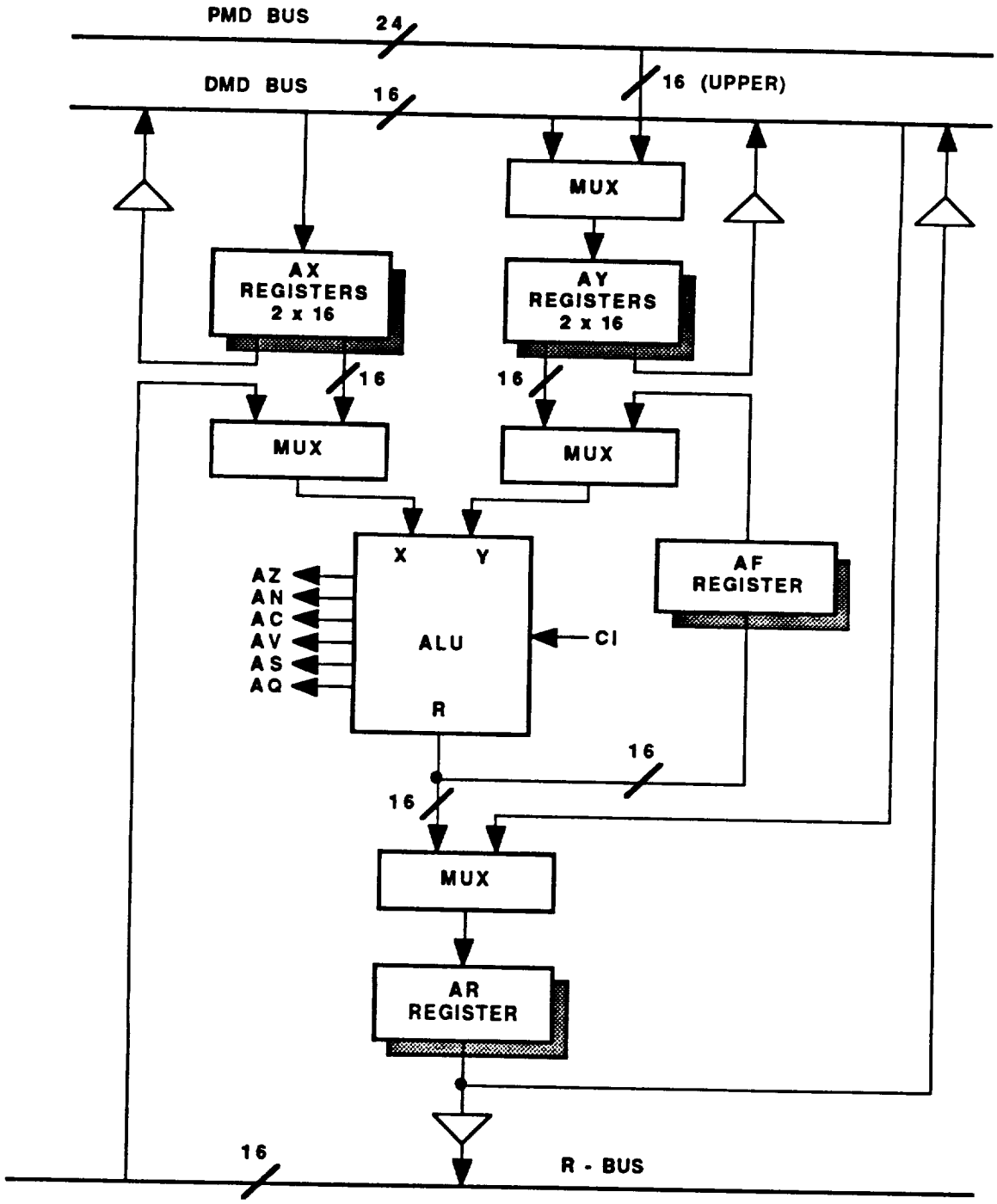
  -- all computational units have provisions for multi-precision operations

SEVEN MAJOR FUNCTIONAL UNITS

- PROGRAM SEQUENCER
- CACHE MEMORY
- TWO ADDRESS GENERATORS
- ARITHMETIC/LOGIC UNIT (ALU)
- MULTIPLIER/ACCUMULATOR (MAC)
- BARREL SHIFTER



2100–UM-2.1-6/86



ALU BLOCK DIAGRAM

ALU FEATURES

- FEEDBACK PATHS
- SIX STATUS FLAGS
- SATURATION
- IMPLEMENTS DIVIDE PRIMITIVES
- PROVISIONS FOR DOUBLE PRECISION
- COMPLETE SET OF BACKGROUND REGISTERS

ALU OPERATIONS

ARITHMETIC

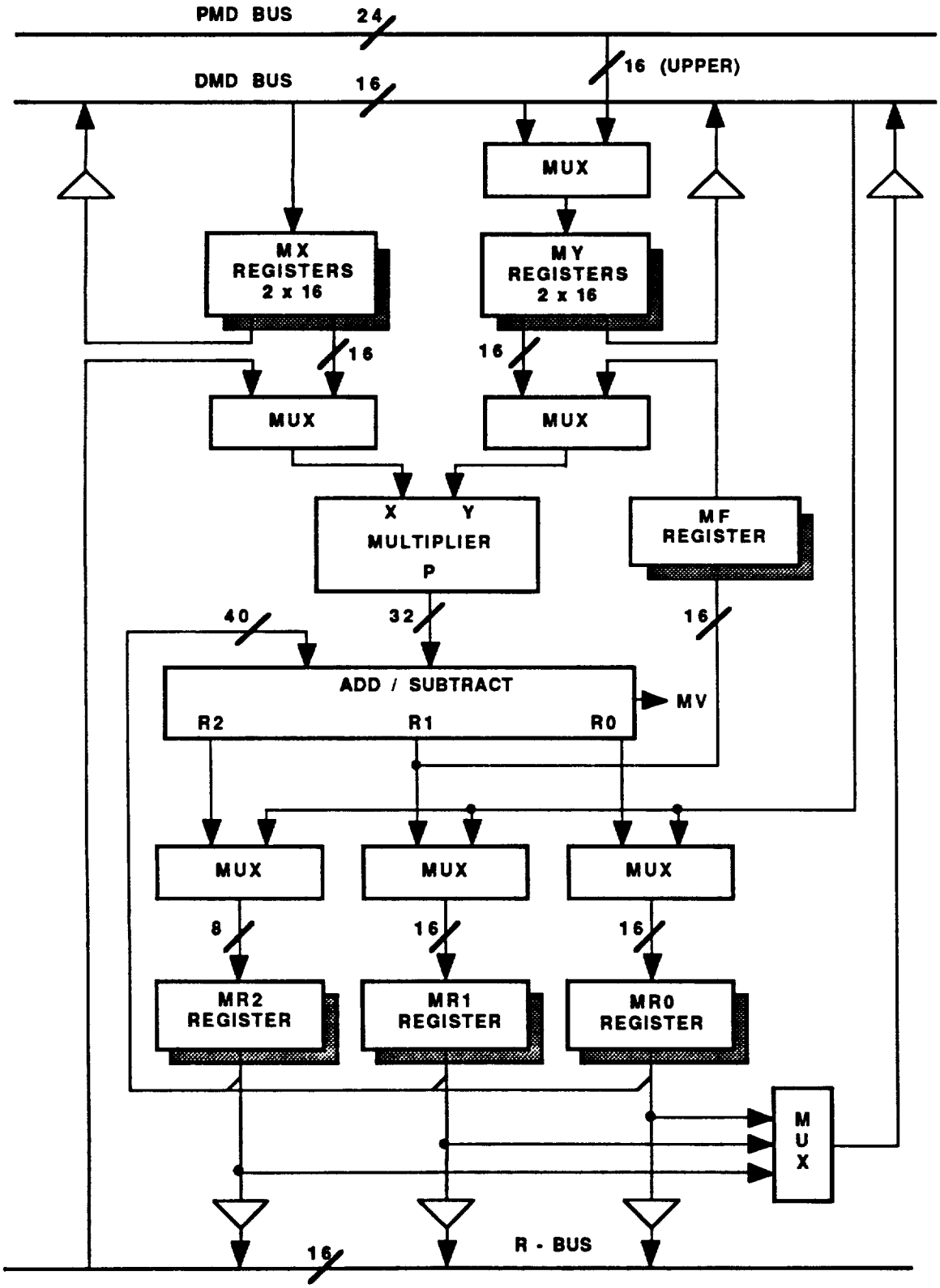
X + Y
X + Y + C
X - Y
X - Y + C - 1
Y - X
Y - X + C - 1
- X
- Y
Y + 1
Y - 1
DIVIDE PRIMITIVES

LOGICAL

X AND Y
X OR Y
X XOR Y
NOT X
NOT Y

OTHER

PASS X
PASS Y
ABS X
CLEAR



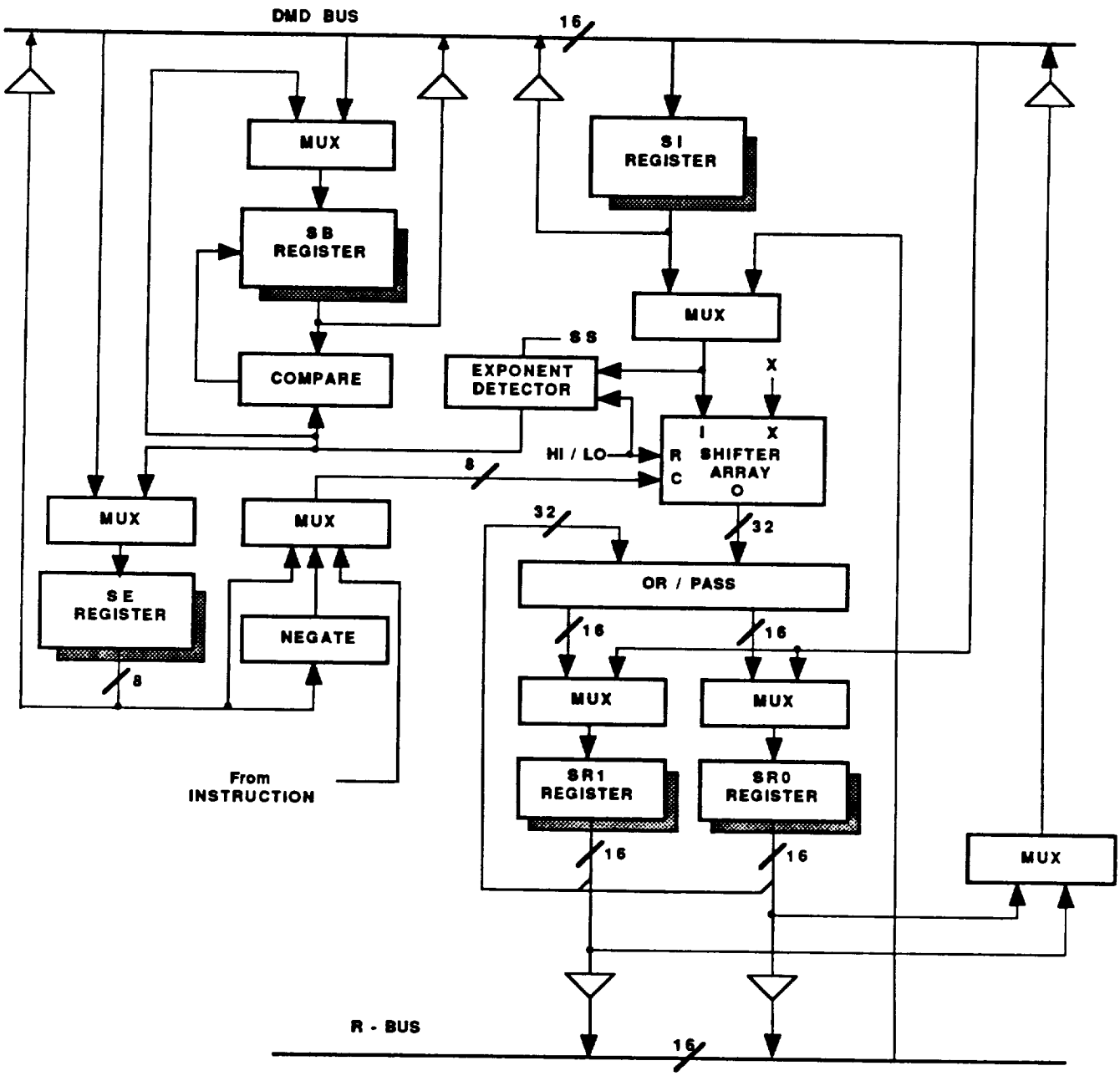
MAC BLOCK DIAGRAM

MAC FEATURES

- FEEDBACK PATHS
- 40-BIT ACCUMULATOR
- SATURATION
- MIXED MODE INPUT OPERANDS
- UNBIASED ROUNDING
- PROVISIONS FOR MULTIPRECISION
- COMPLETE SET OF BACKGROUND REGISTERS

MULTIPLIER/ACCUMULATOR OPERATIONS

OPERATION	INPUT FORMAT
X * Y	SIGNED * SIGNED
MR + X * Y	SIGNED * UNSIGNED
MR - X * Y	UNSIGNED * SIGNED
CLEAR	UNSIGNED * UNSIGNED
	ROUND



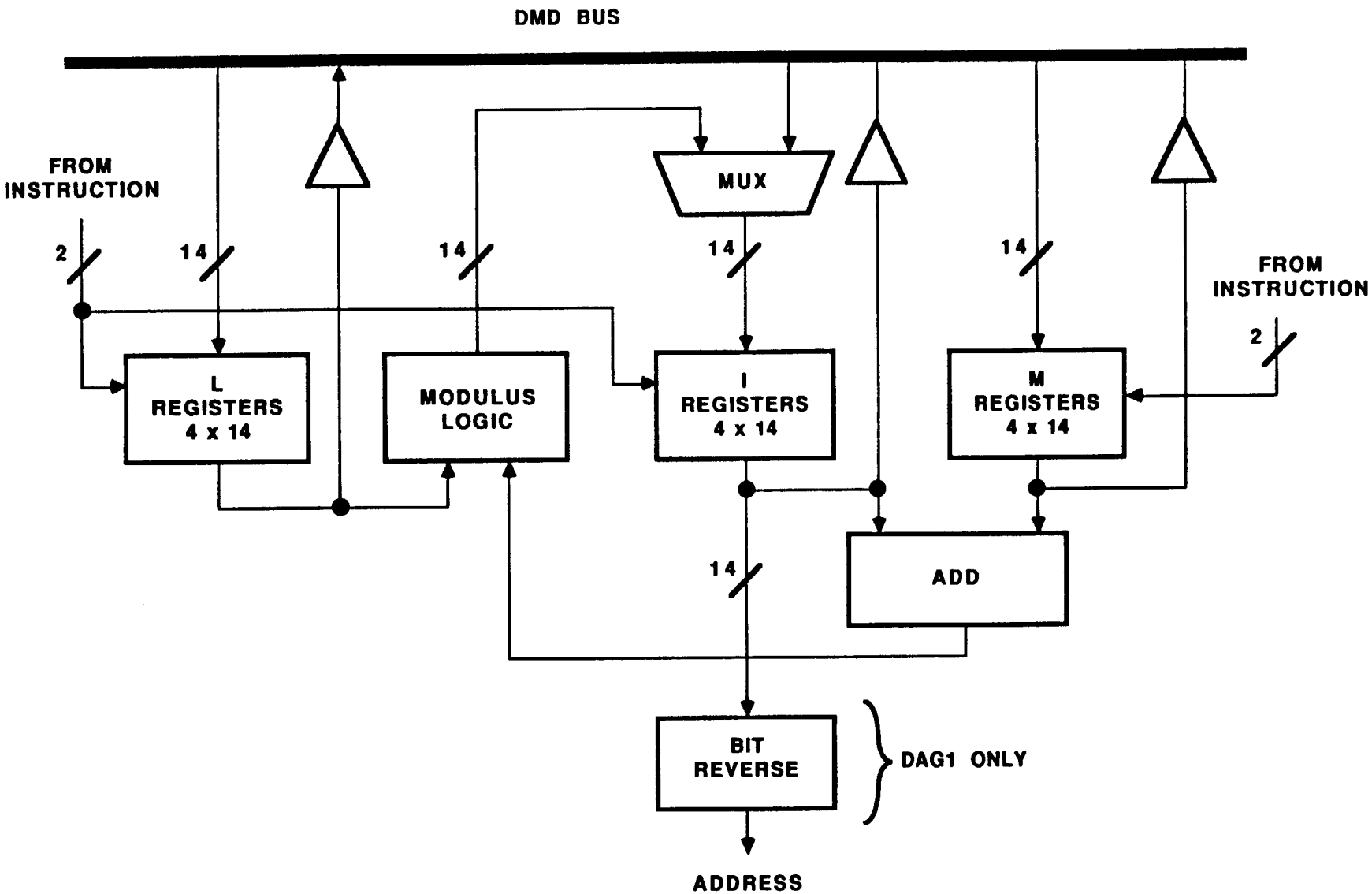
SHIFTER BLOCK DIAGRAM

SHIFTER FEATURES

- ARITHMETIC AND LOGICAL SHIFTS
- LEFT AND RIGHT SHIFTS
- TRUE BLOCK FLOATING POINT
- DIRECT SUPPORT FOR DOUBLE PRECISION
- COMPLETE SET OF BACKGROUND REGISTERS

SHIFTER OPERATIONS

- NORMALIZE
- DENORMALIZE
- SHIFT IMMEDIATE
- DERIVE EXPONENT
- DERIVE BLOCK EXPONENT

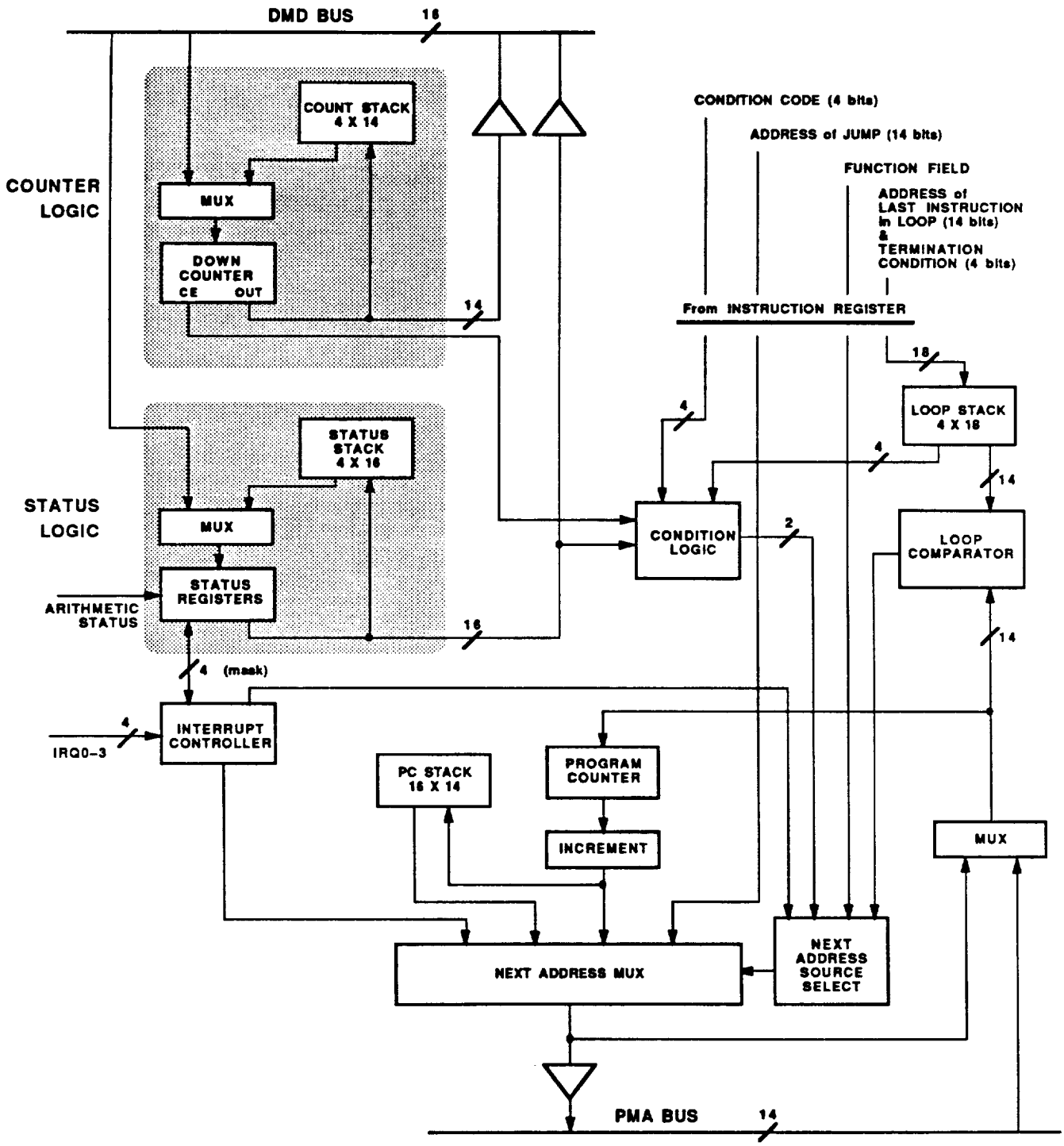

DMD BUS
FROM INSTRUCTION
2
14
14
14
14
MUX
FROM INSTRUCTION
2
L REGISTERS 4 x 14
MODULUS LOGIC
I REGISTERS 4 x 14
M REGISTERS 4 x 14
14
ADD
BIT REVERSE
DAG1 ONLY
ADDRESS

Data Address Generators

- Two independent Data Address Generators: DAG#1 and DAG#2
- DAG#1 has "bit reverse" option, and provides addressing for data memory
- DAG#2 provides addressing for both program and data memory, including instruction addresses for computed jump operations
- Each Data Address Generator contains:

  -- Four Indirect (I) registers
  -- Four Modify (M) registers
  -- Four Length (L) registers

  allowing the maintenance of eight data address pointers at a time
- Automatic modulo addressing for circular buffer implementation



SEQUENCER BLOCK DIAGRAM

Program Sequencer

- Four sources for the next instruction address:

  -- PC incrementer
  -- PC stack
  -- instruction register
  -- interrupt controller

- Four stacks:

  -- 16 x 14 PC Stack (16 levels of subroutines)
  -- 4 x 16 Status Stack (4 levels of interrupts)
  -- 4 x 18 Loop Stack (4 levels of nested loops)
  -- 4 x 14 Count Stack (5 levels of counters)

- Low interrupt processing overhead

  -- Two-cycle interrupt response time
  -- Interrupt nesting and lockout modes available
  -- Secondary data register set for single-cycle context switching

- Zero-overhead sequencing of nested DO UNTIL loops of arbitrary length

ADSP-2100 INSTRUCTION SET

INSTRUCTION CATEGORIES

- DATA MOVE INSTRUCTIONS
- COMPUTATIONAL INSTRUCTIONS
- PROGRAM FLOW INSTRUCTIONS
- MISCELLANEOUS

DATA MOVE INSTRUCTIONS

REGISTER	⟷	REGISTER
REGISTER	⟷	DATA MEMORY
REGISTER	⟷	PROGRAM MEMORY
IMMEDIATE VALUE	→	REGISTER
IMMEDIATE VALUE	→	DATA MEMORY

EXAMPLES:

```
MX0 = AY1;
AX0 = DM (ADDRESS);
AY0 = PM (I4, M6);
SI = H#30F8;
DM (I0, M0) = 167;
```

COMPUTATIONAL INSTRUCTIONS

CONDITIONAL **ALU, MAC,** OR **SHIFT** OPERATION

EXAMPLE: IF MV SR = ASHIFT MR1 BY -1 (HI);

ALU, MAC, OR **SHIFT**

- REGISTER ↔ REGISTER
- OR
- REGISTER ↔ DATA MEMORY
- OR
- REGISTER ↔ PROGRAM MEMORY

EXAMPLE: AR = AR + AY0, AY0 = DM (I0, M1);

ALU OR **MAC**

- DATA MEMORY → REGISTER
- AND
- PROG. MEMORY → REGISTER

EXAMPLE:

MR = MR + MX0 * MY0 (SS), MX0 = DM(I1, M3), MY0 = PM(I7, M4);

PROGRAM FLOW CONTROL

- CONDITIONAL JUMP
- CONDITIONAL SUBROUTINE CALL
- CONDITIONAL RETURN
- CONDITIONAL TRAP
- CONDITIONAL *DO UNTIL*

MISCELLANEOUS

- SATURATE ACCUMULATOR
- MODIFY INDEX REGISTER
- PUSH STATUS STACK
- POP STATUS/LOOP/COUNTER/PC STACK
- MODE CONTROL
- NOP

DEVELOPMENT TOOLS

- SYSTEM BUILDER
- ASSEMBLER
- LINKER
- PROM SPLITTER
- C COMPILER
- SIMULATOR/DEBUGGER
- EMULATOR
- EVALUATION BOARD

SYSTEM BUILDER

- ALLOWS THE USER TO SPECIFY TARGET HARDWARE
- USES HIGH LEVEL CONSTRUCTS

FIR FILTER
ARCHITECTURE DESCRIPTION

```
.SYSTEM                          FILTER_SYSTEM;
.SEG/ROM/ABS=0/PM/CODE           PROG_MEM[4096];
.SEG/ROM/ABS=4096/PM/DATA        COEFF_STOR[256];
.SEG/RAM/ABS=0/DM/DATA           DELAY_LINE[256];
.PORT/ABS=H#3FFE                 AD_SAMPLE;
.PORT/ABS=H#3FFF                 DA_DATA;
.ENDSYS;
```

ASSEMBLER

- SUPPORTS HIGH LEVEL CONSTRUCTS
- SUPPORTS FLEXIBLE MACRO PROCESSING
- ENCOURAGES MODULAR CODE DEVELOPMENT
- PROVIDES A FULL RANGE OF DIAGNOSTICS

FIR FILTER
ASSEMBLY PROGRAM

```
.MODULE/ROM     FIR_ROUTINE;
.CONST          TAPS=15;
.PORT           AD_SAMPLE;
.PORT           DA_DATA;
.ENTRY          FIR_START;

FIR_START:      CNTR=TAPS-1;
                SI=DM(AD_SAMPLE);
                DM(I0,M0)=SI;
                MR=0,MY0=PM(I4,M4),MX0=DM(I0,M0);

                DO CONVOLUTION UNTIL CE;
CONVOLUTION:    MR=MR+MX0*MY0(SS),MY0=PM(I4,M4),MX0=DM(I0,M0);
                MR=MR+MX0*MY0(RND);
                IF MV SAT MR;
                DM(DA_DATA)=MR1;
                RTI;
.ENDMOD;
```

LINKER

- SUPPORTS MULTI-MODULE LINKING
- MAPS ASSEMBLER OUTPUT TO SYSTEM BUILDER

SIMULATOR

- USER-FRIENDLY INTERFACE: INTERACTIVE AND SYMBOLIC
- SUPPORTS FULL SYMBOLIC ASSEMBLY/DISASSEMBLY
- SIMULATES HARDWARE CONFIGURATION
- SIMULATES I/O HANDLING
- FLAGS ILLEGAL OPERATIONS
- DISPLAYS THE INTERNAL OPERATIONS OF THE ADSP-2100

ADSP-2100 Simulator Screen

```
ADSP-2100 Simulator V1.4-00 Analog Devices Inc.      FIR_SYSTEM
                ALU                                   Address Generator #1
AX0 uuuu                          AC 0  AQ 0        I0 0009   M0 0001   L0 000F
AX1 uuuu                AR uuuu   AN 0               I1 uuuu   M1 uuuu   L1 0000
AY0 uuuu                AF uuuu   AV 0               I2 uuuu   M2 uuuu   L2 0000
AY1 uuuu                          AZ 0  AS 0        I3 uuuu   M3 uuuu   L3 0000
            Multiplier-Accumulator                                      Addr
MX0 2000                                              Address Generator #2
MX1 uuuu     MR2 00     MR1 004A     MR0 0644        I4 1000   M4 0001   L4 000F
MY0 FE8C                MF  0000                     I5 uuuu   M5 uuuu   L5 0000
MY1 uuuu                          MV 0               I6 uuuu   M6 uuuu   L6 0000
                Shifter                               I7 uuuu   M7 uuuu   L7 0000
SI  2000     SE uu      SR1 uuuu     SR0 uuuu                  Addr
             SB uu                   SS 0            Data DM 004A  PM 3C00E5  PX 00
CNTR 0000  Astat 00  Mstat 0  Sstat 44   Imask 0  Icntl 00  PC 0015  Time 125200
                halt  : Module: FIR_ROUTINE                 : emulat  register hex
dspp_sim()-CMD_INP_ACKL:

>
```

BASIC SCREEN

ADSP-2100 Simulator Screen

```
ADSP-2100 Simulator V1.4-00 Analog Devices Inc.      FIR_SYSTEM

PM Addr        Contents     Disassembled Contents

0011  rom      14012E       do h#0012  until ce;
0012  rom      A00000       dm(i0,m0)=^DATA_BUFFER/h#0000 ;
0013  rom      3C00F3       imask=h#000F ;
0014  rom      18014F       jump h#0014 ;
0015  rom pc>  3C00E5  FIR_START: cntr=h#000E ;
0016  rom      83FFE8       si=dm(AD_SAMPLE);
0017  rom      680080       dm(i0,m0)=si;
0018  rom      E89800       mr=mx0*0(ss),mx0=dm(i0,m0),my0=pm(i4,m4);
0019  rom      1401FE       do CONVOLUTION until ce;
001A  rom      E90000  CONVOLUTION:mr=mr+mx0*my0(ss),mx0=dm(i0,m0),my0=pm(i4,m4);
CNTR 0000  Astat 00  Mstat 0  Sstat 44  Imask 0  Icntl 00  PC 0015  Time 125200
                 halt  : Module: FIR_ROUTINE            : emulat  pmemory hex
dspp_sim()-CMD_INP_ACKL:
>
>
>
>
> PM
> Address:fir_start
>
```

PROGRAM MEMORY

ADSP-2100 Simulator Screen

```
ADSP-2100 Simulator V1.4-00 Analog Devices Inc.      FIR_SYSTEM

DM Addr        Contents

0000  ram      030E    DATA_BUFFER:
0001  ram      F698
0002  ram      EB0E
0003  ram      0000
0004  ram      14F1
0005  ram      0967
0006  ram      FCF1
0007  ram      0F37
0008  ram      2000
0009  ram      EB0E
CNTR 0000  Astat 00  Mstat 0  Sstat 44  Imask 0  Icntl 00  PC 0015  Time 125200
                 halt  : Module: FIR_ROUTINE              : extend  dmemory hex
dspp_sim()-CMD_INP_ACKL:
>
>
>
>
> DM
> Address:
>
```

DATA MEMORY

ADSP-2100 Simulator Screen

```
ADSP-2100 Simulator V1.4-00 Analog Devices Inc.      FIR_SYSTEM
     ADDSYMBOL     ALTERNATE      BEEP           BACKUP         CACHE
     CHIPRESET     CLEARBREAK     CLEARSTOPTIME  CLEARTIME      CLEARWATCH
     CLOSE         COMMFILE       COUNT          CYCLETIME      DECIMAL
     DELETE        DM             DUMPDM         DUMPPM         EMULATOR
     EXECUTE       EXIT           EXTEND         FINDDM         FINDPM
     FORWARD       HARDWARE       HELP           HEXADECIMAL    INTERRUPT
     LOAD          MODULES        NOBEEP         NOSYMBOLIC     OPEN
     PATCH         PLOTDM         PM             POWERUP        PRIMARY
     READIMAGE     READSYMBOL     REGISTER       REMOVESYMBOL   RESETSTACK
     RUN           SETBREAK       SETDM          SETMODULE      SETPC
     SETPM         SETREGISTER    SETSTOPTIME    SETWATCH       SINGLESTEP
     STACK         STATUS         SYMBOLIC       TOGGLE         TRACE
     WIPE          XREFERENCE
CNTR 0000  Astat 00  Mstat 0  Sstat 44  Imask 0  Icntl 00  PC 0018  Time  10200
                 halt  : Module: FIR_ROUTINE              : emulat trace  hex
dspp_sim()-CMD_INP_ACKL:
>
>
>
>
> help
> help_on?:
```

HELP

EMULATOR

- IN-CIRCUIT EMULATION
- NO DEGRADATION IN PROCESSOR PERFORMANCE
- INTERFACES TO EXTERNAL INSTRUMENTATION
- SELF-EMULATION OF PROCESSOR
- SUPPORTS FULL SYMBOLIC DISASSEMBLY
- SAME INTERACTIVE, SYMBOLIC USER INTERFACE AS THE SIMULATOR

EVALUATION BOARD FEATURES

- LOW COST DEVELOPMENT TOOL
- EVALUATE ADSP-2100A IN REAL TIME APPLICATIONS
- ANALOG I/O CHANNEL
- LINEAR DAC
- SAME USER INTERFACE AS EMULATOR AND SIMULATOR
- 4K DATA MEMORY, 2K PROGRAM MEMORY
- EXPANDABLE TO 32K DATA AND 16K PROGRAM MEMORY

C COMPILER

- COMPLIES WITH ANSI DRAFT STANDARD XJ311
- VALIDATED WITH COMMERCIALLY AVAILABLE VALIDATION SUITE
- PRODUCES ROMABLE CODE
  - LITERALS CAN BE PLACED IN RAM/ROM AND DM/PM
  - SWITCH STATEMENT JUMP TABLES RAM/ROM AND DM/PM
  - RUN TIME STACK RAM/ROM AND DM/PM
- NEW DATA TYPE FRACT (1.15 FORMAT)
- IN LINE ASSEMBLY CODE SUPPORTED
- ASSEMBLY SUBROUTINES CALLABLE
- COMPLETE FLOATING POINT EMULATION LIBRARY

BULLETIN BOARD SERVICE

- ALLOWS UPLOAD AND DOWNLOAD OF PROGRAMS AND PRODUCT INFO
- INFORMATION ON ALL DSP PRODUCTS
- MAIL FACILITY PERMITS COMMUNICATION WITH APPLICATIONS ENGINEERING
- QUESTION AND ANSWER FACILITY
- AUTOBAUD SUPPORTS 300, 1200, 2400 BAUD RATES

(617) 528 - 5012
NO PARITY, 1 STOP BIT
FULL DUPLEX

BENCHMARK PERFORMANCE OF THE ADSP-2100

(12.5 MHz VERSION)

ROUTINE	EXECUTION TIME
FIR FILTER	80 nS PER TAP
COMPLEX FIR FILTER	320 nS PER TAP
BIQUAD FILTER SECTION	560 nS PER SECTION
LATTICE FILTER SECTION	400 nS PER SECTION
256-POINT RADIX-2 FFT	0.88 mS
256-POINT RADIX-4 FFT	0.59 mS
1024-POINT COMPLEX RADIX-2 FFT	4.2 mS
1024-POINT COMPLEX RADIX-4 FFT	2.9 mS
4096-POINT COMPLEX RADIX-2 FFT	19.8 mS
4096-POINT COMPLEX RADIX-4 FFT	14.2 mS
256-TAP LMS ADAPTIVE FILTER UPDATE	41.5 μS
TENTH ORDER LPC ANALYSIS (240-Point rectangular window)	0.36 mS

AVAILABLE ADSP-2100 LITERATURE

- ADSP-2100 USERS MANUAL
- ADSP-2100 CROSS-SOFTWARE MANUAL
- ADSP-2100 APPLICATIONS HANDBOOK VOL. 1
- ADSP-2100 APPLICATIONS HANDBOOK VOL. 2
- ADSP-2100 EMULATOR MANUAL
- ADSP-2100 EVALUATION BOARD MANUAL

TS68930/31

DIGITAL SIGNAL PROCESSOR

ORIGINAL PAGE IS
OF POOR QUALITY

TELECOM MARKETING

OCTOBRE 86

SGS-THOMSON
MICROELECTRONICS

PRECEDING PAGE BLANK NOT FILMED

- TS 68930 -

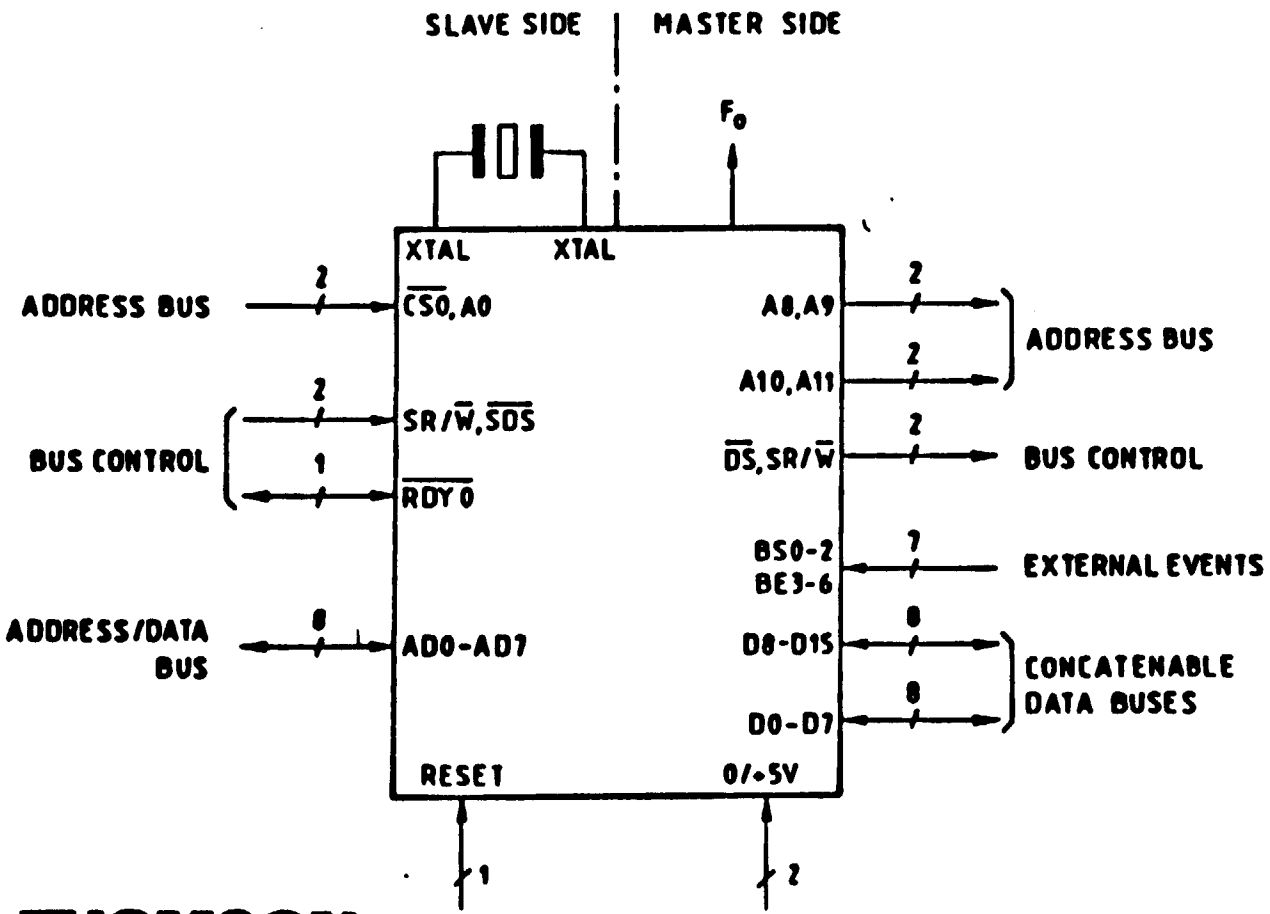
16/32 BIT MICROCOMPUTER:

• INSTRUCTION CYCLE:	160/ 320
• COMP. MODES:	REAL, COMPLEX, DOUBLE PREC
• INSTRUCTION WIDTH:	32 BIT
• INSTRUCTION FORMATS:	8
• DATA BUSES:	16 BITS X 3
• DATA RAM:	2 X 128 X 16/2 X 64 X 32
• DATA ROM:	1 X 512 X 16/1 X 256 X 32
• EXT. MEMORY:	4 K X 16 / 2 K X 32
• HARD MULTIPLIER:	12,5 M X (16 X16)/ SEC
• INT PROGRAM ROM:	1 K X 32 BITS
• EXT PROG ROM (84 PIN):	64 K X 32 BITS
• PACKAGING:	- 48 PIN MICROCOMPUTER (TS 68930)
	- 84 PIN MICROPROCESSOR (TS 68931)
• TECHNOLOGY:	2 µM N-MOS - TANTALLUM SILICIDE
• COMPLEXITY:	120 000 DEVICES
• SILICON AREA:	57 MM^2
• POWER:	1,5 W (48 PIN)

PRECEDING PAGE BLANK NOT FILMED

T E L E C O M

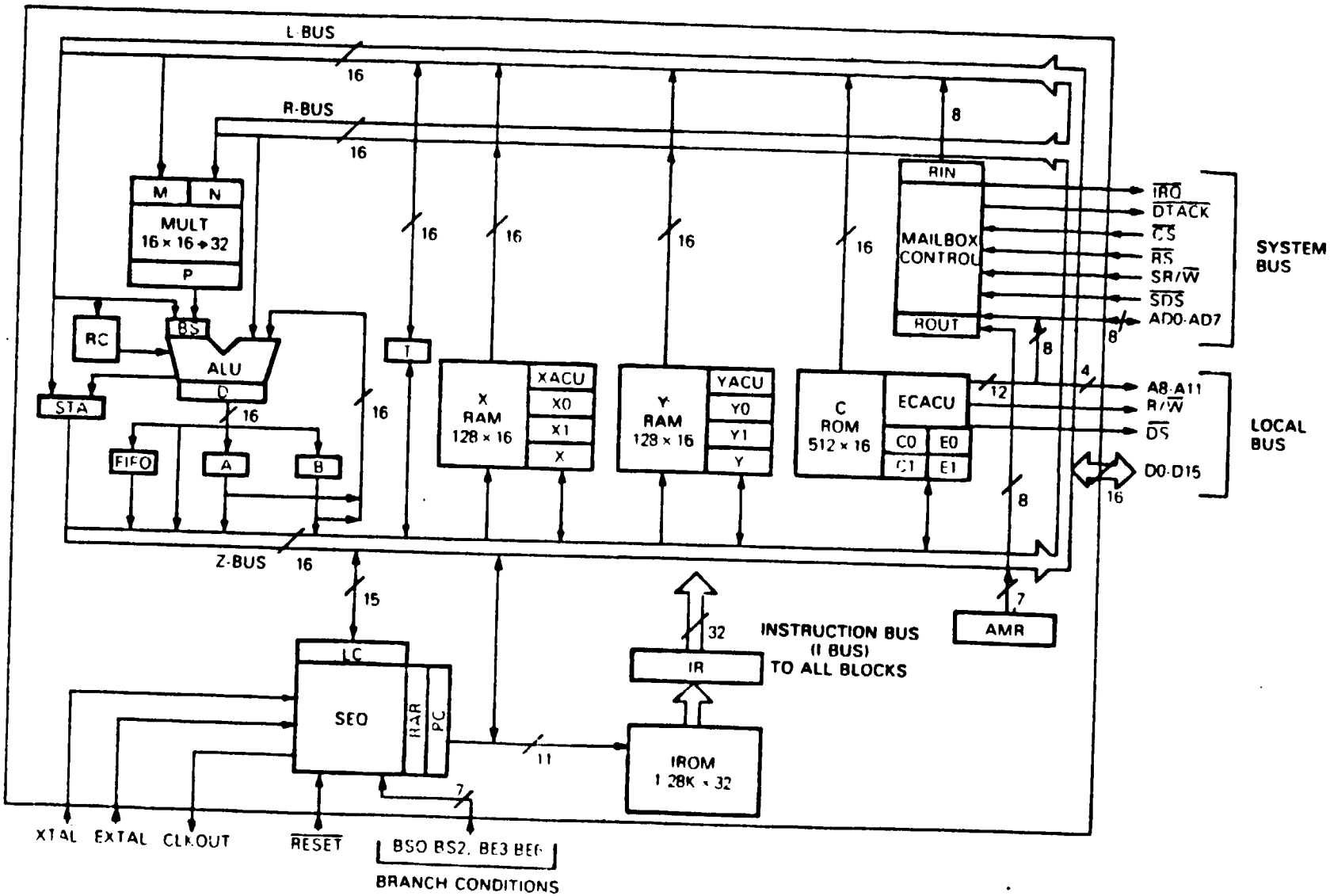
- TS 68930 - PIN- OUT (48 PIN VERSION) -



SGS-THOMSON MICROELECTRONICS

AUTEUR:	DATE:
LUC MARY	MARS 1986

BLOCK DIAGRAM



T E L E C O M

TS 68931: EXTERNAL MEMORY VERSION -

EQUIVALENT TO TS 68930 MICROCOMPUTER EXCEPT:

- EXTERNAL PROGRAM MEMORY: 64 K x 32 (55 NS RAM)
- EXTERNAL COEFFICIENT MEMORY: 512 x 16 (55 NS RAM)
- 84 PIN, LEADLESS CERAMIC CHIP CARRIER
- 1,8W POWER CONSUMPTION (1,5W FOR TS 68930)

APPLICATION COVERAGE:

- PROTOTYPING WITH EPROM (SMALL BOARD PLUGGED IN PLACE OF TS 68930)
- LARGE PROGRAM APPLICATIONS (TS 68930 LIMITED TO 1,2 K x 32)
- LOW QUANTITY (NO MASKING EXPENSE)
- DEVELOPMENT TOOLS (HDS, EVA PSI)

FUNCTION	Nb. OF INST. ROMP	DATA ROMC	EXECUTION TIME
Fixed real FIR	7	1 x 16 bit/TAP	160 nsec/TAP
Fixed complex FIR	7	2 x 16 bit/TAP	320 nsec/TAP
Adaptive real FIR	7	-	320 nsec/TAP
Adaptive complex FIR	9	-	640 nsec/TAP
BIQUAD Cell real	8	4 x 16 bit	960 nsec
BIQUAD Cell complex	8	4 x 16 bit	1920 nsec
Treillis Form	15	2 x 16 bit/Cell	480 nsec/Cell
Exp (jO) (16 bit precision)	15	64 x 16 bit	2720 nsec
FFT Butterfly	4	2 x 16 bit	960 nsec
16 points complex FFT	30	32 x 16 bit	38 µsec
64 points complex FFT	140	128 x 16 bit	265 µsec

- PERFORMANCE OF THE DSP

THE PSI INSTRUCTION SET :

The PSI instruction set is composed of 14 classical or compound instructions that provide for the input, output, manipulation and comparison of data. The instruction set is divided into five functionnal categories. they are :

- 3 calculation instructions : OPIN, OPDI, OPIM
- 4 shift instructions : ASR, LSL, LSR, ROR
- 2 save instructions : SVR, NOP
- 3 branch instructions : BRANCH, JSR, RTS
- 2 initialization instructions : INI, REPEAT

SHIFT INSTRUCTIONS : ASR, LSL, LSR, ROR

They are single-field instructions

SHIFT OPERATION	SHIFT NUMBER	SRC	DEST
ASR	$1 \leq SN \leq 15$	RIN P T Addr.X Addr.Y Addr.E Addr.C	F A
LSL	$0 \leq SN \leq 15$		
LSR	$1 \leq SN \leq 15$		
ROR	$1 \leq SN \leq 15$		

ASR 5 T, A \ (arithmetically shift right by 5 bits T register and store the result in the accumulator A)

LSL 1 P, F \ (logically shift left by 1 bit multiplier result register P and store the result in the fifo F)

ASR 7 45.X, A \ (arithmetically shift right by 7 bits XRAM address 45 and store the result in the accumulator A)

ROR 15 100.Y \ (rotate right by 15 bits YRAM address 100 and store the result in the fifo F)

32 branching conditions are possible :

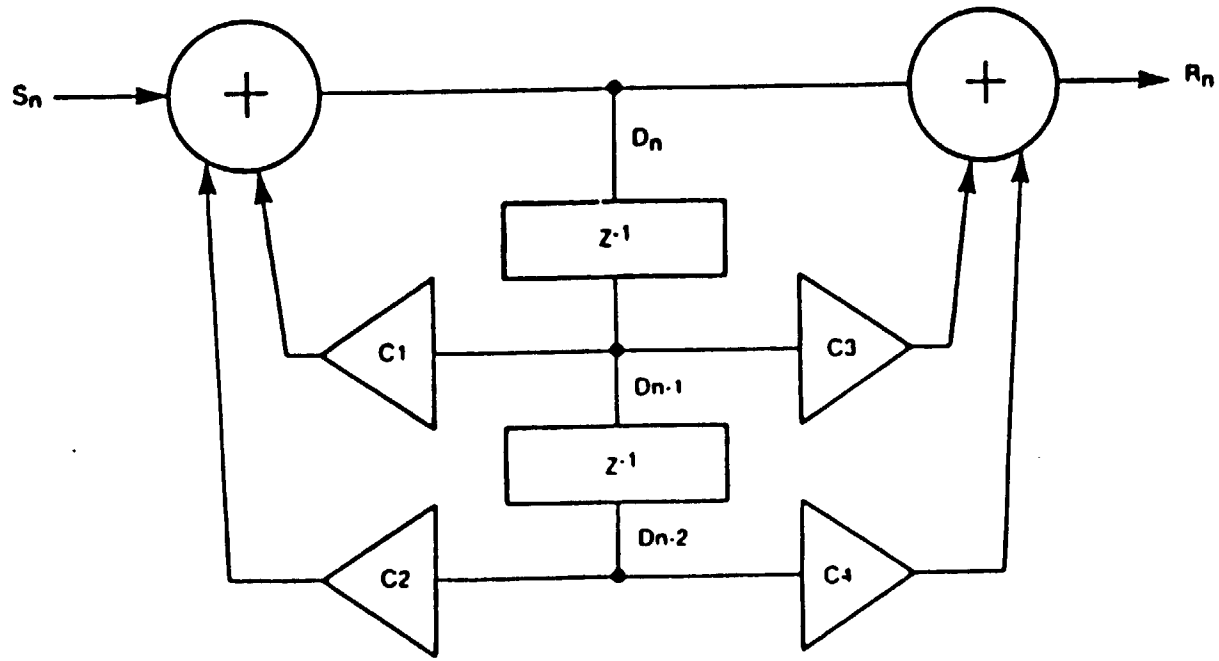
14 STATUS conditions :	(NO)SR, (NO)SI (NO)CR, (NO)CI (NO)Z, (NO)OVF (NO)MOVF
6 BRANCH ON STATE conditions :	(NO)BS0, (NO)BS1 (NO)BS2
8 BRANCH ON EDGE conditions :	(NO)BE3, (NO)BE4 (NO)BE5, (NO)BE6
internal conditions :	(NO)RDYOIN
2 inconditionnal branches :	ALWAYS, NEVER

CONTROL AND INITIALIZATION INSTRUCTION : REPEAT

This instruction is a middle level language instruction, and it is very helpful for the initialization of the loop counter. The machine instruction created is an INI instruction.

examples :

```
REPEAT 23 TIMES : REAL, NOSAT \
NOP \
NOP \
BEGIN :
  NOP \
  NOP \
  NOP \
  NOP \
  NOP \
  NOP \
END :
```



BIQUADRATIC FILTER

SGS-THOMSON MICROELECTRONICS

AUTEUR	DATE
	L19

ALGORITHM :

The 2 equations require the execution of 4 main operations :

- calculate D_n : 4 reads, 2 multiplications, 2 additions
- calculate R_n : 4 reads, 2 multiplications, 2 additions
- replace D_{n-2} by D_{n-1} : 1 write
- replace D_{n-1} by D_n : 1 write

Number of multiplications	= 4	
Number of additions	= 4	
Number of reads	= 8	18 operations
Number of writes	= 2	

The processor takes 6 cycles to realize the biquad filter

THE PIPELINE

A pipelined architecture allows the throughput of the operations to be accelerated but with consequences that some of them will take more than 1 cycle. In the processor the consequences of the pipeline are seen in 2 ways:

- When in instruction IN : The result of a multiplication is available if this multiplication has been started at Instruction IN-2 or before.

- When in instruction IN : if results are transferred back (write bus or Z-BUS) from the RALU to the memories , these results must have been calculated in the instruction IN-1 or before .

PSI PROGRAM FOR A BIQUADRATIC FILTER

```
" BIQUADRATIC FILTER "
INI   REAL, NOSAT ;     ; X0 0 \
INI   REAL, NOSAT ;     ; C0 0 \
OPIN                ; LDL [C0]+, M ; LDR [X0]+, N ; TRA R, F     \
OPIN                ; LDL [C0]+, M ; LDR [X0]-, N ;              \
OPIN                ; LDL [C0]+, M ; LDR [X0]+, N ; ADD P A , A \
OPIN                ; LDL [C0]+, M ; LDR [X0]-, N ; ADD P A , A \
OPIN  ST A, [X0]+ ;                ;               ; ADD P A , A \
OPIN  ST F, [X0]- ;                ;               ; ADD P A , A \
```

NUMBER OF CYCLES : 6 + 2

PSI PROGRAM FOR A CASCADE OF N BIQUADRATICS FILTERS

```
" CASCADE OF N BIQUADRATIC FILTERS ( N > 1)"
INI  REAL, NOSAT   ; EF 0 ; X0 0 \
INI  REAL, NOSAT   ;      ; C0 0 \
OPIN               ; LDL [C0]+, M ; LDR [X0]+, N ; TRA R , F   \
OPIN               ; LDL [C0]+, M ;              ;             \
REPEAT (N-1) TIMES; REAL, NOSAT  \
BEGIN:
OPIN               ; LDL [C0]+, M ; LDR [X0]+, N ; ADD P A , A \
OPIN               ; LDL [C0]+, M ;              ; ADD P A , A \
OPIN  ST A, [X1]+  ;              ;              ; ADD P A , A \
OPIN               ; LDL [C0]+, M ; LDR [X0]+, N ; TRA R , F   \
OPIN  ST F, [X1]+  ; LDL [C0]+, M ;              ; ADD P A , A \
END:
OPIN               ; LDL [C0]+, M ; LDR [X0] , N ; ADD P A , A \
OPIN               ; LDL [C0]+, M ;              ; ADD P A , A \
OPIN  ST A, [X1]+  ;              ;              ; ADD P A , A \
OPIN  ST F, [X1]   ;              ;              ; ADD P A , A \
```

- NUMBER OF CYCLES : 5x(NUMBER OF FILTERS) + 1 + 3

TS68950/51/52

MODEM ANALOG FRONT END (MAFE)

TELECOM MARKETING

OCTOBRE 86

SGS-THOMSON MICROELECTRONICS

TS 68951

THE TS 68951 IS THE RECEIVE SECTION OF THE ANALOG MODEM FRONT-END WHICH, ASSOCIATED WITH A DIGITAL SIGNAL PROCESSOR, AS THE TS 68930, REALISES VOICE-GRADE MODEM APPLICATIONS CONFORMING TO CCITT AND BELL RECOMMENDATIONS.

MAIN CHARACTERISTICS :

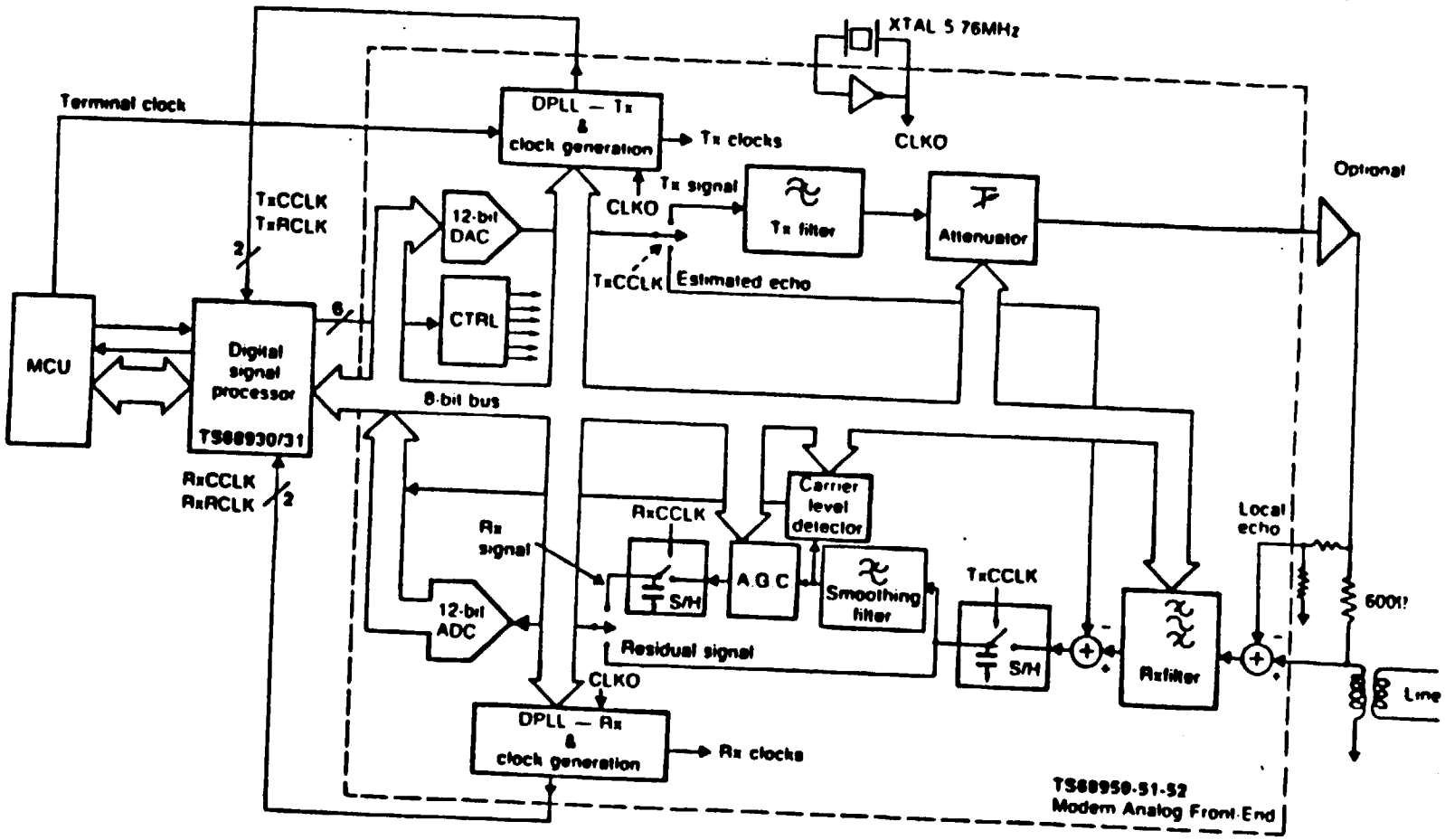
- A/D CONVERTER 12 BITS MULTIPLEXED ON TWO PLESIOCHRONOUS CHANNEL (ON CHANNEL FOR ECHO-CANCELLATION)
- 7 TH. ORDER BAND-PASS PROGRAMMABLE FILTER
- BACK-CHANNEL REJECTION AND SWITCHED SMOOTHING FILTER
- CONTINUOUS-TIME ANTI-ALIASING AND SMOOTHING FILTER
- 0 TO 46.5 dB PROGRAMMABLE GAIN AMPLIFIER
- THRESHOLD PROGRAMMABLE CARRIER DETECT LOGIC
- 8 BITS STANDARD BUS FOR DIGITAL INTERFACE
- KIT WITH TS 68950 AND TS 68952
- CMOS SI-GATE TECHNOLOGY
- +/-5 V POWER SUPPLY
- 24 PIN PLASTIC PACKAGE

TELECOM MARKETING

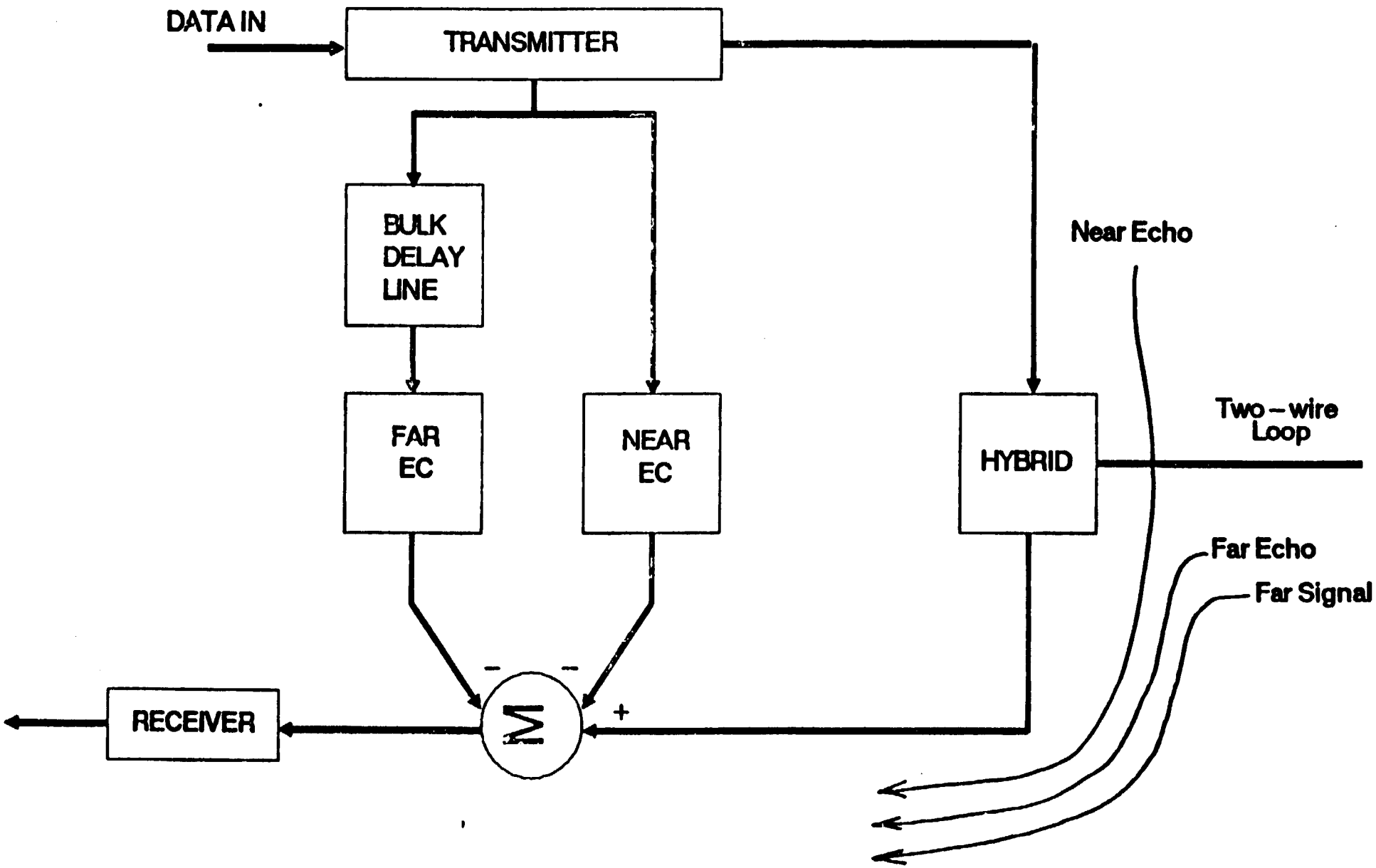
OCTOBRE 86

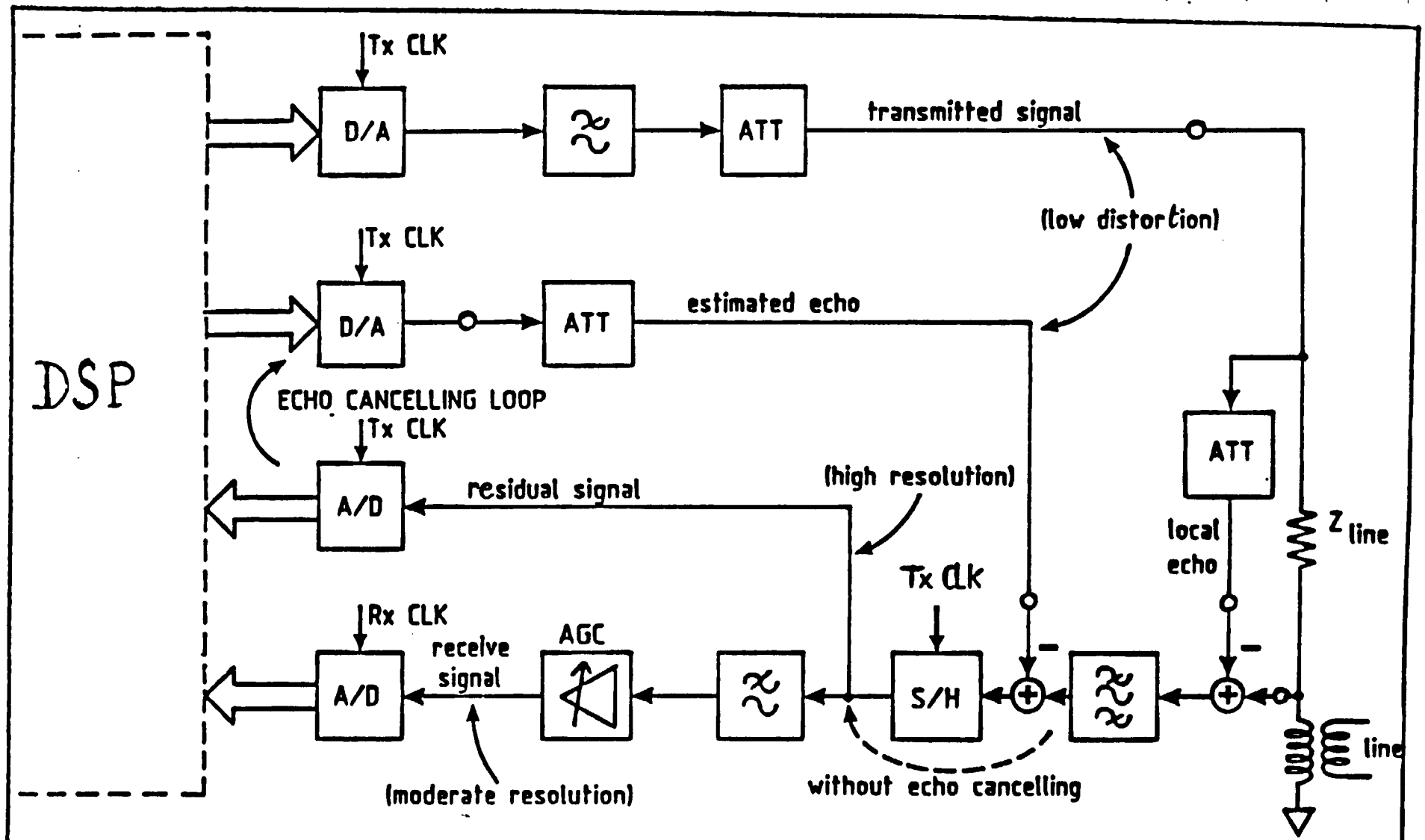
SGS-THOMSON MICROELECTRONICS

TS68930/PSI


XTAL 5.76MHz
Terminal clock
DPLL — Tx & clock generation
Tx clocks
CLKO
Tx signal
Tx filter
Attenuator
Optional
TxCCLK
TxRCLK
2
12-bit DAC
CLKO
TxCCLK
Estimated echo
CTRL
MCU
Digital signal processor
TS68930/31
6
8-bit bus
RxCCLK
RxRCLK
2
Carrier level detector
RxCCLK
Rx signal
S/H
A.G.C
Smoothing filter
TxCCLK
Local echo
600Ω
12-bit ADC
Residual signal
S/H
Rx filter
Line
CLKO
DPLL — Rx & clock generation
Rx clocks
TS68950-51-52
Modem Analog Front-End

Echo Cancellation – based System





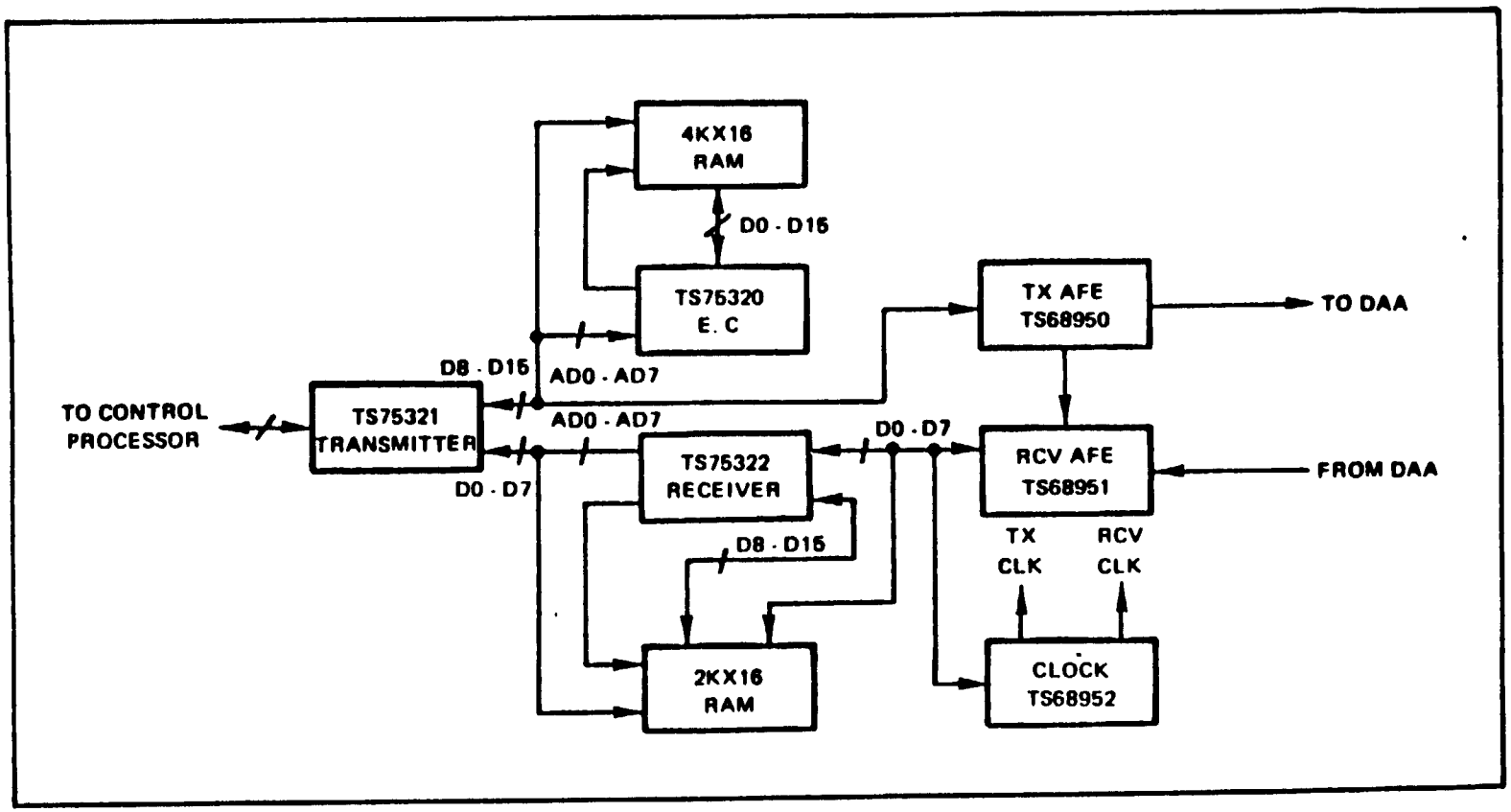
SIMPLIFIED ANALOG FRONT-END DIAGRAM

SGS-THOMSON MICROELECTRONICS

TS7532

V.32 MODEM CHIP SET

BLOCK DIAGRAM



DEVELOPMENT TOOLS

HDS – PSI : STAND ALONE SOFTWARE & HARDWARE DEVELOPMENT STATION

EVA – PSI : LOW COST STAND ALONE EVALUATION BOARD

SOFTWARE :

- LIGNE BY LIGNE ASSEMBLER/DESASSEMBLER
- MACRO ASSEMBLER
- FUNCTIONNAL SIMULATOR

C-6

SGS-THOMSON MICROELECTRONICS

MACRO ASSEMBLER

A/ EFFICIENCY OF THE CODE

- ASSEMBLY LANGUAGE WITH 4 FIELDS
  - L BUS SOURCE
  - R BUS SOURCE
  - OP CODE
  - Z BUS DESTINATION
- KEEP ADVANTAGE OF PARALLEL ACHITECTURE
- USE ALL AVAILABLE RESSOURCES AT MAXIMUM SPEED

B/ EASE OF PROGRAMMING

- WHERE SPEED IS NOT A MUST
- "68000 LIKE" ASSEMBLY LANGUAGE

FUNCTIONNAL SIMULATOR *PSIMUL*

- RUNS ON IBM-PC,XT,AT AND VAX (VMS OR UNIX) , MICROVAX (ULTRIX)
- 1000 INSTRUCTIONS/S ON VAX
- STEP BY STEP EXECUTION
- BREAKPOINT ON VALUE OR ADRESS
- TRACE
- ACCESS TO ALL INTERNAL REGISTERS AND MEMORIES
- INPUT-OUTPUT SIMULATION THROUGHT FILES
  - FOR ADC & DAC ,EXTERNAL MEMORY
  - FOR BRANCH INPUT
  - FOR SYSTEM BUS (MAILBOX FIFO IN & OUT)

AUTEUR

REFERENCE DATE

SGS-THOMSON MICROELECTRONICS

EVA–PSI
LOW COST EVALUATION & EMULATION BOARD

. SINGLE 26 cm x 24 cm BOARD

. 68000 BASED MONITOR

. LINE/LINE ASSEMBLER/DESASSEMBLER

. PROGRAM DOWN–LOADING FROM HOST OR HDS–PSI

. SOPHISTICATED BREAK POINTS

. TWO RS232 SERIAL LINKS FOR :
- . TERMINAL
- . HOST
- . PRINTER

. FREE WRAPPING AREA (80 cm2) FOR APPLICATION

TELECOM MARKETING

OCTOBRE 86

SGS-THOMSON MICROELECTRONICS

THE ASSEMBLER AND THE REVERSE ASSEMBLER

INCLUDED IN THE EVAPSI FIRMWARE.

IT GIVES A QUICK WAY TO DEBUG AN APPLICATION DIRECTLY ON THE EMULATOR.

IT ALLOWS THE PROGRAMMER TO PATCH THE PROGRAM MEMORY OF THE EMULATOR IN ASSEMBLY LANGUAGE WITHOUT MODIFY THE SOURCE FILE AND WITHOUT USING THE MACRO ASSEMBLER.

THE REVERSE ASSEMBLER ALLOWS THE USER TO READ MEMORY PROGRAM CONTENT IN ASSEMBLY LANGUAGE RATHER THAN HEXADECIMAL.

TS68930 – 31 DEVELOPMENT TOOLS

HDS PSI EMULATOR

- INTERACTIVE MENU DRIVEN INTERFACE
- BACKUP OF HDSPSI CONFIGURATIONS IN CMOS RAM (64K)
- MULTIPROCESSING – UP TO 4 EMULATOR\L.A CONTROLLED BY A USER
- MULTIUSER – UP TO 4 USERS CONTROLLING INDEPENDANT EMULATOR\L.A.
- REAL TIME EMULATION – INTERNAL,EXTERNAL OR APPLICATION CLOCK
- 64 K x 32 INSTRUCTION MEMORY
- SOPHISTICATED BREAKPOINTS (20 ON ADRESS,8 COMPLEX (N ADR.1 FOLLOWED BY M ADR.2))
- PSEUDO REALTIME TRACE PROVIDING INTERNAL REGISTER EXAMINATION WITHOUT PERTURBING EXECUTION
- 4K x16 BITS EXTERNAL DATA MEMORY – MAPPING ON A WORD BASIS

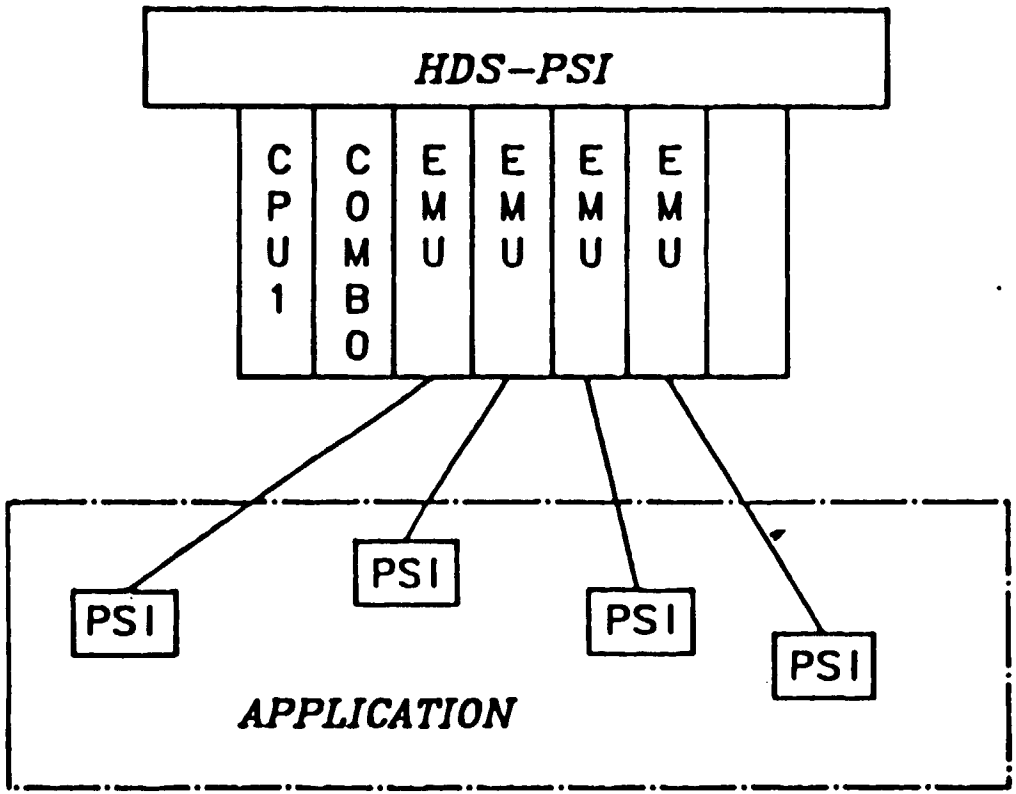
AUTEUR

REFERENCE DATE

SGS-THOMSON MICROELECTRONICS

TS68930/PSI

MULTIPROCESSOR EMULATION



SGS-THOMSON MICROELECTRONICS

AUTEUR	DATE

D.S.P. and MAFE AVAILABILITY PLAN

IC'S AVAILABILITY :

	ENGINEERING SAMPLES	QUALIFICATION SAMPLES	PRODUCTION
- TS68930 - D.S.P. WITH INTERNAL ROM	(not applicable)	ROM code accepted now	Available
- TS68931 - D.S.P. ROMLESS	Available	Available	Available
- TS68950/51/52 - MAFE KIT	Available	Available	Available

TELECOM MARKETING

OCTOBRE 86

SGS-THOMSON MICROELECTRONICS

D.S.P. and MAFE AVAILABILITY PLAN

SOFTWARE DEVELOPMENT TOOLS

MACROASSEMBLER	IBM - PC(XT,AT) VAX - VMS VAX - UNIX VAX - ULTRIX	AVAILABLE
SIMULATOR	IBM - PC VAX - VMS VAX - UNIX,ULTRIX	AVAILABLE
ROUTINE LIBRARY	IBM - PC	AVAILABLE

SGS-THOMSON MICROELECTRONICS

BACKS

D.S.P. AND MAFE AVAILABILITY

MODEM ORIENTED SOFTWARE

- V22 bis (2400 BPS MODEM) Q3 87

  This program will have in back mode the V21,V22,V23,BELL103 and BELL212A.
  An application note will be issued.

- V32 (9600 BPS MODEM) END 87

  1) Echo cancellation software application note will be provided (mid 87)
  2) We 'll provide a full V32 modem chip set (end 87)

PRECEDING PAGE BLANK NOT FILMED

D.S.P. and MAFE AVAILABILITY PLAN

HARDWARE DEVELOPMENT TOOLS

- EVAPSI - STAND-ALONE EMULATOR	AVAILABLE
- HDSPSI - MULTIPROCESSOR HARDWARE DEVELOPMENT SYSTEM WITH LOGIC ANALYSIS	AVAILABLE
- EVM MAFE - MODEM ANALOG FRONT END EVALUATION BOARD	AVAILABLE
- PSI REPROM - REPROM MODULE WITH ROMLESS TS68931	AVAILABLE

NOTES:

(1) EVAPSI emulator and IBM-PC macroassembler are the minimum tools for the development of an application

(2) HDSPSI : up to 4 (emulation + logic analysis) boards

(3) PSI REPROM module is pluggable in a 48 DIP (TS68930 D.S.P. ROM VERSION) socket. Developped for prototyping

TS68930 – 31 SUPPORT TOOLS

PSI – REPROM MODULE

- MODULE WITH ON BOARD
  - ➡ TS68931 : D.S.P. MICROPROCESSEUR VERSION
  - ➡ REPROM FOR PROGRAM (2K*32)
  - ➡ REPROM FOR COEFFICIENT (512*16)
- PLUG IN TS68930 – 48 DIL SOCKET
- SMALL SIZE (6CM) ; VERTICAL OR BENDED VERSION
- APPLICATIONS:
  - ➡ PROTOTYPING OF TS68930
  - ➡ PRESERIES FOR FIELD VALIDATION

AUTEUR

REFERENCE DATE

SGS-THOMSON MICROELECTRONICS

TS68930 – 31 DEVELOPMENT TOOLS

HDS PSI LOGIC ANALYSER

- LOGIC ANALYSER ASSOCIATED WITH EMULATOR (UP TO 4)

- REAL TIME TRACE;2K DEPTH,95 BITS WIDTH

- SYNCHRONOUS ON SYNC. I\O,ASYNCHRONOUS ON SYSTEM BUS

- TRIGERRING FEATURES ;
  - ADRESS BUS (SIMPLE ADRESS OR AFTER N ADR.1 FOLLOWED BY M ADR.2)
  - VALUE ON D BUS
  - BRANCH CONFIGURATION
  - EXTERNAL INPUTS OR START OF MAILBOX EXCHANGE

- BEFORE , AFTER , AND BETWEEN BOUNDS TRIGERRING

- TIME MEASUREMENT FONCTION

AUTEUR

REFERENCE DATE

SGS-THOMSON MICROELECTRONICS

DEBUG MONITOR

THE HDS PSI AND EVAPSI ARE CONTROLLED BY THE MONITOR PROGRAM.

DEBUG MONITOR IS INTERACTIVE MENU ORIENTED.

EASY TO USE, NO LEARNING TIME

KEY FEATURES :

- READ AND MODIFY ALL MEMORIES, INTERNAL REGISTERS AND INPUTS/OUTPUTS RELATIVE TO THE PSI.
- CONTROL PROGRAM EXECUTION, BREAKPOINTS, STEP BY STEP.
- INFORM THE USER AT ANY TIME OF EMULATOR STATE.
- LOAD OBJECT FILES FROM HOST.
- SUPPORT ASSEMBLER AND REVERSE ASSEMBLER.

EVAPSI STAND-ALONE EMULATOR
KEY FEATURES

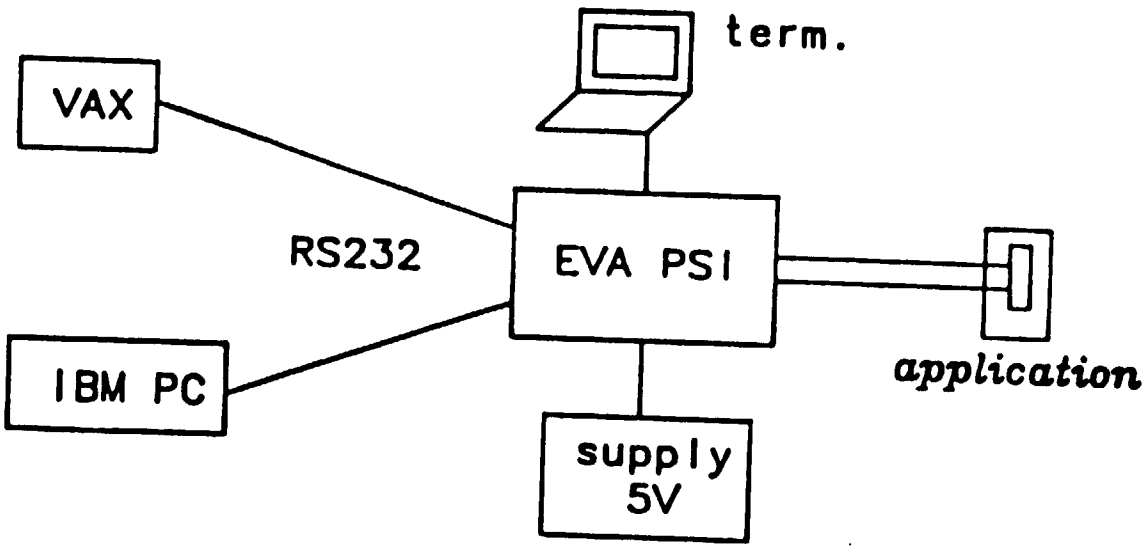
- FULL SPEED EMULATION (Tcycle=160ns)
- PROGRAM MEMORY 2K 32 bits INSTRUCTIONS
- 2K EXTERNAL LOCAL D.S.P. MEMORY
- MAILBOX EMULATION
- HIGH LEVEL BREAKPOINTS : 20 BREAK AT ADR. N
  8 BREAK AFTER N ADR.1 FOLLOWED BY M ADR.2
- PSEUDO-REAL TIME TRACE PROVIDES INTERNAL REGISTER EXAMINATION WITHOUT PERTURBING THE APPLICATION
- STEP BY STEP MODE
- PROBES FOR 48 DIP TS68930 AND 84 PGA TS68931
- USER FRIENDLY MENU
- 2 RS232 LINKS FOR HOST,PRINTER,PROM PROGRAMMER,....
- PROVIDED WITH CMOS RAM WITH BATTERY FOR BACKUP

DEVELOPMENT TOOLS
2 SOLUTIONS

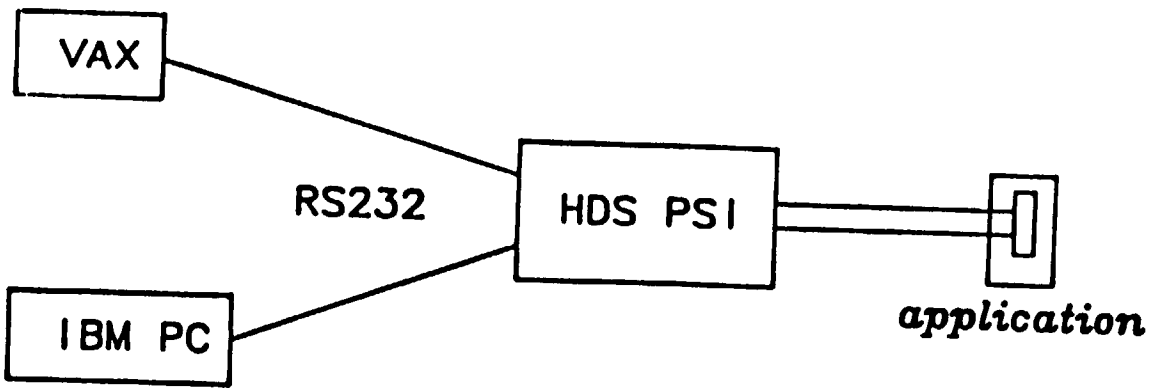
AVAILABILITY

NOW

1



2



MACROASSEMBLER PSIMAC

- RUNS ON IBM-PC,XT,AT AND VAX (VMS OR UNIX) OR MICROVAX (ULTRIX)
- MACRO DEFINITION AND MACRO CALLS
- CONDITIONNAL ASSEMBLY
- HIGH LEVEL LANGUAGE LOOP FACILITY (REPEAT , BEGIN , END)
- MULTIPLE OUTPUT FORMAT GENERATION
- DATA MEMORY FILES GENERATION
- SIMULATION FILES GENERATION
- POWERFULL EXPRESSION FACILITIES

ORIGINAL PAGE IS OF POOR QUALITY

SGS-THOMSON MICROELECTRONICS

AUTEUR

REFERENCE DATE

MACRO ASSEMBLER

- MACRO DEFINITION AND MACRO CALL
- CONDITIONAL ASSEMBLY
- HIGH LEVEL LANGUAGE LOOP FACILITY
- MULTIPLE OUTPUT FORMAT GENERATION
- DATA MEMORY GENERATION
- SIMULATION FILE GENERATION
- POWERFULL EXPRESSION FACILITY

T E L E C O M

- MODEM APPLICATION COVERAGE -

STANDARD	SPEED (BPS)	DSP TS 68930	MAFE TS 68950,1,2	STANDARD MEMORY (55 ns, C-MOS)	OCTAL BUFFER
B 201/ V 26	2400 HD	1	1	0	0
B 208/ V 27	4800 HD	1	1	0	0
B 209/ V 29	9600 HD	1	1	0	0
V 33	14400 HD	2	1	2 x 2K x 8	0
V 26 TER	2400 FD/EC	1	1	2 x 2K x 8	1
V 32 (4800)	4800 FD/EC	2	1	2 x 2K x 8	0
V 32 (9600)	9600 FD/EC	3	1	4 x 2K x 8	0
	9600 HD/FP	2	1	0	0

HD: 2W HALF DUPLEX OR 4W FULL DUPLEX
FD: 2W FULL DUPLEX
EC: ECHO CANCELLATION
FP: FAST POLL (15 MS ON LEASED LINES)

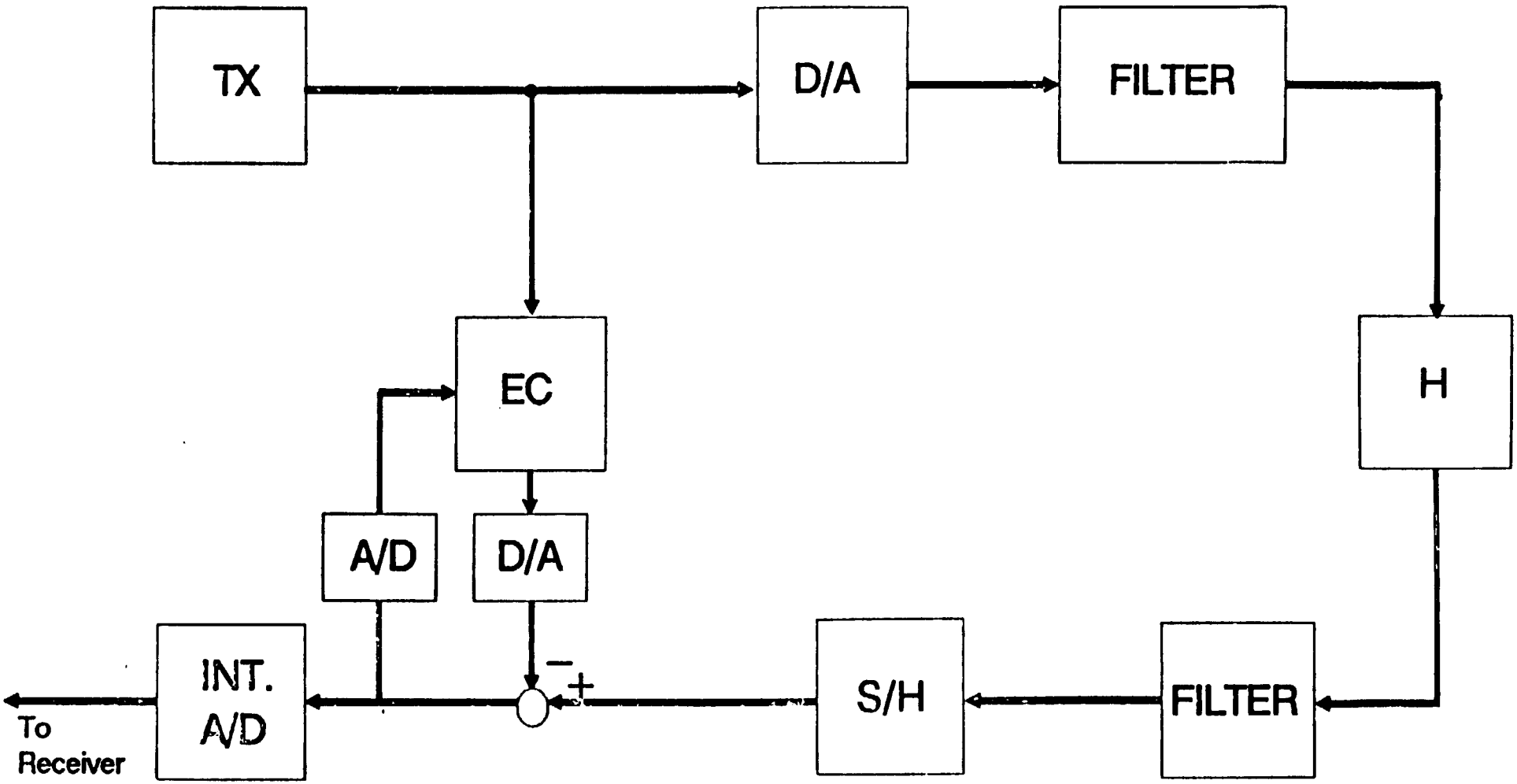
TS7532

V.32 MODEM CHIP SET

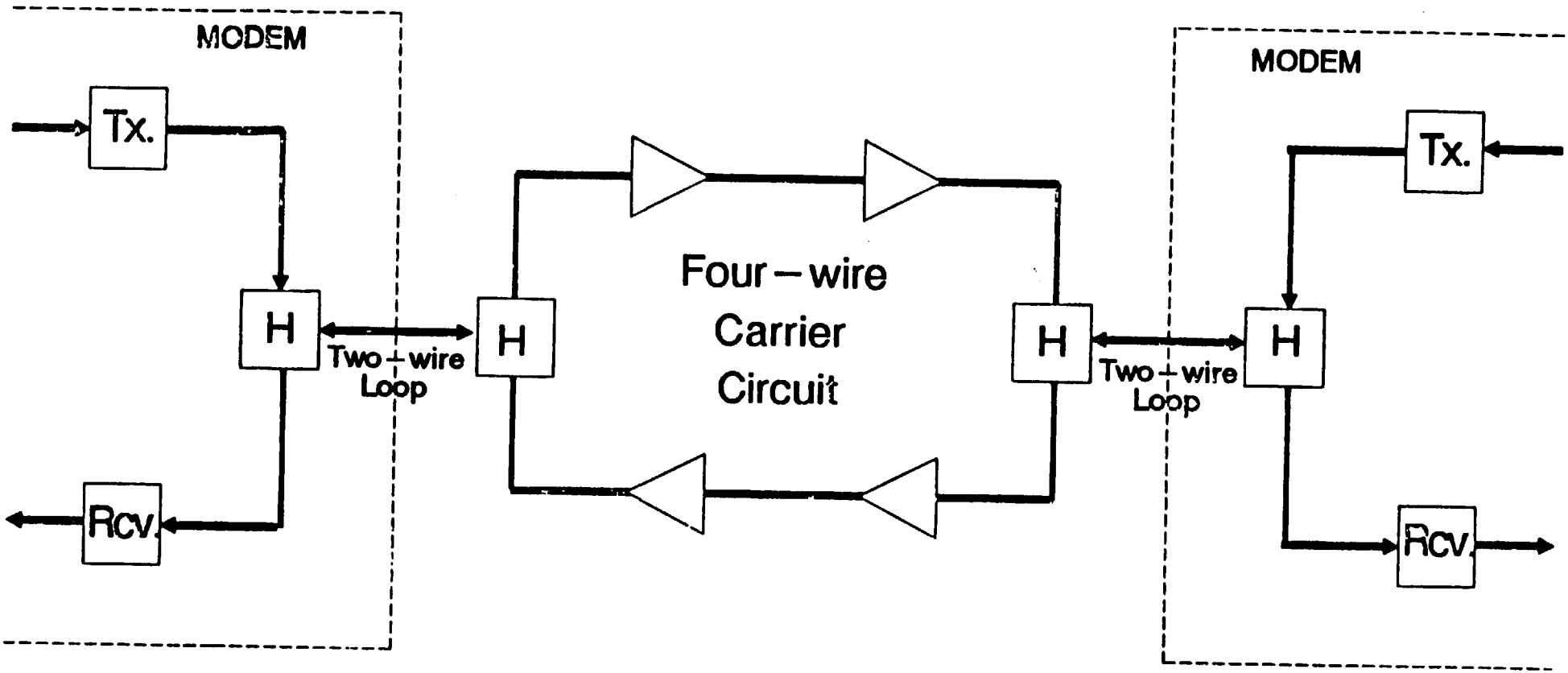
FEATURES -

- Integrated implementation on three DSP and three MAFE chips.
- CCITT V.32 compatible modem chip set.
- Full duplex operation at 9600 and 4800 bps.
- Full implementation of the V.32 handshake.
- Single DSP cancellation of both near-end and far-end echoes.
- Two satellite hops and frequency offset capabilities for the far end echo canceller.
- Trellis encoding and Viterbi decoding.
- 12.5% roll-off raised cosine transmitter pulse shaping.
- High performance passband fractionally spaced adaptive equalizer.
- Signal quality monitoring.
- Parallel interface to standard microprocessors.
- Bit rate data clocks provided for synchronous data transfer.
- Full diagnostic capability.
- DTMF generation.
- Call progress tone detection.
- Future upgrades to include V.22bis, V.22, B212A and FSK without hardware modification.

Digital Echo Canceller with Analog Cancellation



Typical Dialed Connection



TS 68952

THE TS 68952 SYNTHETISES ALL THE REQUESTED CLOCK FREQUENCIES FOR VOICE-GRADE MODEM APPLICATIONS CONFORMING TO CCITT AND BELL RECOMMENDATIONS. IT PARTICULARLY SUPPLIES ALL CLOCKS REQUESTED BY THE TRANSMIT (TS 68950) AND THE RECEIVE (TS 68951) SECTION OF THE MODEM ANALOG FRONT-END AND BY THE DIGITAL SIGNAL PROCESSOR (TS 68930)

MAIN CHARACTERISTICS :

- DPLL TRANSMIT AND RECEIVE TIME-BASE
- FREQUENCY PROGRAMMATION AND SYNCHRONIZATION OF DPLL THROUGH 8 BIT STANDARD BUS
- POSSIBLE SYNCHRONIZATION OF TRANSMIT TIME-BASE ON DATA TERMINAL EQUIPMENT CLOCK OR ON EXTRACTED RECEIVE CLOCK
- FOUR AVAILABLE CLOCKS (PLESIOCHRONOUS TRANSMIT AND RECEIVE) : BIT-CLOCK, RATE-CLOCK, SAMPLING AND MULTIPLEXING CLOCK
- KIT WITH TS 68950 AND TS 68951
- CMOS SI-GATE TECHNOLOGY
- SINGLE + 5V. POWER SUPPLY
- 28 PIN PLASTIC PACKAGE

TS 68950

THE TS 68950 IS THE TRANSMIT SECTION OF THE ANALOG MODEM FRONT-END WHICH, ASSOCIATED WITH A DIGITAL SIGNAL PROCESSOR, AS THE TS 68930, REALISES VOICE-GRADE MODEM APPLICATIONS CONFORMING TO CCITT AND BELL RECOMMENDATIONS.

MAIN CHARACTERISTICS :

- D/A CONVERTER 12 BITS MULTIPLEXED ON TWO CHANNELS (ONE CHANNEL FOR ECHO-CANCELLATION)
- 6 TH. ORDER LOW-PASS FILTER AND SMOOTHING CONTINUOUS-TIME FILTER
- 0 TO 22 dB PROGRAMMABLE ATTENUATION
- 8 BITS STANDARD BUS FOR DIGITAL INTERFACE
- KIT WITH TS 68951 AND TS 68952
- CMOS SI-GATE TECHNOLOGY
- +/-5 V POWER SUPPLY
- 24 PIN PLASTIC PACKAGE

SGS-THOMSON MICROELECTRONICS

Example 2 : $D_n = Y_n + C_n . X_n$

(Fig 2.3.2B)

In example 1 the operation takes 4 cycles. If the operation is repeated (as it is the case in signal processing) the delay remains the same but the throughput is a totally different matter. It will depend on the number of operations possible in 1 cycle.

N	NOTES	L BUS	R BUS	MULT	RALU	Z BUS
1		C_0	X_0	C_0 X_0		
2			Y_0		Y_0	
3		C_1	X_1	C_1 X_1	$C_0 . X_0 + Y_0$	
4			Y_1		Y_1	D_0
5		C_2	X_2	C_2 X_2	$C_1 . X_1 + Y_1$	
6			Y_2		Y_2	D_1
7		etc			$C_2 . X_2 + Y_2$	
8						D_2

PRECEDING PAGE BLANK NOT FILMED

Example 1 : D = Y + C.X

This equation will take 4 cycles (1 to read, 2 for the multiplication and 1 to write).

N	NOTES	L BUS	R BUS	MULT	RALU	Z BUS
1		C	X	C X		
2						
3					C X + Y	
4						D

IMPLEMENTATION

The algorithm is shown for the first sample (n = 0)

(1) $D_0 = S_0 + C1.D_{-1} + C2.D_{-2}$

(2) $R_0 = D_0 + C3.D_{-1} + C4.D_{-2}$

giving

S_0	= input samples
D_{-1} ,D_{-2}	= intermediary results
R_0	= output results
C1, C2, C3, C4	= coefficients

The variables and coefficients are located as follows :

S_0	= input sample in A accumulator
D_{-1} , D_{-2}	= intermediary results stored in XRAM
R_0	= output result , will be left in A accumulator
C1, C2, C3, C4	= coefficients in CROM

The equations of the biquadratic filter (commonly called biquad) can be written as :

(1) $D_n = S_n + C1.D_{n-1} + C2.D_{n-2}$

(2) $R_n = D_n + C3.D_{n-1} + C4.D_{n-2}$

with

S_n	= input samples
D_{n-1} , D_{n-2}	= intermediary results
R_n	= output results
C1, C2, C3, C4	= coefficients

EXAMPLE OF PSI PROGRAM : A BIQUADRATIC FILTER

CONTROL AND INITIALIZATION INSTRUCTION : INI

This instruction has 3 fields :

- MODE FIELD

This field provides the initialization of the MODE and SAT bits of the STATE REGISTER.

operating modes :

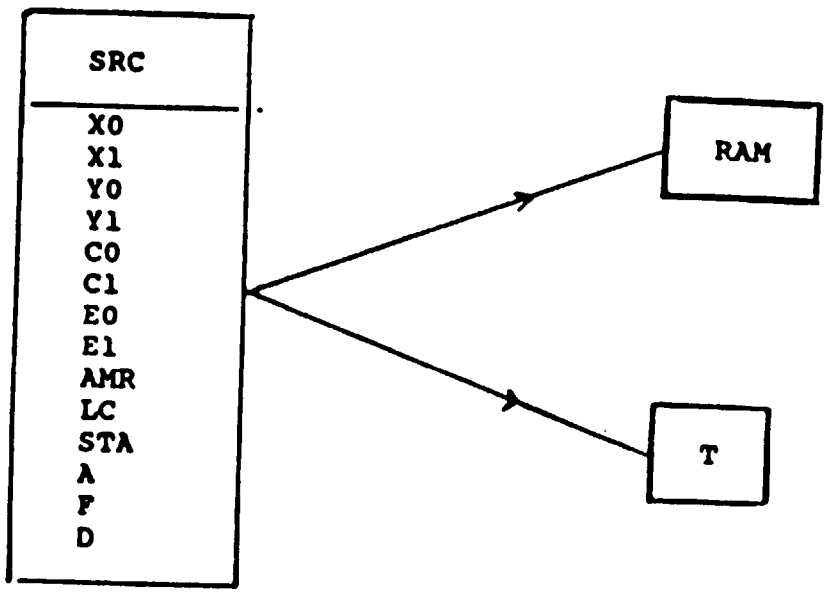
3 operating modes can be chosen :
. REAL mode (16 bits lenght data and ALU)
. CPLX mode (2x16 bits lenght data and ALU)
. DBPR mode (32 bits lenght data and ALU)

saturation mode :

If the saturation mode is chosen (SAT=1), the ALU result is forced to the greatest positive value ($7FFF) in case of positive overflow, or to the greatest negative value ($8000) in case of negative overflow.

SAVE INSTRUCTIONS : SVR, NOP

SVR and NOP are single-field instructions
(NOP \ is equivalent to SVR D, \ ).



X0, X1 cannot be saved in XRAM.
Y0, Y1 cannot be saved in YRAM.

CALCULATION INSTRUCTIONS : OPIN, OPDI, OPIM

They are three or four field instructions :

1) Z field, which allows transfer of previously obtained results towards other resources (specially in RAM), by means of bus Z.

2) L field and 3) R field which allow the multiplier inputs to be loaded and the operands to be brought at ALU input.

4) The ALU field which completes ALU operations on the operands, and updates the status register.

Writing an OPIN, OPDI, OPIM instruction should be carried out respecting the general and special rules stated below :

- ERAM (external RAM) and CROM (coefficients ROM) cannot be simultaneously addressed in an instruction.

- An memory cannot be a source and a destination in the same instruction.

- Two different addresses of a same memory cannot appear in the same instruction.

PROGRAMMING WITH THE PSI (TS 68930 - TS 68931)

SGS-THOMSON MICROELECTRONICS

AUTEUR	DATE

T E L E C O M

- DSP DEVELOPMENT - TOOLS -

FOR EACH STAGE, APPROPRIATE TOOLS ARE PROVIDED:

- 1) HOST COMPUTER SIMULATIONS:
  - ASSEMBLER (PSI ASM)
  - SIMULATOR (PSI MUL)
- 2) REAL TIME EMULATION:
  - HARDWARE DEVELOPMENT STATION (HDS)
  - EVALUATION BOARD (EVA PSI)
- 3) PROTOTYPING:
  - TS 68931 WITH FAST C-MOS SRAM (55 NS)

TS68930/31 KEY POINTS OVERVIEW

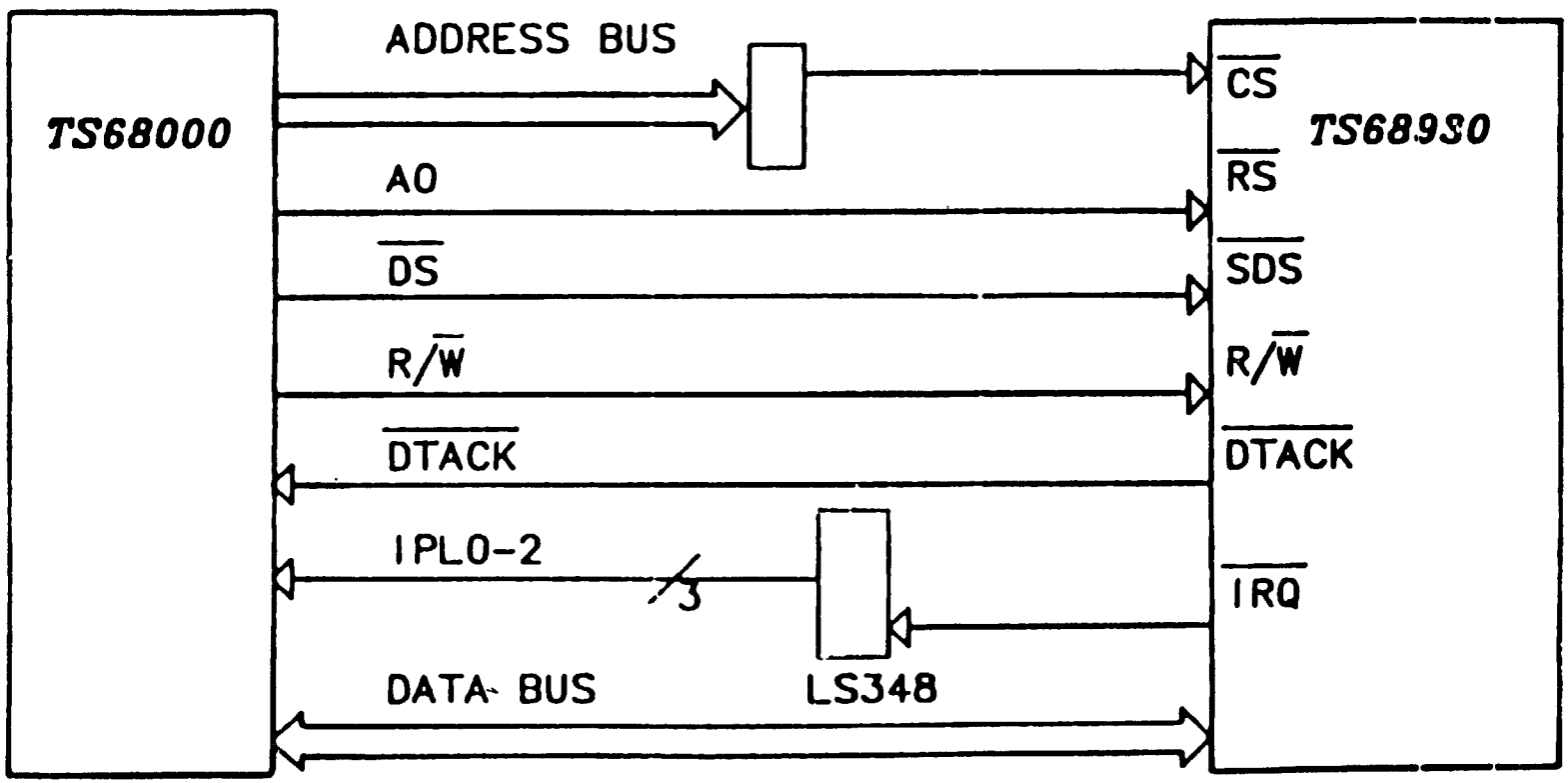
FEATURES	BENEFITS
TRUE HARVARD ARCHITECTURE	SAME EXECUTION TIME MCU\MPU VERSIONS NO INSTRUCTION \ DATA BOTTLENECK
3 DATA BUSES	2 READ , 1 WRITE IN ONE CYCLE (MULT\ACC & UPDATE COEF IN ADAPTIVE FILTERS)
ON-CHIP RESSOURCES (RAM's,COEFF. ROM , PROG. ROM)	MINIMUM PART COUNT
COMPLEX DATA PROCESSING	1 COMPLEX MULT.\320ns 12.5 MEGA REAL M\S
REPEAT INSTRUCTION BLOCS	CODE COMPACTION
LOCAL BUS (16 bits or 2x8 bits)	EXTERNAL DATA READ or WRITE AT FULL SPEED NO GLUE REQUIRED TO CONNECT PERIPHERALS
SYSTEM BUS (and MAILBOX)	STRAIGTHFORWARD MULTIPROCESSOR CONFIGURATIONS CONNECTION WITH HOST CPU WITH DIFFERENT SPEED NO GLUE

AUTEUR

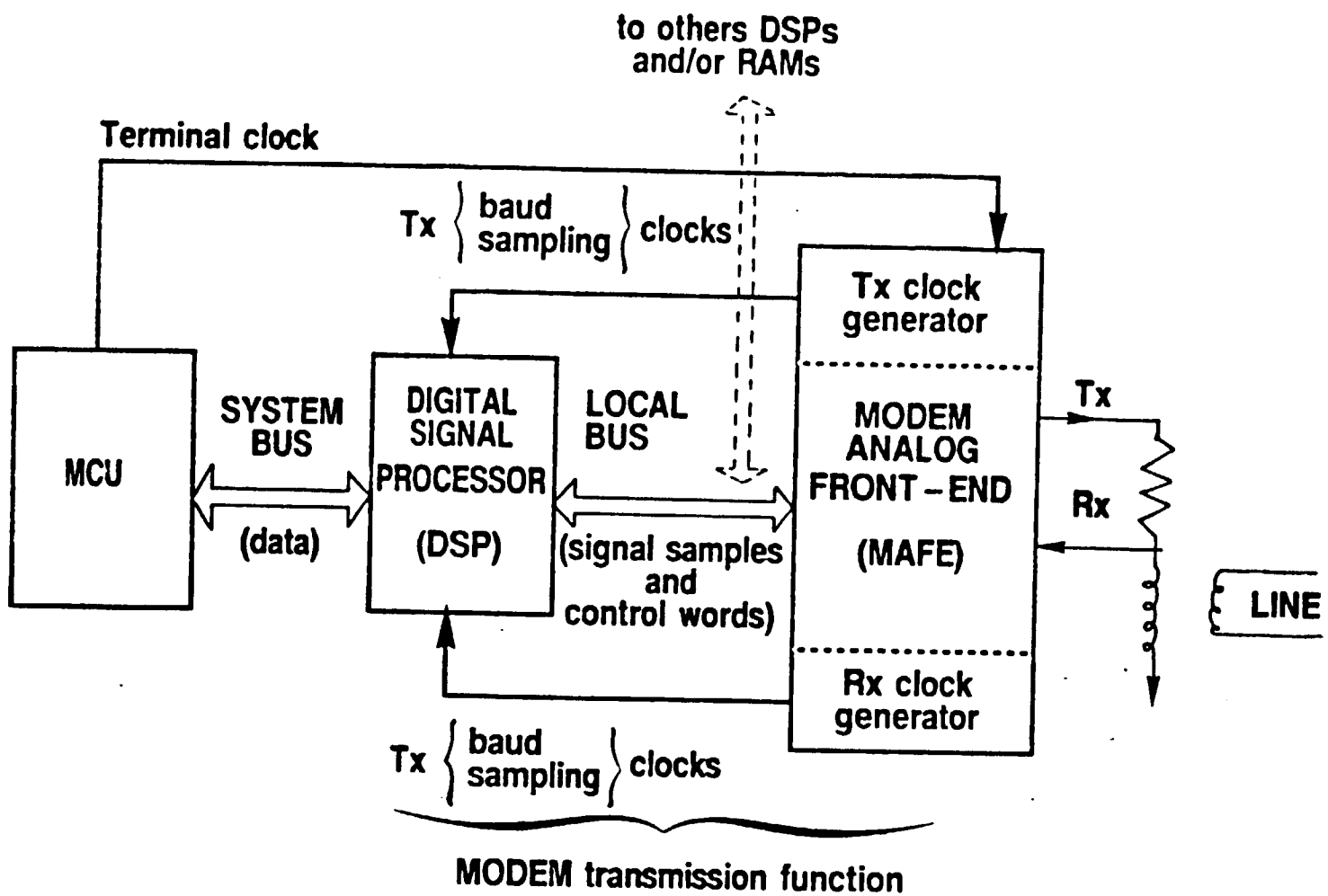
REFERENCE DATE

SGS-THOMSON MICROELECTRONICS

SYSTEM CONFIGURATION



SYSTEM BUS



MODEM IMPLEMENTATION WITH DSP AND ANALOG FRONT-END

SGS-THOMSON
MICROELECTRONICS

THOMSON SIGNAL PROCESSING

TS68930/TS68931 (PSI)

- TS68930 SIGNAL PROCESSOR WITH INTERNAL ROM
- TS68931 SIGNAL PROCESSOR WITH EXTERNAL PROGRAM MEMORY
- TS68950/51/52 MODEM ANALOG FRONT END

TELECOM MARKETING

OCTOBRE 86

SGS-THOMSON MICROELECTRONICS

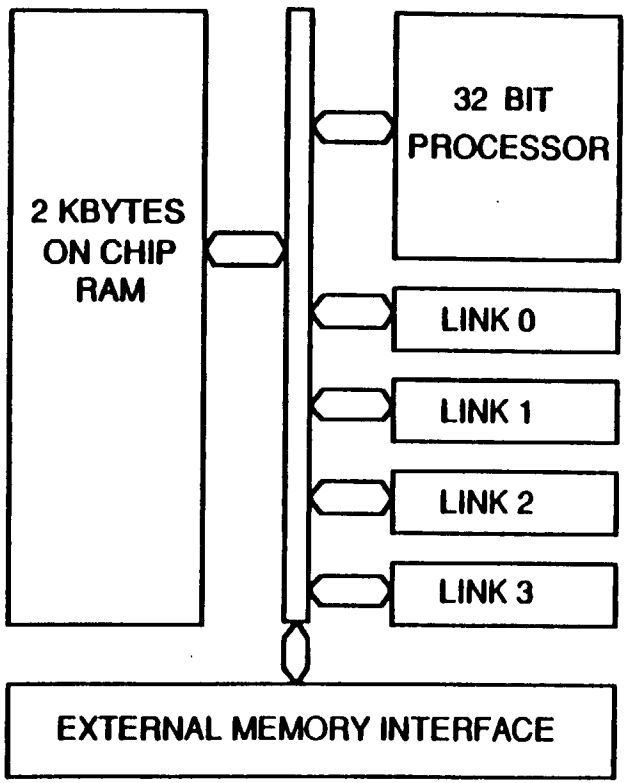
inmos

The IMS T414 transputer and the IMS A100 transversal filter

Neill Shepherd
Senior Microcomputer Applications Engineer
P.O Box 16000
Colorado Springs, CO 80935

inmos

The IMS T414-20

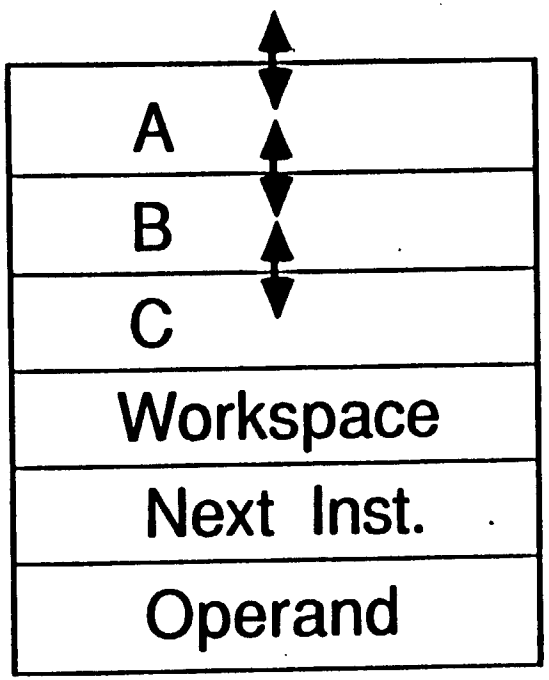


- full 32 bit processor
- 2 KBytes 50ns cycle RAM
- four inter-processor communication links
- configurable memory interface
- standard 5 volt supply
- dissipates less than 1 watt
- Only requires 5Mhz external Clock

PRECEDING PAGE BLANK NOT FILMED

inmos

Transputer Registers



E Error Flag

- expressions are evaluated on a short stack
- compiler introduces necessary temporary variables
- operand register to build up operands

inmos

Instruction Format

7 4	3 0
Function	Data

Function is e.g. :-

- load,
- store,
- jump,
- add constant etc.

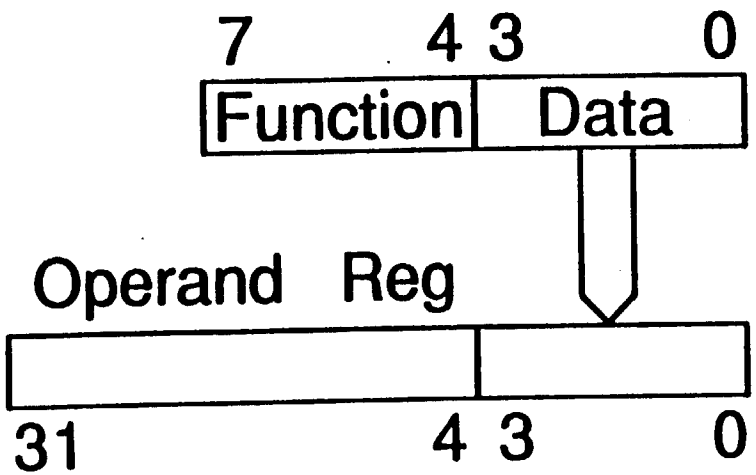
all instructions are 1 byte long

larger operands are built up using operand register

more complicated instructions are built up using operand register

inmos

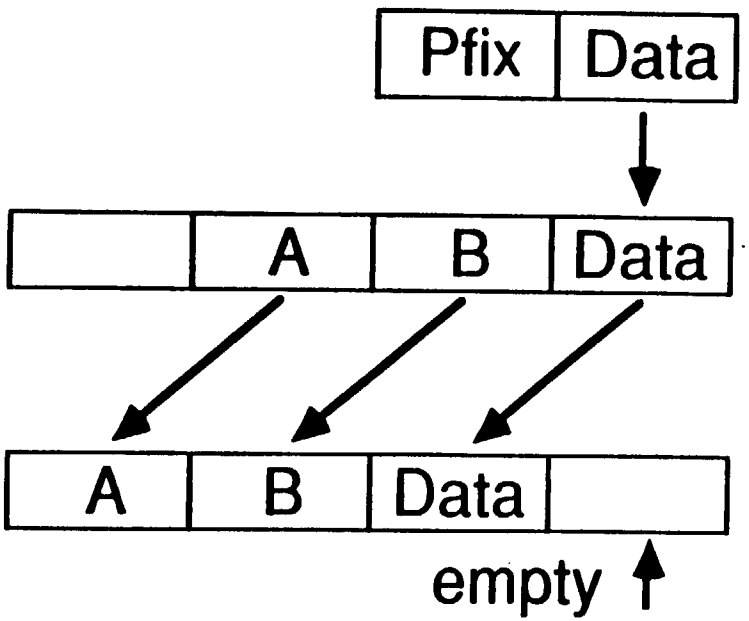
Operand Register



- all instructions load their data into least 4 bits of the operand register
- all instructions then use the entire operand register
- existing bits in higher end of operand register will be used

inmos

Prefix Instruction



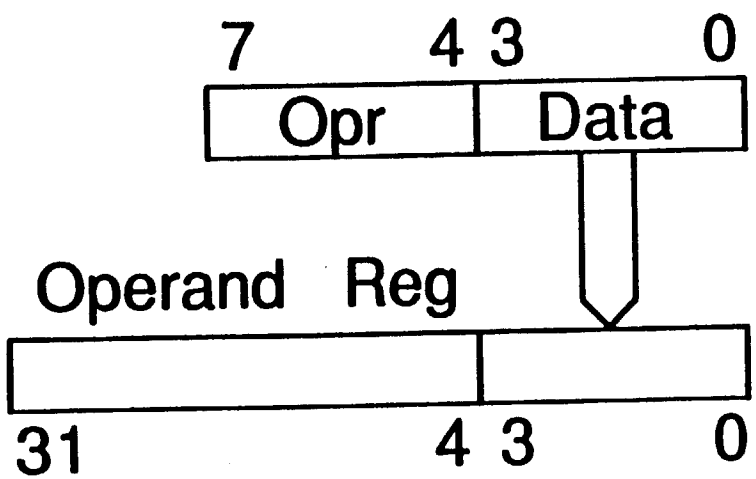
Operand register at start of prefix instruction

Operand register at end of prefix instruction

Negative prefix inverts register after load

inmos

Operate Instruction



- operate instruction loads its data into least 4 bits of the operand register
- processor then decodes the operand register as a new instruction
- 31 instructions available within one byte
- remainder of instruction set uses two bytes (accessed by using prefix then Opr)

inmos

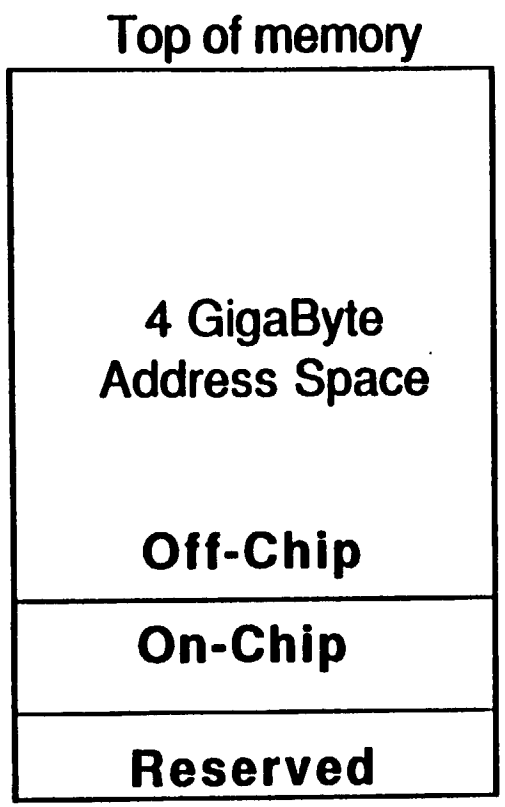
Code size

Single byte format reflects frequency of

local variable access
small jumps
small immediate operands

Typical HLL programs require less than 30% of instructions to be prefixed, resulting in very compact code.

T414 Address Space



linear address space

byte addressed

off chip accesses can cycle at 150ns.

on chip RAM cycles at 50ns.

performance gain from using high speed RAM.

inmos

The Interprocessor Communication Links

- point to point communication
- bidirectional
- serial
- Up to 20 Mbits per second in each direction
- no need for common clock
- DMA supported
- automatic acknowledgements per byte

inmos

Data and Acknowledge Packet

Start		byte of data	
1	1	10101010	0
	type		stop

- data packet

Start	
1	0
	type

- acknowledge packet

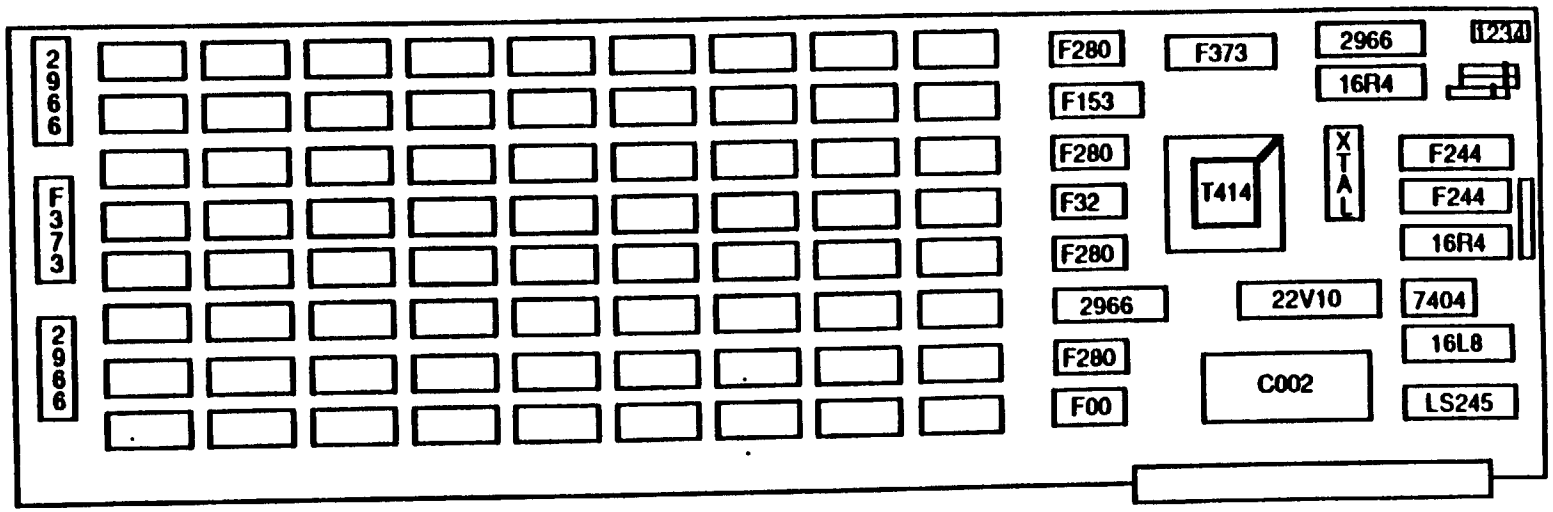
inmos

Interrupts

- want to force an interrupt service routine to run on receipt of an "interrupt" signal
- can do this by communicating with a high priority process which is currently waiting
- EventReq and EventAck form a handshaking pair of signals, which can cause a process to be scheduled
- 2900 ns maximum latency using internal RAM

inmos

Example Small DRAM System



B004 Transputer Evaluation Board - IBM PC

- 2 MBytes Dynamic RAM with parity
- T414 32 bit transputer
- C002 link adaptor

inmos

IMS T414 program execution

Language construct	Program size (Bytes)	Execution time (Cycles)
Operators		
+,-	1	1
*	2	39
/	2	40
=,<>,<,>,<=,>=	2	3
Expressions		
variables on chip	1	2
expressions as subscript	1	1
constants	1.3	1
Assignments		
to on chip variable	1	1
vector size n on chip	4	2n + 4

Average Figures

IMS T414-20 executes 20 cycles per microsecond

inmos

Single transputer benchmarks

Dhrystones/sec		Whetstones/sec	
IMST414-20	**11,345**	**IMSF424-20**	**4,000,000**
I80386-16	6,133	I80386-16+ I80387-16	1,800,000
MC68020-16	4,746	MC68020-16+ MC68881-12	755,000
		IMST414-20	**647,000**
Typical integer performance		Single precision floating point Single chip F424 INMOS estimate Chip set 386/387 Intel estimate	

inmos

Transputer Development Systems

IBM PC or VAX/VMS based tools

Mixed language development
-occam for concurrent programming
-C,Pascal and Fortran to preserve investment

Source level debug

Use with evaluation boards or your own design

inmos

Transputers and DSP

The transputer allows DSP algorithms to be distributed amongst several processors.

Essentially linear price / performance tradeoffs .

For floating DSP, the F424 should be used, as it incorporates an on-chip floating point engine.

For improved performance dedicated hardware, e.g. IMS A100, can be memory mapped.

inmos

The IMS A100

Cascadable Signal Processor

A next generation signal processing system component

inmos

INMOS Design Goals

Address high performance DSP market (3 - 3000 MOPs)

Overcome traditional DSP hardware design complexity for high performance systems

Eliminate requirement for separate language for programming DSP devices

Produce a complete DSP systems design methodology

Develop a range of products which offer a genuine price/performance tradeoff with extensibility

Ship and support a complete solution for system engineers

inmos

Key DSP System Engineering Problems

Input/Output bottlenecks

Interfacing to system host processors

Design time i.e. ease of use

Support tools

Goal: *Enable the systems engineer to concentrate on the algorithms, not the hardware or software required to implement them.*

Conventional Approaches

Most DSP chips, wether building blocks (MAC, bit slices), or single chip solutions, concentrate on solving:

$$y = \sum_{i=0}^{N} C_i X_i$$

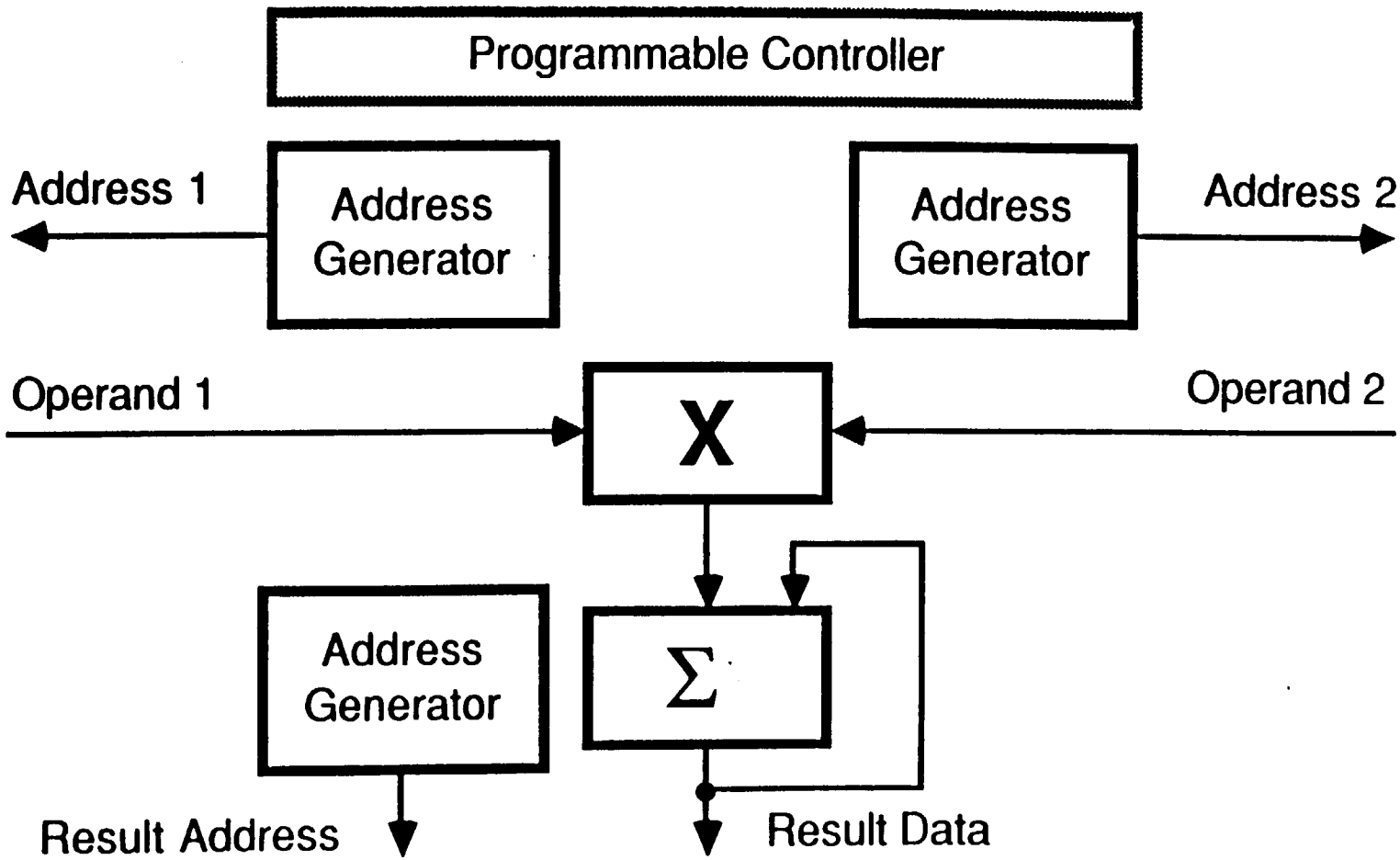
where:

C_i is some algorithm related coefficient, and

X_i is an input data sample

inmos

Conventional Approach Block diagram



INMOS Approach

Most DSP algorithms can in fact be expressed in the general form:

$$y(kT) = C(0)x(kT) + C(1)x((k-1)T) + \ldots$$
$$\ldots + C(N-1)x((k-N+1)T)$$

where:

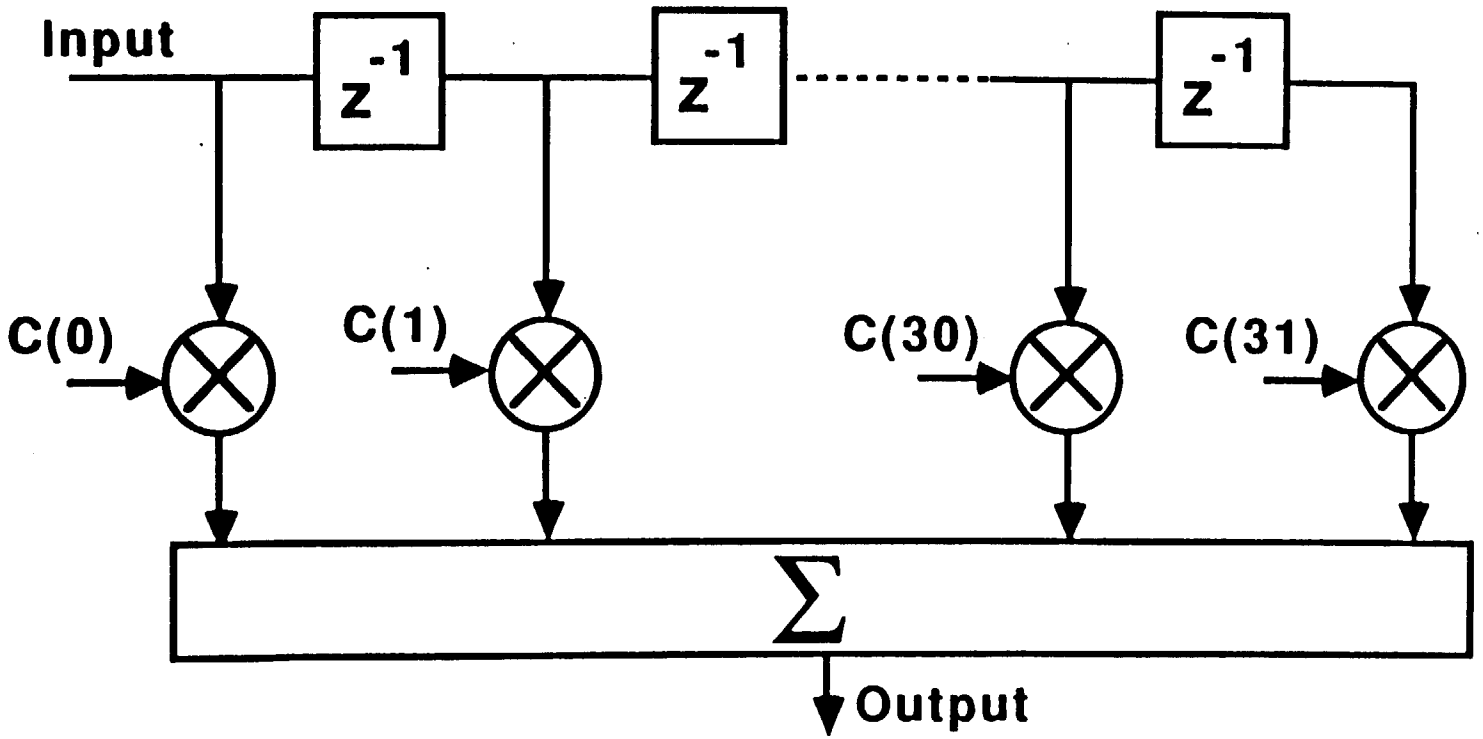
x(kT) is the current time sample,

x((k-1)T) is the previous time sample, etc

The key is to recognize that in evaluating this time sequence, the only data of any use is the final answer.

inmos

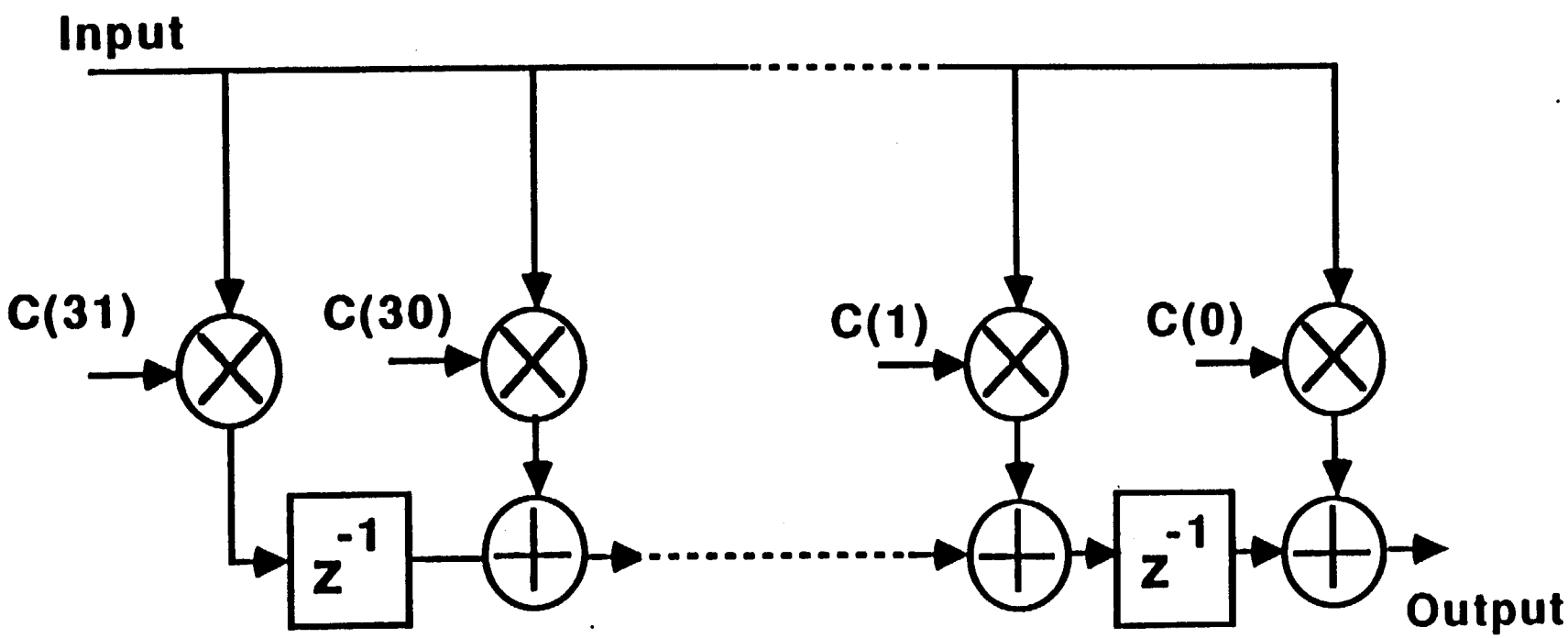
Canonical transversal filter architecture



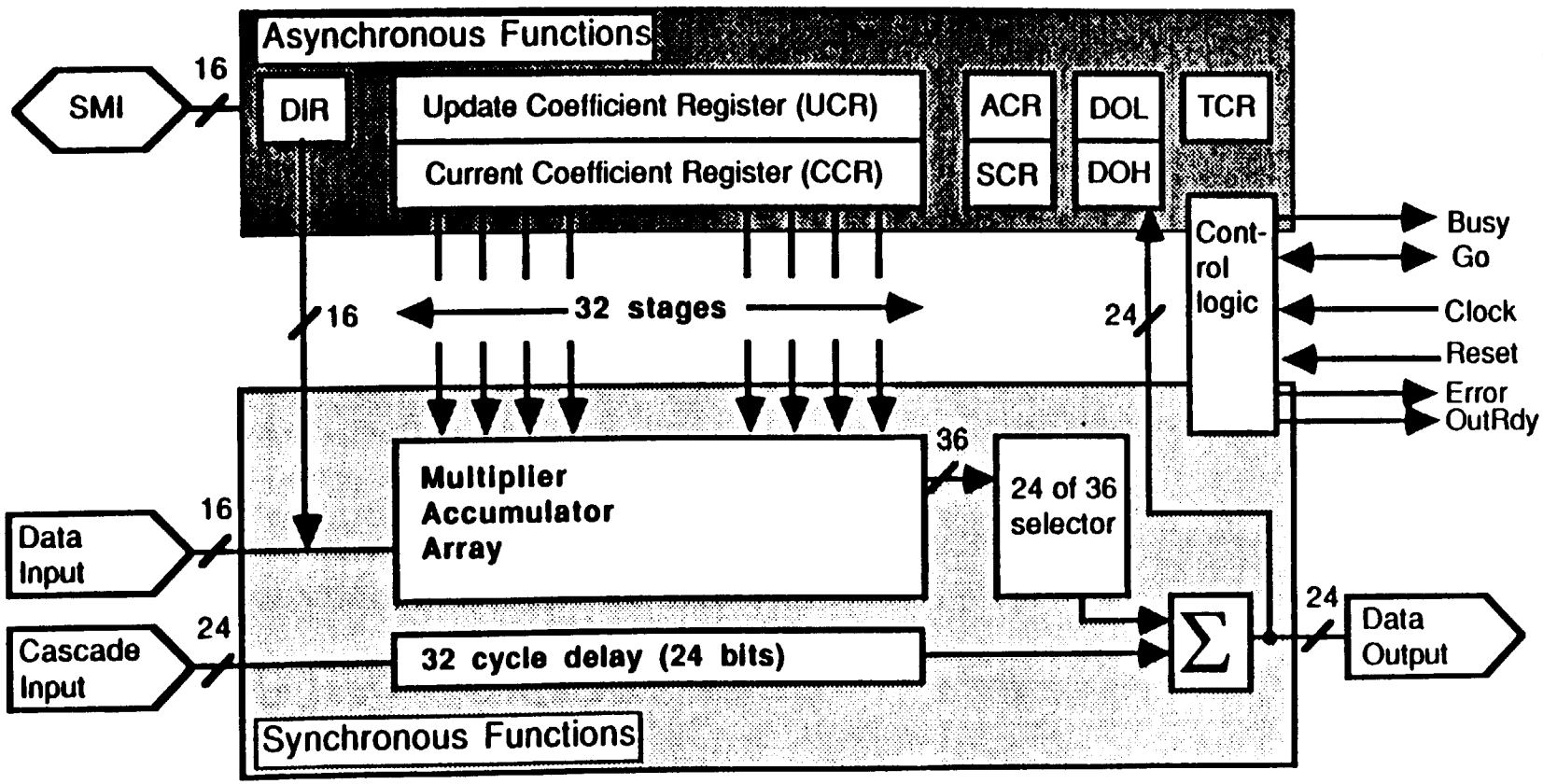
z^{-1} Represents one delay stage

inmos

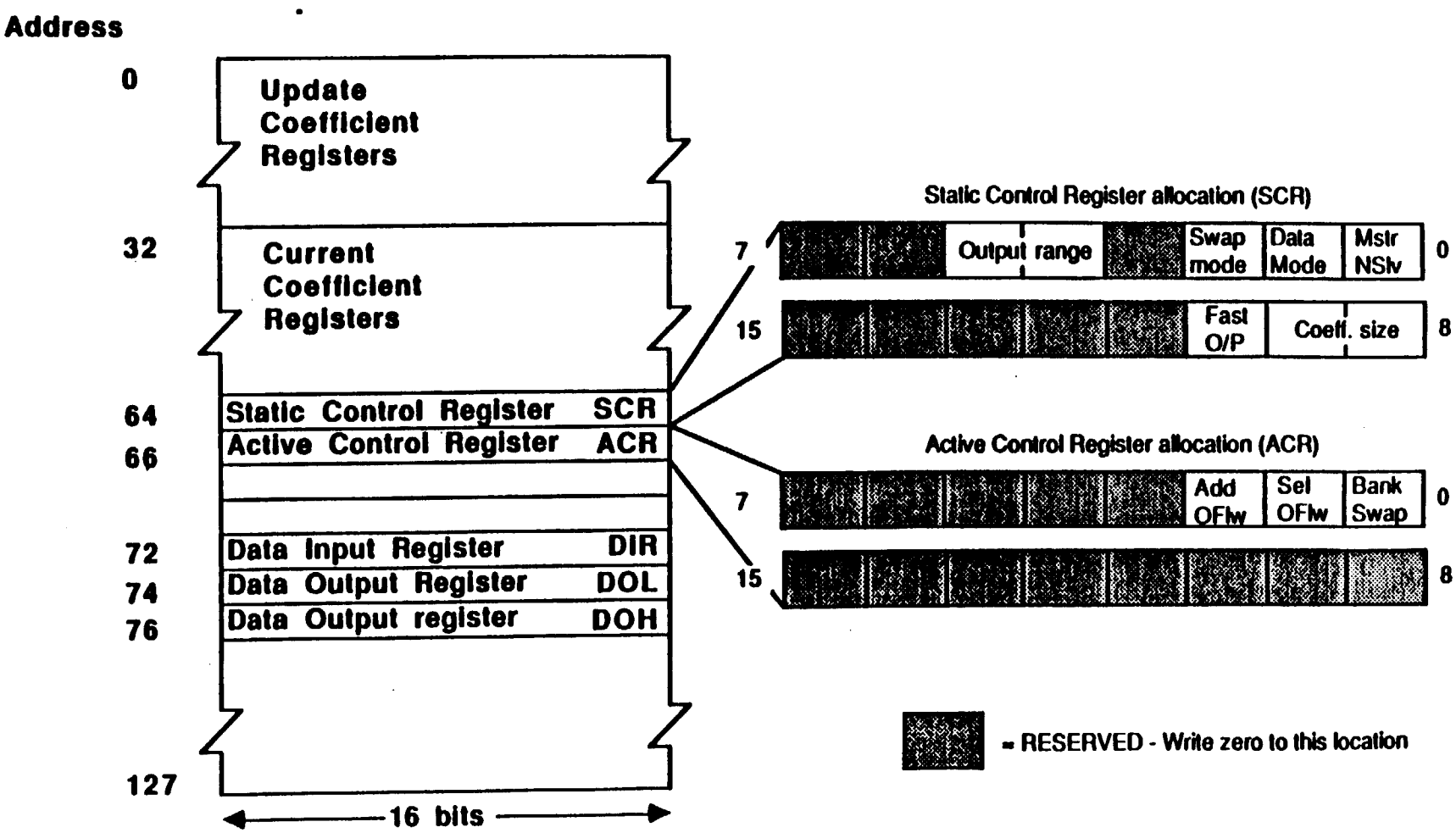
Modified transversal filter architecture



IMS A100 User Model

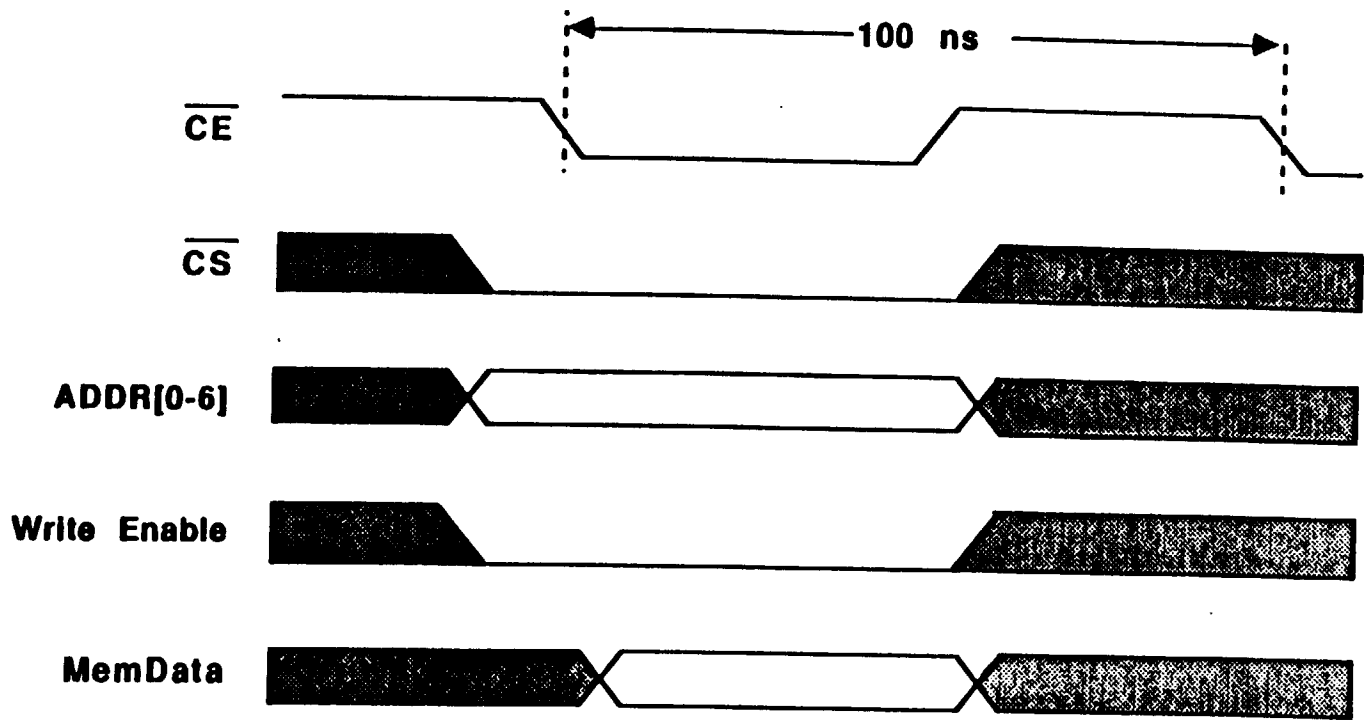


IMS A100 memory map



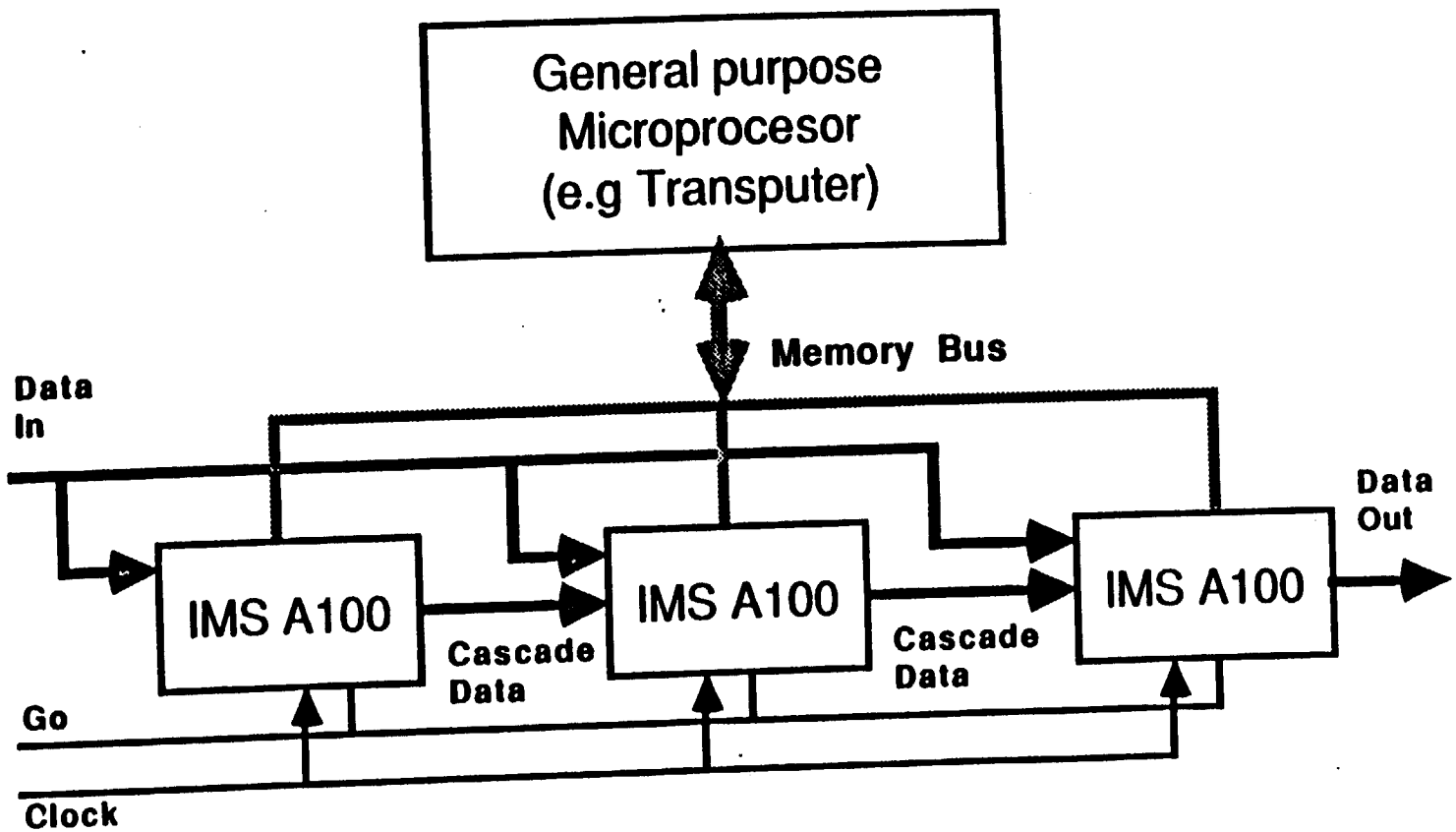
inmos

IMS A100 Write Cycle



inmos

A simple IMS A100 based system



inmos

Primary Functions

Fixed and adaptive FIR or IIR filters up to 10M Samples/s

Correlation and convolution

Discrete Fourier Transforms

Pulse compression and beamforming

Fixed point matrix multiplcation

Note : *All algorithms can be use either real or complex data and coefficients*

inmos

Application areas

Communications

Radar and sonar systems

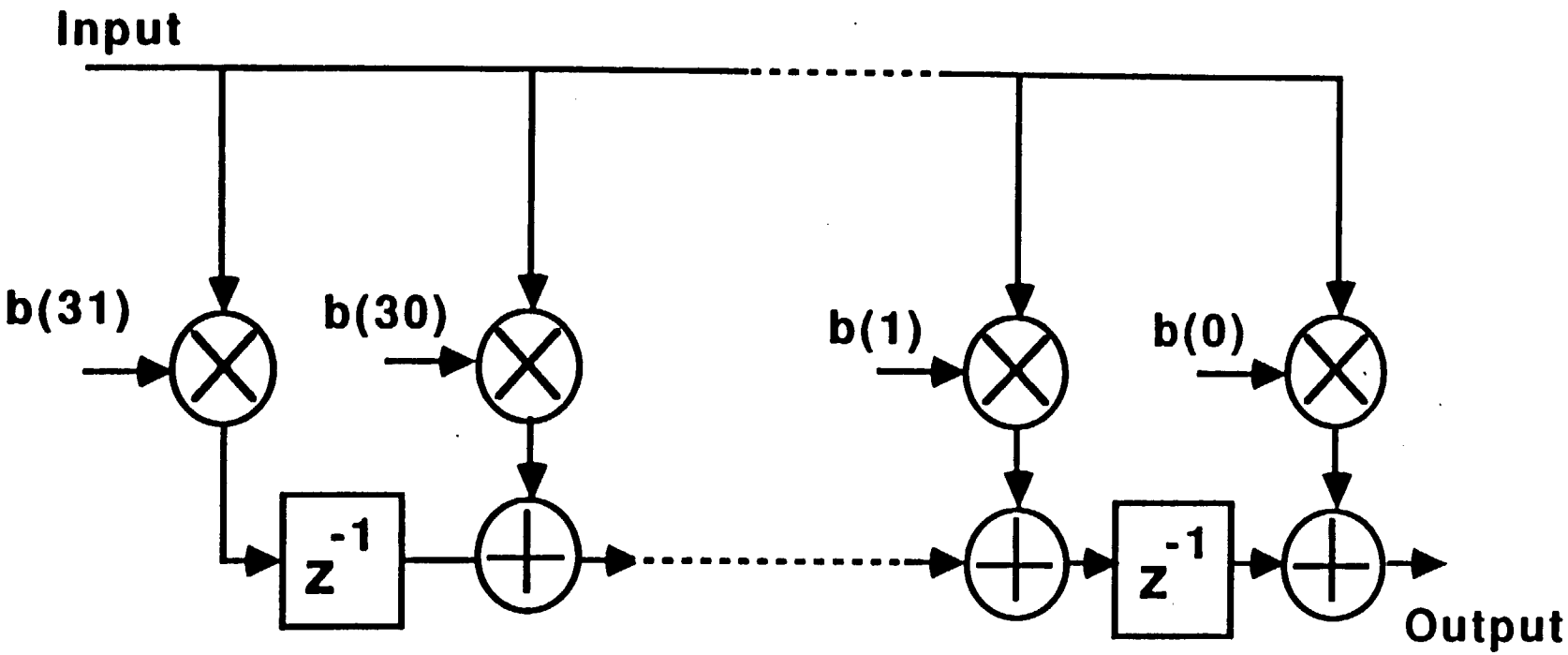
Image processing

Control systems

High speed real time instrumentation

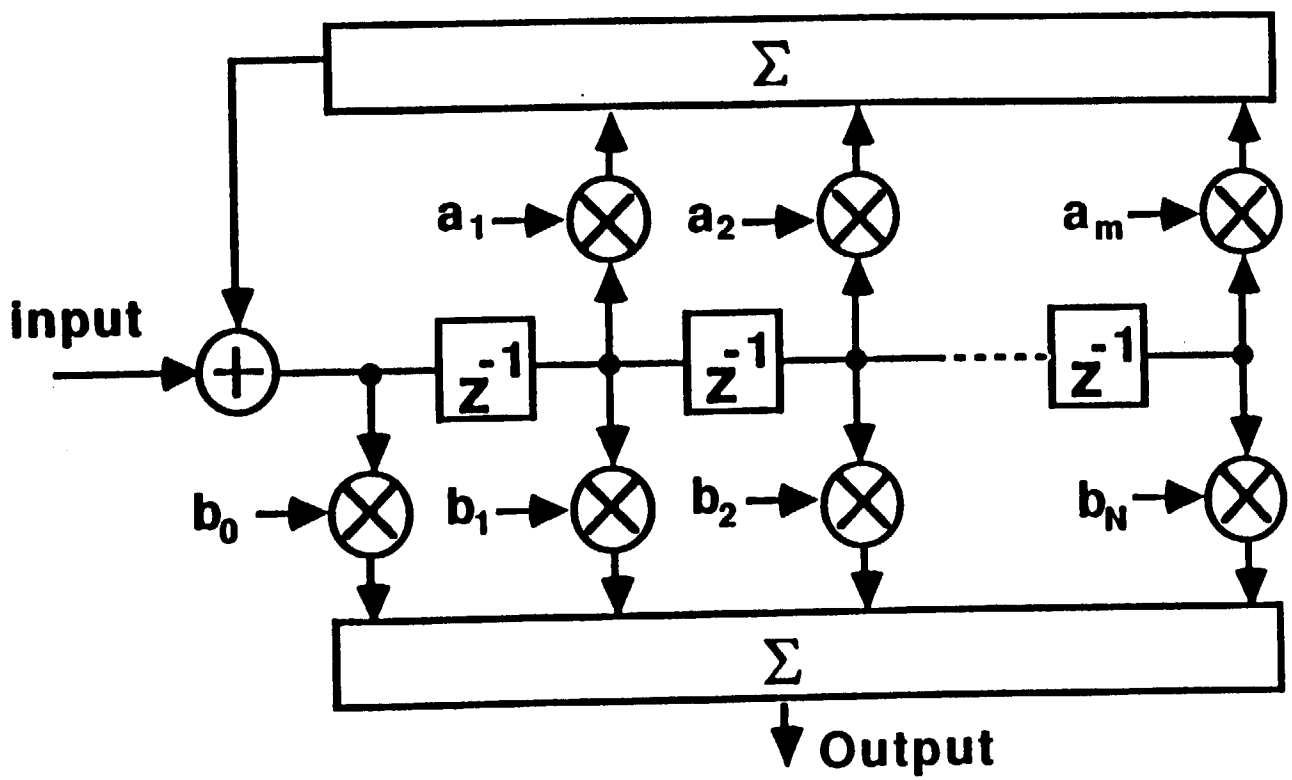
Workstation turbocharger

A100 FIR filter implementation



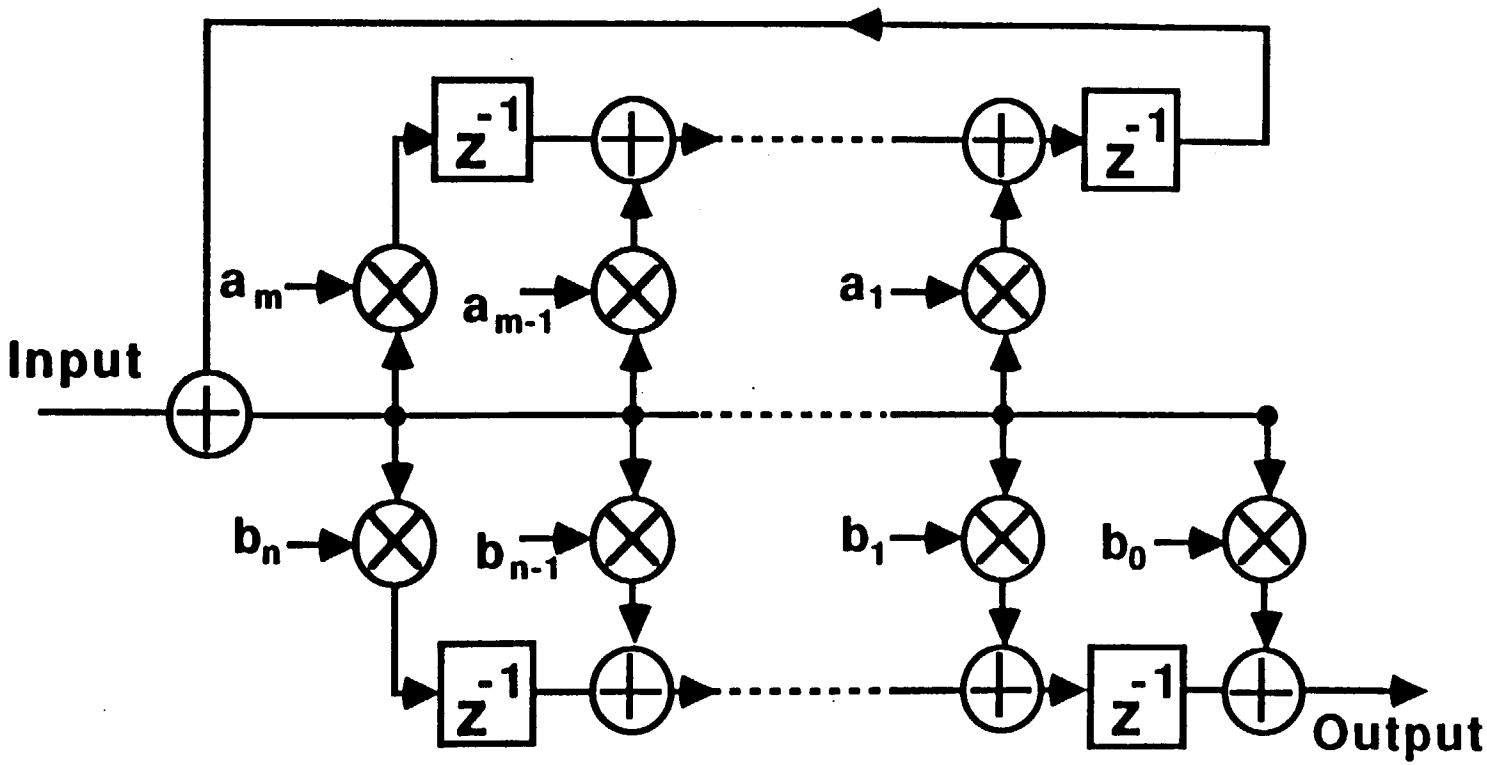
inmos

Recursive (IIR) digital filter



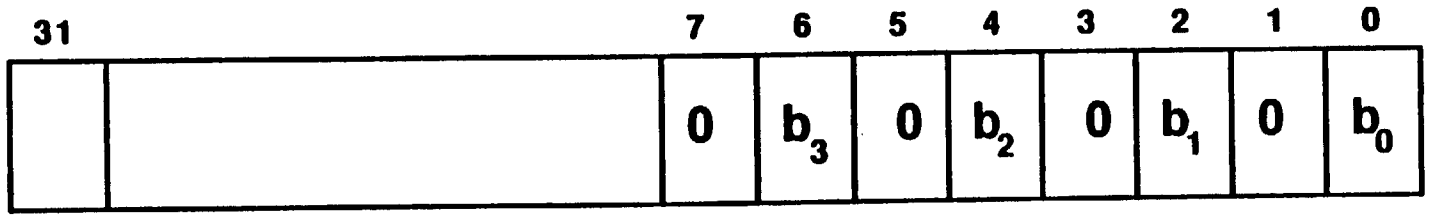
inmos

Modified IIR filter design

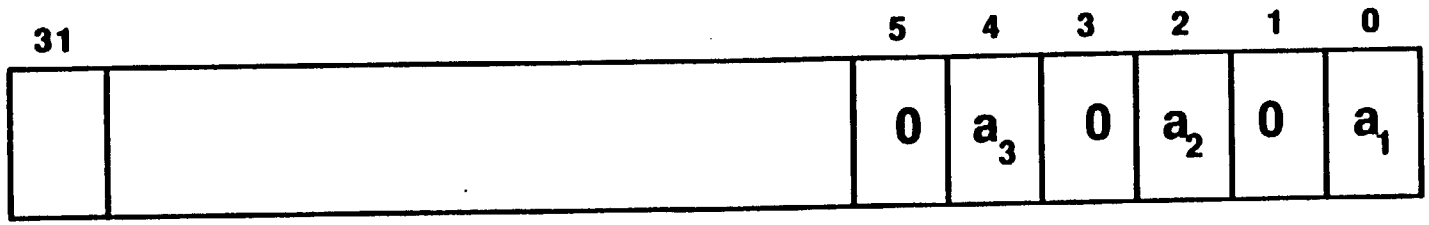


inmos

Coefficient Allocation for IIR filter



Memory Bank A



Memory Bank B

Mode set to : 'Continuous Bank Swap'

inmos

Correlation and Convolution

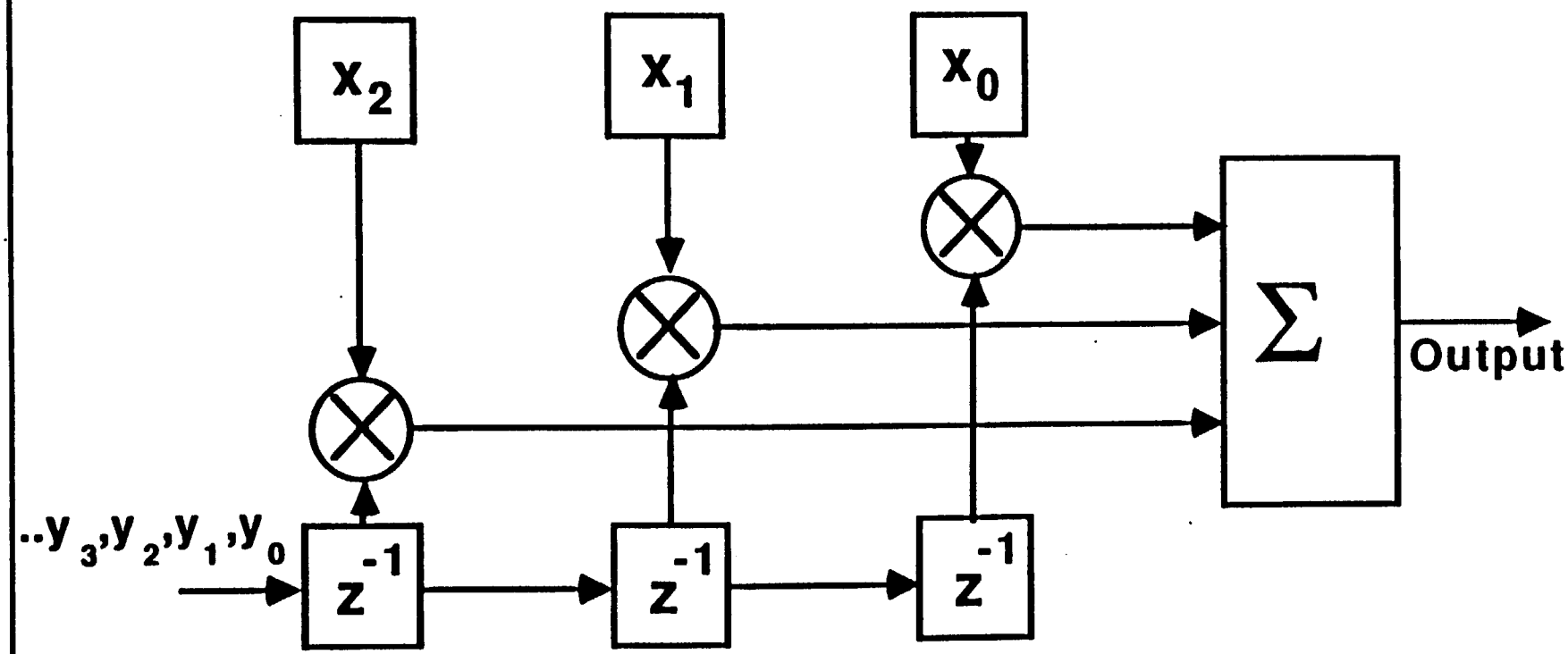
$$R_{xy}(\tau) = \frac{1}{T_0}\int_{\frac{-T_0}{2}}^{\frac{+T_0}{2}} x(t' - \tau)y(t)dt \qquad C_{xy}(\tau) = \lim_{\tau \to \infty} \frac{1}{T}\int_{\frac{-T}{2}}^{\frac{+T}{2}} x(t)y(\tau - t)dt$$

become in discrete time

$$R_{xy}(mT) = \frac{1}{N}\sum_{k=0}^{N-1} x((k - m)T)y(T)dt \qquad C_{xy}(mT) = \frac{1}{N}\sum_{k=0}^{N-1} x(kT)y((m - k)T)dt$$

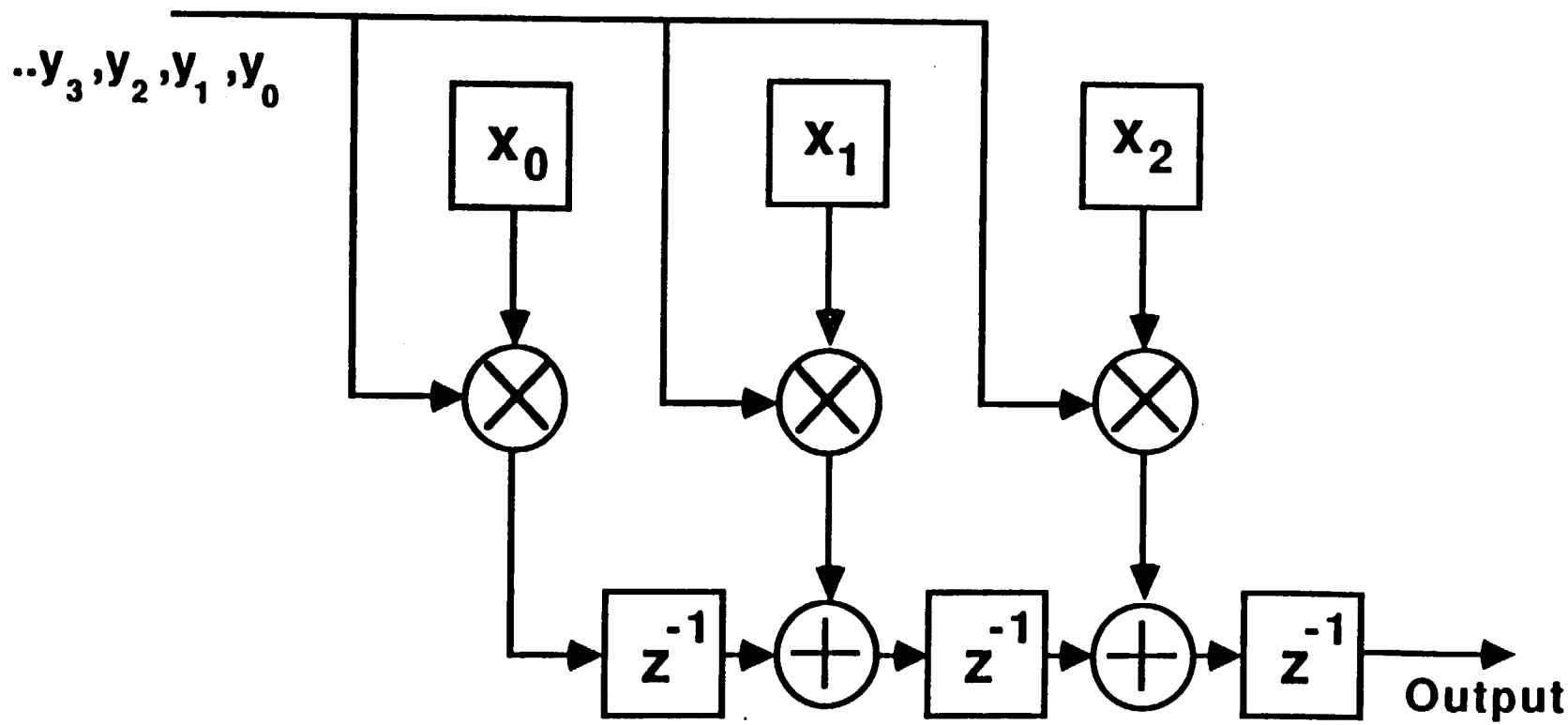
inmos

Conventional Correlator Structure



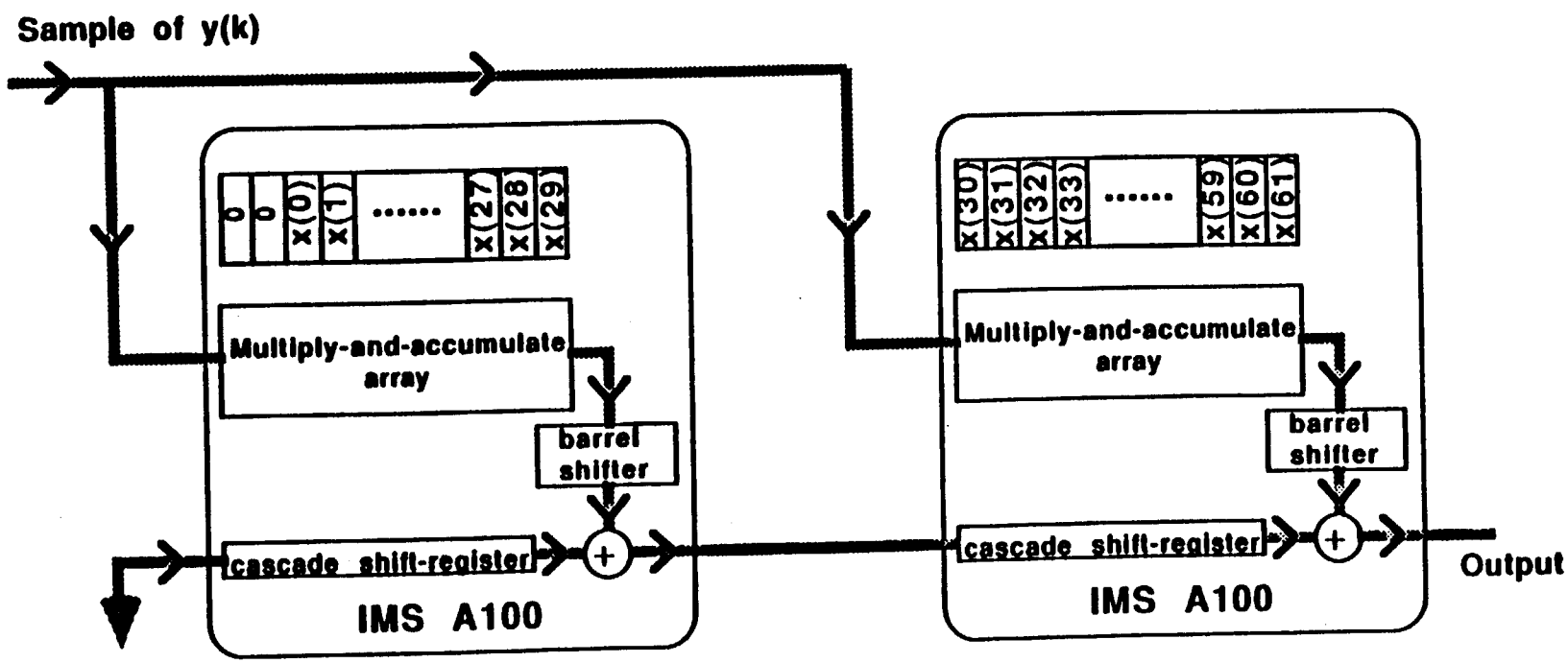
z^{-1} Represents one delay stage

Modified Correlator Structure (A100)

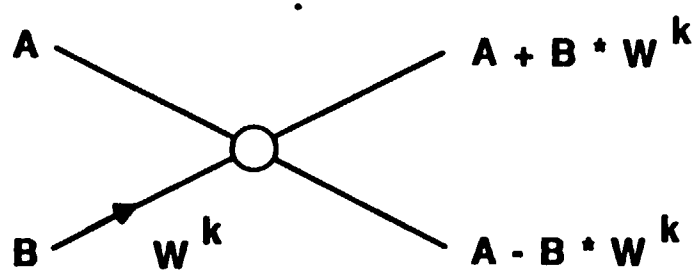


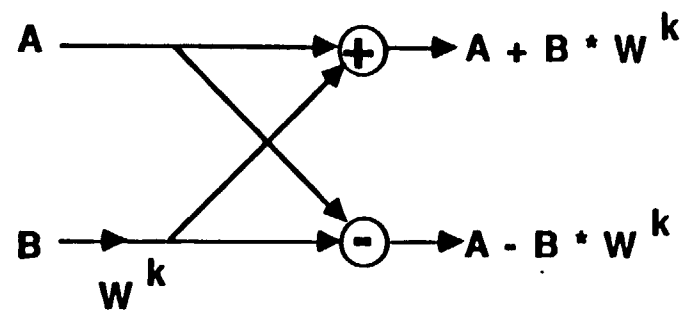
z^{-1} Represents one delay stage

62 point correlator with 2 A100s

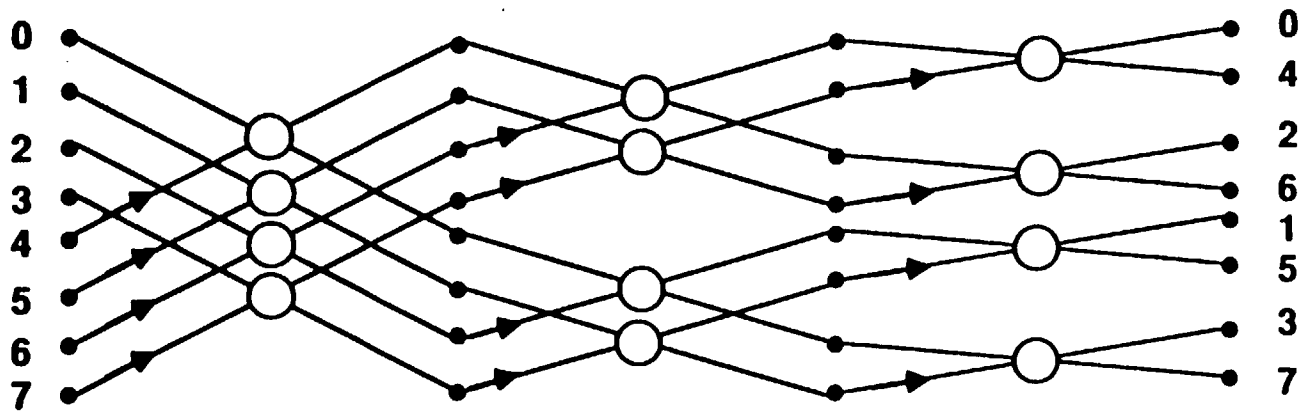


FFT Butterflies



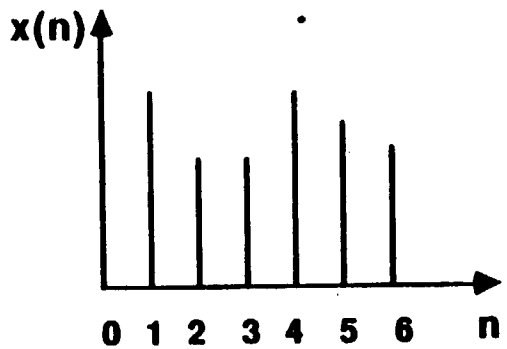


FFT BUTTERFLY



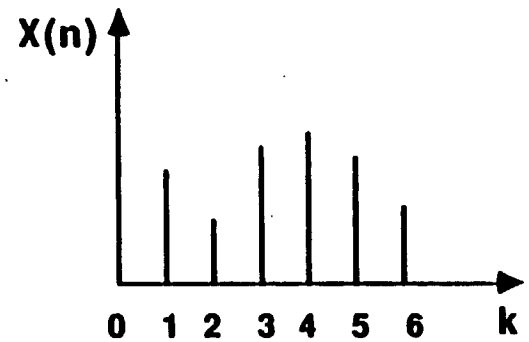
8-POINT FFT

DFT Computation



$$X(k) = \sum_{0}^{N-1} x(n)\, W_N^{nk}$$

where $W_N = e^{-2\pi j/N}$

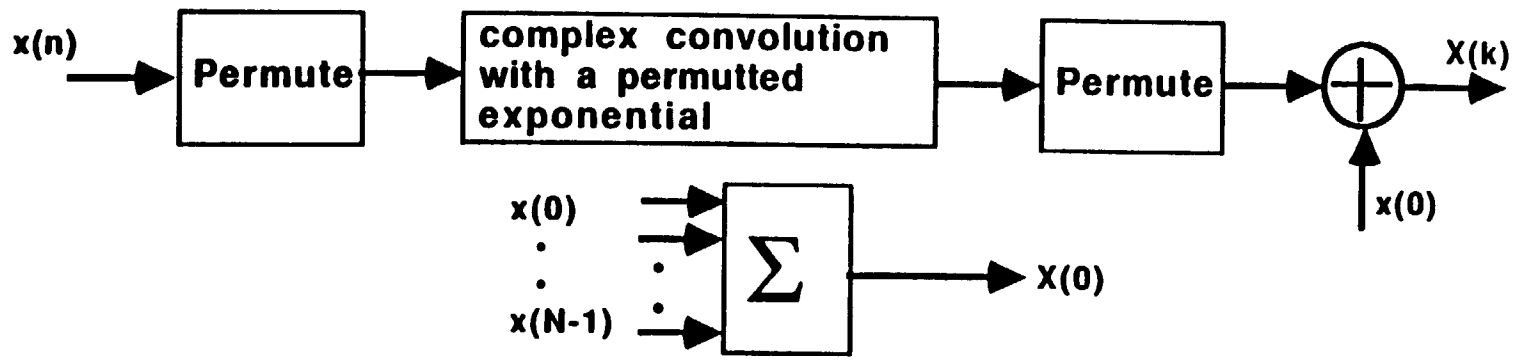


$$\begin{vmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \\ X(4) \\ X(5) \\ X(6) \end{vmatrix} = \begin{vmatrix} W^0 & W^0 & W^0 & W^0 & W^0 & W^0 & W^0 \\ W^0 & W^1 & W^2 & W^3 & W^4 & W^5 & W^6 \\ W^0 & W^2 & W^4 & W^6 & W^1 & W^3 & W^6 \\ W^0 & W^3 & W^6 & W^2 & W^5 & W^1 & W^4 \\ W^0 & W^4 & W^1 & W^5 & W^2 & W^6 & W^3 \\ W^0 & W^5 & W^3 & W^1 & W^6 & W^4 & W^2 \\ W^0 & W^6 & W^5 & W^4 & W^3 & W^2 & W^1 \end{vmatrix} \begin{vmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \\ x(4) \\ x(5) \\ x(6) \end{vmatrix}$$

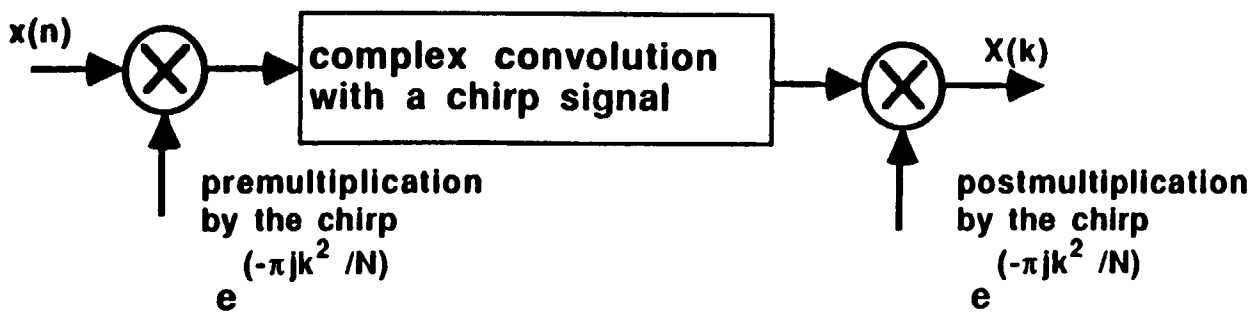
Requires N^2 complex multiplications (49 in this case)

Mapping DFTs onto the A100

a) Radar's Prime Number transform (N prime)



b) Chirp-Z-Transform (Any N)



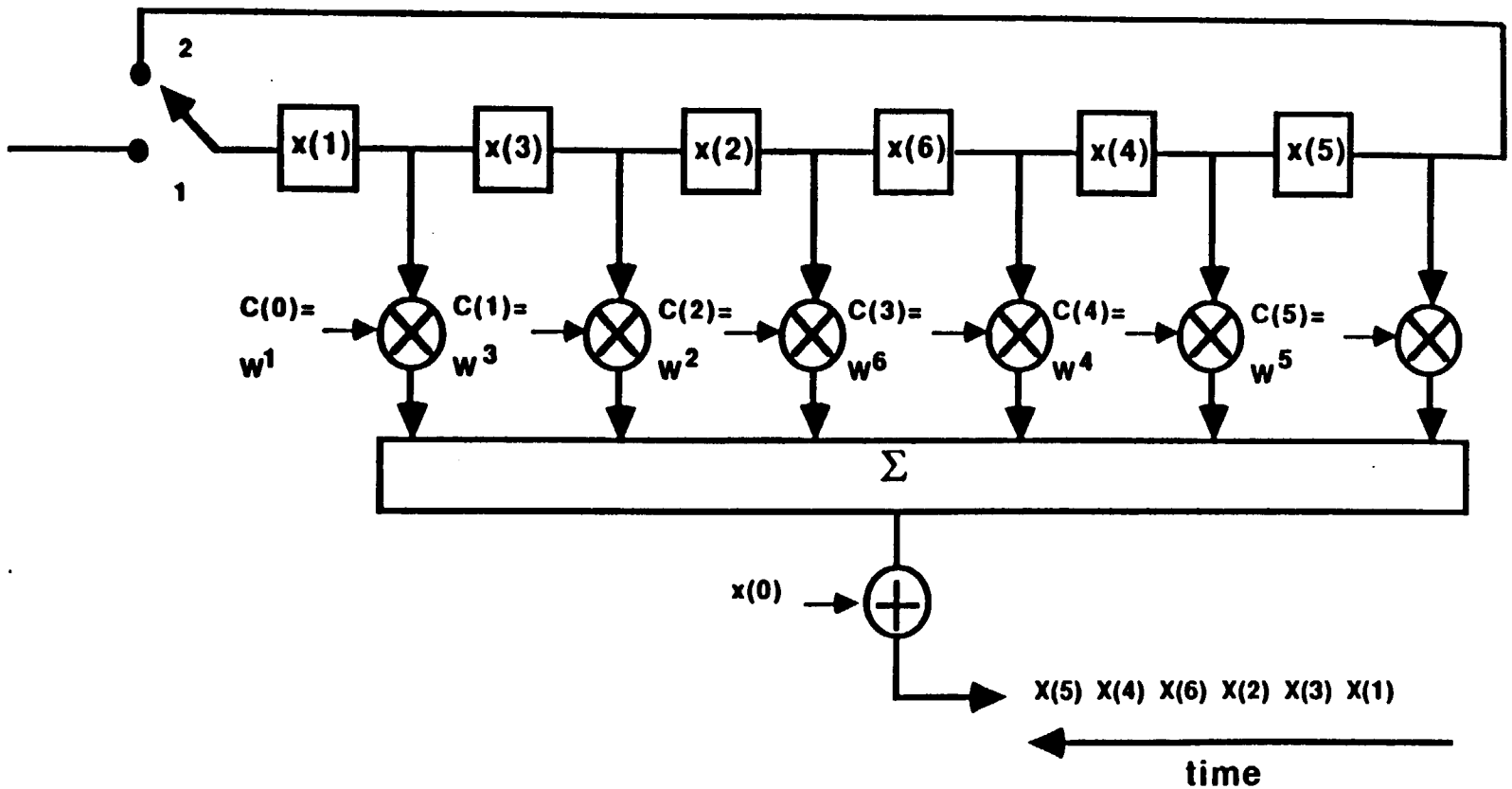
inmos

Radar's Prime Number Transform

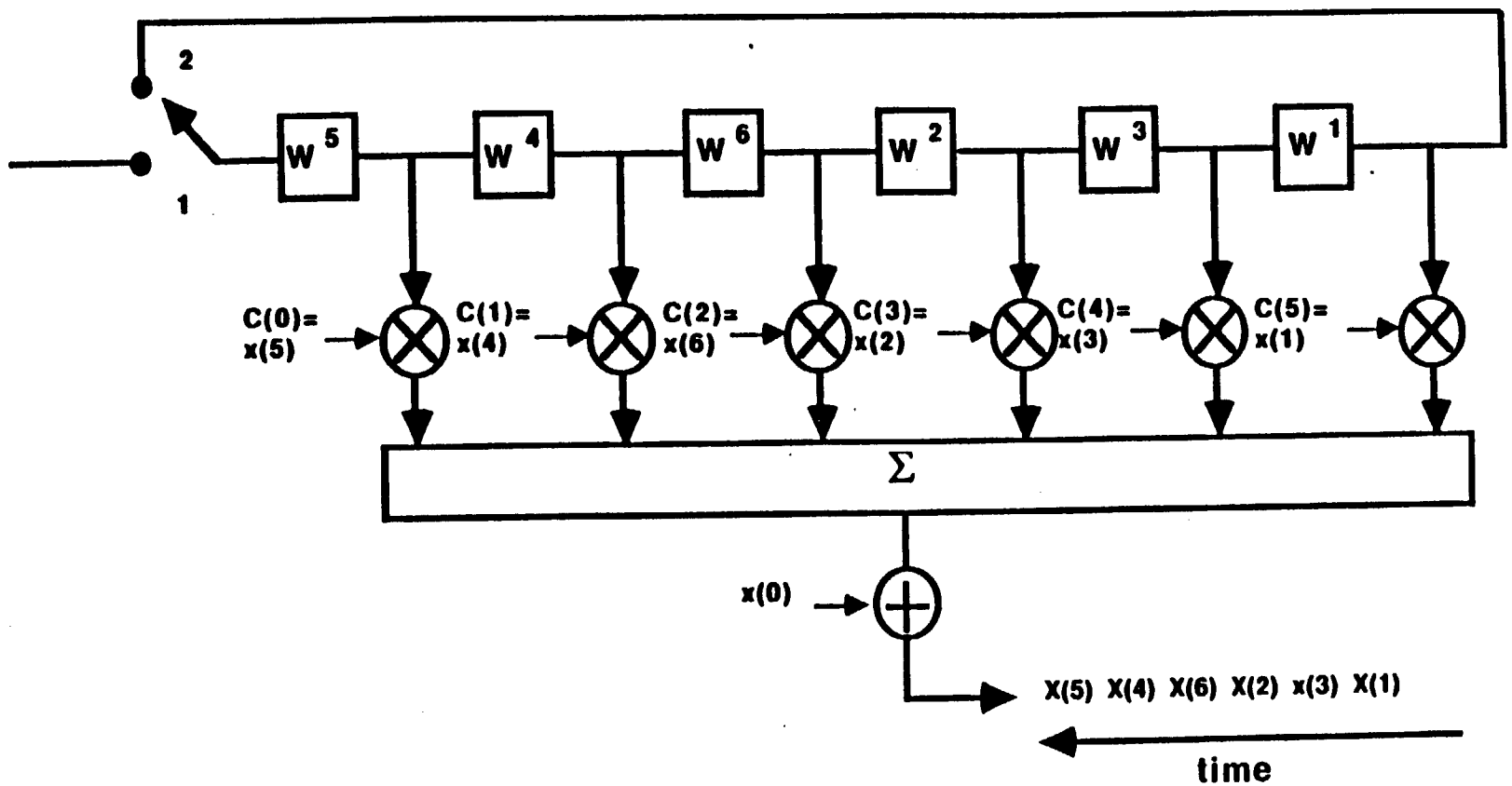
$$\begin{vmatrix} X(1) \\ X(2) \\ X(3) \\ X(4) \\ X(5) \\ X(6) \end{vmatrix} = \begin{vmatrix} W^1 & W^2 & W^3 & W^4 & W^5 & W^6 \\ W^2 & W^4 & W^6 & W^1 & W^3 & W^6 \\ W^3 & W^6 & W^2 & W^5 & W^1 & W^4 \\ W^4 & W^1 & W^5 & W^2 & W^6 & W^3 \\ W^5 & W^3 & W^1 & W^6 & W^4 & W^2 \\ W^6 & W^5 & W^4 & W^3 & W^2 & W^1 \end{vmatrix} \begin{vmatrix} x(1) \\ x(2) \\ x(3) \\ x(4) \\ x(5) \\ x(6) \end{vmatrix} + \begin{vmatrix} x(0) \\ x(0) \\ x(0) \\ x(0) \\ x(0) \\ x(0) \end{vmatrix}$$

$$\begin{vmatrix} X(1) \\ X(3) \\ X(2) \\ X(6) \\ X(4) \\ X(5) \end{vmatrix} = \begin{vmatrix} W^1 & W^3 & W^2 & W^6 & W^4 & W^5 \\ W^3 & W^2 & W^6 & W^4 & W^5 & W^1 \\ W^2 & W^6 & W^4 & W^5 & W^1 & W^3 \\ W^6 & W^4 & W^5 & W^1 & W^3 & W^2 \\ W^4 & W^5 & W^1 & W^3 & W^2 & W^6 \\ W^5 & W^1 & W^3 & W^2 & W^6 & W^4 \end{vmatrix} \begin{vmatrix} x(1) \\ x(3) \\ x(2) \\ x(6) \\ x(4) \\ x(5) \end{vmatrix} + \begin{vmatrix} x(0) \\ x(0) \\ x(0) \\ x(0) \\ x(0) \\ x(0) \end{vmatrix}$$

7 Pt DFT- Circulating Permuted input

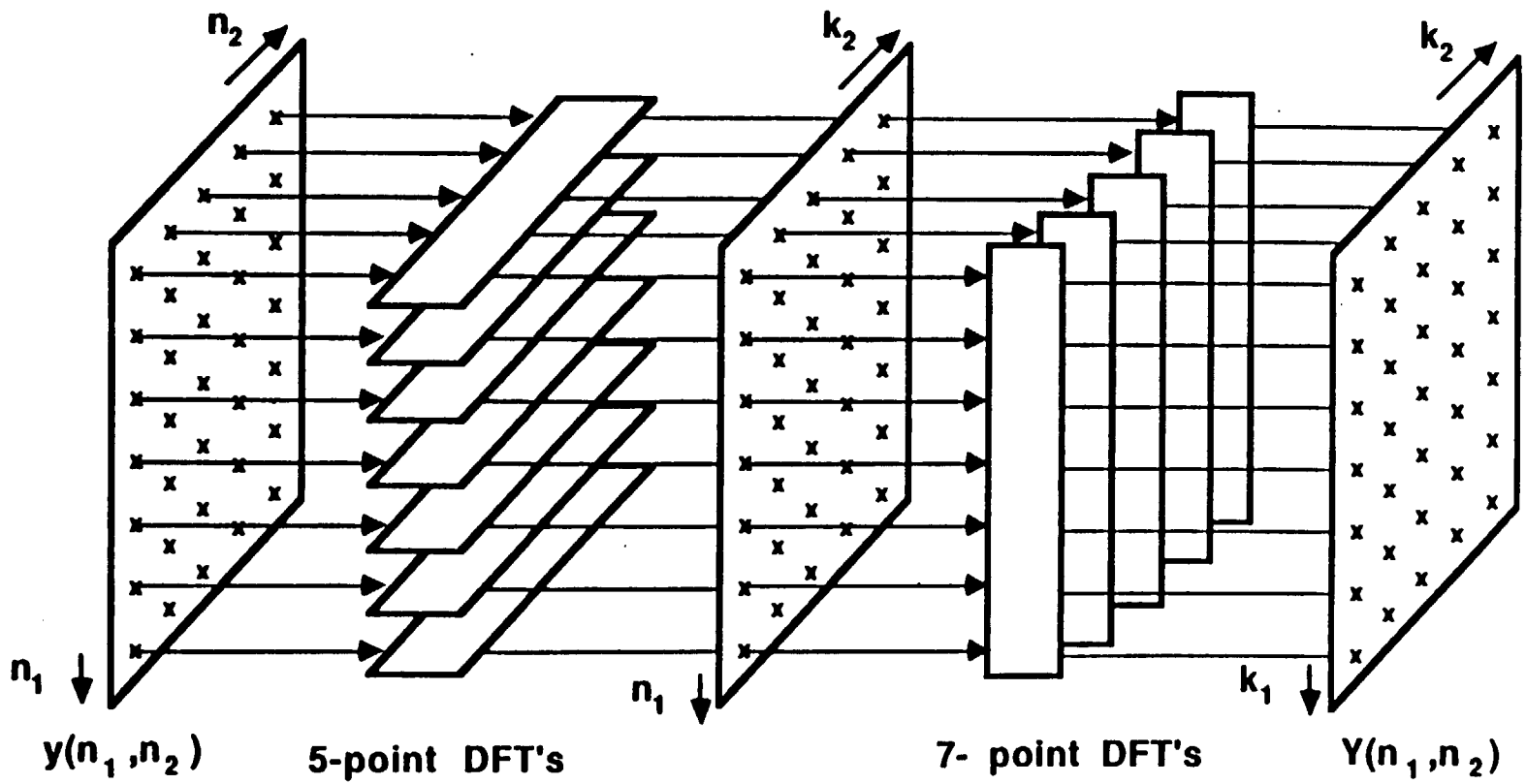


7 Pt DFT- Circulating Twiddle Factors

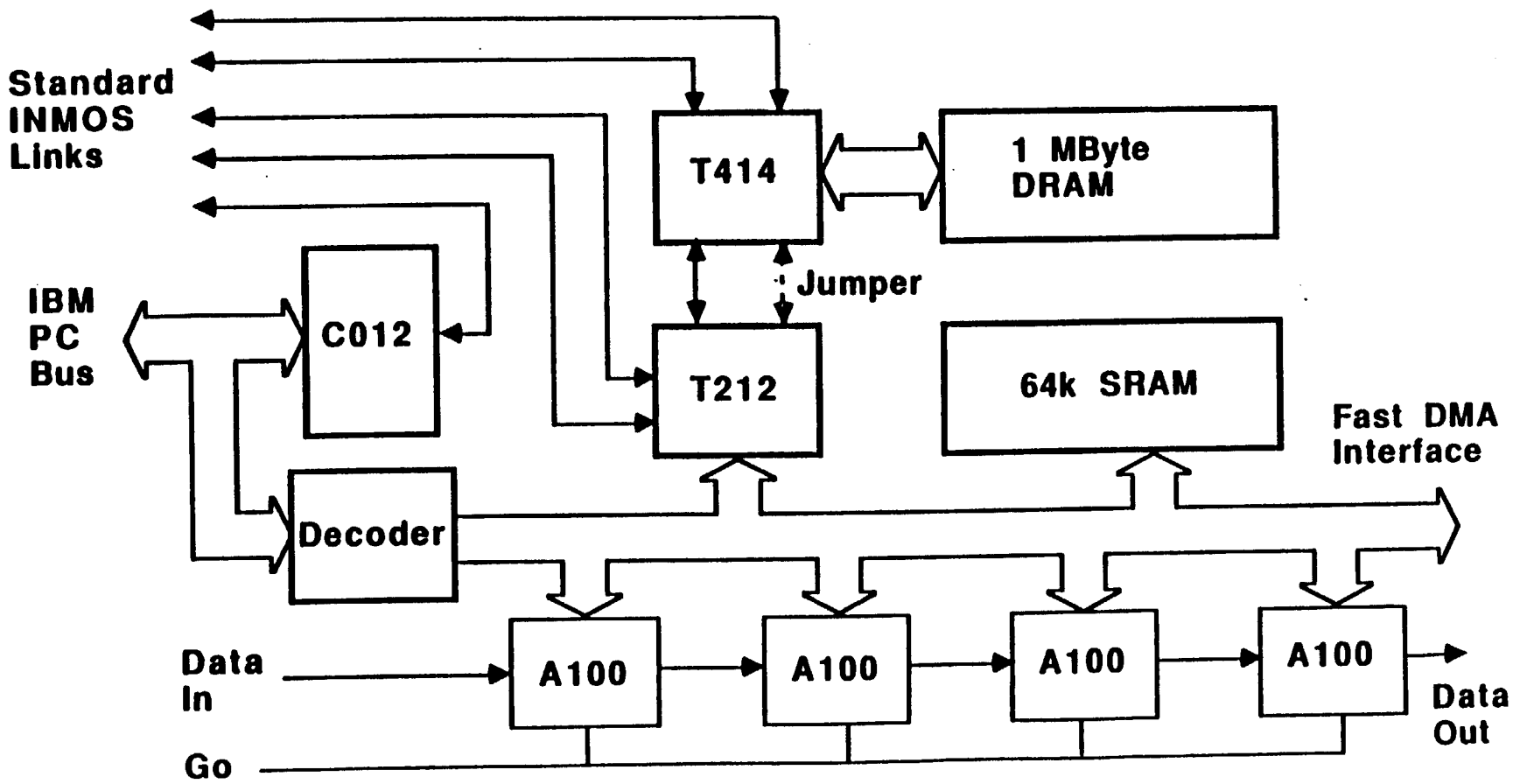


inmos

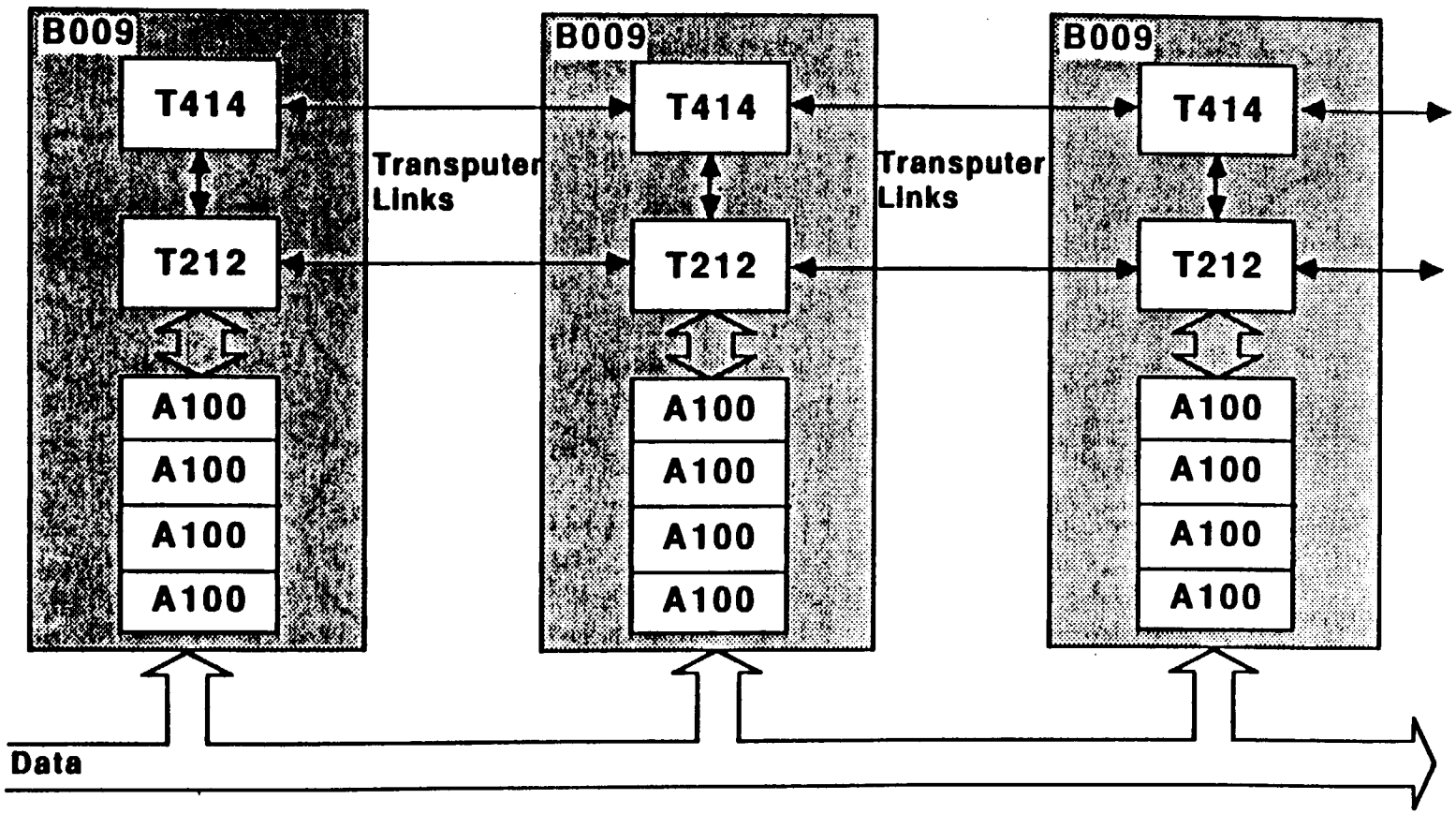
35 Point DFT split into row and column DFTs



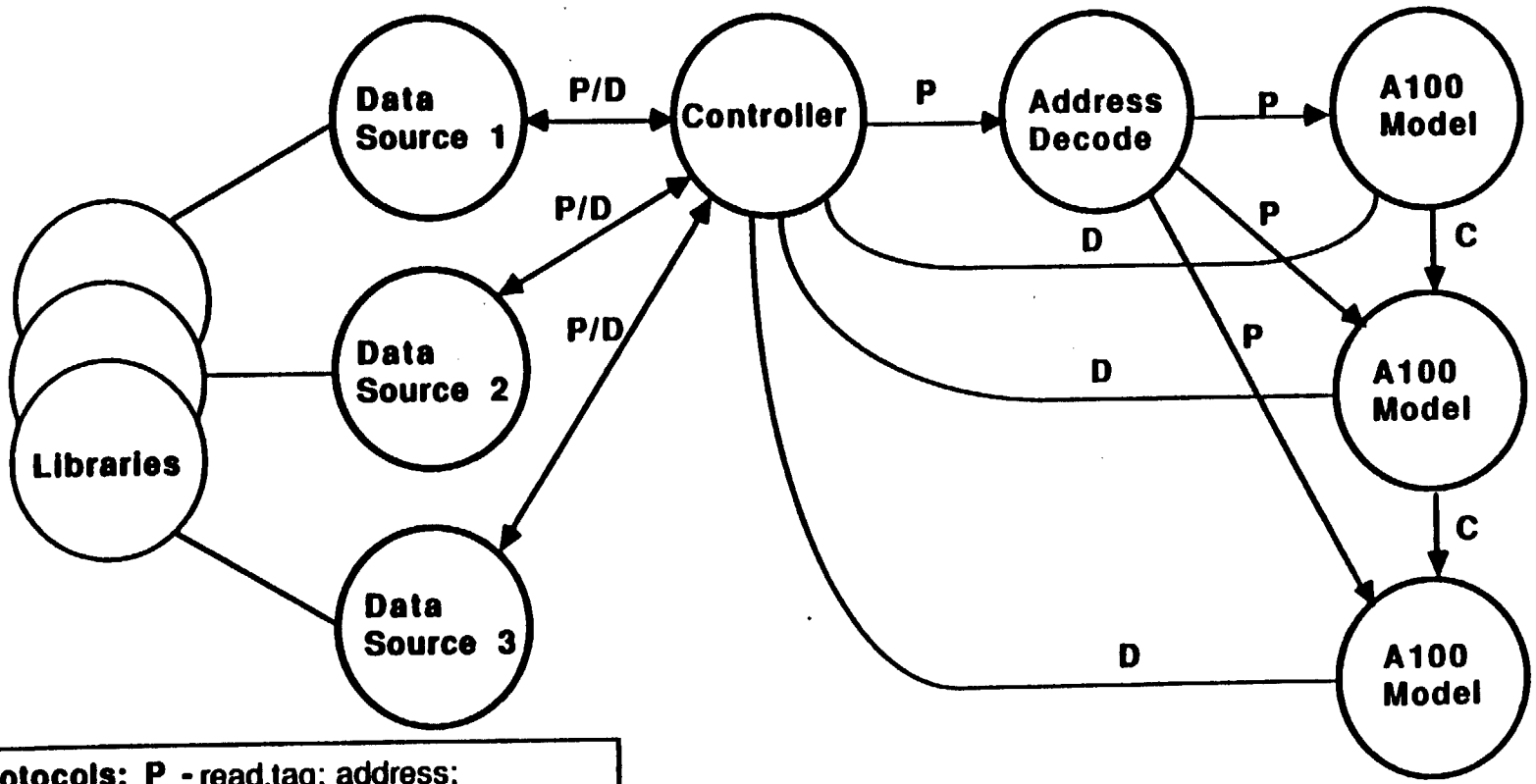
IMS B009: The A100 evaluation Board



Pipieline of B009s



IMS A100 Occam Simulator Overview



Protocols: P - read.tag; address;
write.tag; address; data
D - data(in response to read.tag)
C - cascade data (INT64)

IMS A100 Performance(MOPs)

inmos

Coefficient word size (bits)	Sampling rate (MHz)	Effective number of muliply-and-accumumulates (Millions/sec)		
		one A100	four A100	n A100
4	10	320	1280	320n
8	5	160	640	160n
12	2.3	106	424	106n
16	2.5	80	320	80n

inmos

Performance Comparison

What \ Who	TMS 32020	AMD 29500	IMS A100
32 Stage FIR filter	40kHz	300kHz	2.5MHz (10MHz)
32 Point Correlation	400µS	100µS	16µS (400nS)
1000 Point DFT	20-30mS	3-10mS	2-10mS (<1mS)

inmos

IMS A100 Summary

The IMS A100 changes the rules for high performance DSP by providing unrivalled processing power in a single, easy to use, cascadable device.

INMOS DSP products provide support tools which provide a complete integrated approach to designing high performance DSP systems.

With the IMS A100, INMOS leads the way in a new generation of products which address the real issues in DSP systems engineering.

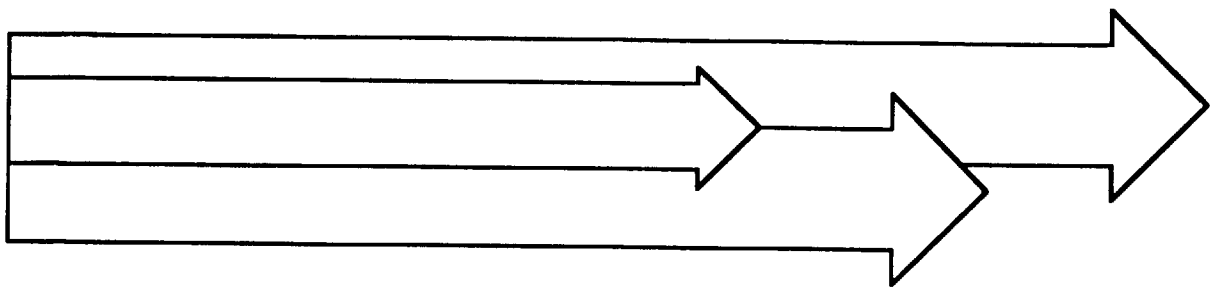
inmos

Component availability

Part No	Description	Price*	Availability
IMS T414	32 bit transputer	420	NOW
IMS T212	16 bit transputer	315	NOW
IMS F424	Floating point transputer	---	Q2 87
IMS A100	Cascadable signal processor	420	Oct 86

* = 100 unit price in $

THE MSM6992 AND MSM699210 DIGITAL SIGNAL PROCESSORS



OKI

SEMICONDUCTOR

INTRODUCTION

I. THE MSM6992 DSP

II. THE MSM699210 DSP

III. COMPARITIVE ANALYSIS OF 22 -BIT FLOATING POINT VERSUS 16-BIT FIXED POINT

IV. PROGRAMMING THE MSM6992/210

V. SUPPORT

VI. EXAMPLE APPLICATION

PRECEDING PAGE BLANK NOT FILMED

I. THE MSM6992 DSP

MSM6992 PHYSICAL PARAMETERS

125,000 TRANSISTORS

2 MICRON DUAL-METAL CMOS

400 mW POWER CONSUMPTION

SINGLE 5V POWER SUPPLY

100/125 NS INSTRUCTION TIME

MAJOR FEATURES

MODIFIED HARVARD EXTERNAL ARCHITECTURE

22 -BIT (16E6) FLOATING POINT FORMAT

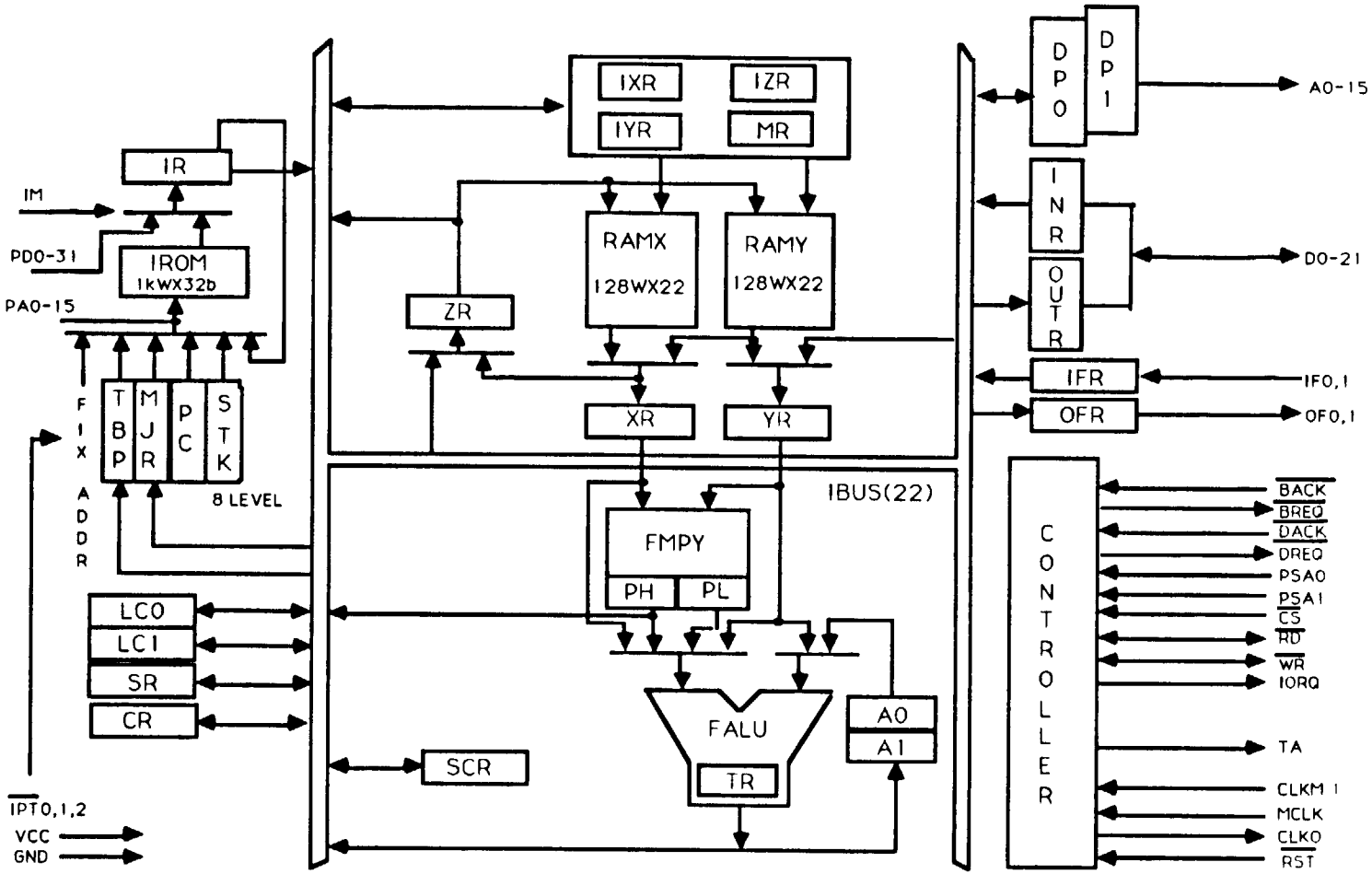
20 MGFLOPS THROUGHPUT

256X22-BIT WORDS INTERNAL DATA RAM

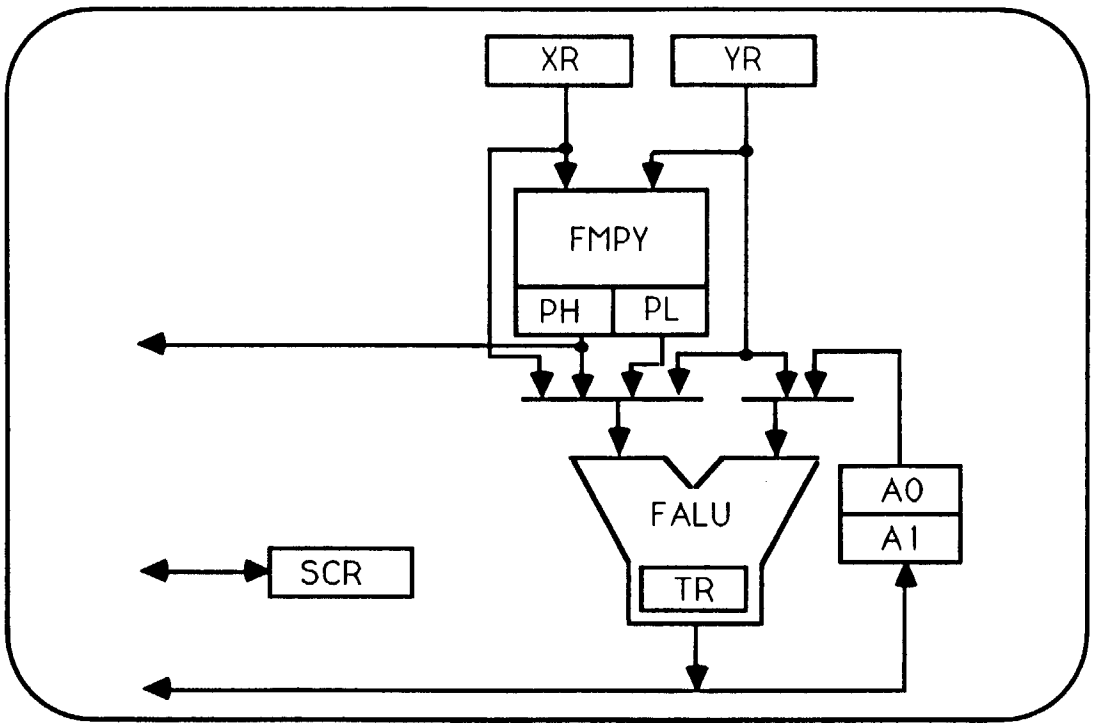
1024X32-BIT WORDS INSTRUCTION ROM

EXTERNAL DATA AND PROGRAM MEMORY
UP TO 64K WORDS EACH

MSM6992 BLOCK DIAGRAM



ARITHMETIC UNIT



FLOATING POINT MULTIPLIER
TWO-STAGE FLOATING POINT ALU
TWO ACCUMULATORS

TWO-STAGE FALU OPERATION EXAMPLE

SA = MA X 2**EA

SB = MB X 2** EB

ALU OPERATION: SA + SB

STAGE 1. BIT ALIGNMENT AND ADDITION (RIGHT SHIFT)

SB = [MB X 2**(EB-EA)] **EA = MC X 2**EA

TR = (MA + MC) X 2**EA = MD X 2**EA

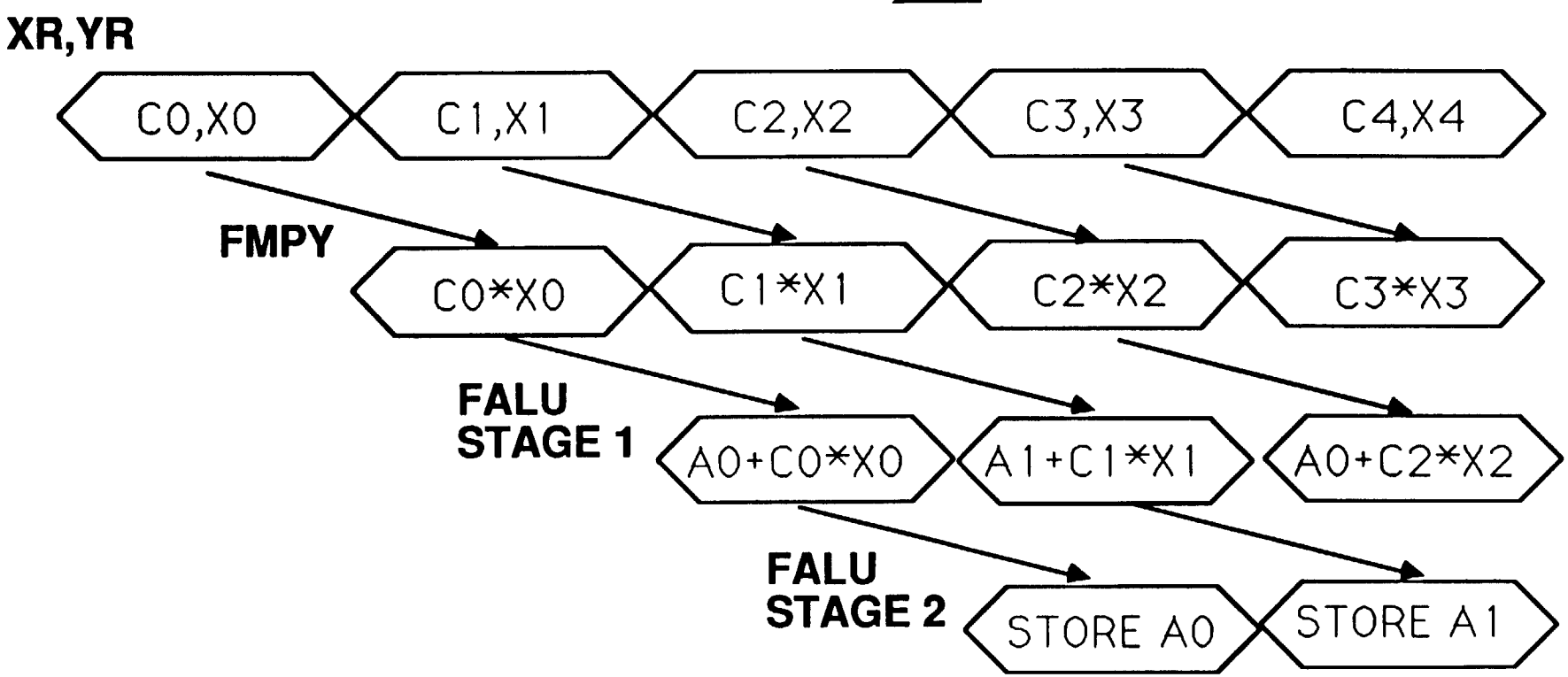
STAGE 2. POST-NORMALIZATION (LEFT SHIFT)

ALU = [MD X SCALE_FACTOR] X 2**EC = ME X 2**EC

NOTE: STAGES ARE PIPELINED, ALLOWING FOR A FULLY POST-NORMALIZED RESULT EVERY 100 NS

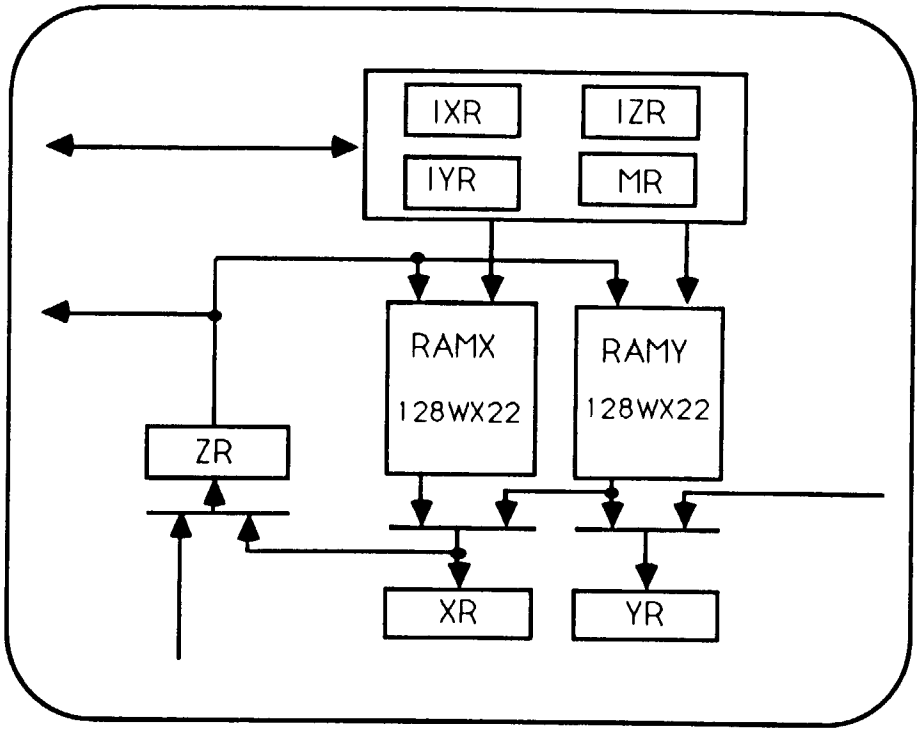
CALCULATION OF A SUM OF PRODUCTS

$$\sum C_i X_i$$



TO COMPLETE THE SUM OF PRODUCTS ADD A0+A1

INTERNAL DATA MEMORY

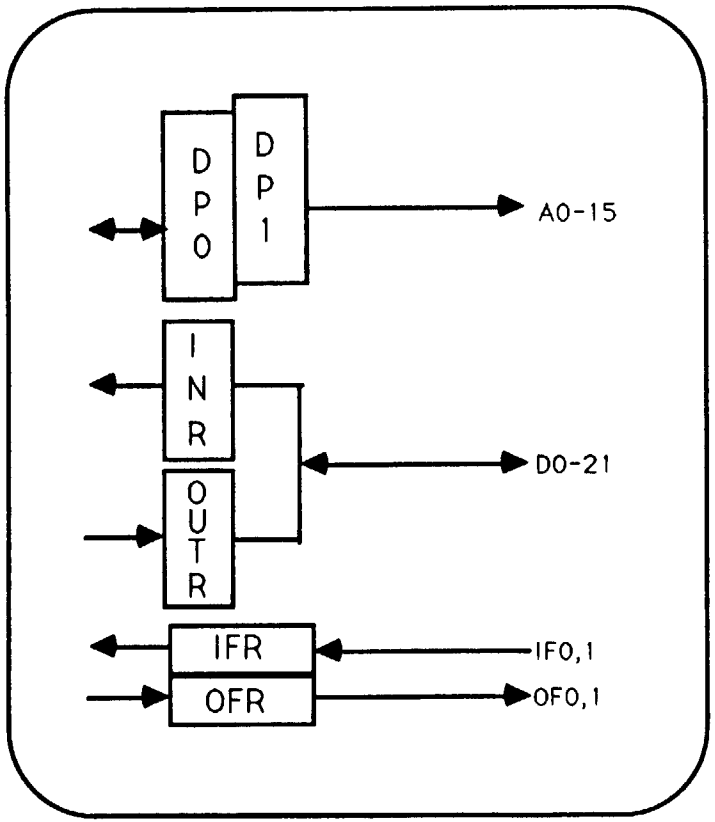


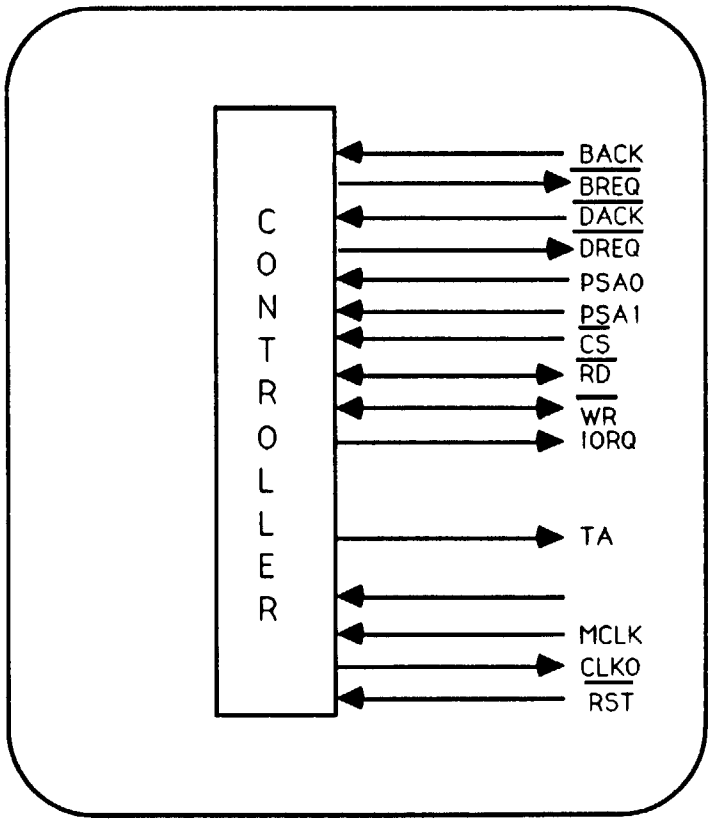
THREE OPERAND MOVES IN ONE INSTRUCTION

DATA SHIFT POSSIBLE IN ONE CYCLE

RAM AND RAMY CAN BE ACCESSED SEPERATELY
OR AS A SINGLE PAGE RAMXY

EXTERNAL INTERFACE



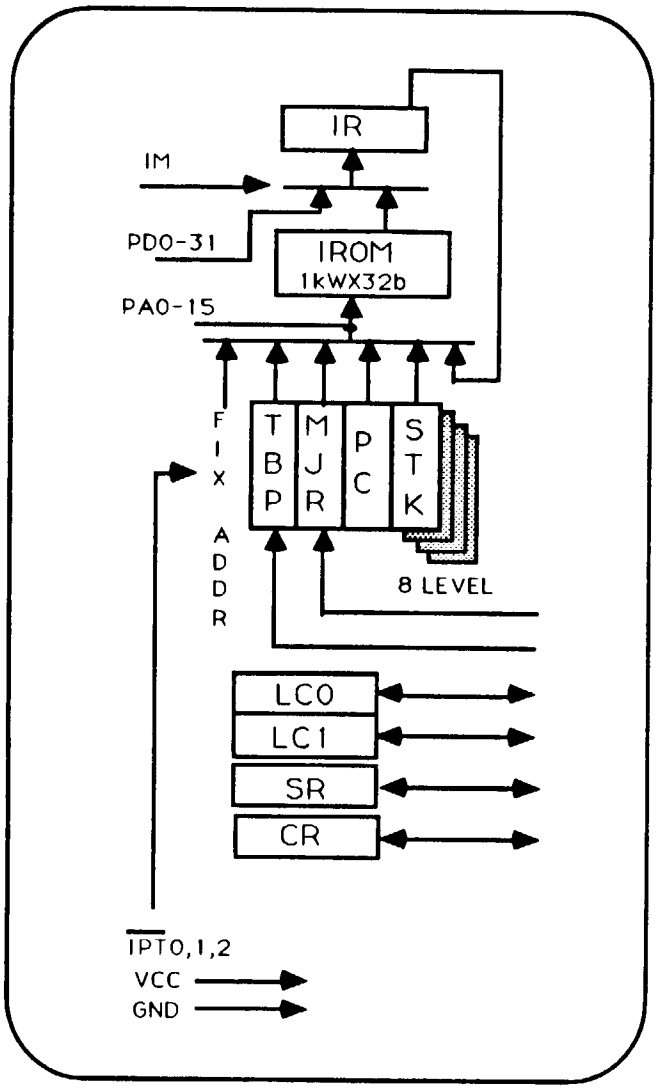


MASTER, SLAVE, AND DMA INTERFACE MODES AVAILABLE

MICRO-SEQUENCECONTROL BLOCK

ZERO-OVERHEAD LOOPING

NO PENALTY FOR USING EXTERNAL PROGRAM MEMORY



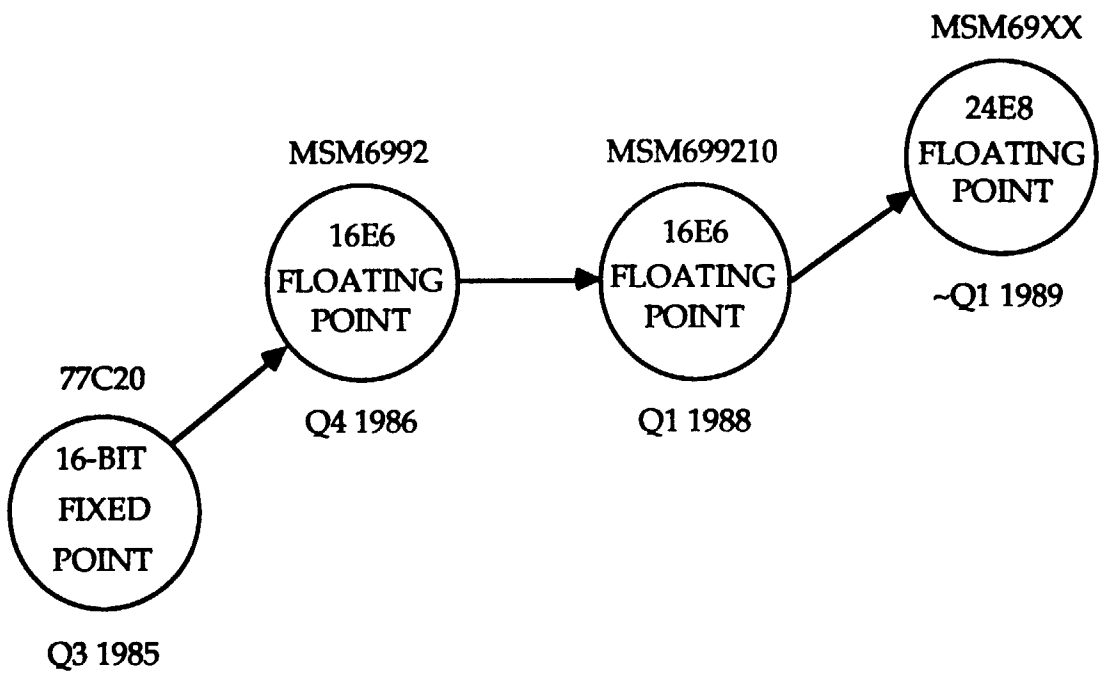
PERFORMANCE

MSM6992 BENCHMARKS	
1024 Complex pt. FFT (rad 2)	6.9 ms
FIR filter	100 ns/tap
2nd order IIR filter	
single	900 ns
pipelined	600 ns
4x4 by 1x4 Matrix Multiply	2100 ns

CODE FOR ALL BENCHMARKS AVAILABLE ON REQUEST

II. THE MSM699210 DSP

OKI'S DSP EVOLUTION



THE MSM699210 DSP

EXTERNAL VON NEUMANN ARCHITECTURE

1.5 MICRON FULL CUSTOM DESIGN

100 NS CYCLE TIME

TWICE AS MUCH INTERNAL RAM AND ROM

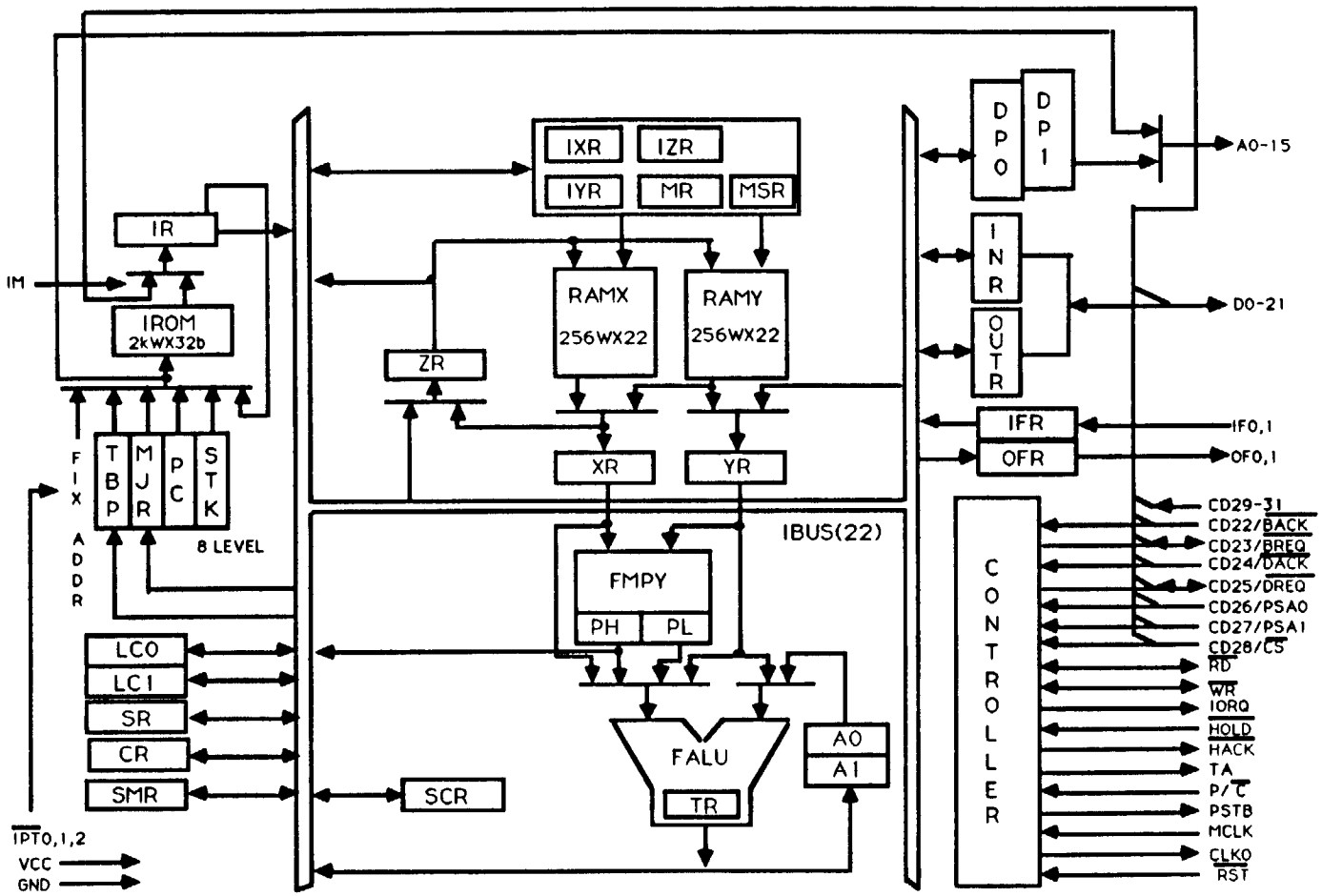
AVAILABLE IN 84-PIN PLCC OR 100-PIN FLAT PACK

PROGRAMMABLE WAIT ST ATES

100% CODE COMPATIBLE WITH MSM6992

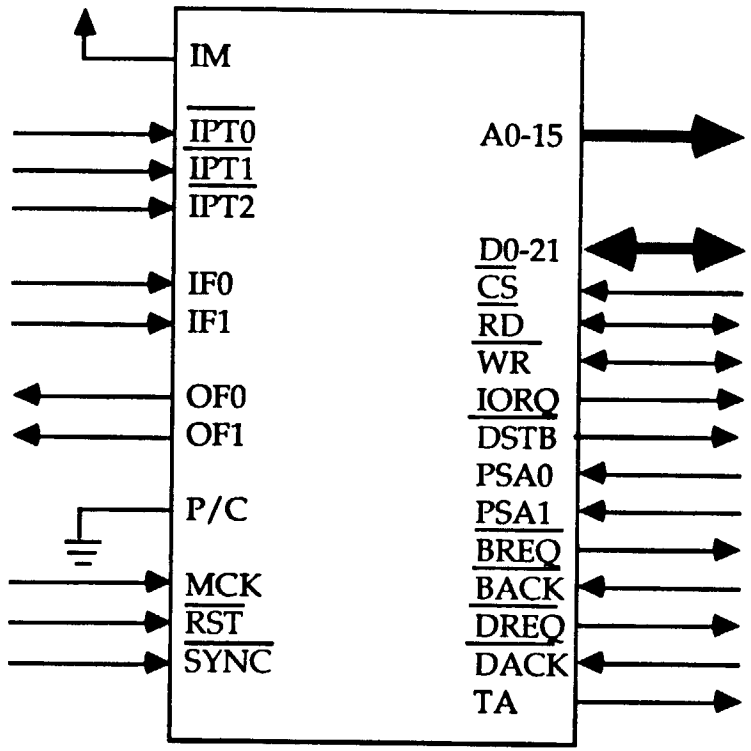
LOW COST

MSM699210 BLOCK DIAGRAM

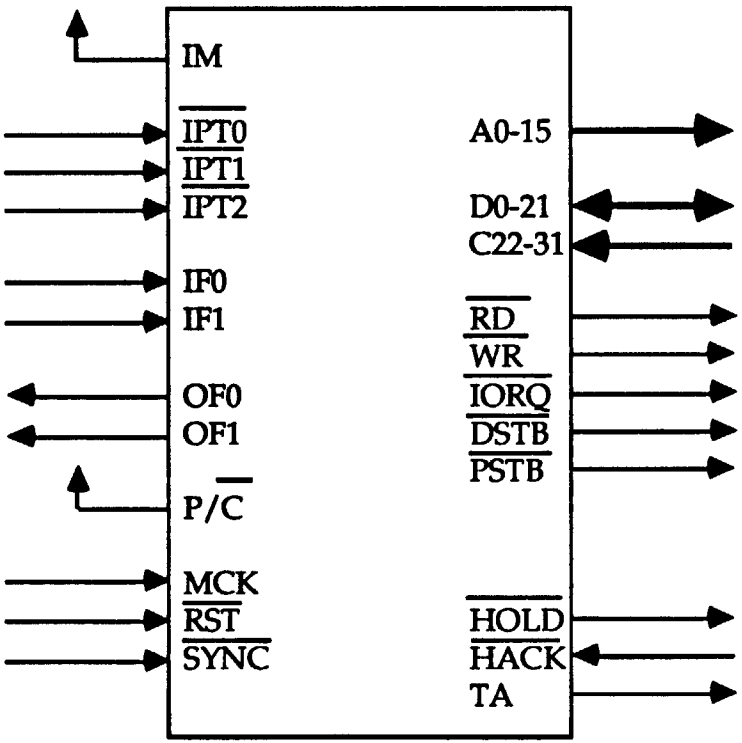


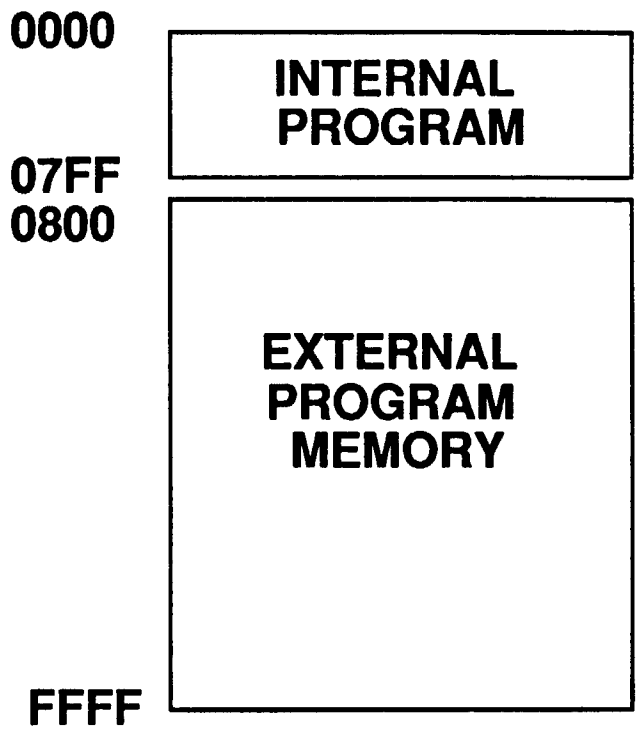
CONTROLLER MODE

USE OF ON-CHIP ROM ONLY



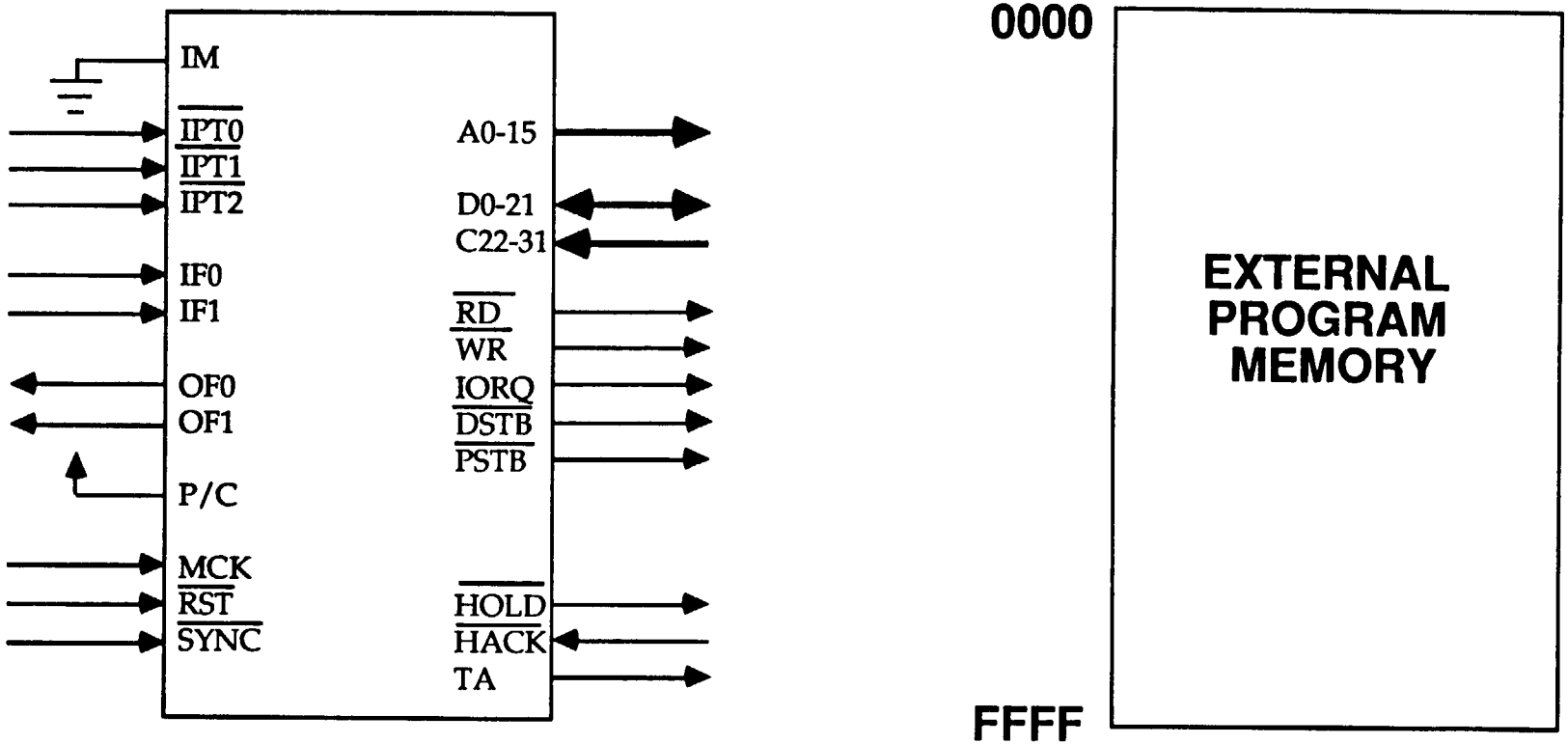
PROCESSOR MODE #1 INTERNAL AND EXTERNAL PROGRAM MEMORY





2 CYCLE EXECUTION WHEN EXTERNAL DATA OPERAND AND EXTERNAL INSTRUCTION FETCH OCCUR SIMULTANEOUSLY

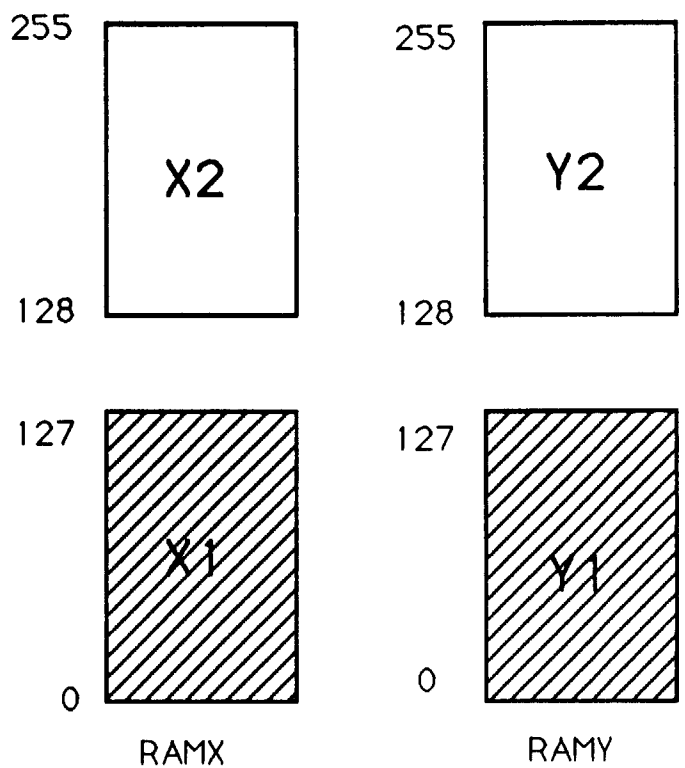
PROCESSOR MODE #2
EXTERNAL PROGRAM MEMORY ONLY



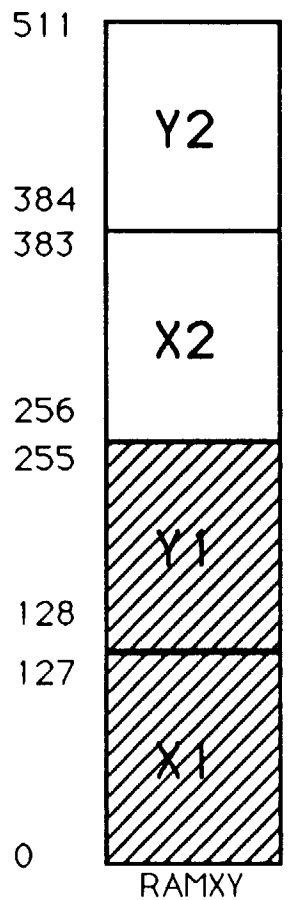
2 CYCLE EXECUTION WHEN EXTERNAL DATA OPERAND FETCH OCCURS

MSM6992 COMPATIBLE MEMORY MODE

I - TYPE

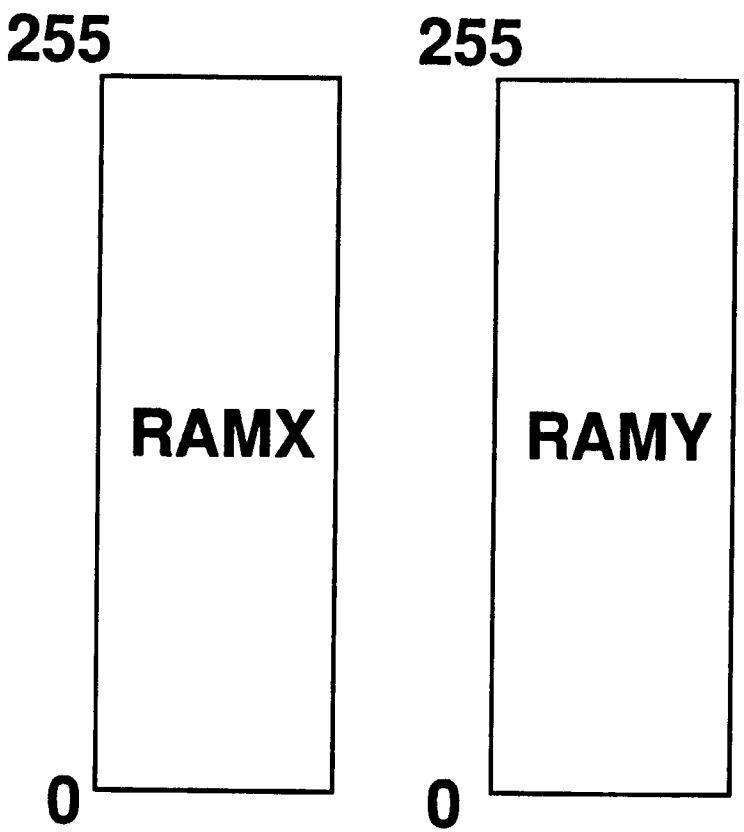


E - TYPE



PAGED MEMORY MODE

I - TYPE



E - TYPE

PR = 00
READ RAMX
WRITE RAMX

PR = 01
READ RAMY
WRITE RAMX

PR = 10
READ RAMX
WRITE RAMY

PR = 11
READ RAMY
WRITE RAMY

WAIT STATES

MODIFIES EXTERNAL DATA ACCESS TIME

ALL REGISTERS FROZEN DURING WAIT STATES

100 NS PER WAIT STATE

MAXIMUM ACCESS TIME :~700 NS

NEW INSTRUCTIONS: EXPANSION OF IRC FIELD OF THE CTL INSTRUCTION

The new instructions are:

(1) Set Power Down Mode (drastically reduces power consumption)

(2) Set Save mode (freezes multiplier, ALU, restricts accumulator access)

(3) Reset Save Mode (resets (2))

(4) Set Normalization Inhibit Mode (inhibits FALU post-normalization)

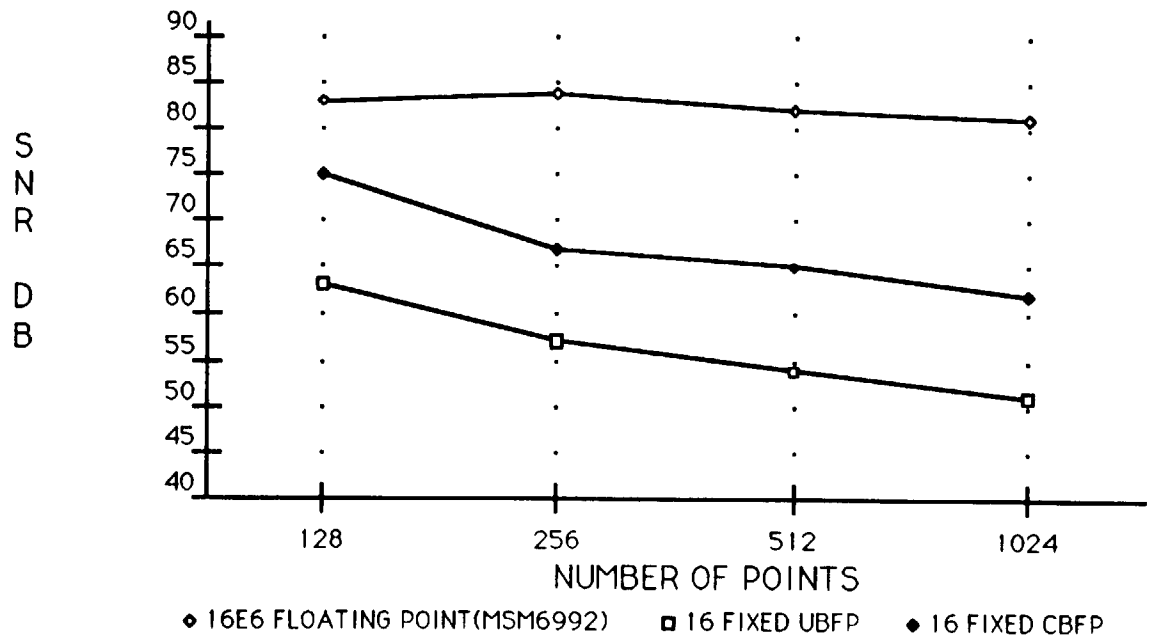
(5) Reset Normalization Inhibit Mode (resets (4))

(6) Set Recover Mode (same as Save, but XR, PH, and PL are not held)

(7) Reset Recover Mode (resets 6)

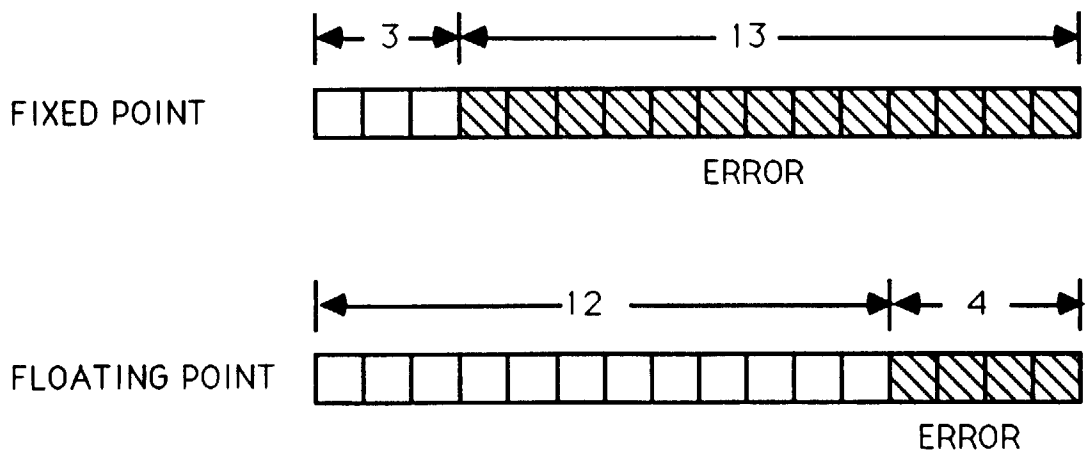
III. COMPARITIVE ANALYSIS OF 22-BIT FLOATING POINT VERSUS 16-BIT FIXED POINT

FFT PERFORMANCE COMPARISON

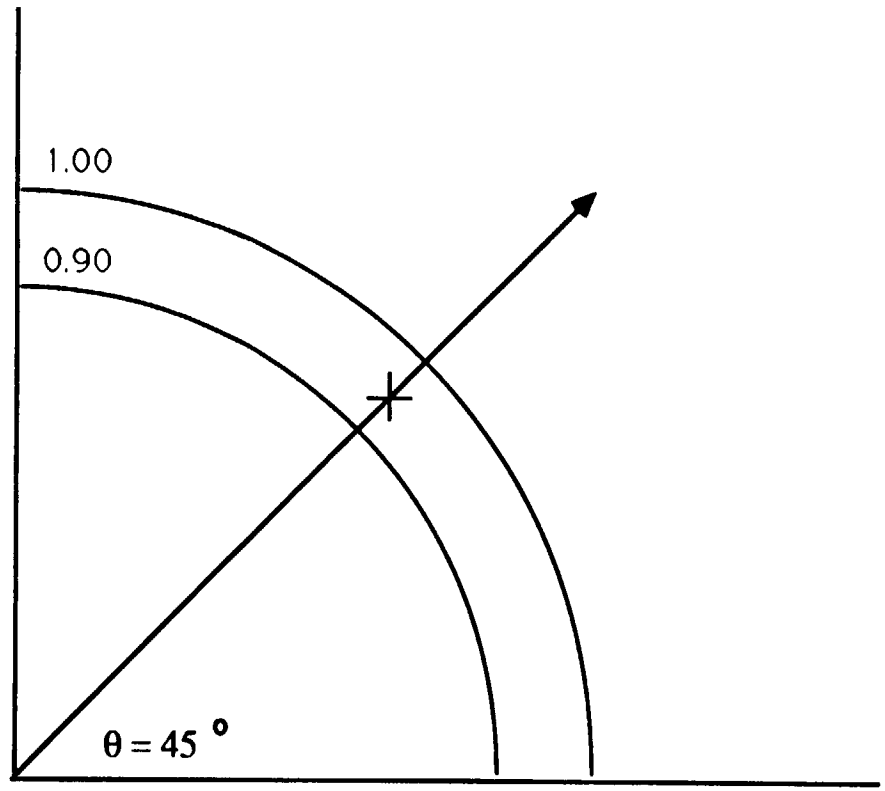


TWO-DIMENSIONAL FFT 512X512 EXPERIMENTAL RESULTS

(16-BIT MANTISSA)

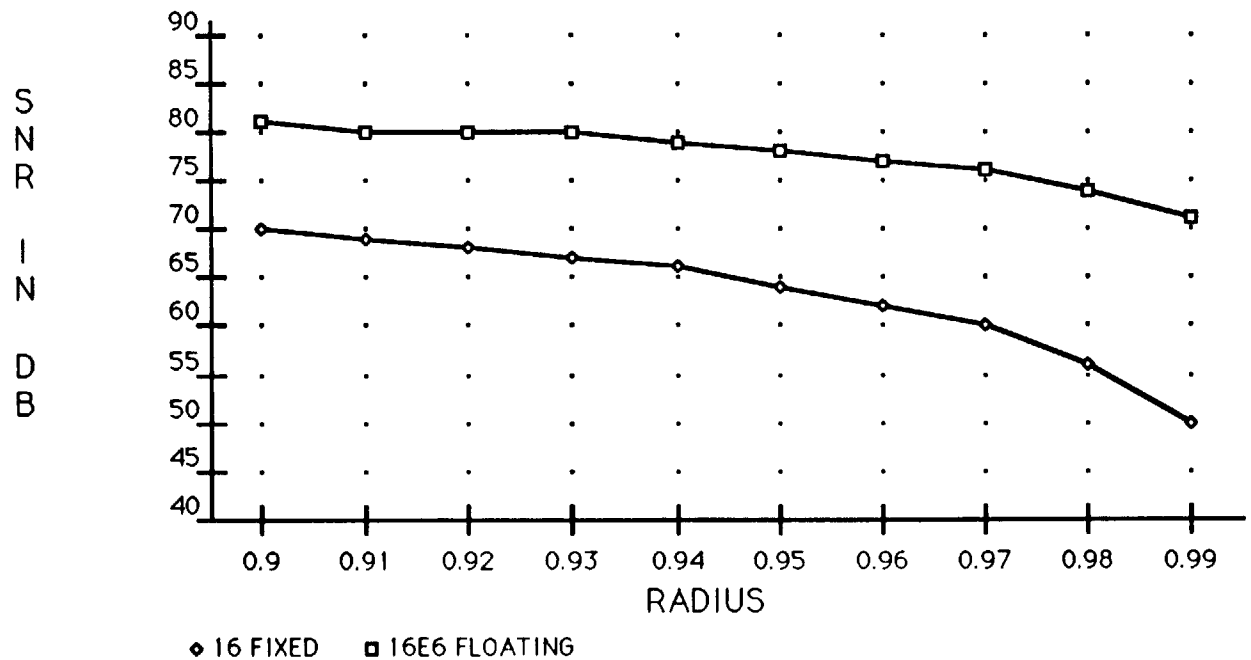


QUANTIZATION NOISE IN IIR FILTERS



UNIT CIRCLE

SNR PERFORMANCE



IV. PROGRAMMING THE MSM6992/210

INSTRUCTION SET

IFLA	Internal Floating Point Arithmetic
IFXA	Internal Fixed Point Arithmetic
ILO	Internal Logical
EFLA	External Floating Point Arithmetic
EFXA	External Fixed Point Arithmetic
ELO	External Logical
MOVE	Register to Register Moves
SFT	Shifting Operations
CTL	Control Instructions
LIT	Literal Instructions
JMP	Transfer of Control Instructions

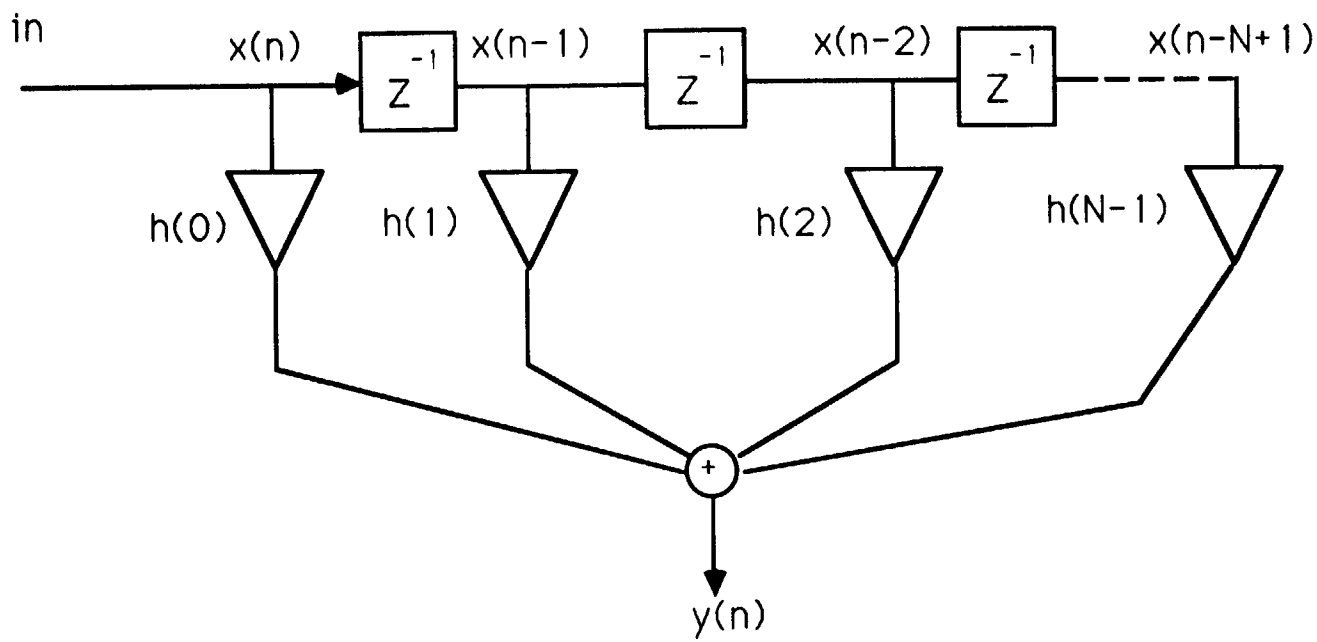
These microinstructions have several fields apiece in order to support several simultaneous operations. Mnemonics are provided for each operation field.

An example of a single 100 ns IFLA instruction follows:

```
LXY    ++1,++1:load XR and YR
FADD PH,A0    :add product register to a0
MOV  ZR,ALU   :move ALU result to ZR reg.
STM  ++1      :store ZR into data memory
LOOP ;        :return to top of code loop
```

(Note that a floating point multiply is executed as well)

FIR FILTER



```
SET     IXR,0H
SET     IYR,80H
SET     IZR,0H      ;
SET     MR,0F3H     ;
LXY     0,0         ;
LXY     ++1,++1
STM     0           ;
LXY     ++1,++1
FLD     PH
STM     ++1         ;
LXY     ++1,++1
FLD     PH
STA     A0
STM     ++1         ;
FADD    PH,A0
STA     A1          ;
FADD    PH,A1
STA     A0          ;
MOV     YR,ALU      ;
FADD    YR,A0       ;
```

	RAMX		RAMY
3H	A	83H	E
2H	B	82H	F
1H	C	81H	G
0H	D	80H	H

IXR	0H	IZR	0H
IYR	80H	MR	?
XR	?	YR	?
A0	?	ZR	?
A1	?		

```
SET     IXR,0H
SET     IYR,80H
SET     IZR,0H      ;
SET     MR,0F3H     ;
LXY     0,0         ;
LXY     ++1,++1
STM     0           ;
LXY     ++1,++1
FLD     PH
STM     ++1         ;
LXY     ++1,++1
FLD     PH
STA     A0
STM     ++1         ;
FADD    PH,A0
STA     A1          ;
FADD    PH,A1
STA     A0          ;
MOV     YR,ALU      ;
FADD    YR,A0       ;
```

	RAMX		RAMY
3H	A	83H	E
2H	B	82H	F
1H	C	81H	G
0H	D	80H	H

IXR	0H	IZR	0H
IYR	80H	MR	0F3H
XR	?	YR	?
A0	?	ZR	?
A1	?		

```
SET     IXR,0H
SET     IYR,80H
SET     IZR,0H      ;
SET     MR,0F3H     ;
LXY     0,0         ;
LXY     ++1,++1
STM     0           ;
LXY     ++1,++1
FLD     PH
STM     ++1         ;
LXY     ++1,++1
FLD     PH
STA     A0
STM     ++1         ;
FADD    PH,A0
STA     A1          ;
FADD    PH,A1
STA     A0          ;
MOV     YR,ALU      ;
FADD    YR,A0       ;
```

RAMX		RAMY	
3H	A	83H	E
2H	B	82H	F
1H	C	81H	G
0H	D	80H	H

IXR	0H	IZR	0H
IYR	80H	MR	0F3H
XR	D	YR	H
A0	?	ZR	D
A1	?		

```
SET     IXR,0H
SET     IYR,80H
SET     IZR,0H      ;
SET     MR,0F3H     ;
LXY     0,0         ;
LXY     ++1,++1
STM     0           ;
LXY     ++1,++1
FLD     PH
STM     ++1         ;
LXY     ++1,++1
FLD     PH
STA     A0
STM     ++1         ;
FADD    PH,A0
STA     A1          ;
FADD    PH,A1
STA     A0          ;
MOV     YR,ALU      ;
FADD    YR,A0       ;
```

	RAMX		RAMY
3H	A	83H	E
2H	B	82H	F
1H	C	81H	G
0H	C	80H	H

IXR	1H	IZR	0H
IYR	81H	MR	0F3H
XR	C	YR	G
A0	?	ZR	C
A1	?		

```
SET     IXR,0H
SET     IYR,80H
SET     IZR,0H      ;
SET     MR,0F3H     ;
LXY     0,0         ;
LXY     ++1,++1
STM     0           ;
LXY     ++1,++1
FLD     PH
STM     ++1         ;
LXY     ++1,++1
FLD     PH
STA     A0
STM     ++1         ;
FADD    PH,A0
STA     A1          ;
FADD    PH,A1
STA     A0          ;
MOV     YR,ALU      ;
FADD    YR,A0       ;
```

	RAMX		RAMY
3H	A	83H	E
2H	B	82H	F
1H	B	81H	G
0H	C	80H	H

IXR	2H	IZR	1H
IYR	82H	MR	0F3H
XR	B	YR	F
A0	?	ZR	B
A1	?		

```
SET     IXR,0H
SET     IYR,80H
SET     IZR,0H      ;
SET     MR,0F3H     ;
LXY     0,0         ;
LXY     ++1,++1
STM     0           ;
LXY     ++1,++1
FLD     PH
STM     ++1         ;
LXY     ++1,++1
FLD     PH
STA     A0
STM     ++1         ;
FADD    PH,A0
STA     A1          ;
FADD    PH,A1
STA     A0          ;
MOV     YR,ALU      ;
FADD    YR,A0       ;
```

	RAMX		RAMY
3H	A	83H	E
2H	A	82H	F
1H	B	81H	G
0H	C	80H	H

IXR	3H	IZR	2H
IYR	83H	MR	0F3H
XR	A	YR	E
A0	D*H	ZR	A
A1	?		

```
SET     IXR,0H
SET     IYR,80H
SET     IZR,0H      ;
SET     MR,0F3H     ;
LXY     0,0         ;
LXY     ++1,++1
STM     0           ;
LXY     ++1,++1
FLD     PH
STM     ++1         ;
LXY     ++1,++1
FLD     PH
STA     A0
STM     ++1         ;
FADD    PH,A0
→ STA   A1          ;
FADD    PH,A1
STA     A0          ;
MOV     YR,ALU      ;
FADD    YR,A0       ;
```

RAMX		RAMY	
3H	A	83H	E
2H	A	82H	F
1H	B	81H	G
0H	C	80H	H

IXR	3H	IZR	2H
IYR	83H	MR	0F3H
XR	?	YR	?
A0	D*H	ZR	?
A1	C*G		

```
SET     IXR,0H
SET     IYR,80H
SET     IZR,0H      ;
SET     MR,0F3H     ;
LXY     0,0         ;
LXY     ++1,++1
STM     0           ;
LXY     ++1,++1
FLD     PH
STM     ++1         ;
LXY     ++1,++1
FLD     PH
STA     A0
STM     ++1         ;
FADD    PH,A0
STA     A1          ;
FADD    PH,A1
STA     A0          ;
MOV     YR,ALU      ;
FADD    YR,A0       ;
```

	RAMX		RAMY
3H	A	83H	E
2H	A	82H	F
1H	B	81H	G
0H	C	80H	H

IXR	3H	IZR	2H
IYR	83H	MR	0F3H
XR	?	YR	?
A0	D*H + B*F	ZR	?
A1	C*G		

```
SET     IXR,0H
SET     IYR,80H
SET     IZR,0H      ;
SET     MR,0F3H     ;
LXY     0,0         ;
LXY     ++1,++1
STM     0           ;
LXY     ++1,++1
FLD     PH
STM     ++1         ;
LXY     ++1,++1
FLD     PH
STA     A0
STM     ++1         ;
FADD    PH,A0
STA     A1          ;
FADD    PH,A1
STA     A0          ;
MOV     YR,ALU      ;
FADD    YR,A0       ;
```

	RAMX		RAMY
3H	A	83H	E
2H	A	82H	F
1H	B	81H	G
0H	C	80H	H

IXR	3H	IZR	2H
IYR	83H	MR	0F3H
XR	?	YR	C*G + A*E
A0	D*H + B*F	ZR	?
A1	C*G		

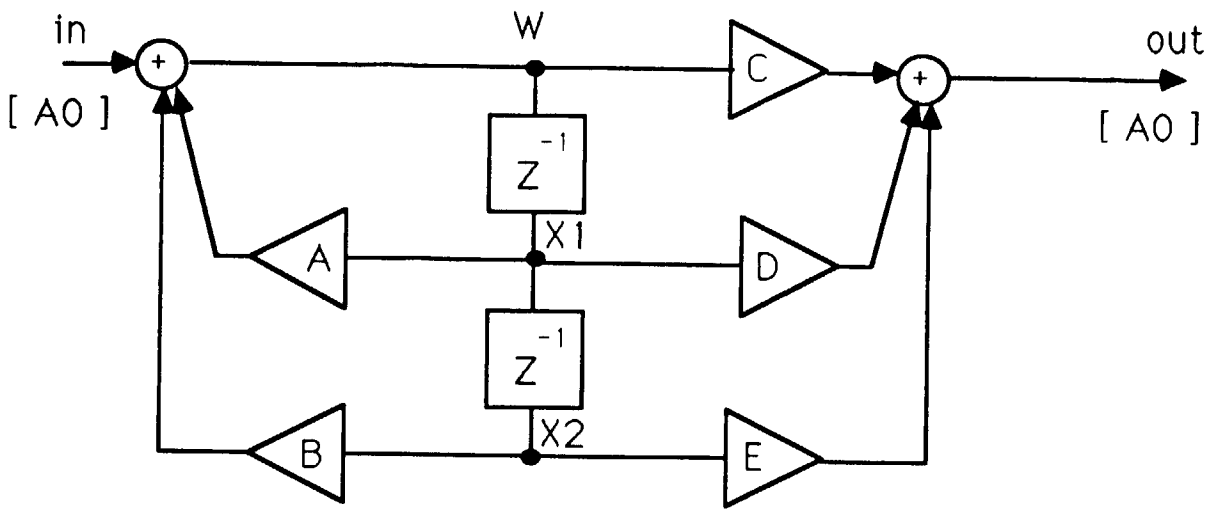
```
SET     IXR,0H
SET     IYR,80H
SET     IZR,0H      ;
SET     MR,0F3H     ;
LXY     0,0         ;
LXY     ++1,++1
STM     0           ;
LXY     ++1,++1
FLD     PH
STM     ++1         ;
LXY     ++1,++1
FLD     PH
STA     A0
STM     ++1         ;
FADD    PH,A0
STA     A1          ;
FADD    PH,A1
STA     A0          ;
MOV     YR,ALU      ;
FADD    YR,A0       ;
```

	RAMX		RAMY
3H	A	83H	E
2H	A	82H	F
1H	B	81H	G
0H	C	80H	H

ALU=A*E+B*F+C*G+D*H END OF NEXT CYCLE

IXR	3H	IZR	2H
IYR	83H	MR	0F3H
XR	?	YR	?
A0	D*H + B*F	ZR	?
A1	C*G		

DIRECT FORM II BIQUAD



W = A*X1 + B*X2 + in
out = C*W + D*X1 + E*X2
XI = W
X2=X1

DIRECT FORM II BIQUAD ASSEMBLY CODE

```
:
IIR     MACRO    X1,X2,A,B,C,D,E
        LXY     X1,A     ;              :  A*X1
        LXY     X1,D     ;              :  D*X1
        LXY     X2,B                    :  B*X2
        FADD    PH,A0   ;               :  W = A*X1 + in
        LXY     X2,E                    :  E*X2
        FLD     PH                      :  out = D*X1
        STA     A0      ;
        LXY     X1,A                    :
        FADD    PH,A0                   :  W = A*X1 + B*X2 + in
        STM     X2                      :  X2 = X1
        STA     A1      ;               :
        STA     A0                      :
        MOV     YR,ALU                  :
        LX      C                       :  C*W
        FADD    PH,A1   ;               :  out = D*X1 +  E*X2
        STA     A1                      :
        FLD     A0      ;               :
        FADD    PH,A1                   :  out = C*W + D*X1 + E*X2
        MOV     ZR,ALU                  :
        STM     X1      ;               :  X1 = W
        STA     A0      ;               :  A0 = out
```

MATRIX MULTIPICATION

$$
\begin{vmatrix} a(1,1) & a(1,2) & \cdots & a(1,N) \\ a(2,1) & a(2,2) & \cdots & a(2,N) \\ \vdots & \vdots & & \\ a(N,1) & \cdot & \cdot & a(N,N) \end{vmatrix}
\times
\begin{vmatrix} b(1) \\ b(2) \\ \cdot \\ \cdot \\ b(N) \end{vmatrix}
=
\begin{vmatrix} c(1) \\ c(2) \\ \cdot \\ \cdot \\ c(N) \end{vmatrix}
$$

MatrixA MatrixB MatrixC

ALTERNATE ACCUMULATOR METHOD

ROW N X COLUMN → A0
ROW N X COLUMN → A1

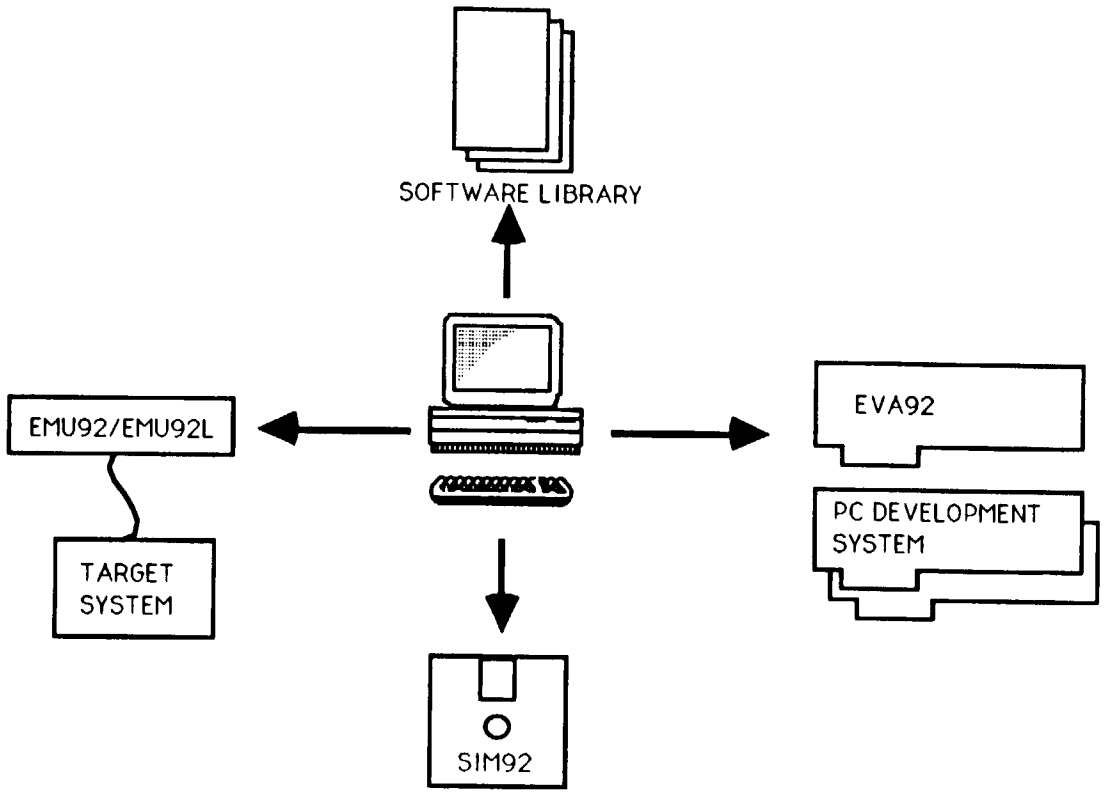
A0 + A1 = RESULT

ALTERNATE ROW METHOD

ROW N X COLUMN → A0
ROW (N+1) X COLUMN → A1

NO NEED TO ADD ACCUMULATORS

V. SUPPORT


SOFTWARE LIBRARY
EMU92/EMU92L
TARGET SYSTEM
EVA92
PC DEVELOPMENT SYSTEM
SIM92

EMU92

8K-WORDS OF EXTERNAL PROGRAM MEMORY

8K-WORDS OF EXTERNAL DATA MEMORY

(2) 4K-WORD I/O BUFFERS

ICE CAPABILITY

RUNS WITH DBG92 SOFTWARE PC-DOS OR MS-DOS (2.1 OR HIGHER)

SOFTWARE TRACE CAPABILITY

EMU92L

2K-WORDS OF EXTERNAL PROGRAM MEMORY

NO EXTERNAL DATA MEMORY OR I/O BUFFERS

ICE CAPABILITY

RUNS WITH DBG92L SOFTWARE PC-DOS OR MS-DOS (2.1 OR HIGHER)

SOFTWARE TRACE CAPABILITY

SINGLE BOARD

EVA92

MSM6992 EVALUATION BOARD

SINGLE PC/XT/AT CARD

16-K WORD DATA MEMORY (8-BIT)

16-K WORD PROGRAM MEMORY

SIM92

SIMULATOR FOR THE MSM6992 AND MSM699210

RUNS ON MS-DOS OR PC-DOS 2.1 AND UP

SAME COMMAND SET AS DBG92(L)

DUMP/RESTART FACILITY

SOURCE COMPARISON

BREAKPOINT ON DATA POINTER RANGE VIOLATION

ABLE TO SIMULATE INTERRUPTS

GENERATES CODE EFFICIENCY PARAMETERS

PC-BASED DEVELOPMENT SYSTEM

CONSISTS OF TWO PC-AT CARDS

12-BIT ADC & 16-BIT DAC

PROGRAMMABLE SAMPLE RATE

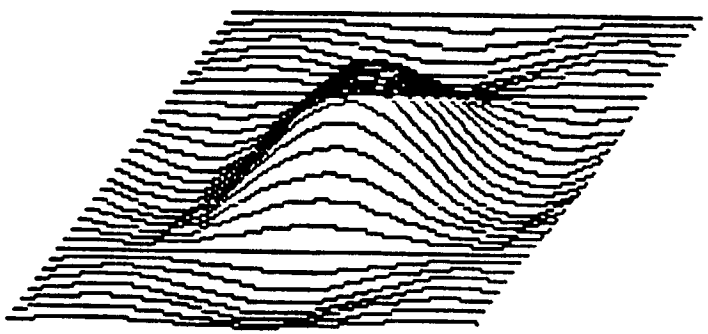
ADJUSTABLE ANTI-ALIASING FILTER

WAVEFORM CAPTURE AND ANALYSIS

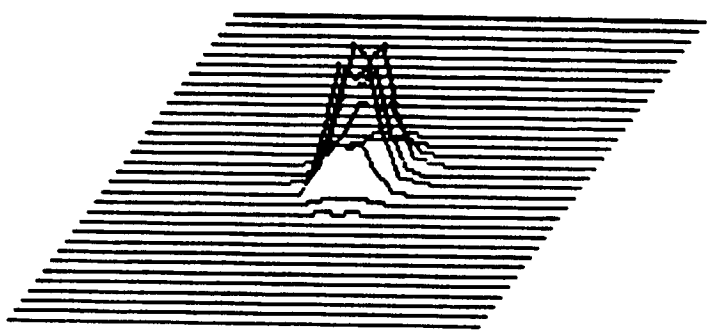
SPECTRAL ANALYSIS SOFTWARE

GRAPHICS CAPABILITY (NEEDS EGA BOARD)

SPATIAL DOMAIN



TRANSFORM DOMAIN



INTERMEDIATE LANGUAGE ASSEMBLER

HELPS EASE PROGRAMMING BY MAKING PIPELINE INVISIBLE TO PROGRAMMER

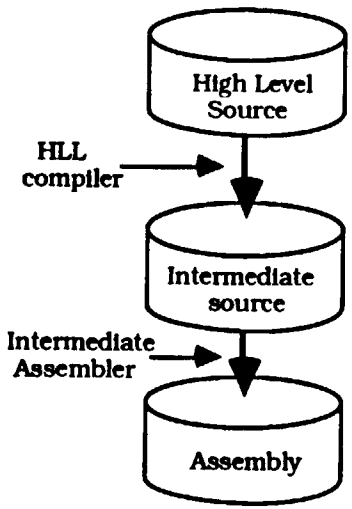
PRODUCES MORE EFFICIENT CODE THAN A HIGH-LEVEL LANGUAGE ASSEMBLER

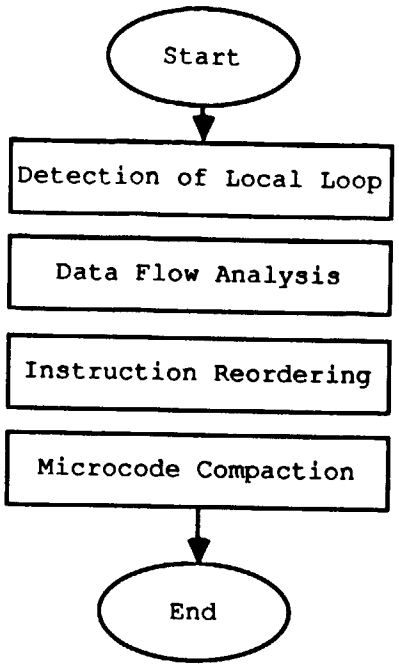
PERFROMS INSTRUCTION REORDERING IN ORDER TO PREVENT PIPELINE INTERLOCKS

EXAMPLE:

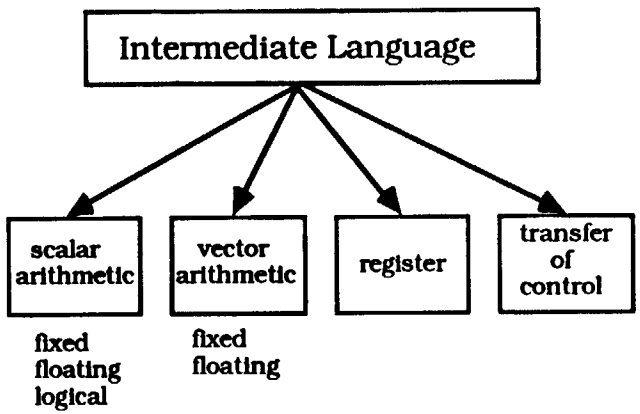
```
LXY A,B ;
NOP ;          →    MADD A,B,A0,A0 ;
FADD PH,A0;
STA A0 ;
```

PLACEMENT OF INTERMEDIATE LANGUAGE





INTERMEDIATE INSTRUCTIONS



: 128 TAP FIR FILTER

```
VSFT   COEFF,128;
VMAD   DATA,COEFF,OUT,128;
```

SOFTWARE LIBRARY

FIR FILTERING

IIR FILTERING

MATRIX ARITHMETIC

HAMMING WINDOWING

DIVISION

RADIX 2 FFT 8 --> 8K PTS

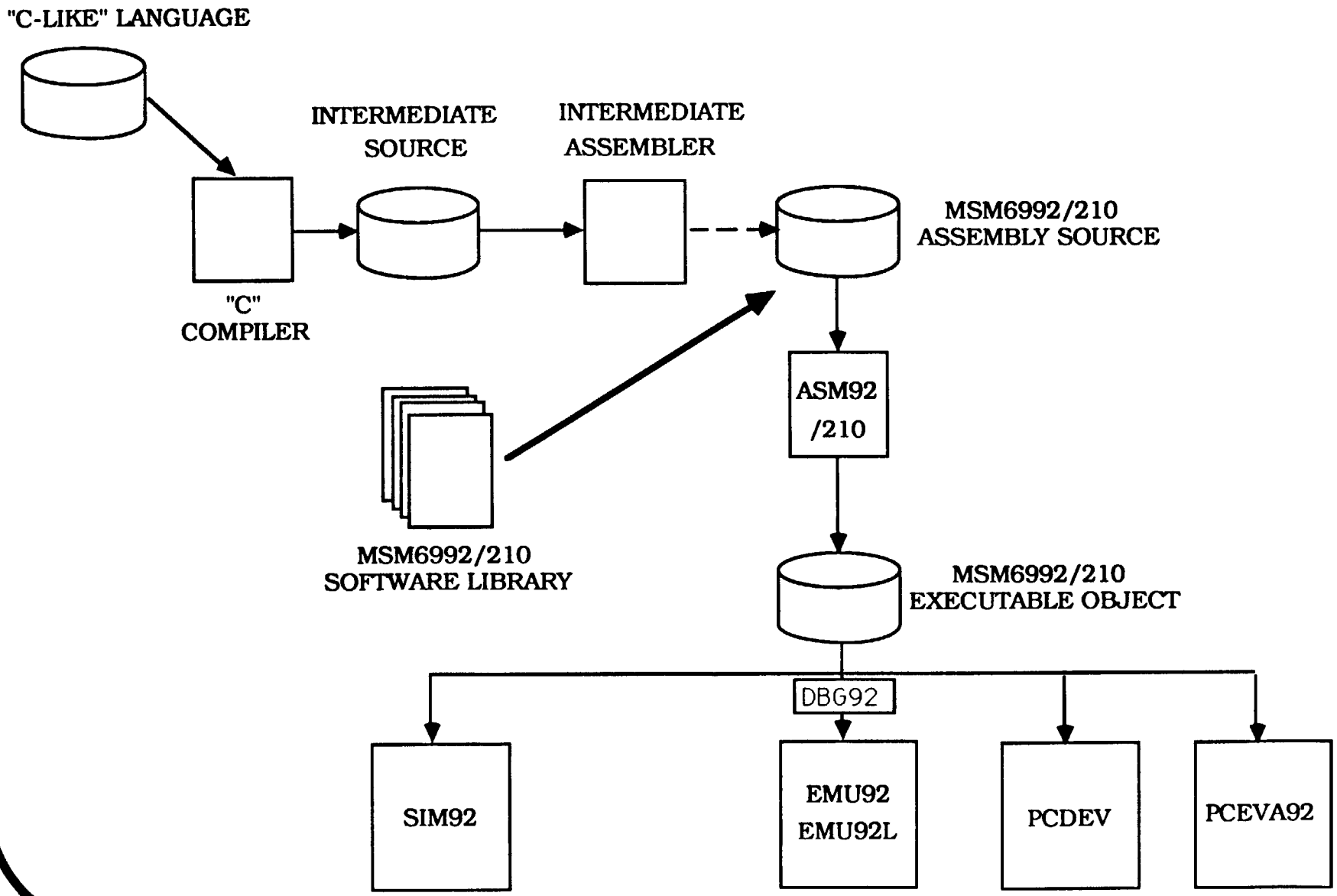
SQUARE ROOT

SIN, COS, AND TANGENT

ADAPTIVE FILTERING

AUTOCORRELATION

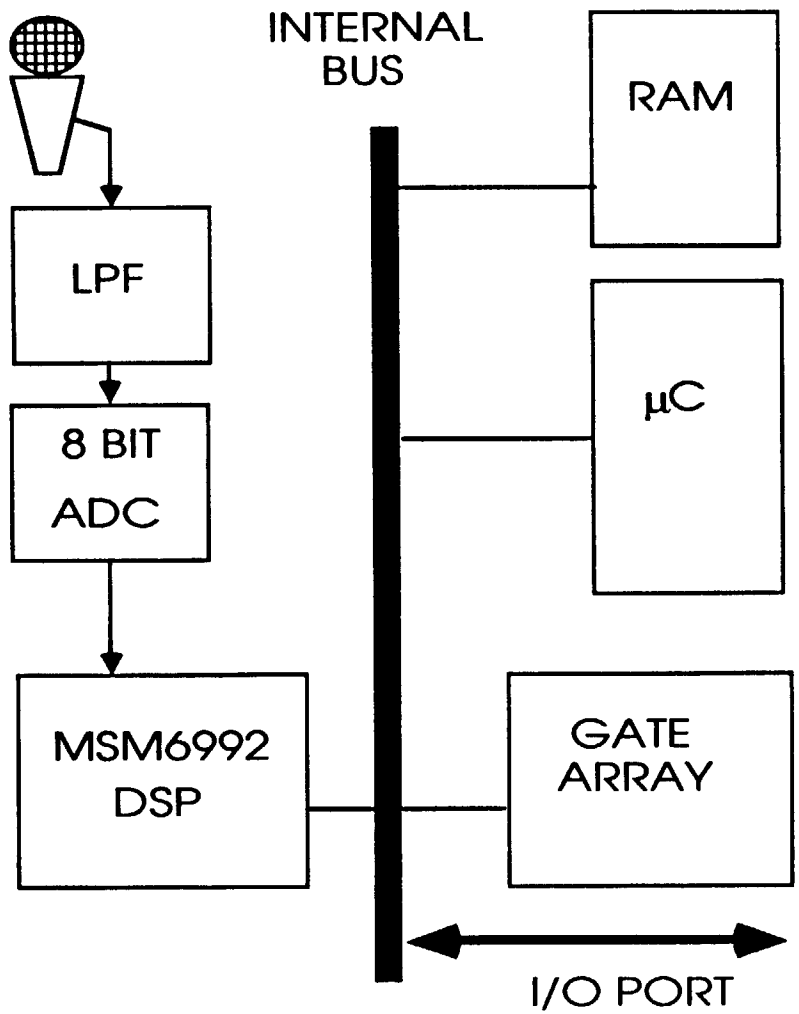
DEVELOPMENT SUPPORT CONFIGURATION



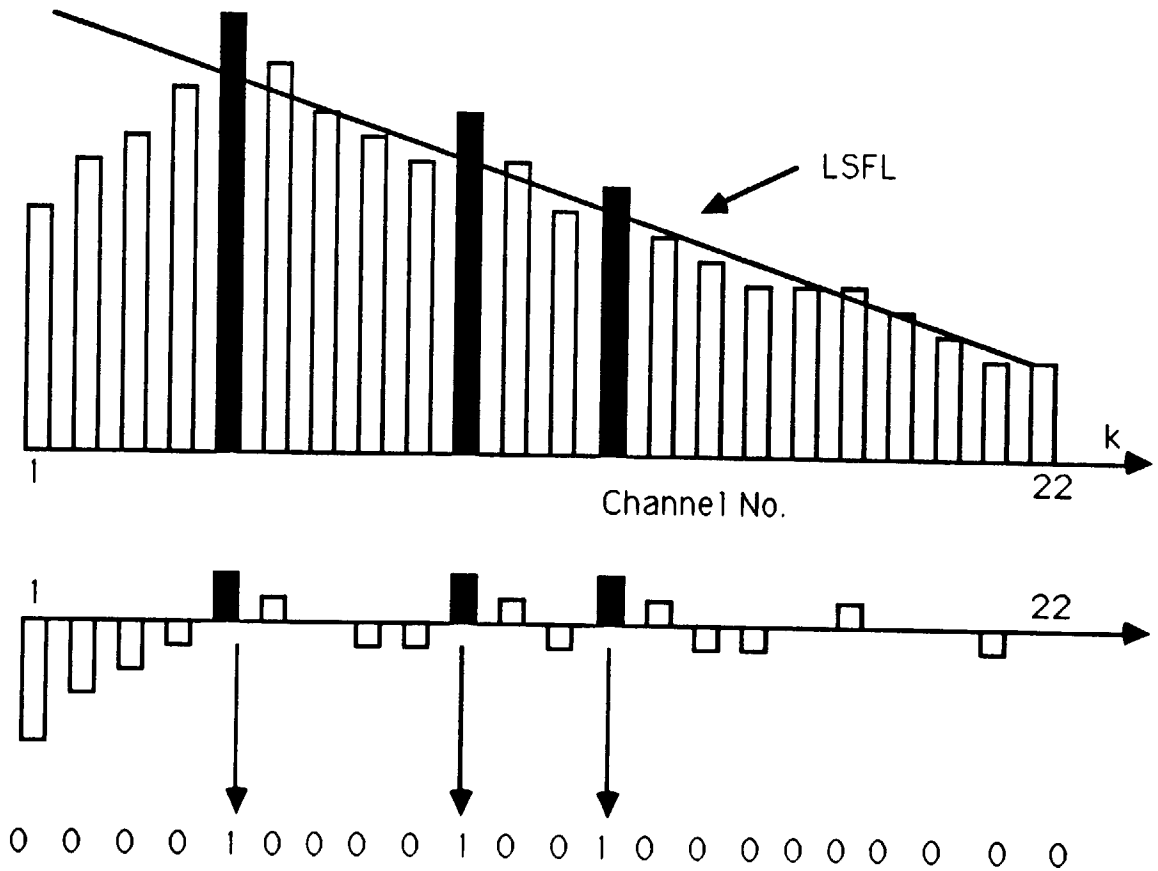
OKI 'S DSP FAMILY

VI. EXAMPLE APPLICATION: SPEECH RECOGNITION BOARD

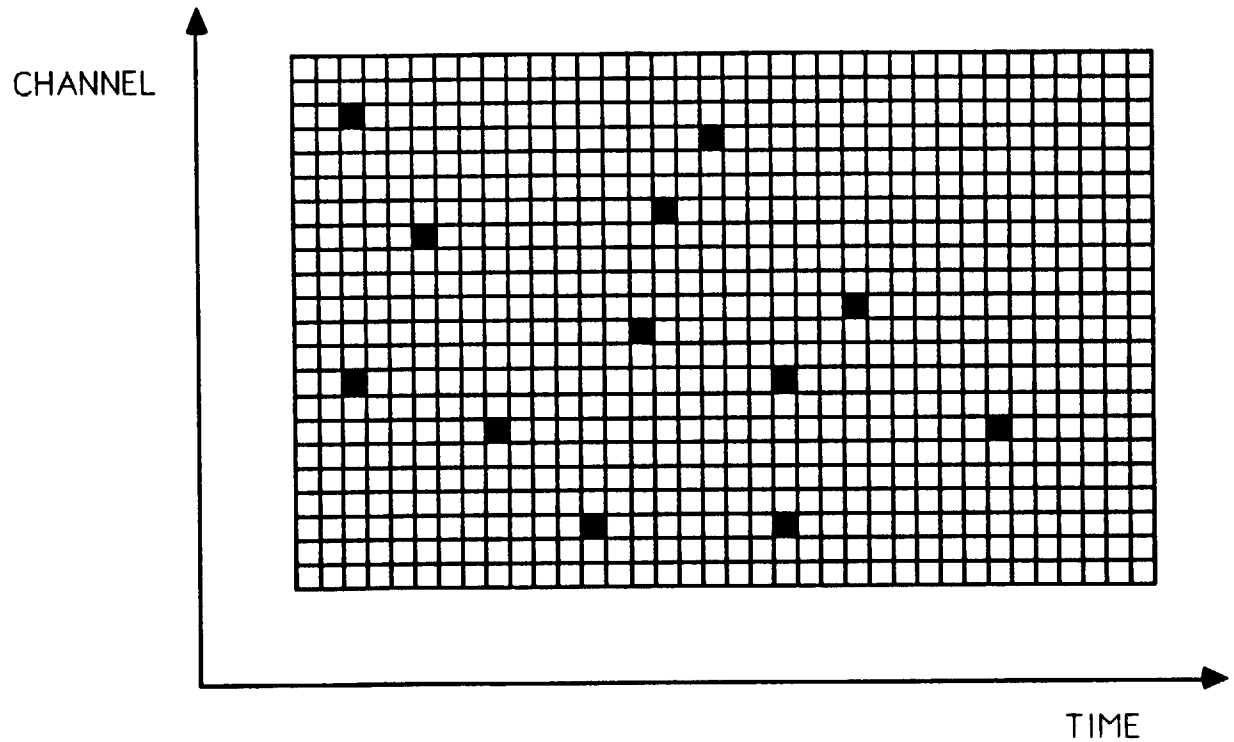
SPEECH RECOGNITION BOARD



EXTRACTION OF LOCAL PEAKS VECTOR



SPECTRAL PEAKS VERSUS TIME



PERFORMANCE

95% RECOGNITION RATE
IN A 10 dB SNR ENVIRONMENT

ZORAN

DSP ASSOCIATES

DIGITAL SIGNAL PROCESSING

SEMINAR

PRECEDING PAGE BLANK NOT FILMED

DSP ASSOCIATES SEMINAR OUTLINE

- Digital Filter Processors
  - Products
  - Architecture
  - Applications
- Vector Signal Processors
  - ZR34161
  - ZR34325
- Image Compression Processors
  - What is Image Compression?
  - ZR36010 ICP
  - Image Compression Examples

PRECEDING PAGE BLANK NOT FILMED

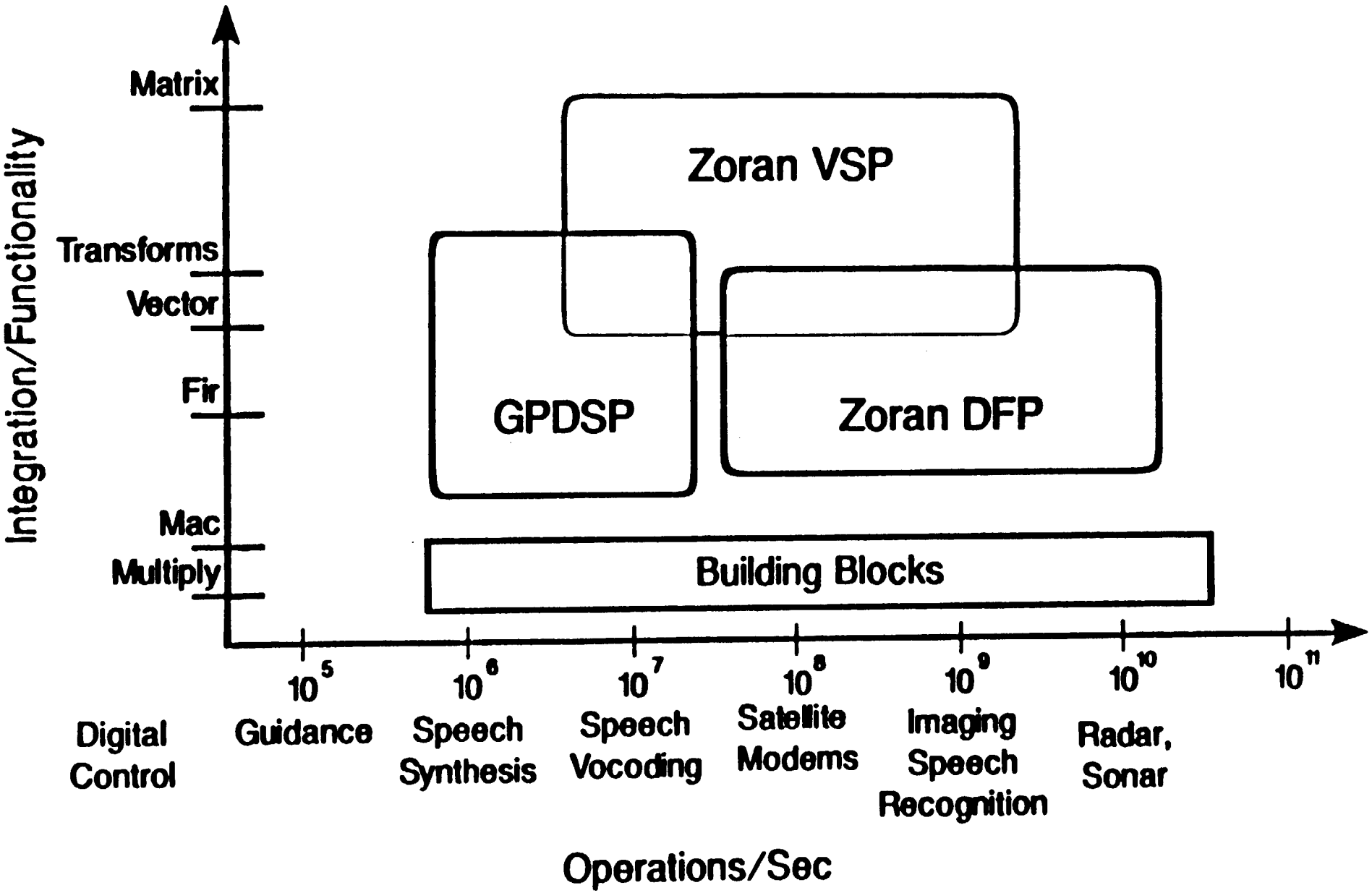
CHARTER

To design, manufacture and market proprietary, monolithic, systems processors and support systems for the high-performance digital signal processing (DSP) market.

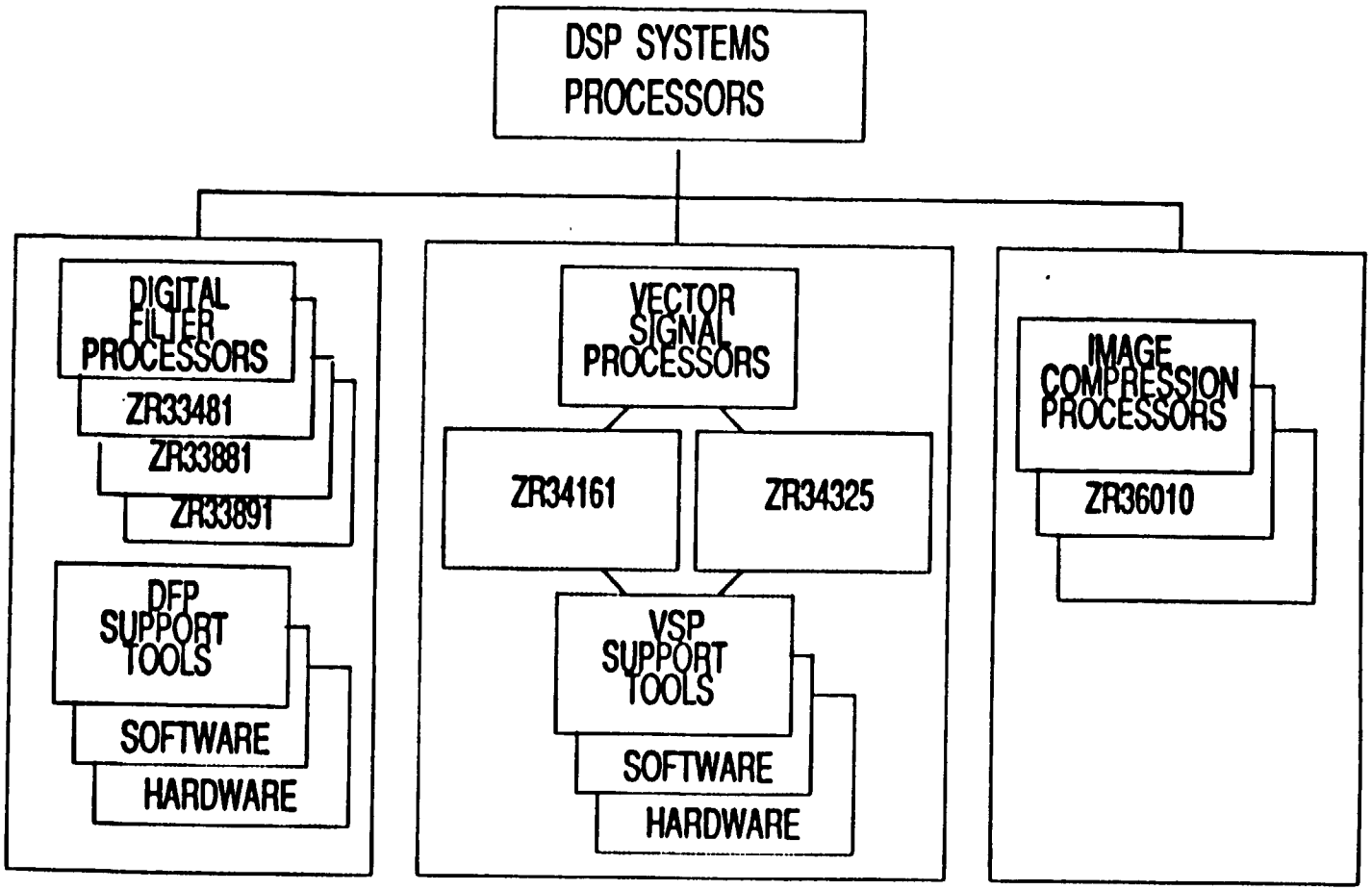
SYSTEM PROCESSORS APPROACH

- Application-driven, top-down design
- Optimized architecture (throughput, vector processing, block floating point, DMA ...)
- Embedded algorithms (FFT, FIR ...)
- High level instructions (FFT, Power Spectrum, Demodulation ...)
- System level tools

ZORAN


Integration/Functionality
Matrix
Transforms
Vector
Fir
Mac
Multiply
Zoran VSP
GPDSP
Zoran DFP
Building Blocks
10^5
10^6
10^7
10^8
10^9
10^{10}
10^{11}
Digital Control
Guidance
Speech Synthesis
Speech Vocoding
Satellite Modems
Imaging Speech Recognition
Radar, Sonar
Operations/Sec

PRODUCT PORTFOLIO--3Q/87



CURRENT PRODUCTS

- **Digital Filter Processors (DFP)**
- ZR34161 Vector Signal Processor (VSP)

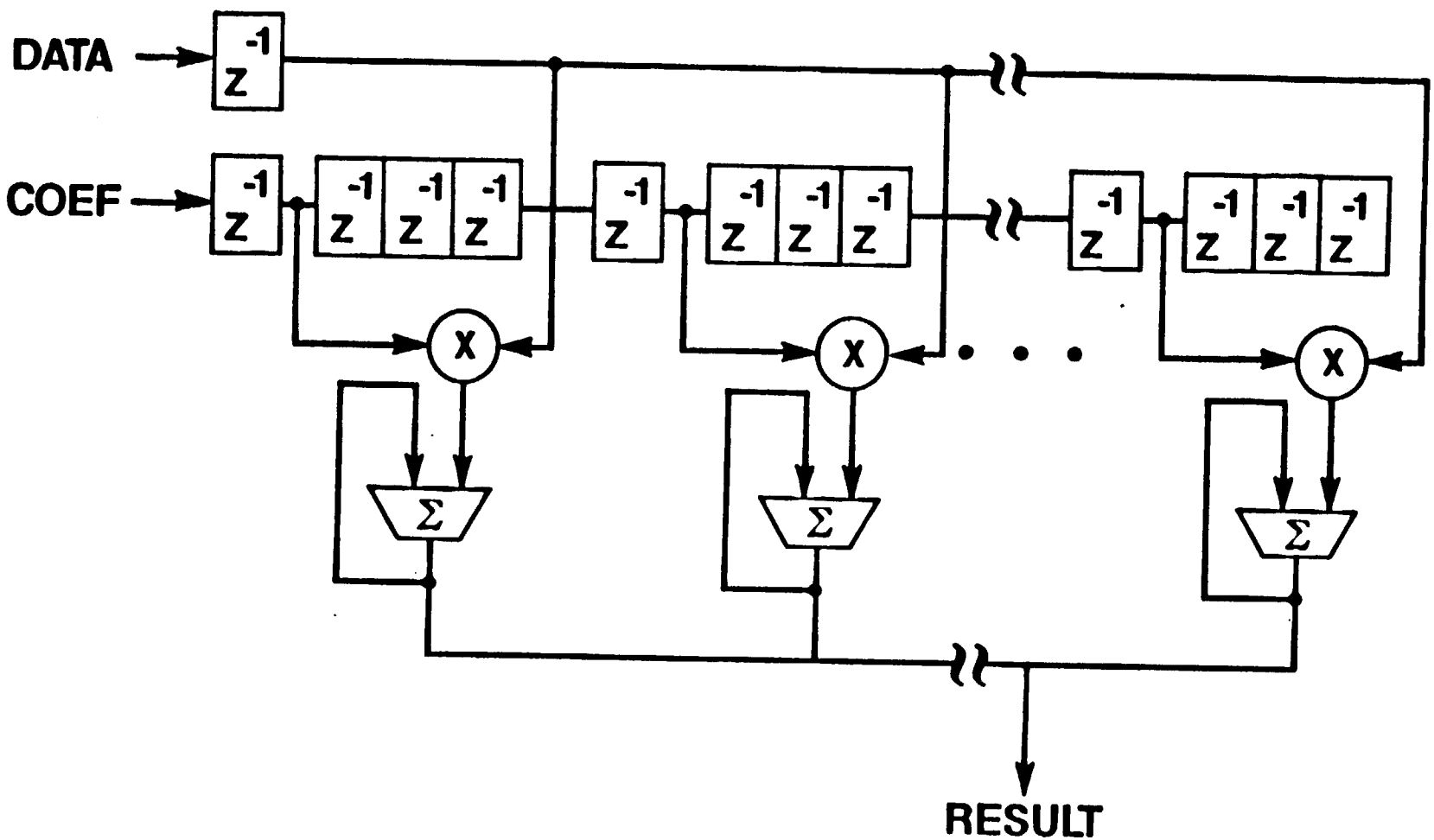
DFP FUNCTIONAL APPLICATIONS

- 1-D Filters
- 2-D FIR Filters
- Correlation/Convolution
- Adaptive Filters
- Image Processing
- Digital Video
- Radar/Sonar

DFP FAMILY FEATURES

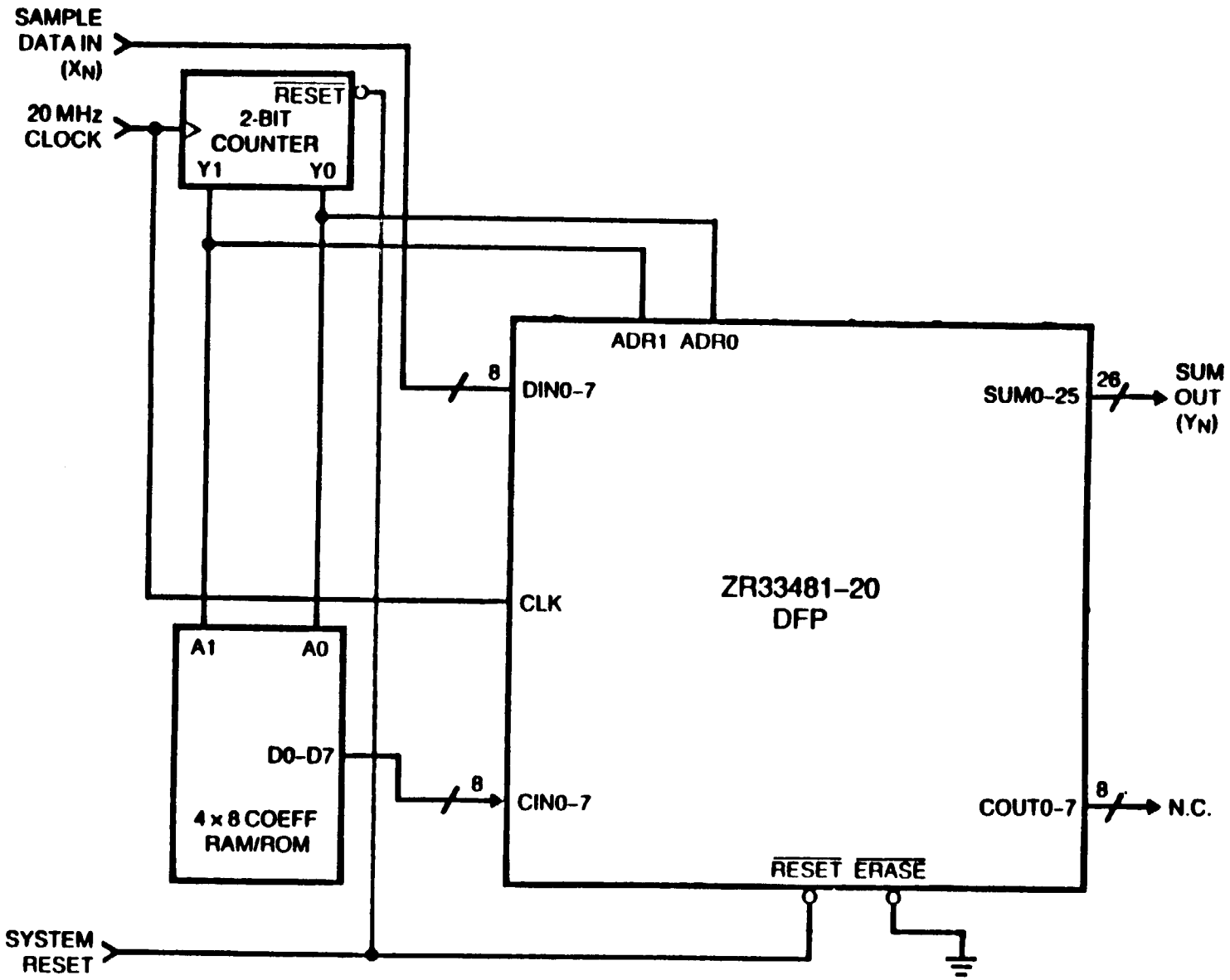
- Complete FIR filter system in single IC
- Paralles, linear array processor architecture
  4 or 8 filter cells per device
- 8 x 8 or 9 x 9 multiplier with 26-bit accumulation in each cell
- Each cell performs 30M mult and 30M add/sec
- Cascadable for longer filters
- Expandable coefficient and data word size

DIGITIAL FILTER PROCESSOR (DFP) FUNCTIONAL BLOCK DIAGRAM



ZR33481 4-TAP FIR SEQUENCE

CLK	CELL0	CELL1	CELL2	CELL3	SUM/CLR
0	$C_3 \bullet X_0$	0	0	0	—
1	$+C_2 \bullet X_1$	$C_3 \bullet X_1$	0	0	—
2	$+C_1 \bullet X_2$	$+C_2 \bullet X_2$	$C_3 \bullet X_2$	0	—
3	$+C_0 \bullet X_3$	$+C_1 \bullet X_3$	$+C_2 \bullet X_3$	$C_3 \bullet X_3$	CELL0 (Y3)
4	$C_3 \bullet X_4$	$+C_0 \bullet X_4$	$+C_1 \bullet X_4$	$+C_2 \bullet X_4$	CELL1 (Y4)
5	$+C_2 \bullet X_5$	$C_3 \bullet X_5$	$+C_0 \bullet X_5$	$+C_1 \bullet X_5$	CELL2 (Y5)
6	$+C_1 \bullet X_6$	$+C_2 \bullet X_6$	$C_3 \bullet X_6$	$+C_0 \bullet X_6$	CELL3 (Y6)
7	$+C_0 \bullet X_7$	$+C_1 \bullet X_7$	$+C_2 \bullet X_7$	$C_3 \bullet X_7$	CELL0 (Y7)


SAMPLE
DATA IN
(XN)
20 MHz
CLOCK
RESET
2-BIT
COUNTER
Y1
Y0
ADR1 ADR0
8
DIN0-7
SUM0-25
26
SUM
OUT
(YN)
CLK
ZR33481-20
DFP
A1
A0
D0-D7
4 x 8 COEFF
RAM/ROM
8
CIN0-7
COUT0-7
8
N.C.
RESET ERASE
SYSTEM
RESET

THE DFP FAMILY

Processor	Filter Cells	Word Length	Sample Rates	Packages
ZR33481-XX	4	8 bits	15 MHz 20 MHz	68 LCC, 68 PLCC
ZR33881-XX	8	8 bits	15, 20 MHz	68 LCC
ZR33891-XX	8	9 bits	15 MHz 20 MHz 25, 30 MHz	68 LCC, 84 PGA, 84 PLCC

DFP CUSTOMER BENEFITS
MODULARITY AND FLEXIBILITY

- Need higher filter order, more taps?
  Cascade DFPs.

- Need high sample rate?
  Use DFPs in parallel.

- Need more word length (16 x 16 bit)?
  Use multiple DFPs.

- Need decimation or interpolation?
  DFP contains efficient mechanisms.

- Need 2-D filters?
  DFP implements efficiently.

DFP BASICS

- **Decimation & Interpolation**
- Extended Filter Lengths
- Extended Sample Rates

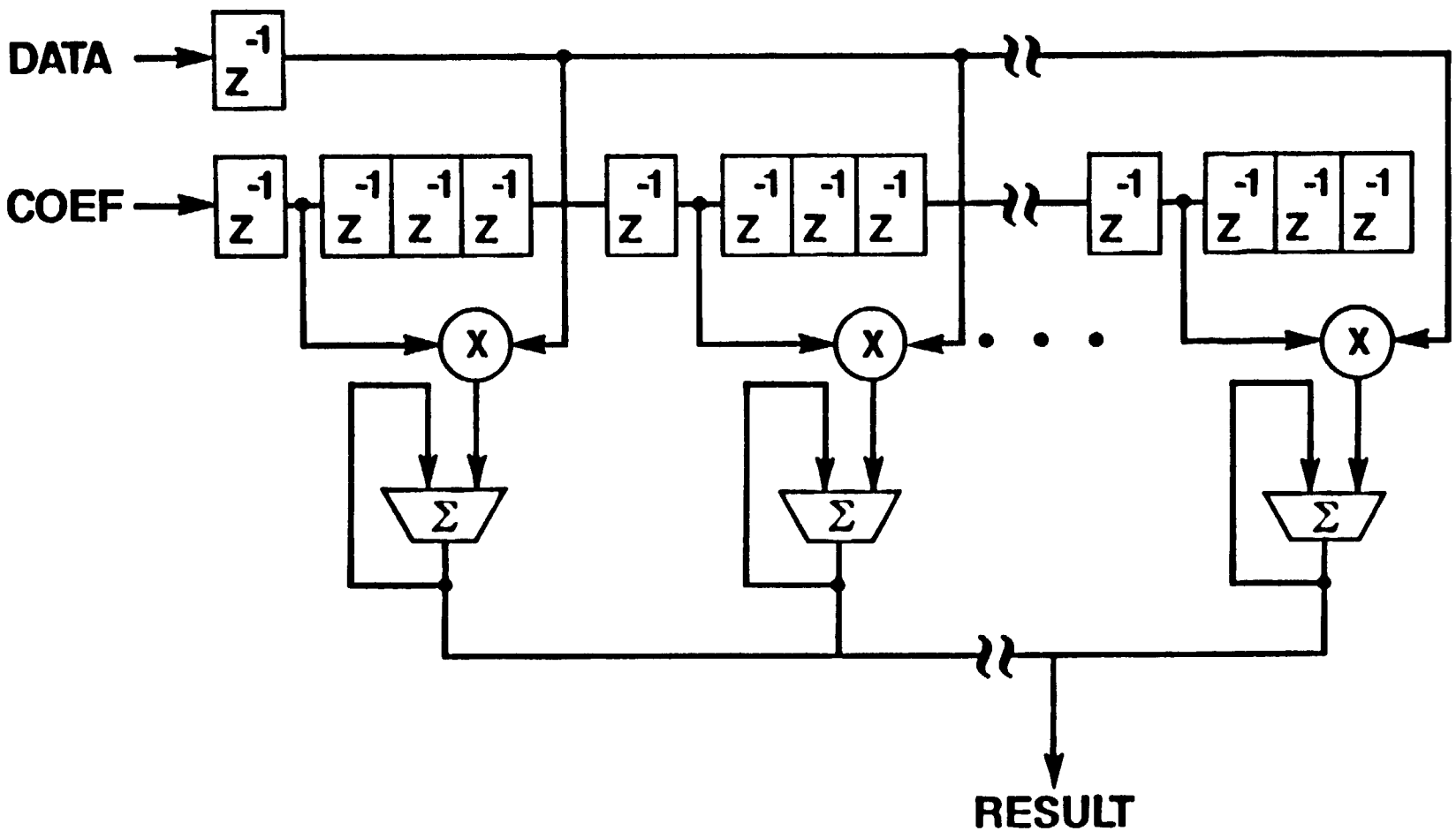
DECIMATION & INTERPOLATION APPLICATIONS

SAMPLE RATE CONVERSION

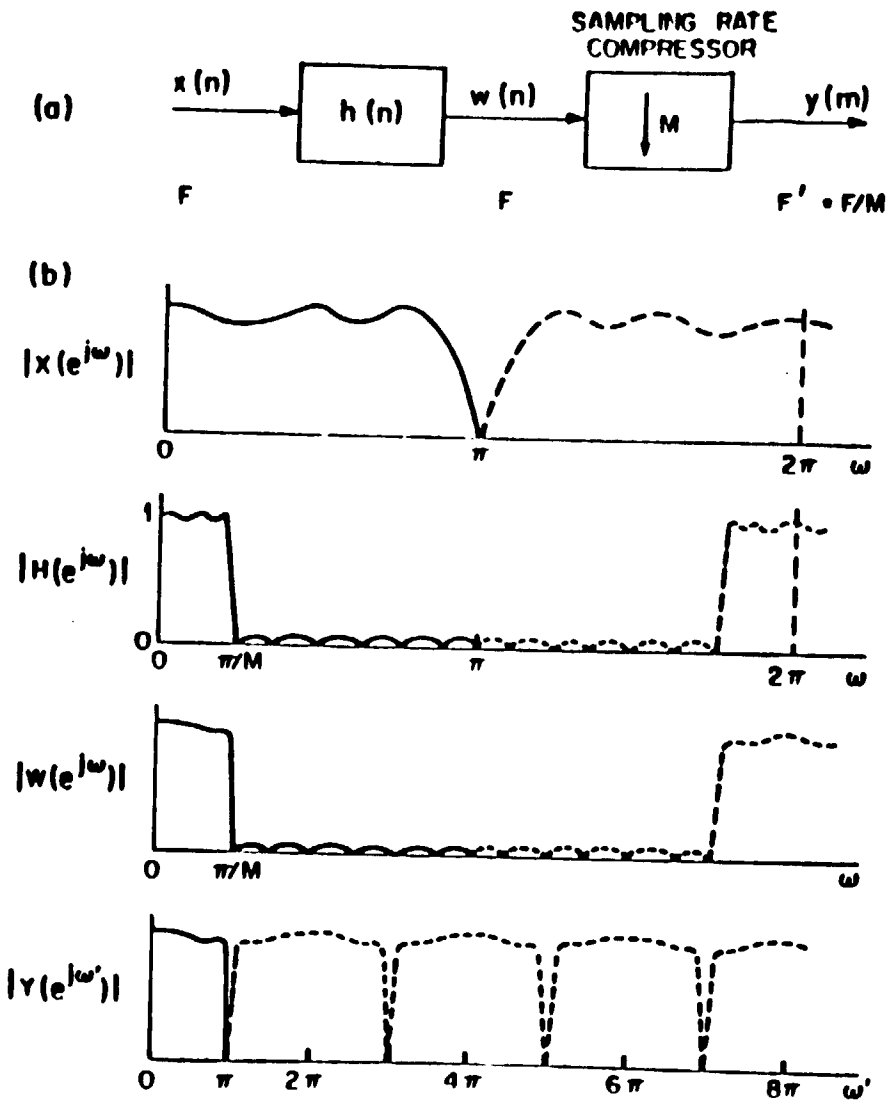
- DIGITAL AUDIO:
  - -- CD (44.1 KHz) « DAT (48 KHz/32 KHz)
- DIGITAL VIDEO:
  - -- Composite (14.3 MHz) « Component (13.5 MHz)
- COMMUNICATIONS:
  - -- TDM/FDM Transmux
  - -- Subband Speech Coding
  - -- Digital Radio & Satellite Modems
- ZOOM SPECTRUM ANALYSIS
- RADAR

REFERENCE: Multirate Digital Signal Processing, Ronald Crochiere, Lawrence Rabiner, Prentice Hall 1983

DIGITIAL FILTER PROCESSOR (DFP) FUNCTIONAL BLOCK DIAGRAM



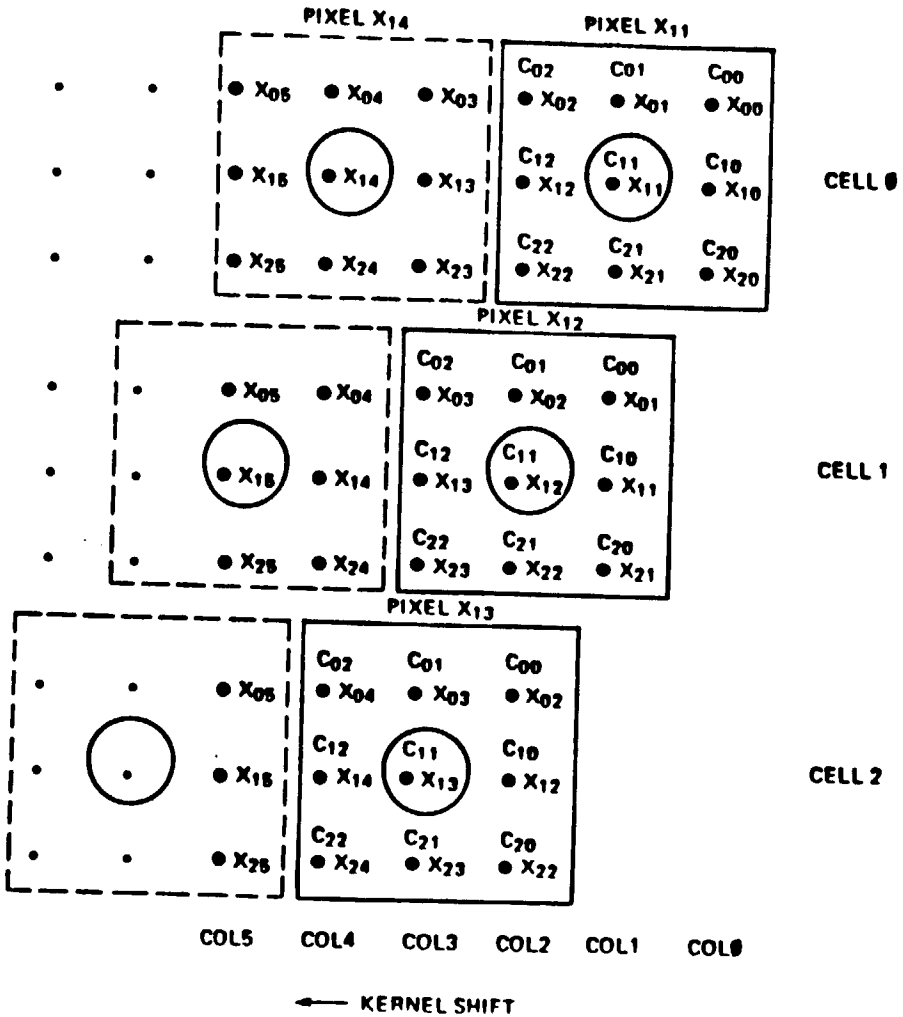
DECIMATION


(a)
x (n)
h (n)
w (n)
SAMPLING RATE COMPRESSOR
M
y (m)
F
F
F' = F/M
(b)
|X(e^jω)|
0
π
2π
ω
|H(e^jω)|
1
0
0
π/M
π
2π
ω
|W(e^jω)|
0
π/M
ω
|Y(e^jω')|
0
π
2π
4π
6π
8π
ω'

DECIMATION-BY-TWO FIR SEQUENCE 20 MHz TO 10 MHz

CLK	CELL0	CELL1	CELL2	CELL3	SUM/CLR
0	$C_7 \bullet X_0$	0			—
1	$+C_6 \bullet X_1$	0			—
2	$+C_5 \bullet X_2$	$C_7 \bullet X_2$			—
3	$+C_4 \bullet X_3$	$+C_6 \bullet X_3$			—
4	$+C_3 \bullet X_4$	$+C_5 \bullet X_4$	$C_7 \bullet X_4$		—
5	$+C_2 \bullet X_5$	$+C_4 \bullet X_5$	$+C_6 \bullet X_5$		—
6	$+C_1 \bullet X_6$	$+C_3 \bullet X_6$	$+C_5 \bullet X_6$	$C_7 \bullet X_6$	—
7	$+C_0 \bullet X_7$	$+C_2 \bullet X_7$	$+C_4 \bullet X_7$	$+C_6 \bullet X_7$	CELL0 (Y7)
8	$C_7 \bullet X_8$	$+C_1 \bullet X_8$	$+C_3 \bullet X_8$	$+C_5 \bullet X_8$	CELL0 (Y7)
9	$+C_6 \bullet X_9$	$+C_0 \bullet X_9$	$+C_2 \bullet X_9$	$+C_4 \bullet X_9$	CELL1 (Y9)
10	$+C_5 \bullet X_{10}$	$C_7 \bullet X_{10}$	$+C_1 \bullet X_{10}$	$+C_3 \bullet X_{10}$	CELL1(Y9)
11	$+C_4 \bullet X_{11}$	$+C_6 \bullet X_{11}$	$+C_0 \bullet X_{11}$	$+C_2 \bullet X_{11}$	CELL2 (Y11)
12	$+C_3 \bullet X_{12}$	$+C_5 \bullet X_{12}$	$C_7 \bullet X_{12}$	$+C_1 \bullet X_{12}$	CELL2 (Y11)
13	$+C_2 \bullet X_{13}$	$+C_4 \bullet X_{13}$	$+C_6 \bullet X_{13}$	$+C_0 \bullet X_{13}$	CELL3 (Y13)
14	$+C_1 \bullet X_{14}$	$+C_3 \bullet X_{14}$	$+C_5 \bullet X_{14}$	$C_7 \bullet X_{14}$	CELL3 (Y13)
15	$+C_0 \bullet X_{15}$	$+C_2 \bullet X_{15}$	$+C_4 \bullet X_{15}$	$+C_6 \bullet X_{15}$	CELL0 (Y15)

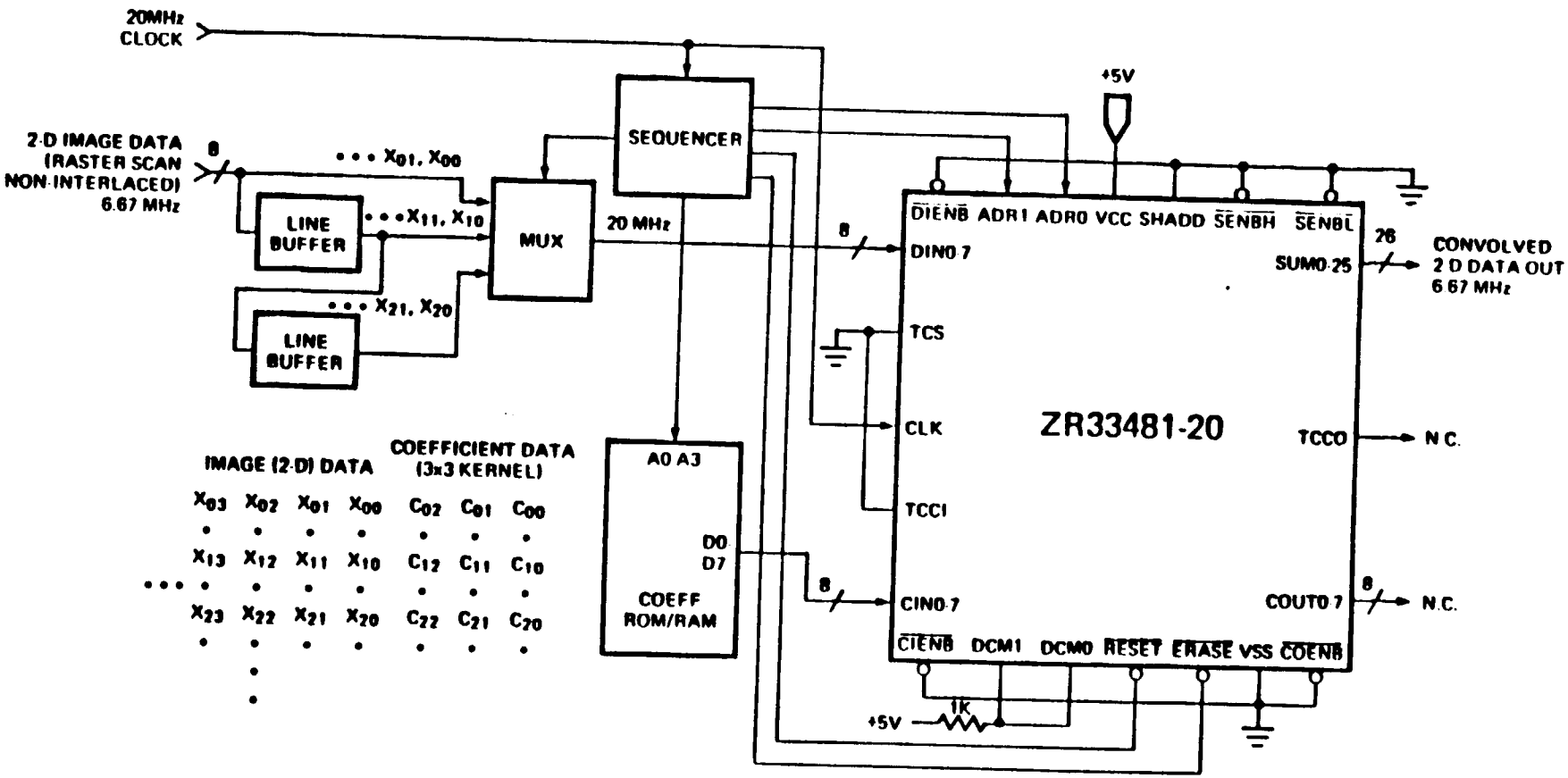
3 X 3 CONVOLUTION



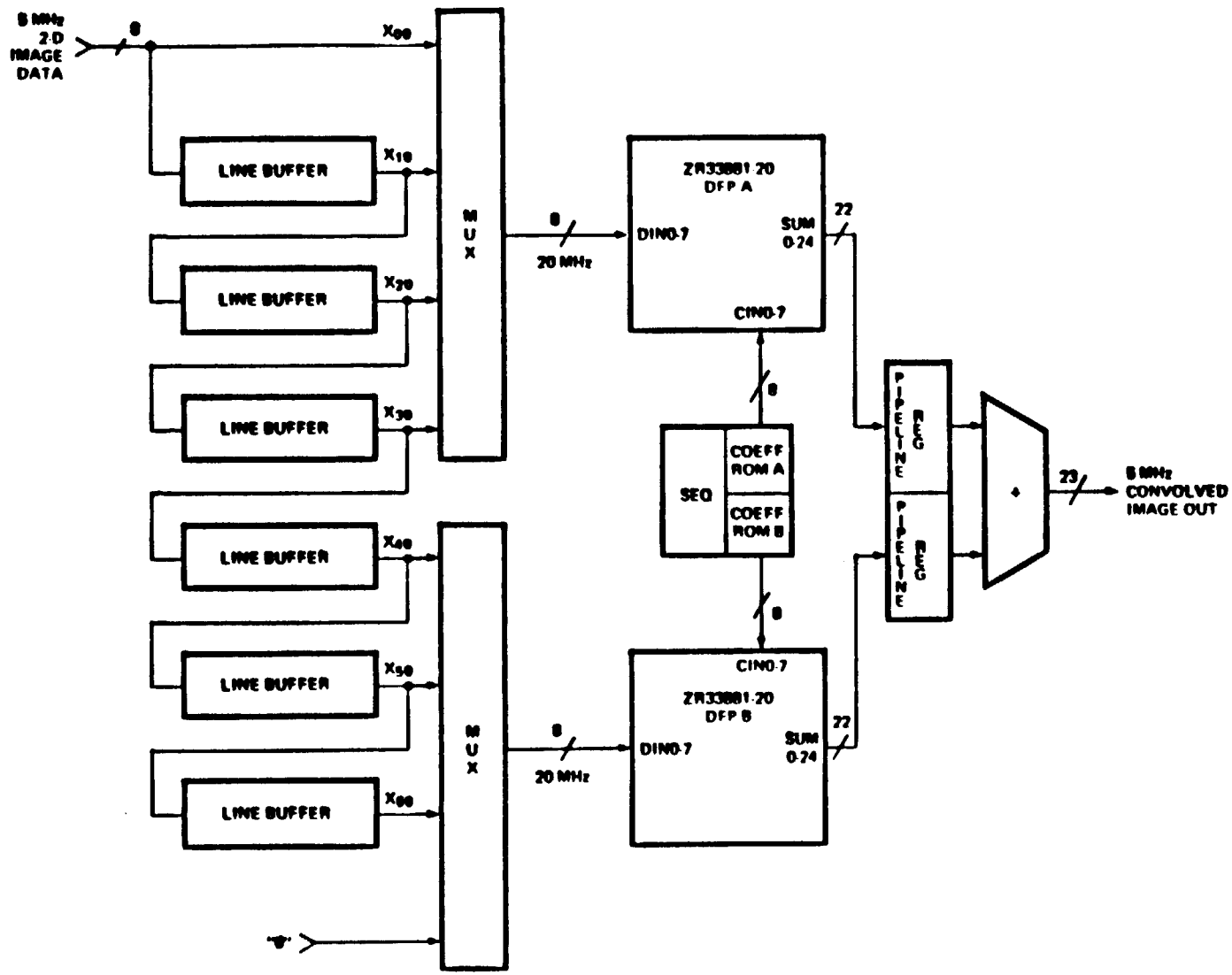
6.67 MHz 3 X 3 CONVOLUTION SEQUENCE

CELL 0	CELL 1	CELL 2		SUM OUT
$C_{00}X_{00}$				
$+C_{10}X_{10}$				
$+C_{20}X_{20}$				
$+C_{01}X_{01}$	$C_{00}X_{01}$			
$+C_{11}X_{11}$	$+C_{10}X_{11}$			
$+C_{21}X_{21}$	$+C_{20}X_{21}$			
$+C_{02}X_{02}$	$+C_{01}X_{02}$	$C_{00}X_{02}$		
$+C_{12}X_{12}$	$+C_{11}X_{12}$	$+C_{10}X_{12}$		
$+C_{22}X_{22}$	$+C_{21}X_{22}$	$+C_{20}X_{22}$		CELL 0
$C_{00}X_{03}$	$+C_{02}X_{03}$	$+C_{01}X_{03}$		
$+C_{10}X_{13}$	$+C_{12}X_{13}$	$+C_{11}X_{13}$		
$+C_{20}X_{23}$	$+C_{22}X_{23}$	$+C_{21}X_{23}$		CELL 1
$+C_{01}X_{04}$	$C_{00}X_{04}$	$+C_{02}X_{04}$		
$+C_{11}X_{14}$	$+C_{10}X_{14}$	$+C_{12}X_{14}$		
$+C_{21}X_{24}$	$+C_{20}X_{24}$	$+C_{22}X_{24}$		CELL 2
$+C_{02}X_{05}$	$+C_{01}X_{05}$	$C_{00}X_{05}$		
$+C_{12}X_{15}$	$+C_{11}X_{15}$	$+C_{10}X_{15}$		
$+C_{22}X_{25}$	$+C_{21}X_{25}$	$+C_{20}X_{25}$		CELL 0
•	$+C_{02}X_{06}$	$+C_{01}X_{06}$		
•	$+C_{12}X_{16}$	$+C_{11}X_{16}$		
•	$+C_{22}X_{26}$	$+C_{21}X_{26}$		CELL 1
	•	$+C_{02}X_{07}$		
	•	$+C_{12}X_{17}$		
	•	$+C_{22}X_{27}$		CELL 2
		•		•
		•		•
		•		•

6.67 MHz 3 X 3 CONVOLUTION BLOCK DIAGRAM



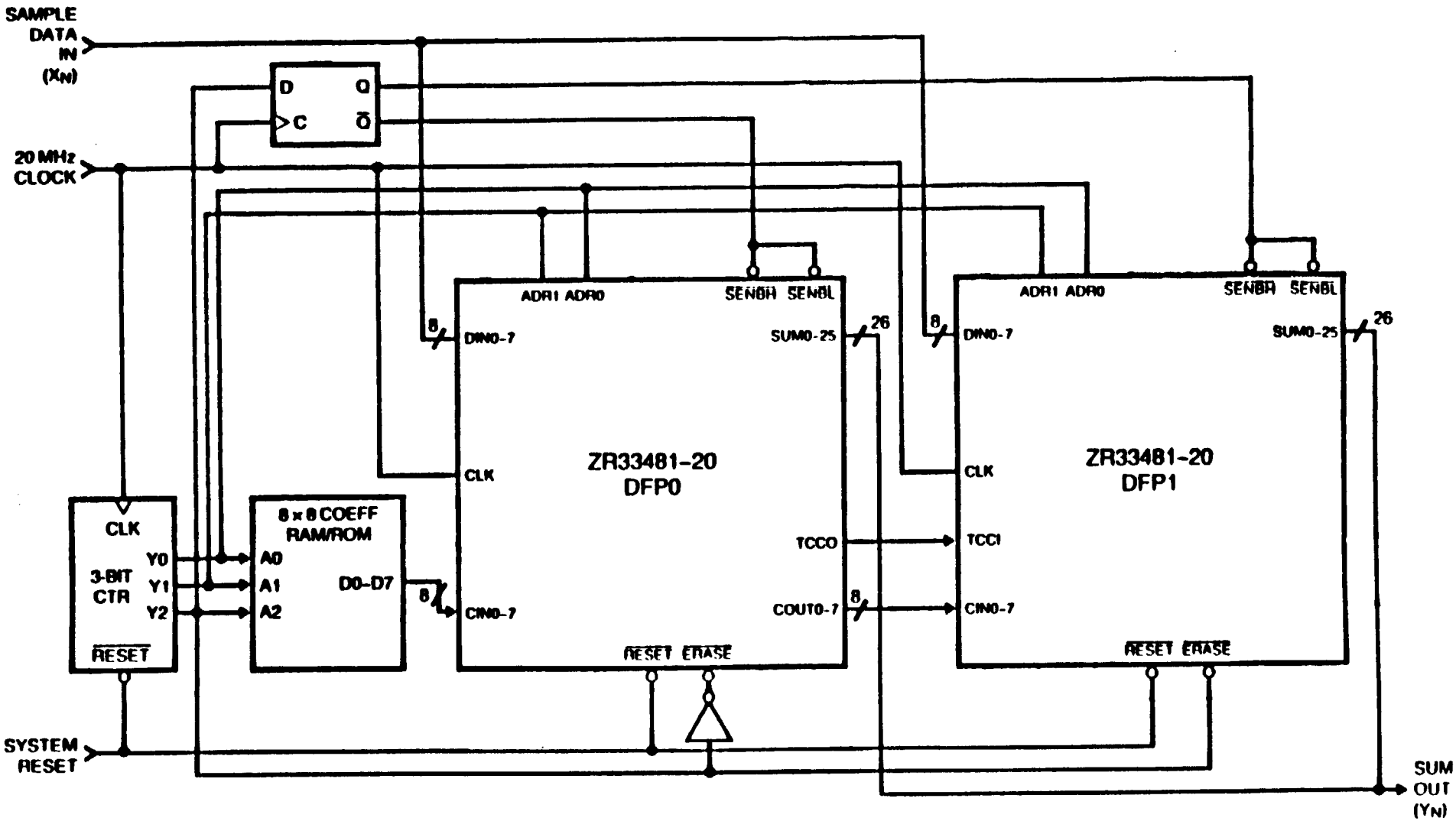
5 MHz 7X7 CONVOLUTION BLOCK DIAGRAM



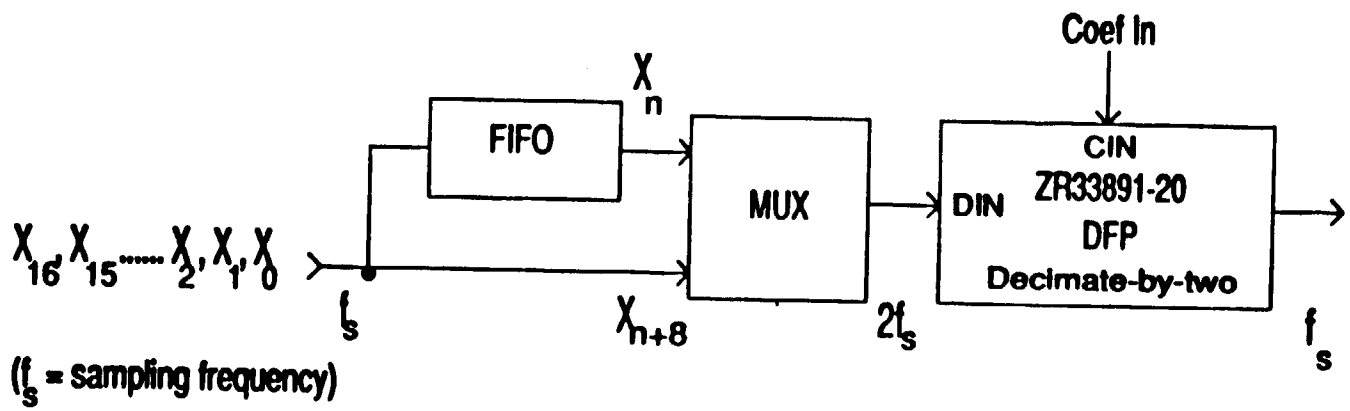
DFP BASICS

- Decimation & Interpolation
- **Extended Filter Lengths**
- Extended Sample Rates

ZR33481 DFP 20MHz 8-TAP, 8 X 8 FIR



Using Decimation-by-Two to Process a 10MHz Signal with a 20MHz DFP



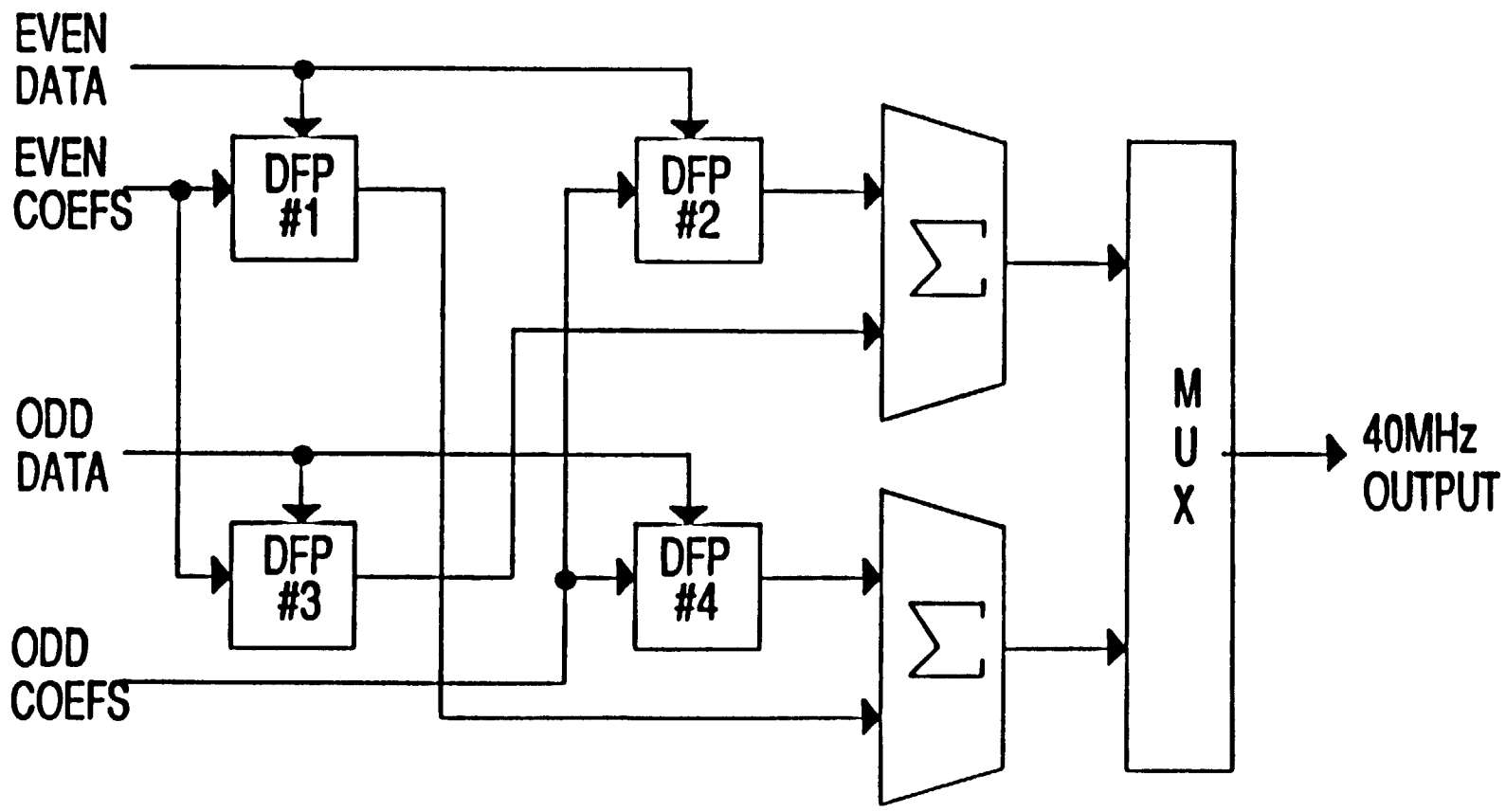
Extended Length Filters Using Decimation Techniques

CELL 0	CELL 1	CELL 2	CELL 3	CELL 4	CELL 5-CELL 7	SUM OUT
C0	C1	C2	C3	C4	••••	
c_0 x_0	---	---	---	---	••••	
c_8 x_8	---	---	---	---	••••	
c_1 x_1	c_0 x_1	---	---	---	••••	
c_9 x_9	c_8 x_9	---	---	---	••••	
c_2 x_2	c_1 x_2	c_0 x_2	---	---	••••	
c_{10} x_{10}	c_9 x_{10}	c_8 x_{10}	---	---	••••	
c_3 x_3	c_2 x_3	c_1 x_3	c_0 x_3	---	••••	
c_{11} x_{11}	c_{10} x_{11}	c_9 x_{11}	c_8 x_{11}	---	••••	
c_4 x_4	c_3 x_4	c_2 x_4	c_1 x_4	c_0 x_4	••••	
c_{12} x_{12}	c_{11} x_{12}	c_{10} x_{12}	c_9 x_{12}	c_8 x_{12}	••••	
c_5 x_5	c_4 x_5	c_3 x_5	c_2 x_5	c_1 x_5	••••	
c_{13} x_{13}	c_{12} x_{13}	c_{11} x_{13}	c_{10} x_{13}	c_9 x_{13}	••••	
c_6 x_6	c_5 x_6	c_4 x_6	c_3 x_6	c_2 x_6	••••	
c_{14} x_{14}	c_{13} x_{14}	c_{12} x_{14}	c_{11} x_{14}	c_{10} x_{14}	••••	
c_7 x_7	c_6 x_7	c_5 x_7	c_4 x_7	c_3 x_7	••••	
$\boxed{c_{15}\ x_{15}}$	c_{14} x_{15}	c_{13} x_{15}	c_{12} x_{15}	c_{11} x_{15}	••••	Y_{15}
c_0 x_8	c_7 x_8	c_6 x_8	c_5 x_8	c_4 x_8	••••	---
c_8 x_{16}	$\boxed{c_{15}\ x_{16}}$	c_{14} x_{16}	c_{13} x_{16}	c_{12} x_{16}	••••	Y_{16}
c_1 x_9	c_0 x_9	c_7 x_9	c_6 x_9	c_5 x_9	••••	---
c_9 x_{17}	c_8 x_{17}	$\boxed{c_{15}\ x_{17}}$	c_{14} x_{17}	c_{13} x_{17}	••••	Y_{17}
c_2 x_{10}	c_1 x_{10}	c_0 x_{10}	c_7 x_{10}	c_6 x_{10}	••••	---
c_{10} x_{18}	c_9 x_{18}	c_8 x_{18}	$\boxed{c_{15}\ x_{18}}$	c_{14} x_{18}	••••	Y_{18}
c_3 x_{11}	c_2 x_{11}	c_1 x_{11}	c_0 x_{11}	c_7 x_{11}	••••	---
c_{11} x_{19}	c_{10} x_{19}	c_9 x_{19}	c_8 x_{19}	$\boxed{c_{15}\ x_{19}}$	••••	Y_{19}
c_4 x_{12}	c_3 x_{12}	c_2 x_{12}	c_1 x_{12}		••••	---

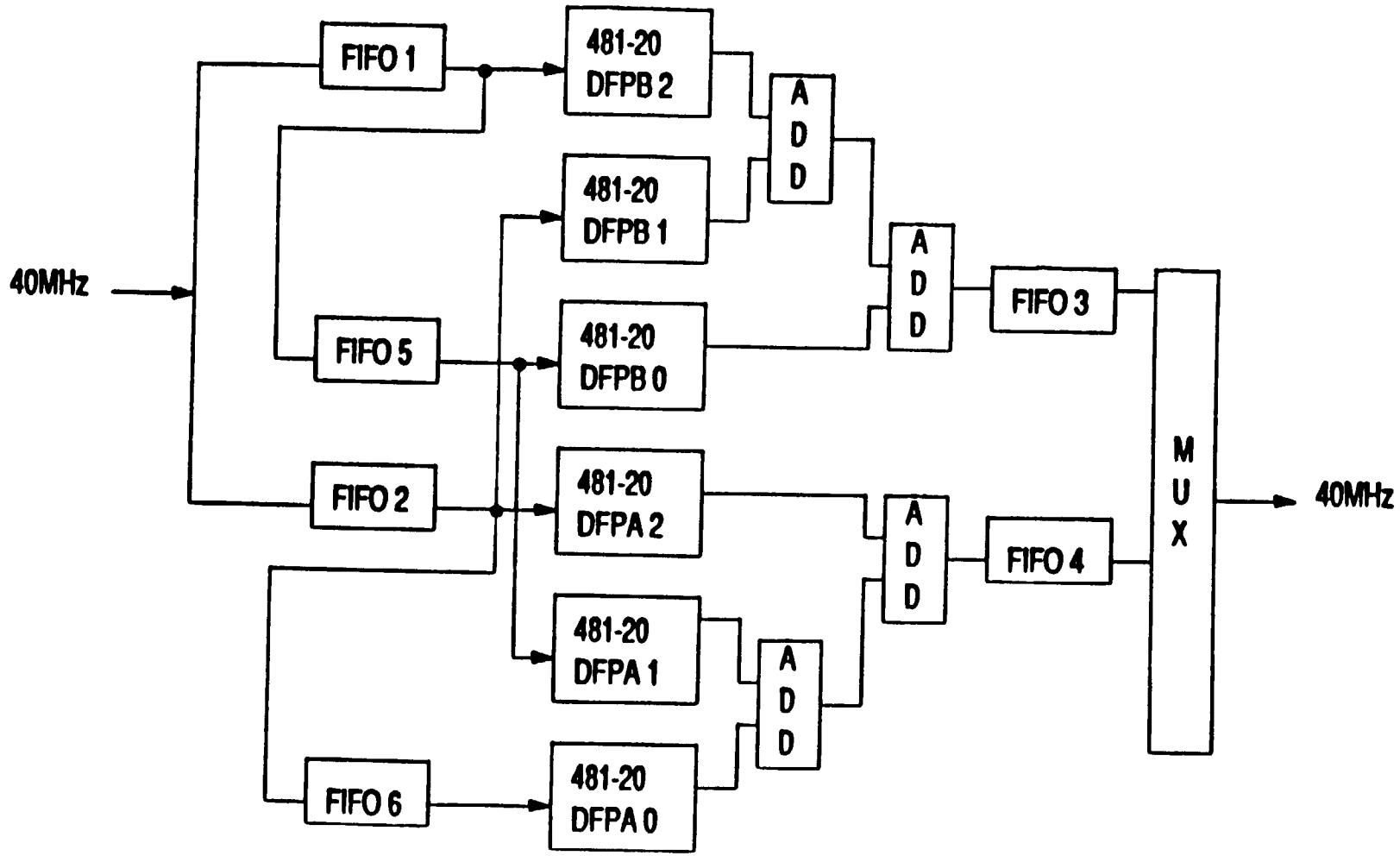
DFP BASICS

- Decimation & Interpolation
- Extended Filter Lengths
- **Extended Sample Rates**

40 MHZ, 16-TAP FILTER WITH FOUR ZR33891 DFPs



40MHz 3 x 3 CONVOLUTION



ZORAN

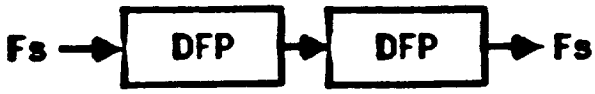
DIGITAL FILTER PROCESSOR SUMMARY GUIDE

C = TOTAL DFP CELLS
H = # DFPs CASCADED HORZ
V = # DFPs CASCADED VERT

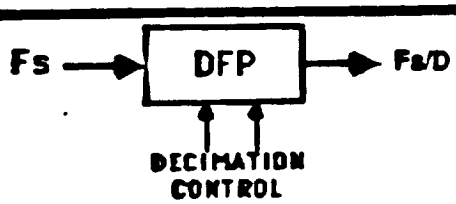
D = DECIMATION FACTOR
I = INTERPOLATION FACTOR
T = TOTAL # DESIRED TAPS

(ASSUME 8-CELL DFP)

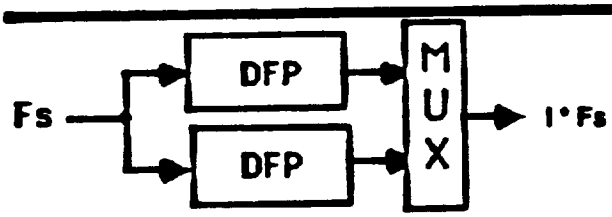
BASIC CONFIGURATION
MAX TAPS = 8
MAX Fs = 20 MHz



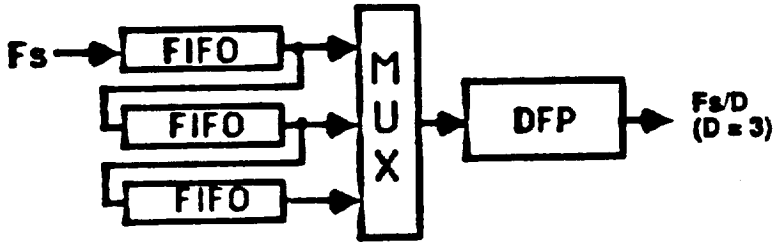
BASIC CASCADE
MAX TAPS = 8 * H
MAX Fs = 20 MHz (27 MHz)



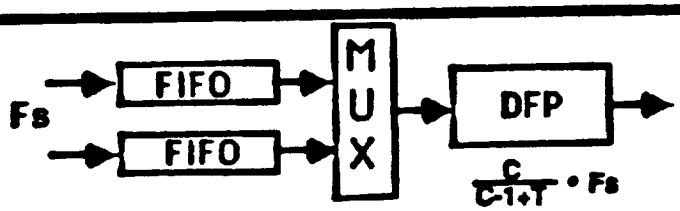
BASIC DECIMATION
MAX TAPS = 8 * D * H
MAX Fs = 20 MHz (27 MHz)



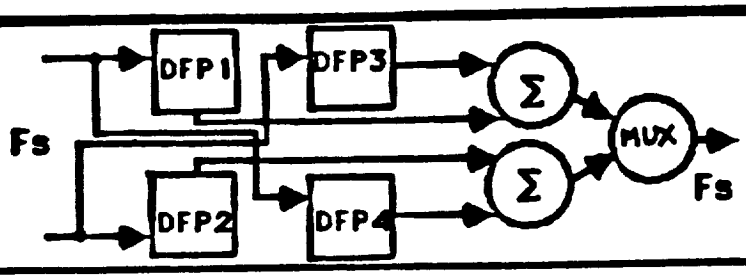
BASIC INTERPOLATION
MAX TAPS = 8 * I * H
MAX Fs(IN) = 20 MHz
MAX Fs(OUT) = I * Fs = 40 MHz



1-D REDUCED SAMPLE RATE
MAX TAPS = 8 * D * H
MAX Fs = 20/D MHz
OR
2-D BASIC CONFIGURATION
MAX HORZ TAPS = 8 * H
MAX VERT TAPS = 4 * V
MAX Fs = 20/D MHz

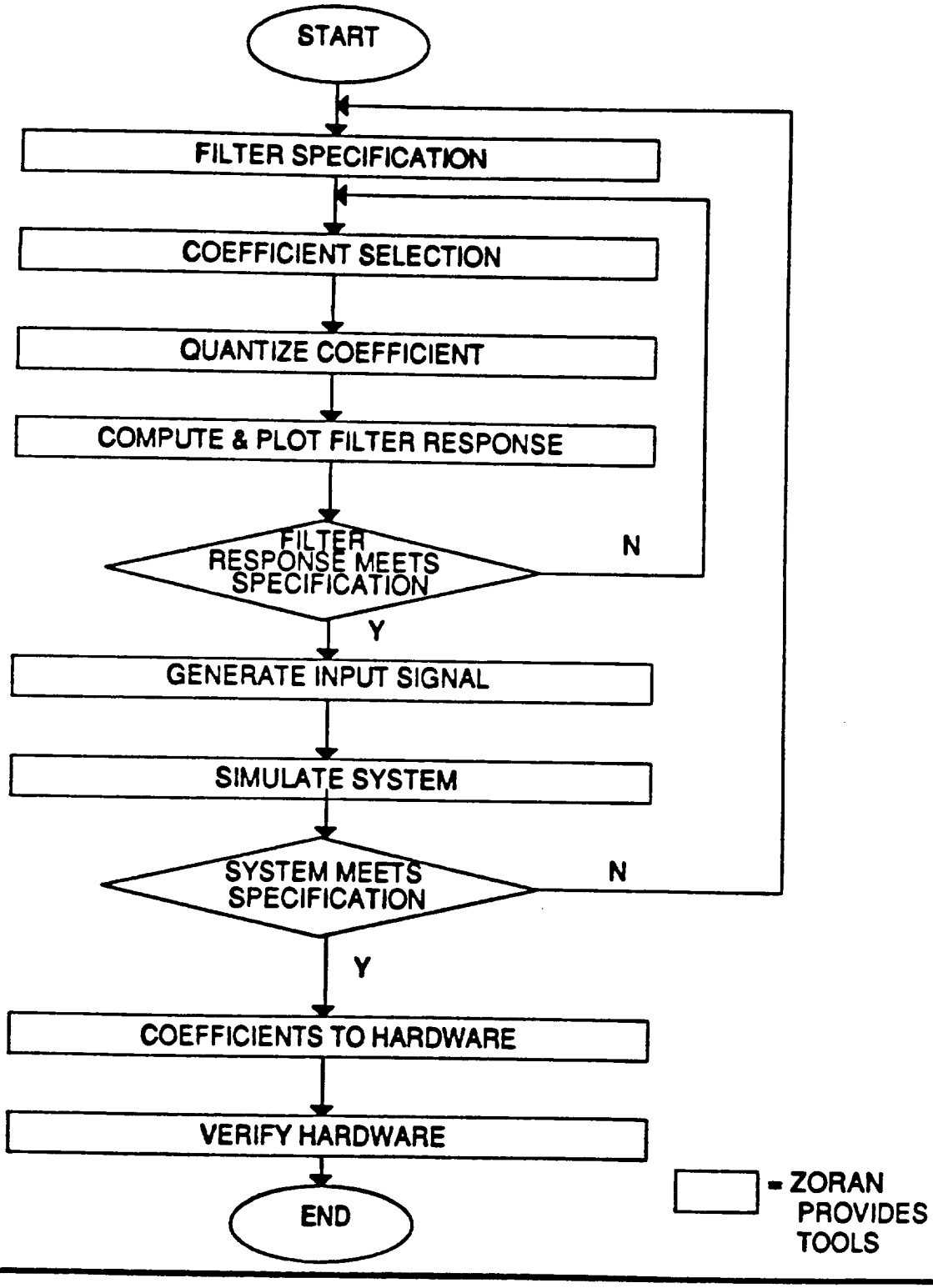


GENERAL CONFIGURATION
MAX Fs(OUT) = $\frac{C}{C-1+T} * Fs$

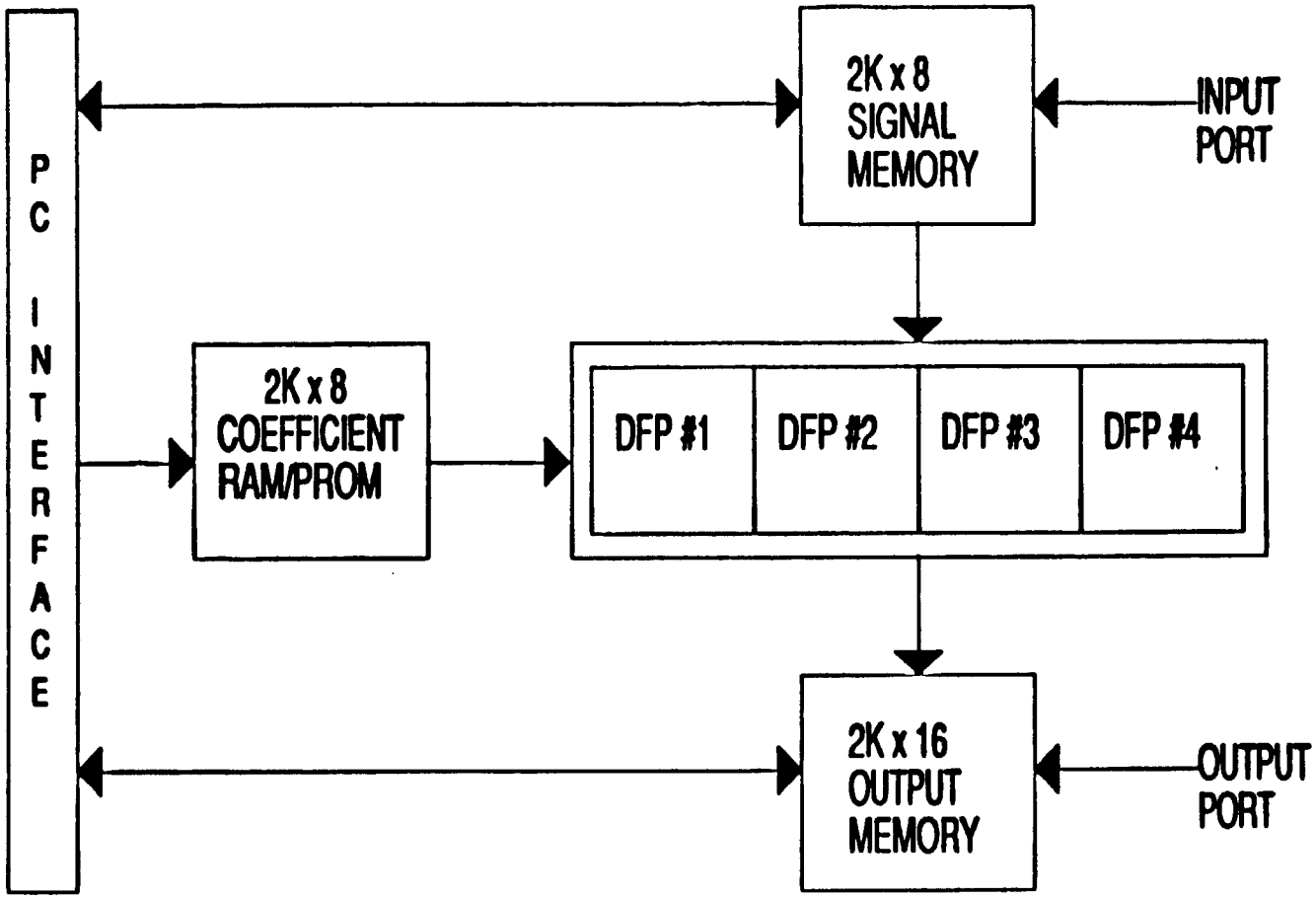


BASIC HIGHER SAMPLE RATE
DFPs = $(Fs/20\ MHz)^2 * H$
TAPS = 8 * (Fs/20) * H

DIGITAL FILTER DESIGN PROCEDURE



DFP DESIGN TOOLS

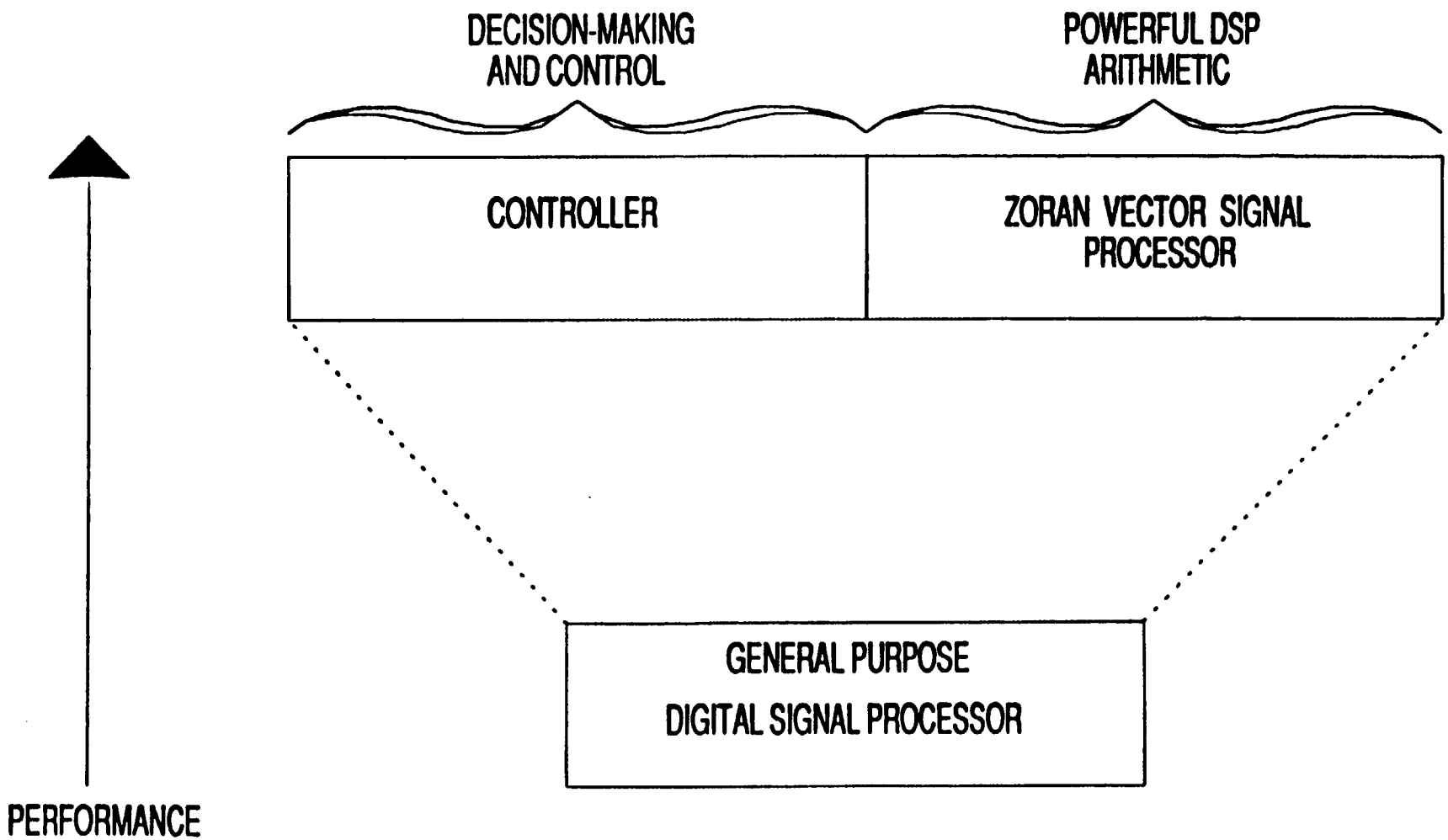


DIGITAL FILTER PROCESSOR BOARD (DFPB)

CURRENT PRODUCTS

- Digital Filter Processors (DFP)
- **ZR34161 Vector Signal Processor (VSP)**

DSP SYSTEM FUNCTIONAL PARTITIONING



ZR34161 VSP FUNCTIONAL APPLICATIONS

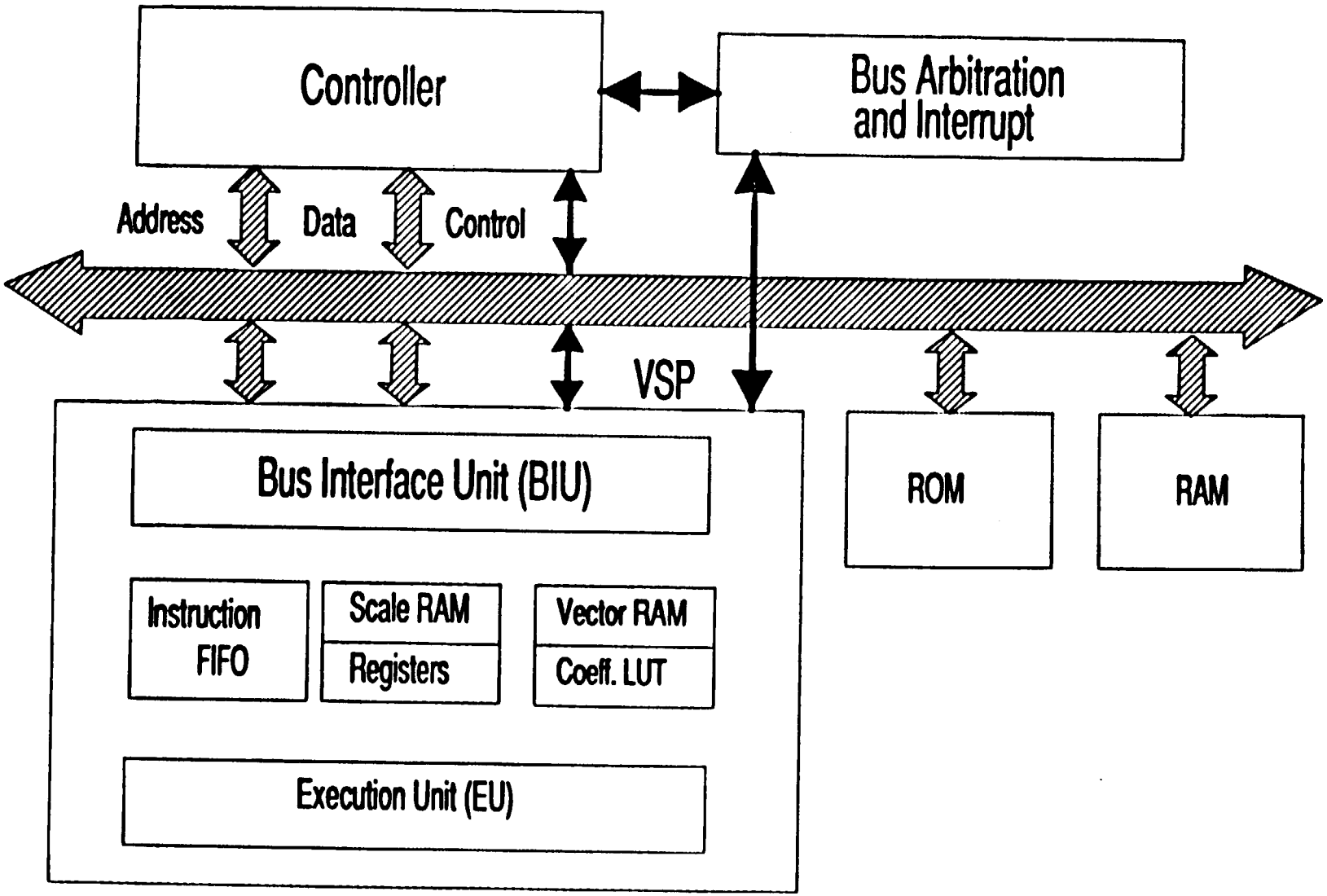
- 1-D and 2-D FFT
- 1-D and 2-D Sine/ Cosine Transforms
- 1-D and 2-D Auto/Cross Correlation
- 1-D and 2-D Convolution
- Filtering
- Windowing
- Modulation/Demodulation
- .
- .
- .
- Spectrum Analysis

ZR34161 VSP PERFORMANCE

(25 MHZ, 80 NSEC CYCLE)

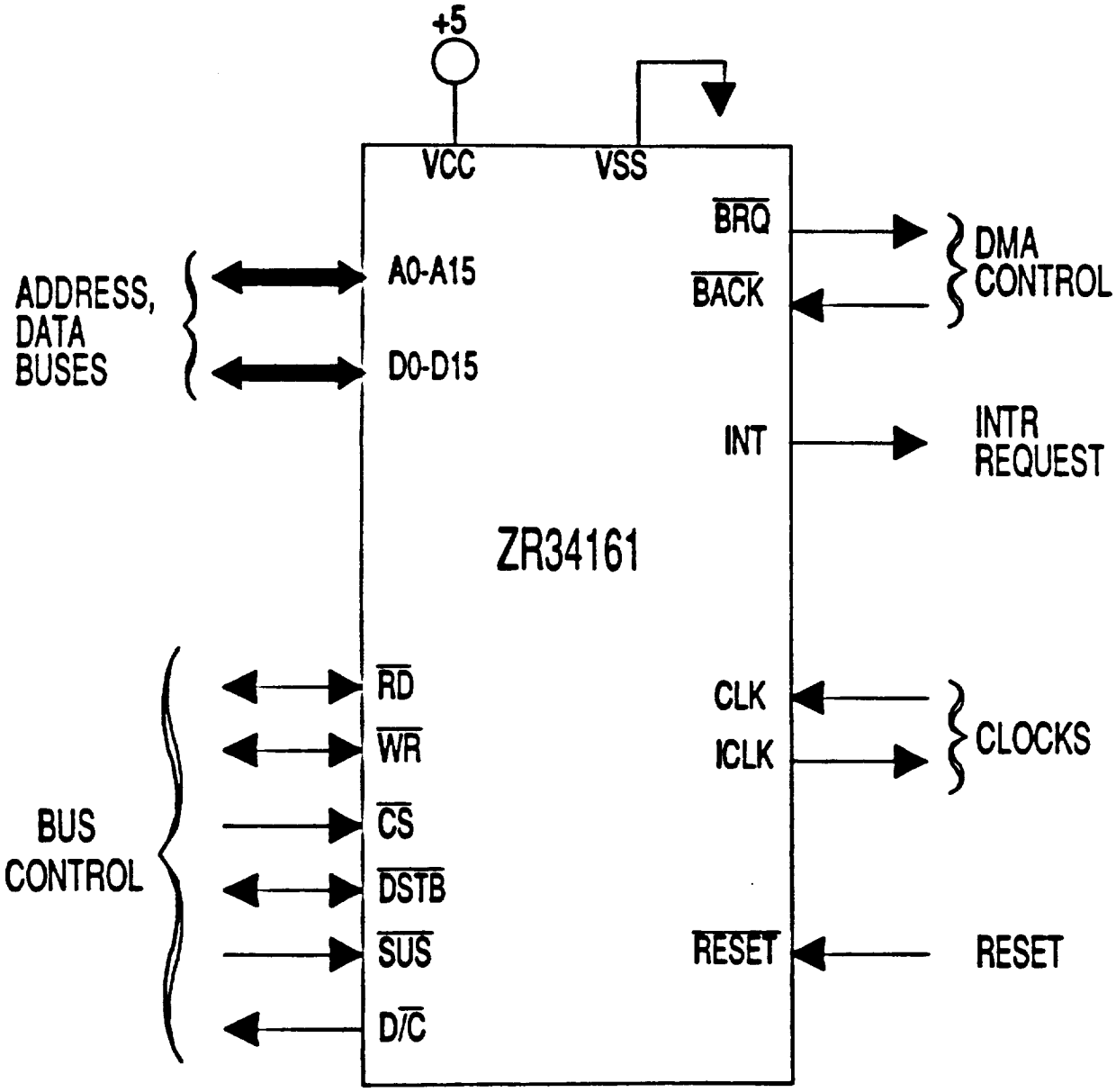
• 1024-point integer complex FFT	2.1 ms
• 1024-point block-floating complex FFT	2.6 ms
• 128-point block-floating complex FFT	190 µs
• 8 x 8-point 2-D complex FFT	131 µs
• 16 x 16-point 2-D FCT	764 µs
• 128-point x 128-point complex vector multiply	42 µs
• 128-point magnitude square/accumulate	23 µs
• 128-point complex demodulation	42 µs
• 4 x 4 matrix multiplication	21 µs
• LD/ST 128 complex data samples	21 µs

VSP - A LOOSELY COUPLED PERIPHERAL PROCESSOR

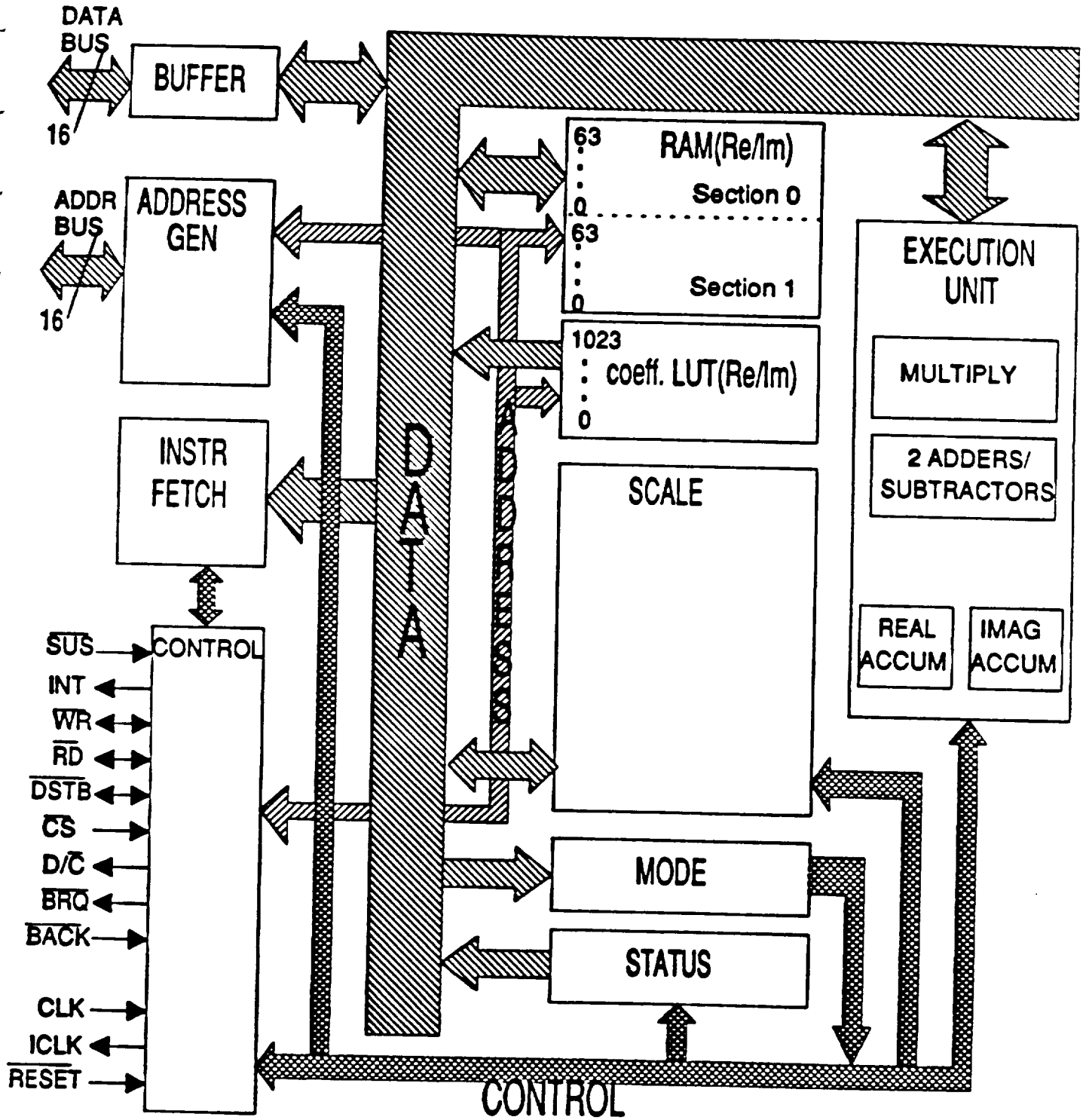


VSP LOGICAL PINOUT

48-PIN DIP



VSP ARCHITECTURE



SIGNIFICANT ARCHITECTURE FEATURES

- High-level "VECTOR" instruction set
- Embedded block floating-point arithmetic for FFT
- Ability to parallel multiple devices

11 INTERNAL ALU INSTRUCTIONS

FFT	(FAST FOURIER TRANSFORM)
MGSQ	(MAGNITUDE SQUARE/ACCUMULATE)
DEMO	(DEMODULATE)
MODLT	(MODULATE)
CMLT	(CROSS MULTIPLY/ACCUMULATE)
CMCN	(COMPLEX CONJUGATE)
ACCR	(ACCUMULATE REAL)
ACCI	(ACCUMULATE IMAGINARY)
ABS	(ABSOLUTE VALUE)
SCL	(SCALE)
SCLT	(SCALE LITERAL)

FFT PRIMITIVE
POWER SPECTRUM PROGRAM

LD NMPT:128, RS:0, MDF:2, ZR:1, MBA:0

FFT NMPT:128, RS:0, FPS:64, LPS:1

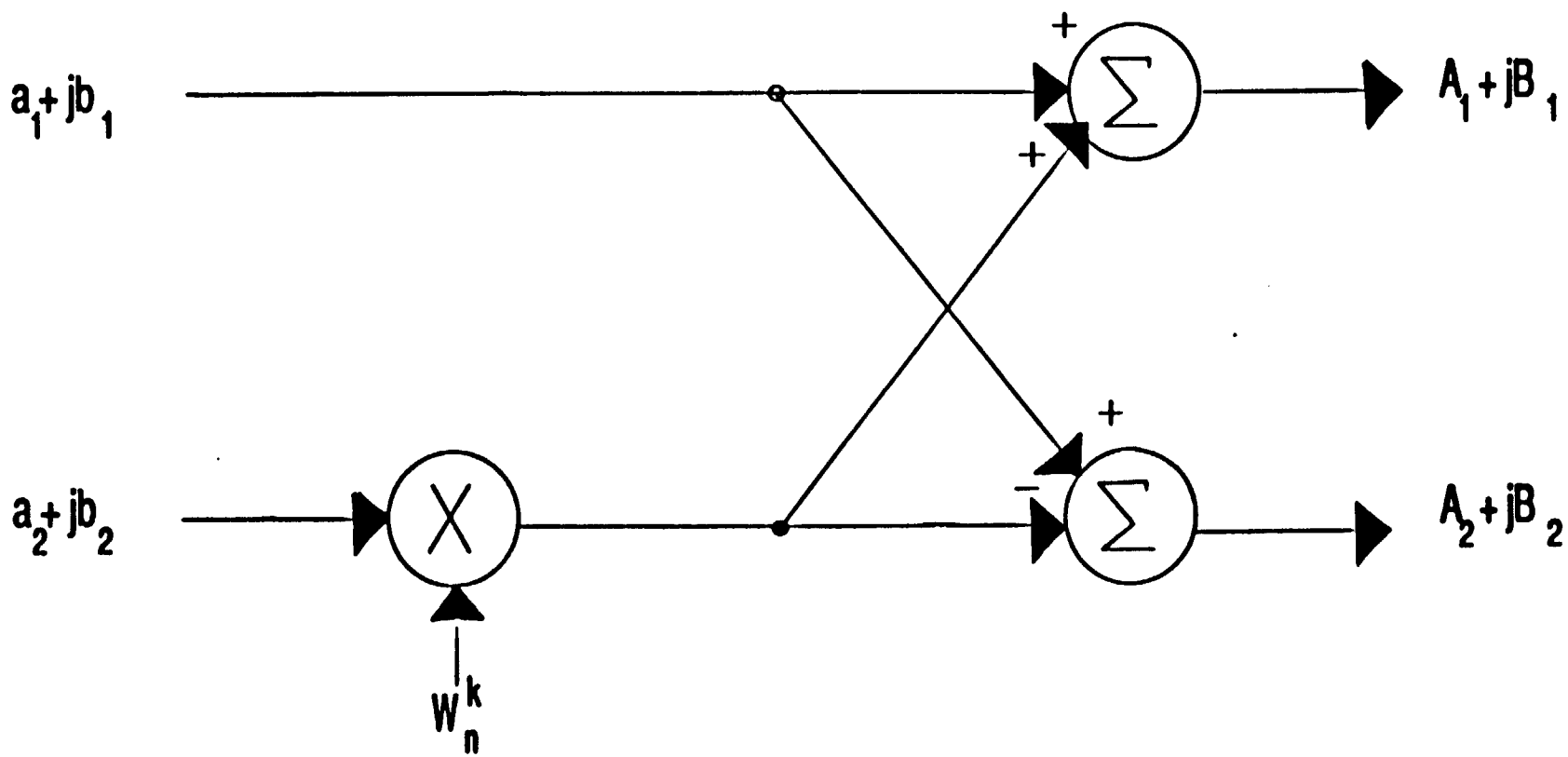
MGSQ NMPT:128, RS:0, ADF:2

ST NMPT:128, RS:0, RV:1, MDF:2, MBA:256

FFT PRIMITIME POWER SPECTRUM PROGRAM EXECUTION TIME

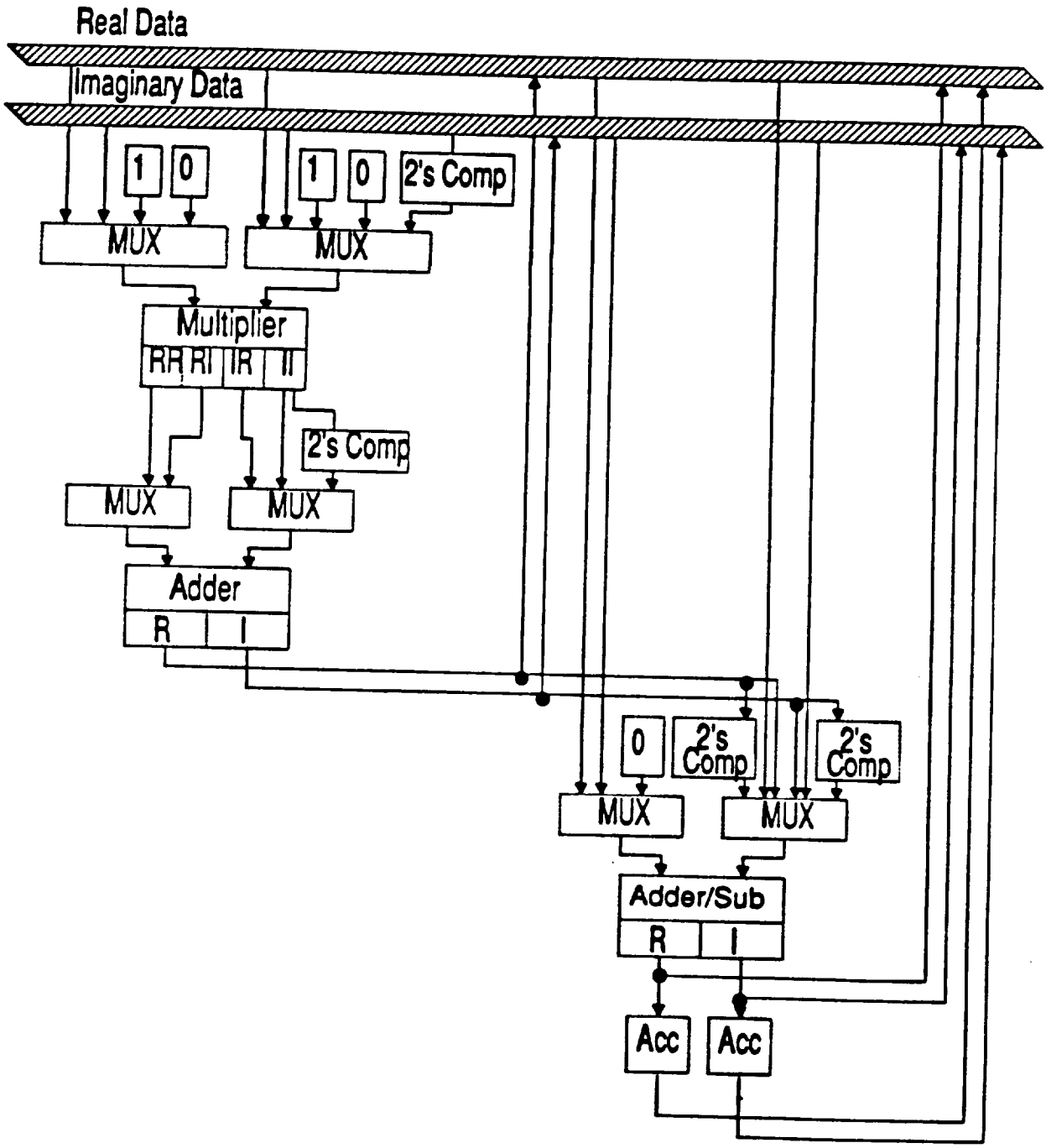
Instr. Fetch	Number Clocks Instr. Fetch	Number Clocks Data Fetch	Execution Time (Clocks)
LD	6	134	134
FFT	6		1850
MGSQ	2		262
ST	6	134	134
Total	20	268	2380

FFT BUTTERFLY



$$W_n^k = \cos\left(2\pi\frac{k}{n}\right) - j\sin\left(2\pi\frac{k}{n}\right)$$

VSP EU


Real Data
Imaginary Data
1
0
1
0
2's Comp
MUX
MUX
Multiplier
RR
RI
IR
II
2's Comp
MUX
MUX
Adder
R
I
0
2's Comp
2's Comp
MUX
MUX
Adder/Sub
R
I
Acc
Acc

EXAMPLE OF VSP ARITHMETIC EFFICIENCY
128-PT COMPLEX FFT

THEORETICAL BEST PERFORMANCE: 1792 CLOCKS

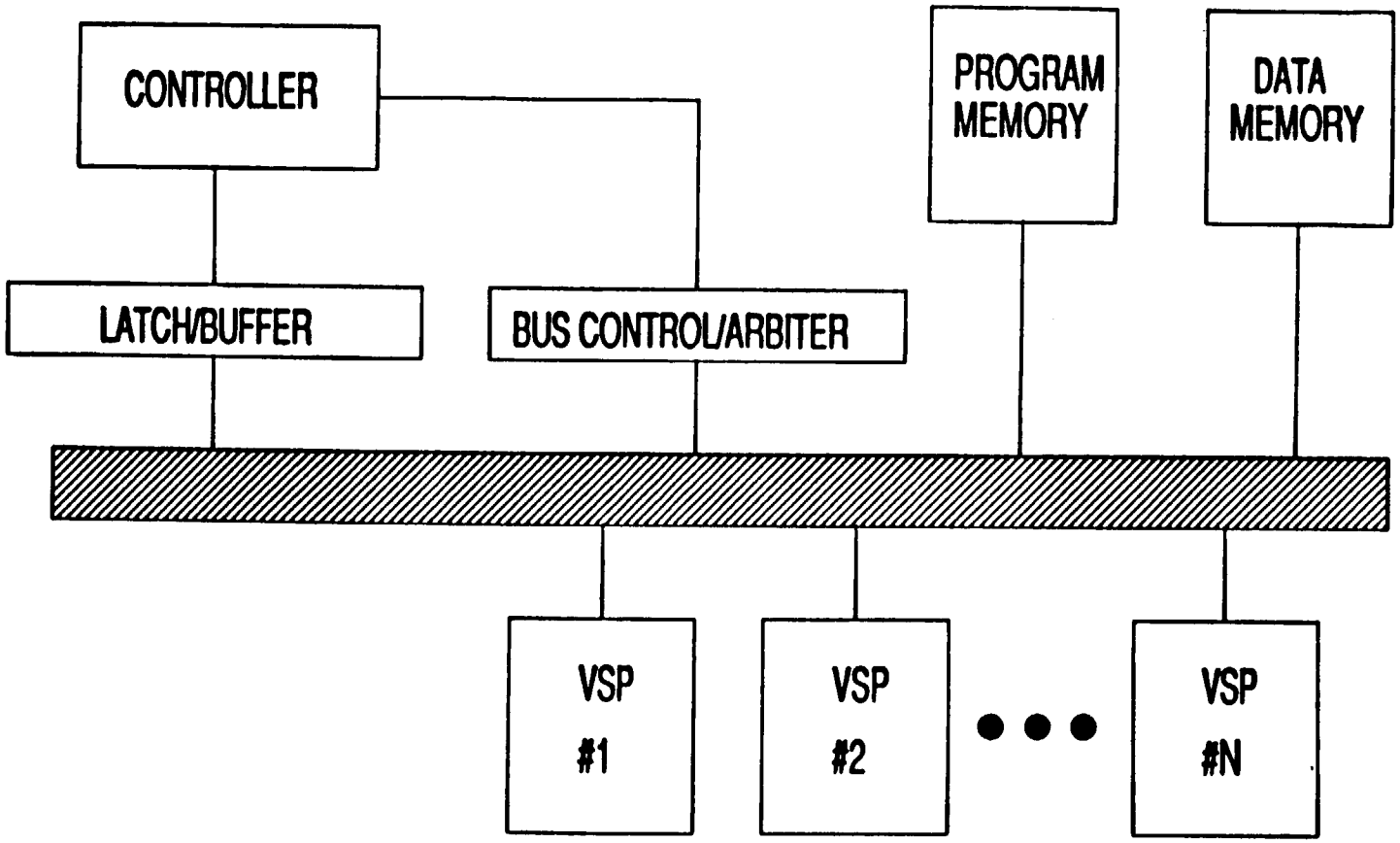
NUMBER OF:

BUTTERFLIES	=	64
PASSES	=	7
REAL MULTIPLIES PER BUTTERFLY	=	4

ACTUAL VSP PERFORMANCE: 1850 CLOCKS

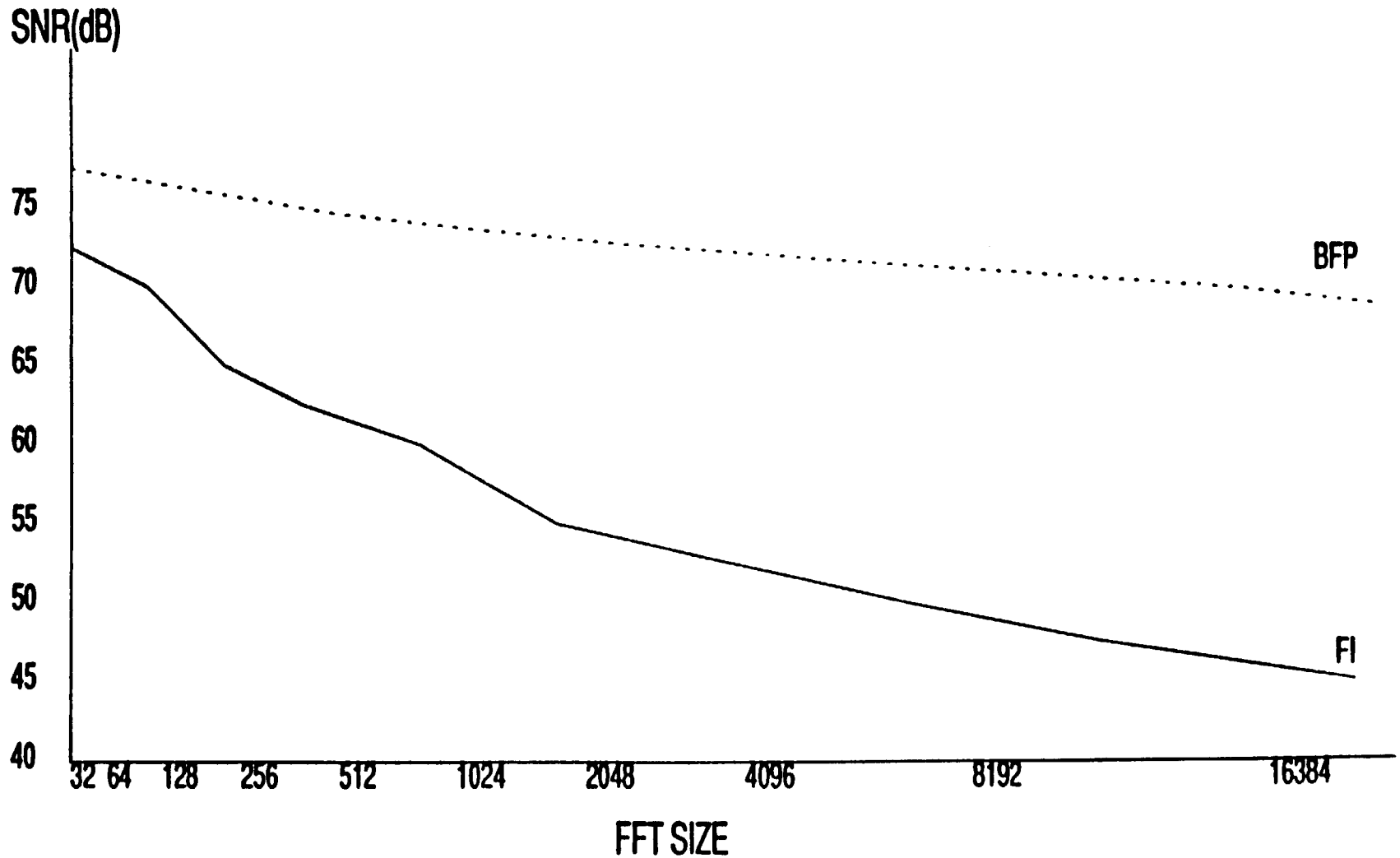
ARITHMETIC EFFICIENCY: 97%

PARALLEL VSP ARCHITECTURE

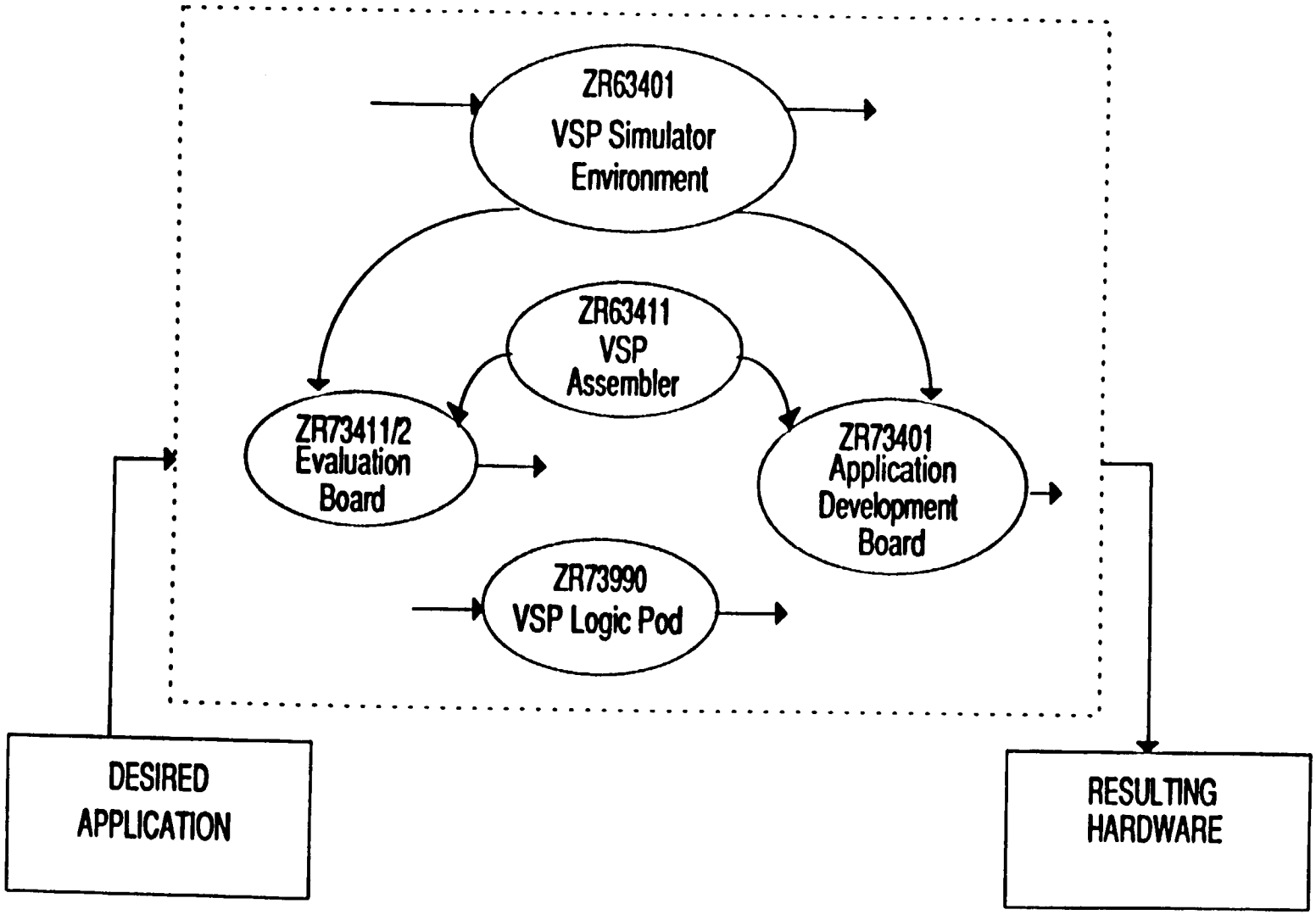


BLOCK FLOATING POINT VS. FIXED INTEGER

ARITHMETIC PERFORMANCE



VSP DEVELOPMENT TOOLS



VSP BASICS

- **Larger FFTs**
- Time-Domain Convolution
- Fast Convolution
- Hardware Interface
- Multi-VSP Configuration

LARGER FFTS

- Factored into smaller FFTs:
  - -- 64 or 128 points
- Up to 1K complex points using internal CLUT
- 2K - 16K complex points using both internal CLUT and external coefficients

VSP-325 FUNCTIONAL APPLICATIONS

ARCHITECTURE AND INSTRUCTIONS ARE OPTIMIZED TO SOLVE GENERAL DSP PROBLEMS:

- 1-D and 2-D Fast Fourier Transforms (FFT)
- Discrete Fourier Transforms (DFT)
- FIR/IIR filters (FIR/IIR)
- Matrix Operations (MTX)
- Thresholding/Comparisons (THR/CMP)
- Min/Max Detection (MIN/MAX)
- Polynomial Expansion (POLY)
- Modulation/Demodulation (MODLT/DEMO)

VSP-325 FEATURES

ARCHITECTURE IS OPTIMIZED FOR:

- Full IEEE (754-1985) 32-bit floating-point arithmetic
- Parallel Processors on a single bus:
  - Multi-processor communication
  - Semaphore handling
- 1-D and 2-D processing
- Dual-ported internal memory
- 16M word external memory address space

VSP-325 FEATURES

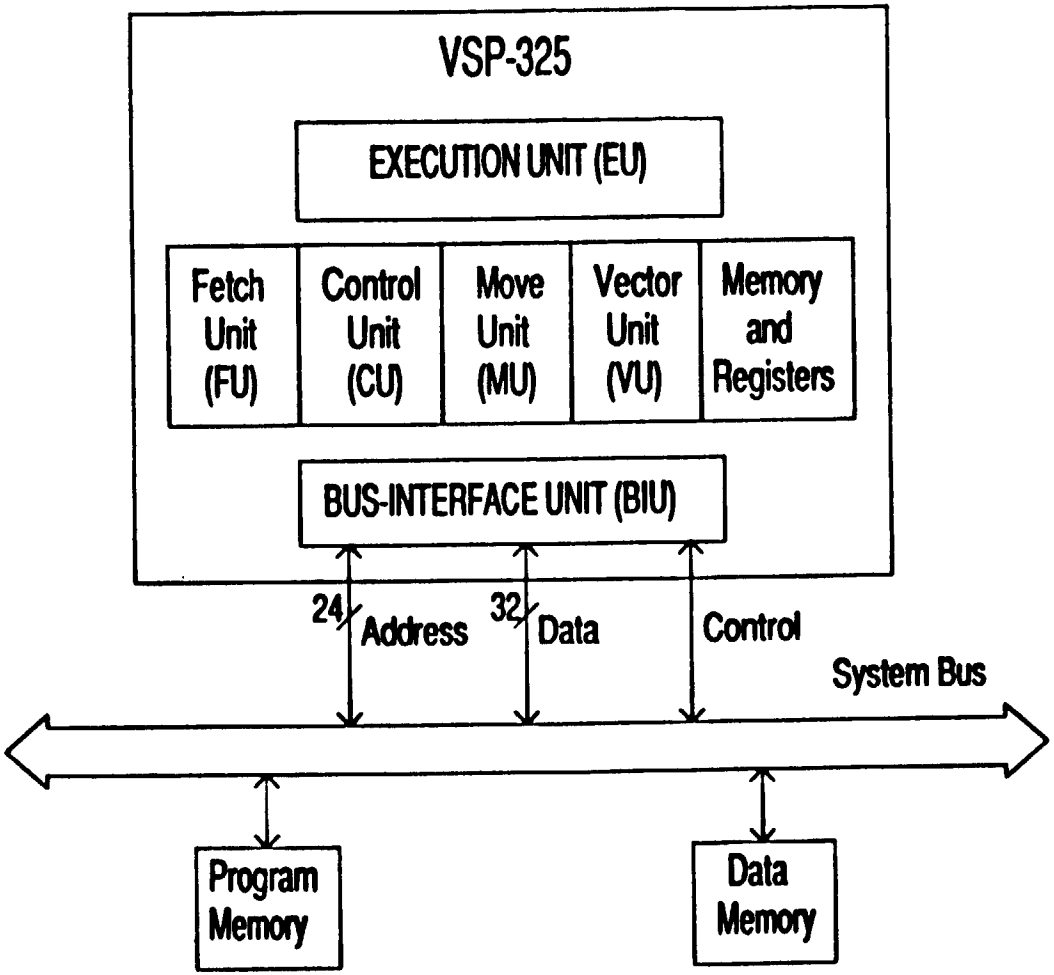
ARCHITECTURE IS OPTIMIZED FOR:

- Flexible addressing modes: Direct, Indirect, Indexed
- Hardware subroutines
- Conditional operations
- Program looping
- Vectorized interrupt and trap
- Start-up initialization

VSP-325 PERFORMANCE (INCLUDING I/O)
(25 MHz Clock / 80ns cycle)

• 64-point complex FFT	83.0 µS
• 128-point real FFT	117.0 µS
• 1024-point complex FFT:	
-- single VSP-325	1731.0 µS
-- two VSP-325s	886.0 µS
• IIR filter (per biquad section)	0.4 µS
• 32-tap FIR of 128 real points	382.0 µS
• Two dimensional 3 x 3 convolution (per output)	1.12 µS
• 4 x 4 real matrices multiplication	8.0 µS
• 10 x 10 real matrices multiplication	132.0 µS
• Log (LN)/per element (POLY)	1.52 µS
• Division/per element (LUT)	0.9 µS
• Square root/per element (LUT)	0.6 µS

VSP-325 SYSTEM CONFIGURATION



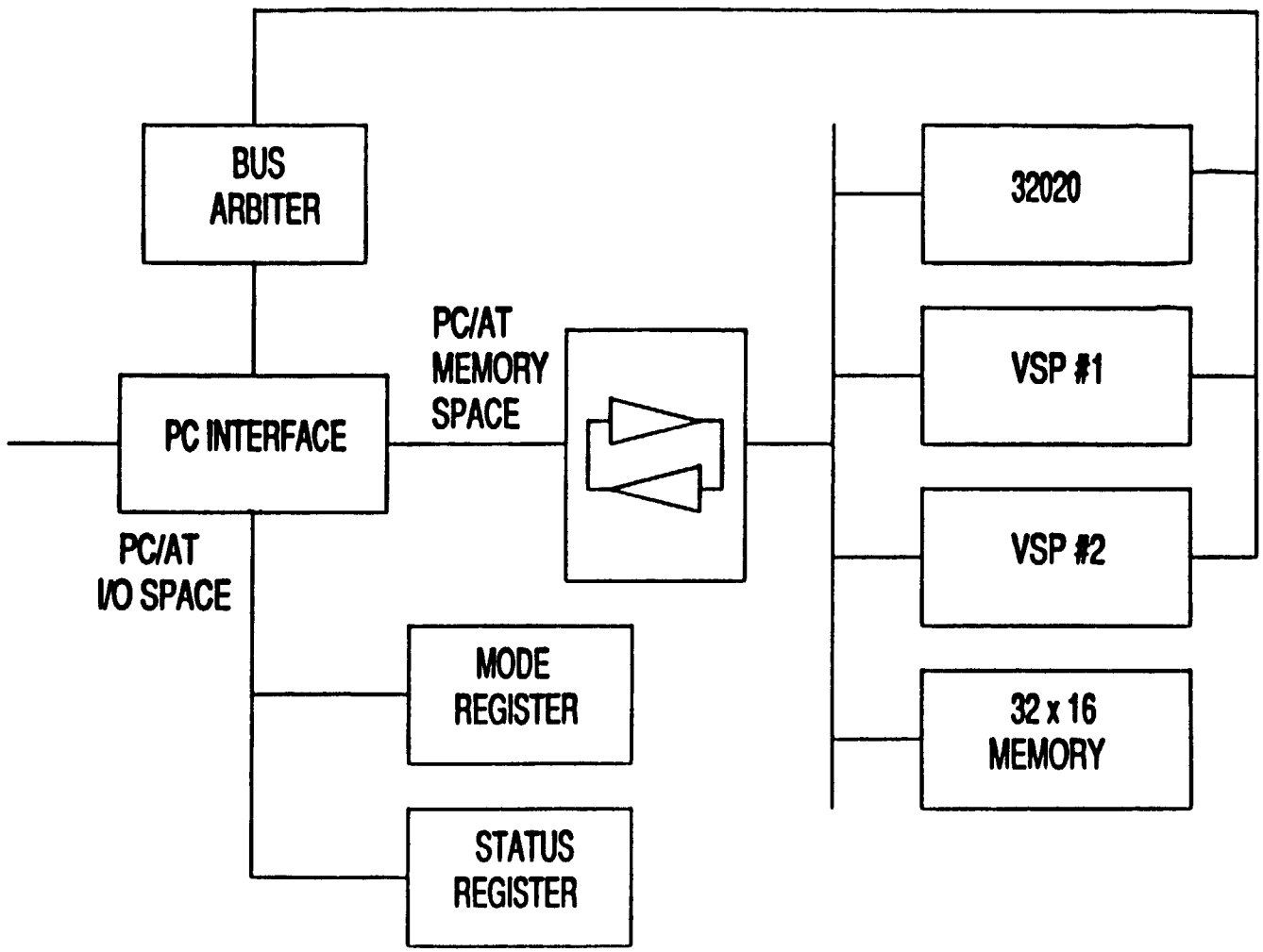
VSP-325 KEY ARITHEMTIC INSTRUCTIONS

- MTX (MATRIX MULTIPLY)
  - -- External matrix up to 1024 x 64
  - -- Internal matrix up to 64 elements

- POLY (POLYNOMIAL EXPANSION)
  - -- Up to 1024 coefficients
  - -- Up to 32 points per instruction
  - -- Trigonometric and transcendental approximations
  - -- Interpolation

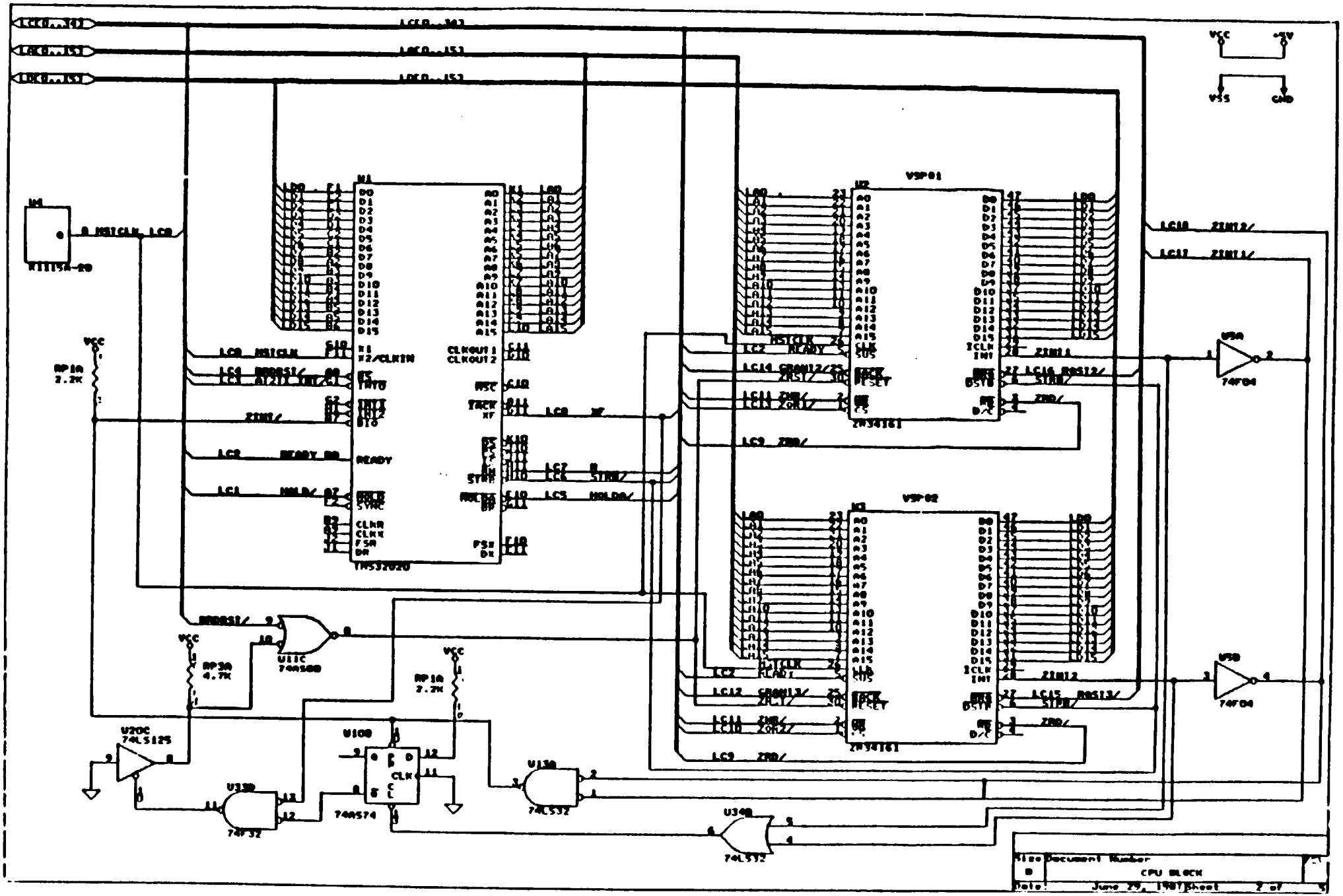
$$P(X) = A_1X^1 + A_2X^2 + \ldots + A_NX^N \; ; \; N < 1024$$

TWO VSP SYSTEM WITH TMS32020 HOST

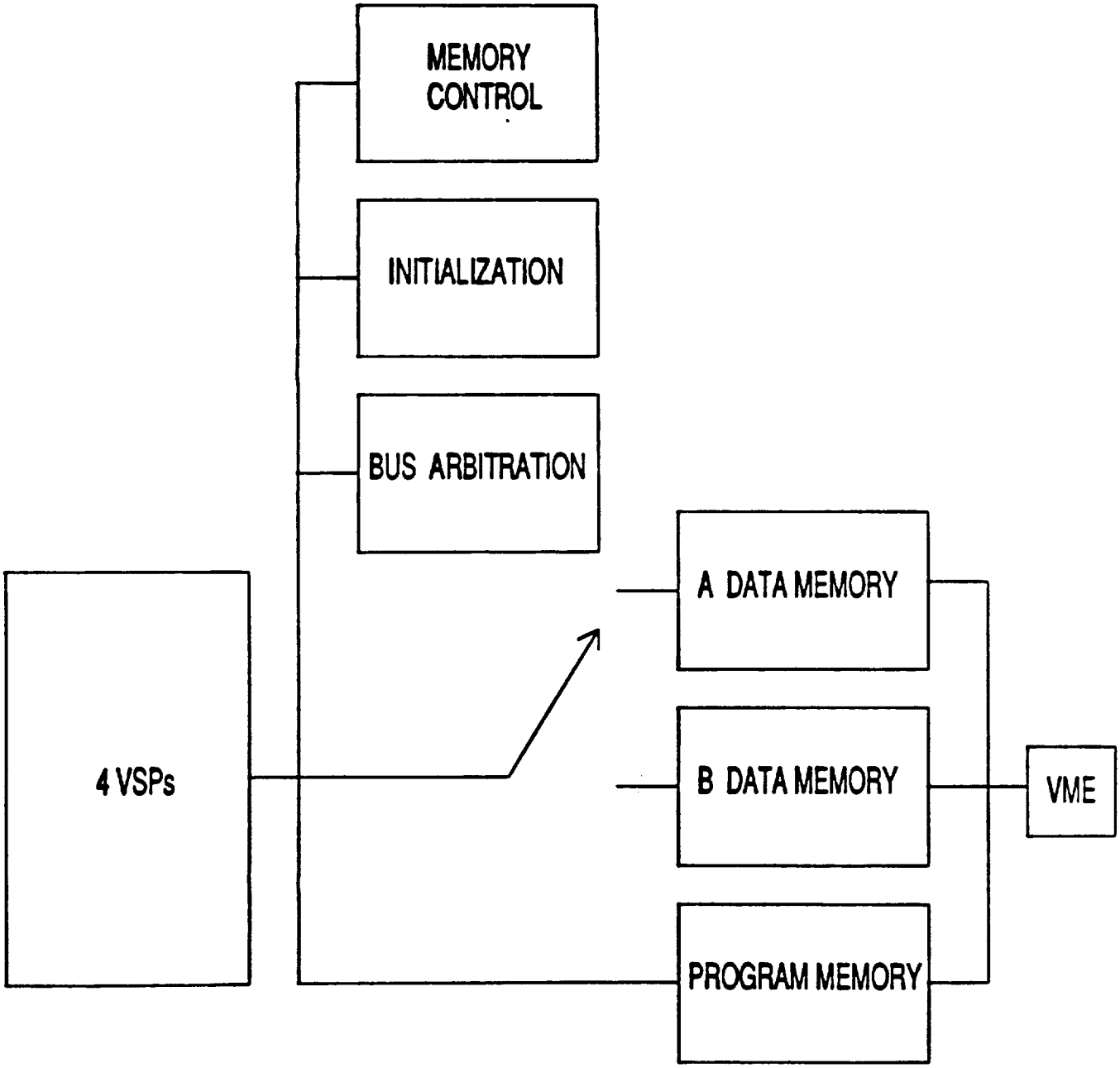


ZORAN

VSP/TMS32020 INTERFACE

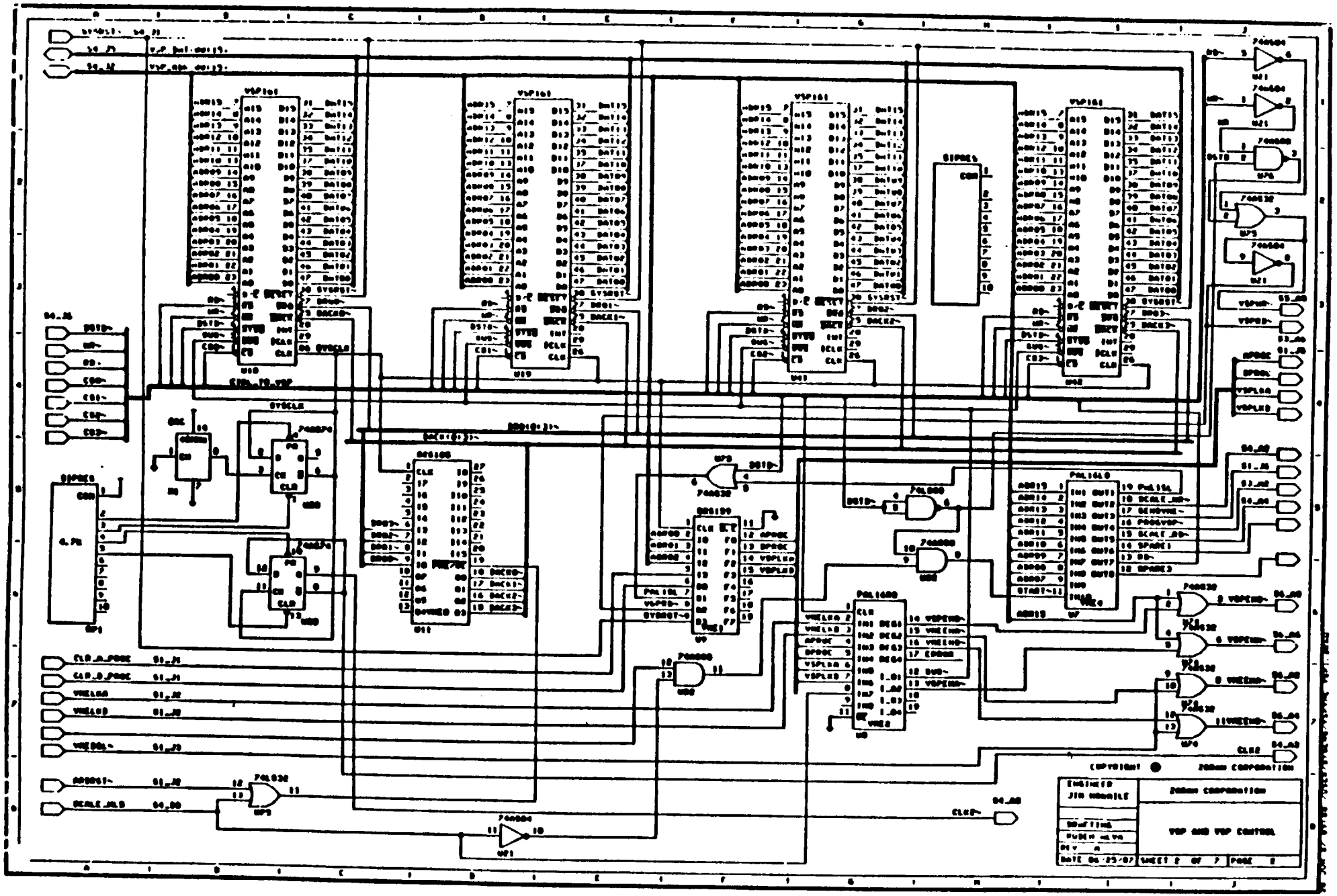


QUADVSP SYSTEM



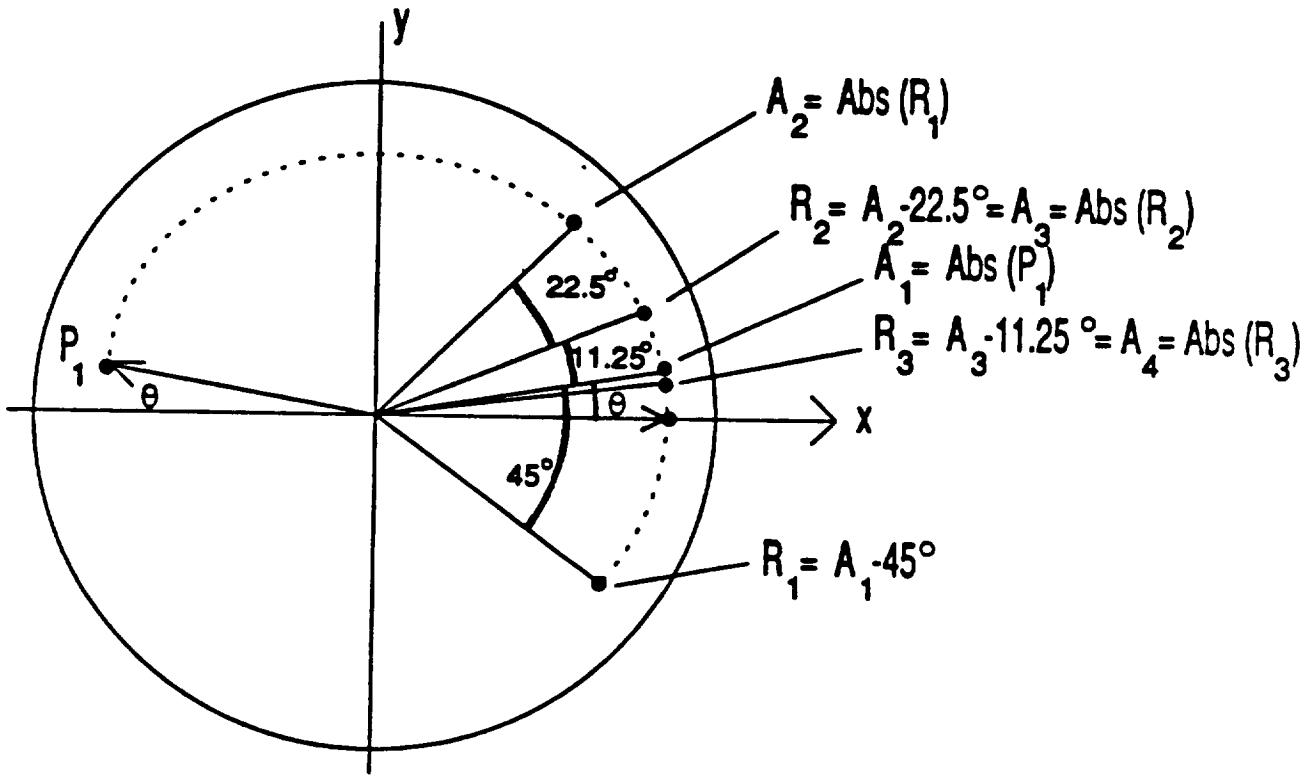
ZORAN

FOUR VSP/VME SYSTEM



SPECTRUM ANALYSIS

MAG (MAGNITUDE) EXAMPLE



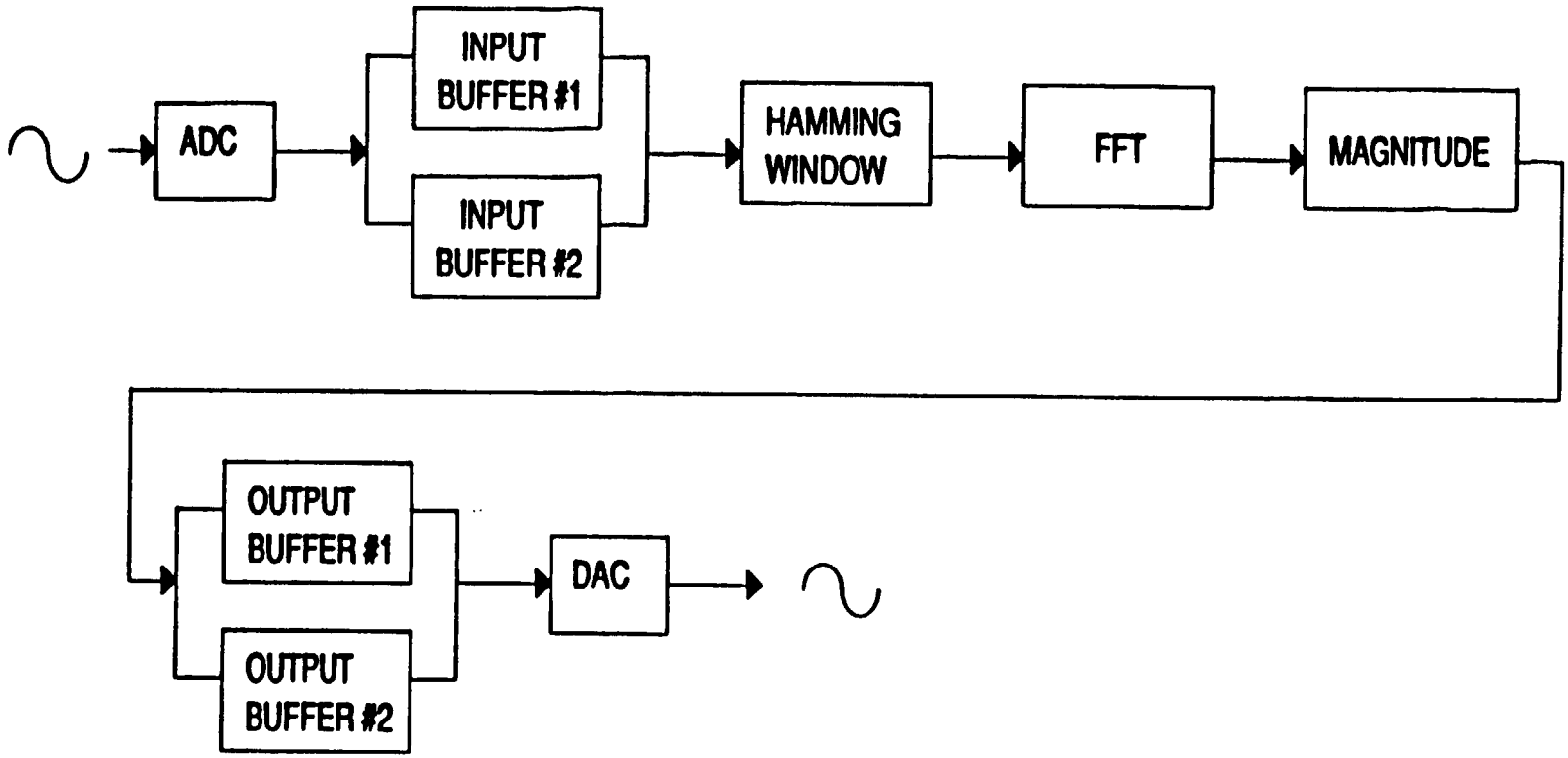
- Performs an iterative vector rotation of point P towards the +x axis

- Maximum Error:
  - 3 iterations < 1.9%
  - 5 iterations < 0.12%

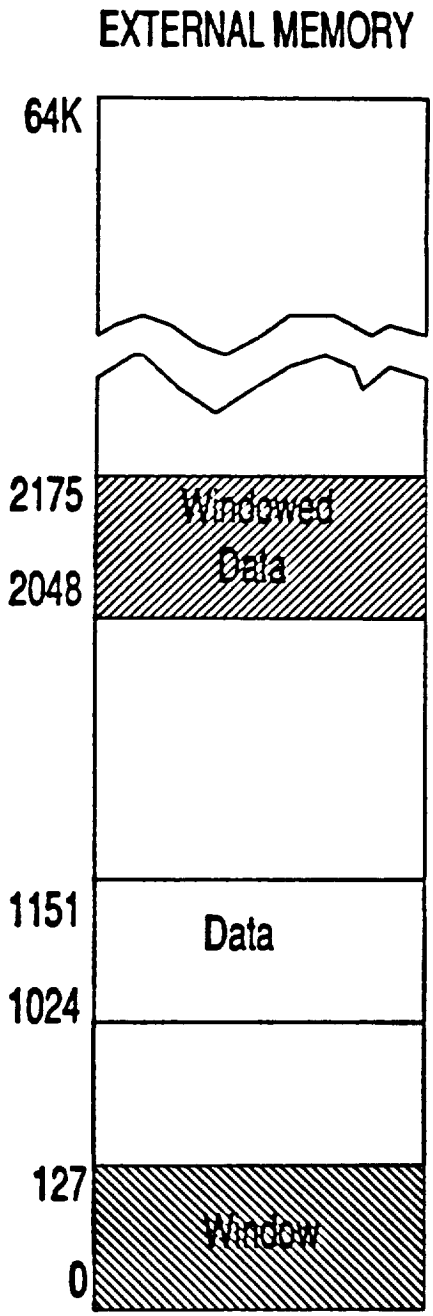
MAGNITUDE CODE

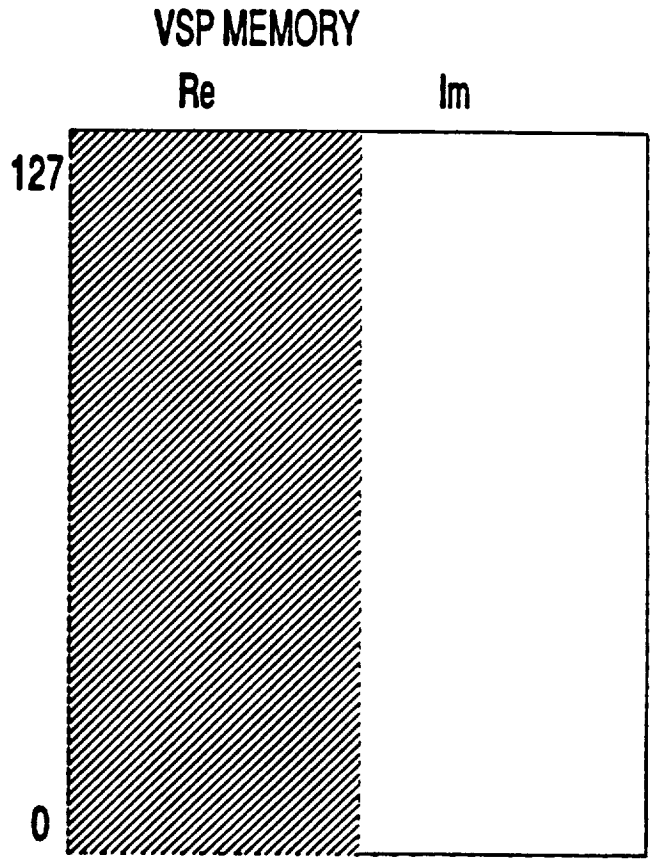
```
LDSM     NMPT:1, MBA:OneRAMSec;
         /* Get data into VSP RAM & reflect into 1st quadrant */
LD       NMPT:VEC_LEN, RS:0, MBA:W;
ABS      NMPT:VEC_LEN, ADF:3, RS:0;
         /* Rotate by 45 deg.  Note:  3150 = 10*(360-45) */
         /* Note:  Sh:1 (divide by 2) to prevent overflow */
MODLT    NMPT:VEC_LEN, ADF:3, RS:0, RIA:0, SH:1, RBA:3150;
ABS      NMPT:VEC_LEN, ADF:3, RS:0;
         /* Rotate by 22.5 deg. pp*/
MODLT    NMPT:VEC_LEN, ADF:3, RS:0, RIA:0, SH:0, RBA:3375;
ABS      NMPT:VEC_LEN, ADF:3, RS:0;
         /* Rotate by 11.25 deg. */
MODLT    NMPT:VEC_LEN, ADF:3, RS:0, RIA:0, SH:0, RBA:3488;
ST       NMPT:VEC_LEN, MDF:2, RS:0, MBA:REALBUF;
```

SPECTRAL ANALYSIS USING THE VSP



WINDOWING PRIMITIVE





LD NMPT:128, RS:0, MDF:2, MBA:0

MLTR NMPT:128, RS:0, ADF:2, MBA:1024

ST NMPT:128, RS:0, MDF:2, MBA:2048

NEW PRODUCTS

- **ZR34325 Vector Signal Processor (VSP)**
- ZR36010 Image Compression Processor (ICP) (see Imaging section)

VSP-325 SYSTEM APPLICATIONS

- Laboratory/Scientific Instruments
- Spectral Analysis
- Medical Imaging
- Radar/Sonar Processing
- Image Processing
- Array/Matrix Processing
- Graphics Processing
- Communications Systems
- Industrial Inspection
- IC/Board Testers

MATRIX MULTIPLY

[C] = [A]X[B]

```
LDR   SC:1, SAR:MATRIX_OUT;
LD    NMPT:64, EMA:b_mtx;
MTX   NMPT:64, ROW:8, REPEAT:1024, EMA:a_mtx;
```

EXAMPLE WITH:

- [A]=1024 rows by 8 columns
- [B]=8 rows by 8 columns
- [C]=1024 rows by 8 columns

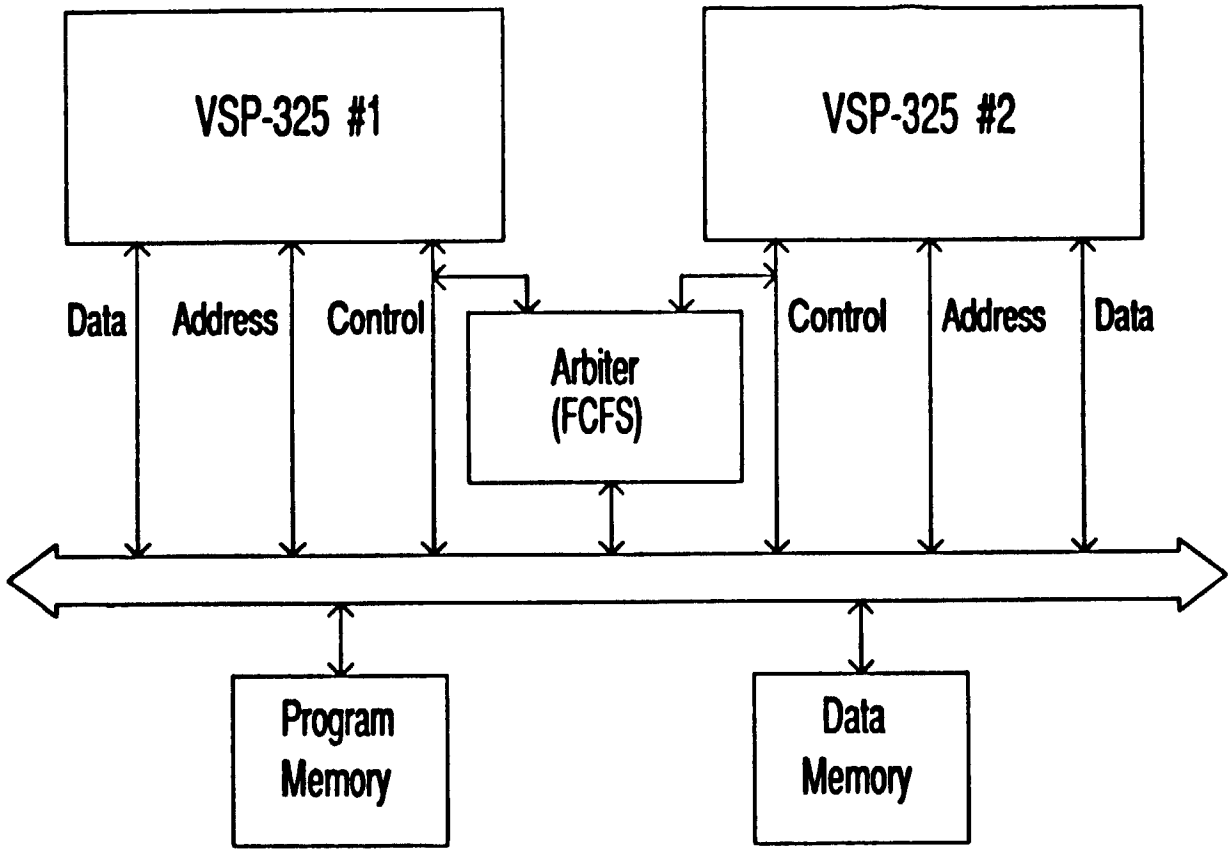
64-POINT WINDOWED POWER SPECTRUM WITH PEAK DETECTION

```
LDR     SAR:peak_out

LD      NMPT:64, IDF:2, ZF:1, RV:1, EMA:in_addr

MULT    NMPT:64, EMA:wind_addr;

FFT     NMBT:64, FPS:32, LPS:1, R:1, RBA:0;

MGSQ    NMPT:64;

MAX     NMPT:64, CO:0, SE:0;

ST      NMPT:64, ODF:2, EMA:out_addr;
```

1024-POINT COMPLEX FFT

```
(1)  LDR  MR:(XOR=1), LC:31 EMA:Scratch;
/* begin first wave of FFT processing */
(2)  LD   NMPT:32, RS:1, EMA:0;
(3)  FFT  NMBT:32, RS:1, FPS:16, LPS:1, RBA:0;
/* beginning of loop */
(4)  LD   NMPT:32, RS:0, UB:1, EMA:2;
(5)  FFT  NMBT:32, RS:0, FPS:16, LPS:1, RBA:0;
(6)  ST   NMPT:32, RS:1, UB:1, EMA:0;
(7)  LOOP     LENGTH:3;
/* end of loop */
(8)  ST   NMPT:32, RS:1, EMA:2;
/* end of first wave of FFT processing */
(9)  LDR  MR:(XOR=1), LC:31 EMA:Scratch;
/* begin second wave of FFT processing */
(10) LD   NMPT:32, RS:1, EMA:0;
(11) FFT  NMBT:32, RS:1, FPS:16, LPS:1, RBA:0;
/* beginning of loop */
(12) LD   NMPT:32, RS:0, UB:1, EMA:2;
(13) FFT  NMBT:32, RS:0, FPS:16, LPS:1, UR:1, RBA:(pi/32);
(14) ST   NMPT:32, RS:1, UB:1, EMA:0;
(15) LOOP     LENGTH:3;
/* end of loop */
(16) ST   NMPT:32, RS:1, EMA:2;
/* end of second wave of FFT processing */
```

PARALLEL VSP-325 SYSTEM CONFIGURATION



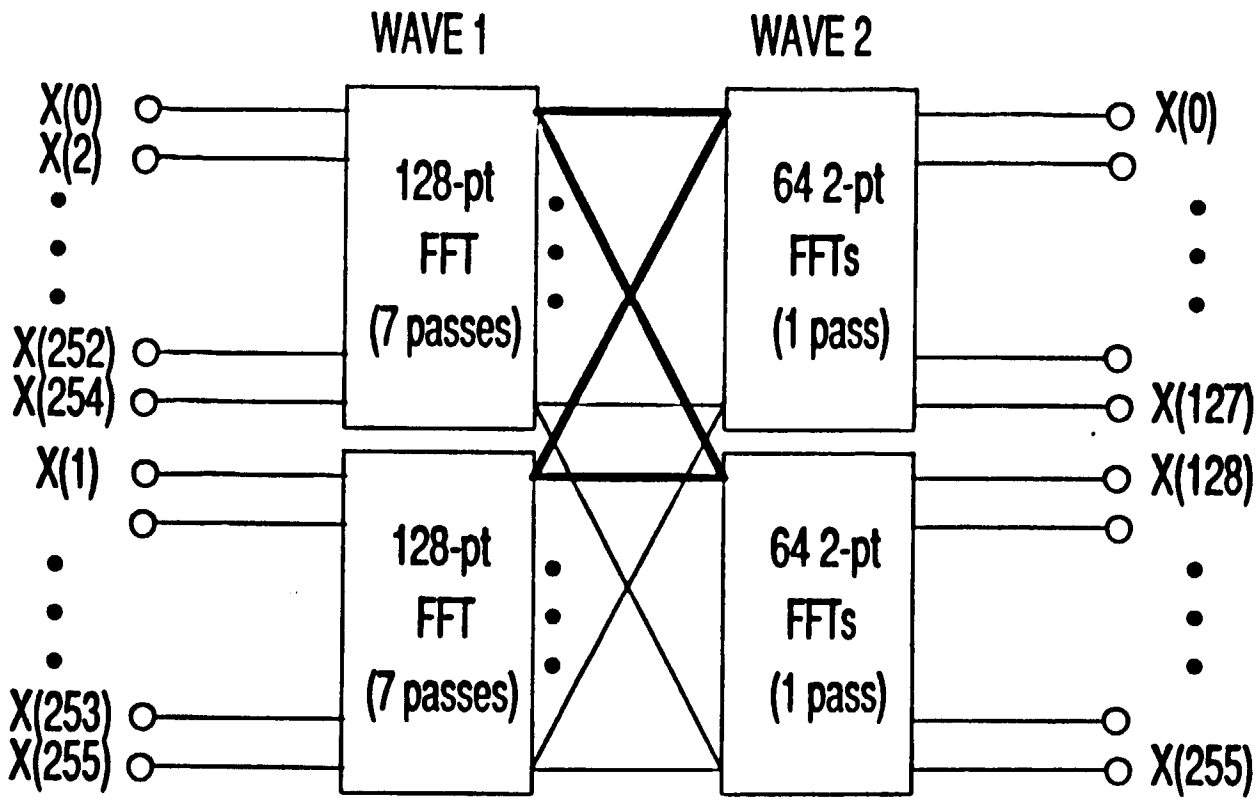
WHY IMAGE COMPRESSION?

- Reduce Data Storage
  - one color image (512 x 512 x 8 x 3) = 787 kilobytes
  - compressed 10 to 1 = 78.7 kilobytes
- Reduce Transmission Bandwidth
  - one color image (512 x 512 x 8 x 3) transmitted over a 64K bps channel takes 98 seconds
  - compressed 10 to 1, transmission takes 9.8 seconds

IMAGE COMPRESSION APPLICATIONS

- Electronic Publishing (Encyclopedias ...)
- Parts/Maintenance Catalogs
- Still-Frame Video Teleconferencing
- Electronic Cameras
- Medical (X-Rays, CT scanners)
- Military Satellite Reconnaissance

CONCEPTUAL 256-PT FFT DIAGRAM



* Each block consists of LD, FFT, ST

256-POINT FFT

```
DEFAULT RS:0, INTRP:0, EI:0, MDF:3, ZR:0, ZP:0;

DEFAULT FSIZ:128, AS:1, I:0, R:0;

LDSM   NMPT:1,    UP:0,     MD:1,                    MBA:16383,
LD     NMPT:128,  MBS:1,    MSS:2,     RV:0          MBA:IN;
FFT    NMBT:128,  FPS:64    LPS:1,     RBA:0;
ST     NMPT:128,  MBS:1,    MSS:2,     RV:0,         MBA:IN;
LD     NMPT:128,  MBS:1,    MSS:2,     RV:0,         MBA:IN+2;
FFT    NMBT:128,  FPS:64,   LPS:1,     RBA:0;
ST     NMPT:128,  MBS:1,    MSS:2,     RV:0,         MBA:IN+2;
LD     NMPT:128,  MBS:128,  MSS:128,   RV:0,         MBA:IN;
FFT    NMBT:128,  FPS:1,    LPS:1,     RBA:0;
ST     NMPT:128,  MBS:1,    MSS:2,     RV:1,         MBA:OUT;
LD     NMPT:128,  MBS:128,  MSS:128,   RV:0,         MBA:IN+258;
FFT    NMBT:128,  FPS:1,    LPS:1      RBA:14;
ST     NMPT:128,  MBS:1,    MSS:2,     RV:1,         MBA:OUT+2;
```

PROGRAM PERFORMANCE

NUMBER OF INSTRUCTIONS	NUMBER CLOCKS INSTRUCTION FETCH	EXECUTION TIME (CLOCKS)
13	78	6535

VSP-325 DEVELOPMENT TOOLS

- SOFTWARE SIMULATOR

  -- Signal processing library/HL language:
  FFTs, FIR, IIR Filters, Matrix Operations, etc.
  -- Allows algorithm development/evaluation
  -- Signal generation and plotting
  -- Parser/assembler

- HARDWARE DEVELOPMENT BOARD

- IN-CIRCUIT EMULATION CAPACITY

NEW PRODUCTS

- ZR34325 Vector Signal Processor (VSP)
- **ZR36010 Image Compression Processor (ICP) (see Imaging section)**

FIR FILTER PERFORMANCE

(METHOD 2)

Number of taps	4	8	16	32	64
ICLKs/pt	17	21	29	45	77
Throughput (kHz)	580	475	345	222	130

- 13 ICLKs overhead
  -- independent of number of taps
- Can use slow memory (<145 ns)

VSP BASICS

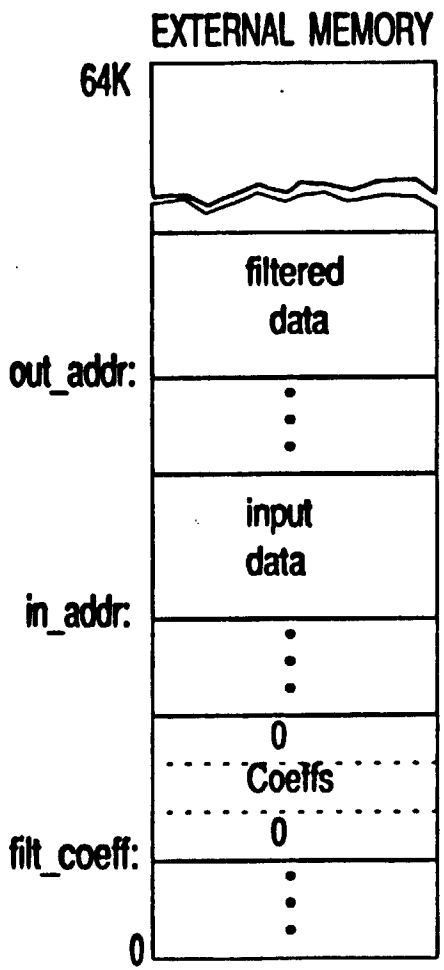
- Larger FFTs
- Time-Domain Convolution
- **Fast Convolution**
- Hardware Interface
- Multi-VSP Configuration

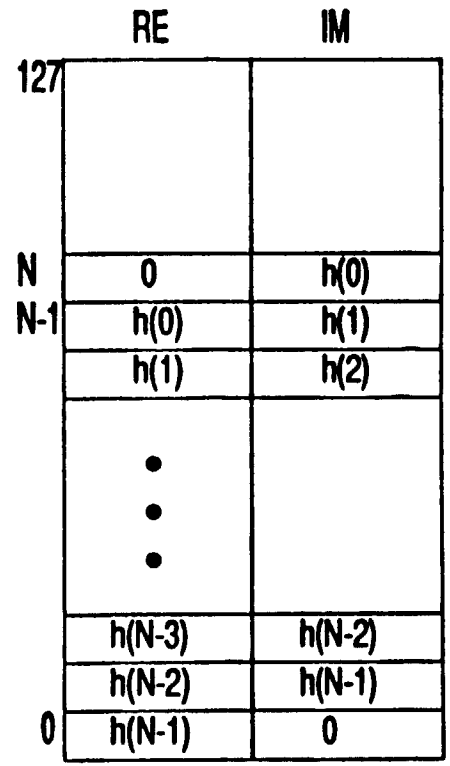
VSP BASICS

- Larger FFTs
- **Time-Domain Convolution**
- Fast Convolution
- Hardware Interface
- Multi-VSP Configuration

VSP-161 CONVOLUTION EXAMPLE

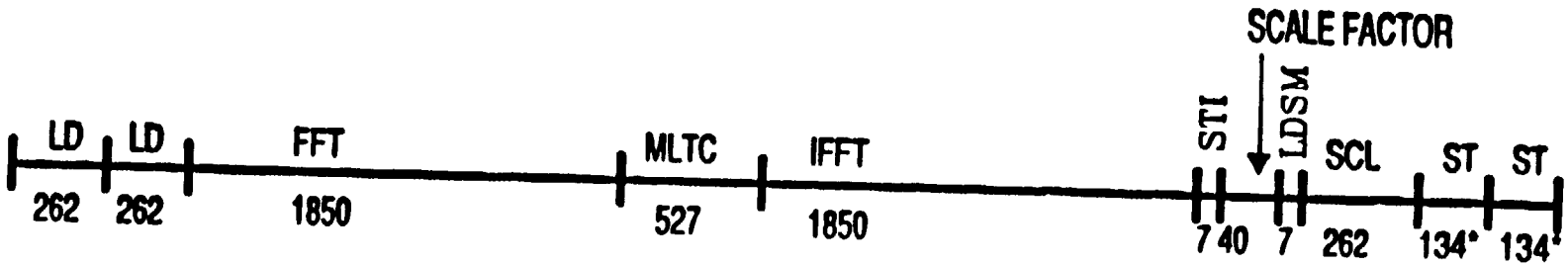
(METHOD 2)






```
LD NMPT:N+1, MDF:2, MBA: filt_coeff+1;
LD NMPT:N+1, MDF:1, MBA: filt_coeff;
for (k=0; k < num_samples; k++)
{MLTR NMPT:N+1, ADF:0, MBA: in_addr+k;
STI NMPT:4, STR:1, MBA: out_addr + 4k; }
```

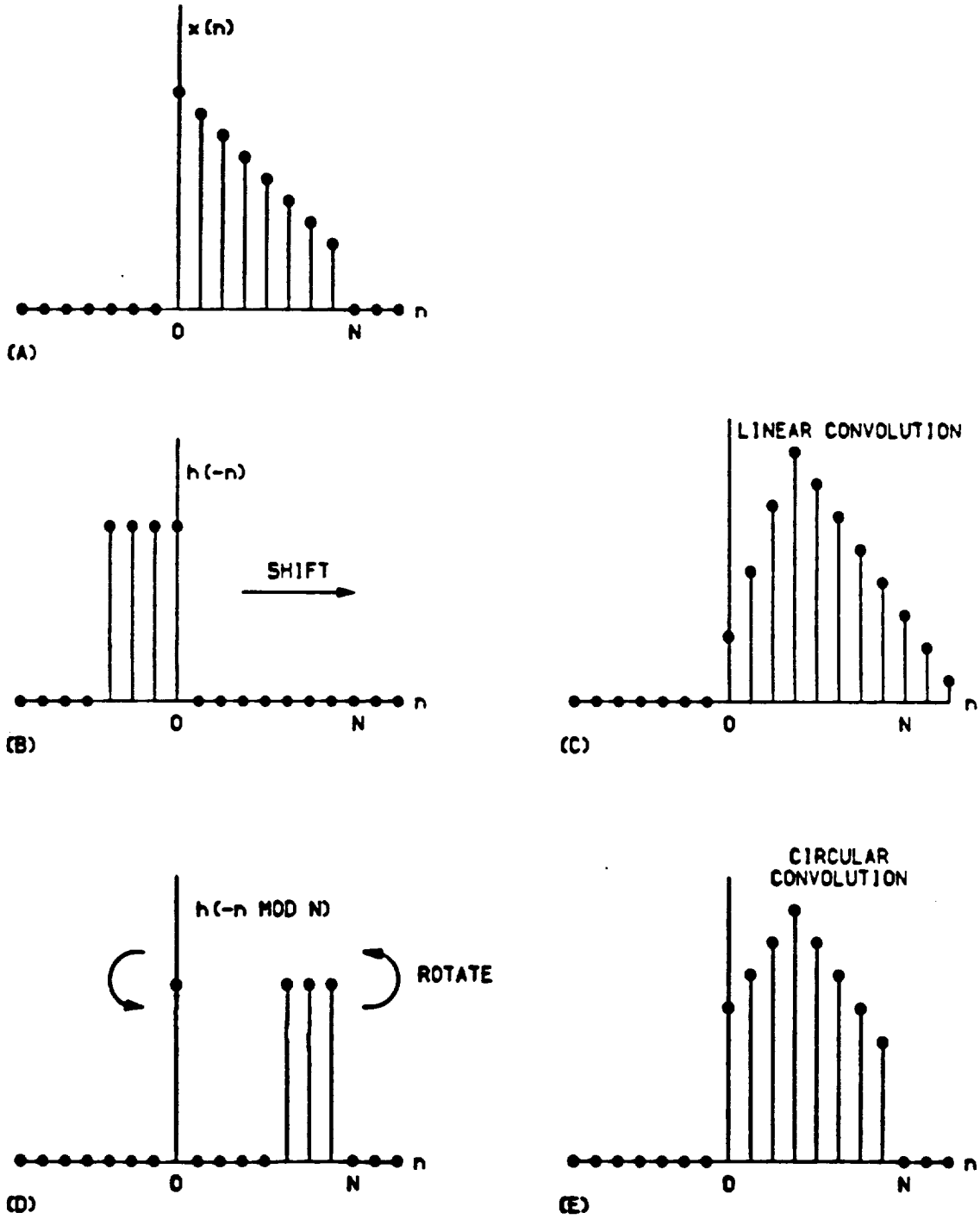
FFT-BASED FAST CONVOLUTION TIMING



- Total Execution Time < 5079 ICLKS OR < 508us
- Number of output points : 258-2M

*Actually 134-2(M-1)

LINEAR VS. CIRCULAR CONVOLUTION

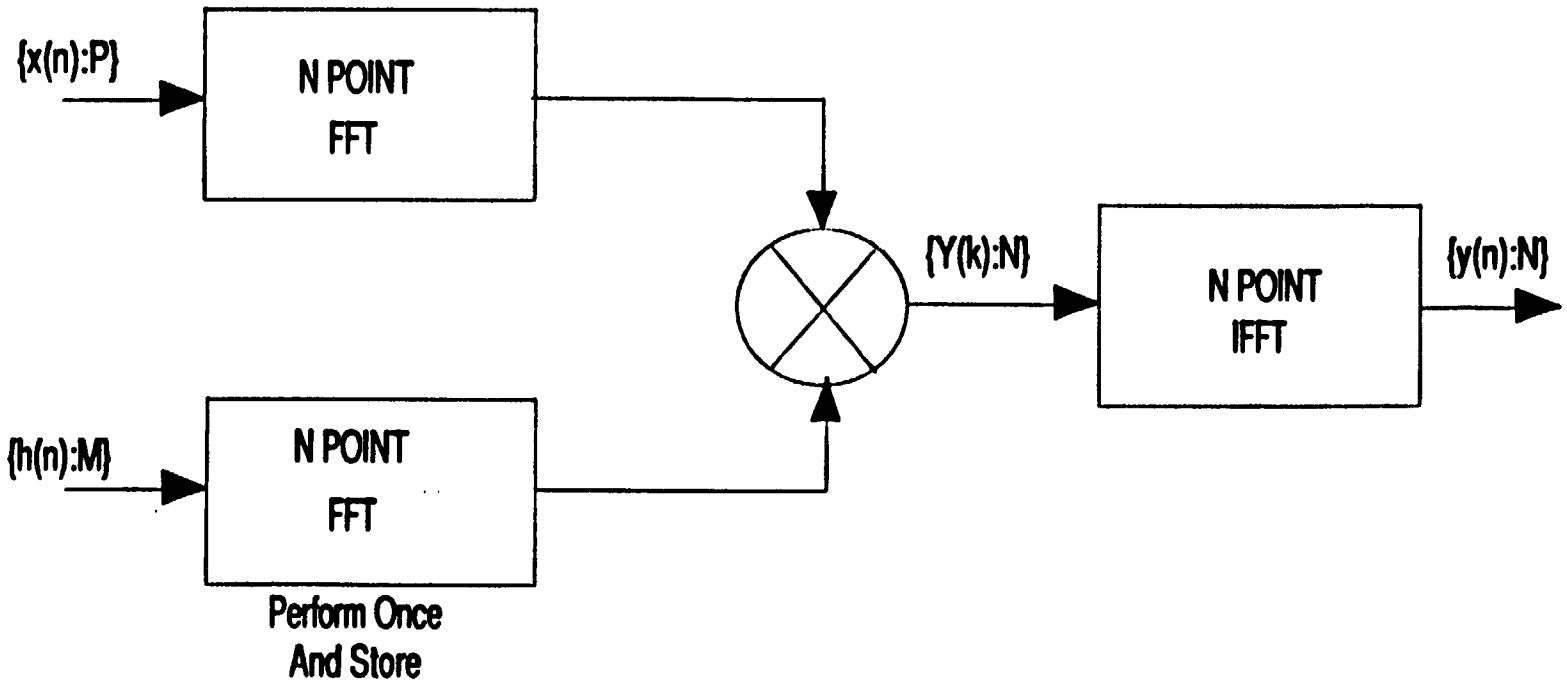


FAST CONVOLUTION

- Use of DFT $\Rightarrow$ Circular convolution
- To get linear convolution use:
  - -- Overlap & Add

    or
  - -- Overlap & Save (Discard)

FLOW DIAGRAM OF FFT BASED FAST CONVOLUTION



$N = M + P - 1$

TRANSFORM THEOREM

CONVOLUTION IN THE TIME DOMAIN IS EQUIVALENT

TO

MULTIPLICATION IN THE FREQUENCY DOMAIN

$$Y(n) = \sum_{k=0}^{N-1} x(n-k)\, h(k)$$

$$Y(n) = \mathrm{IFFT}\,[X(f)H(f)]$$

EXAMPLES OF PROCESSING IN THE FREQUENCY DOMAIN

Number of Clocks

256-Pts 1-D Filter	Time Domain	VSP CLOCKS Frequency Domain	FREQUENCY DOMAIN Speed Improvement
64 Taps	16384	10600	1.55
128 Taps	32768	14000	2.34

256 x 256 Image

2-D Filter	Spatial Domain	Frequency Domain	
64 x 64 Kernel	153M	7M	22.0

PARALLEL- VSP BENCHMARKS

1K-PT COMPLEX BLOCK-FLOATING FFT

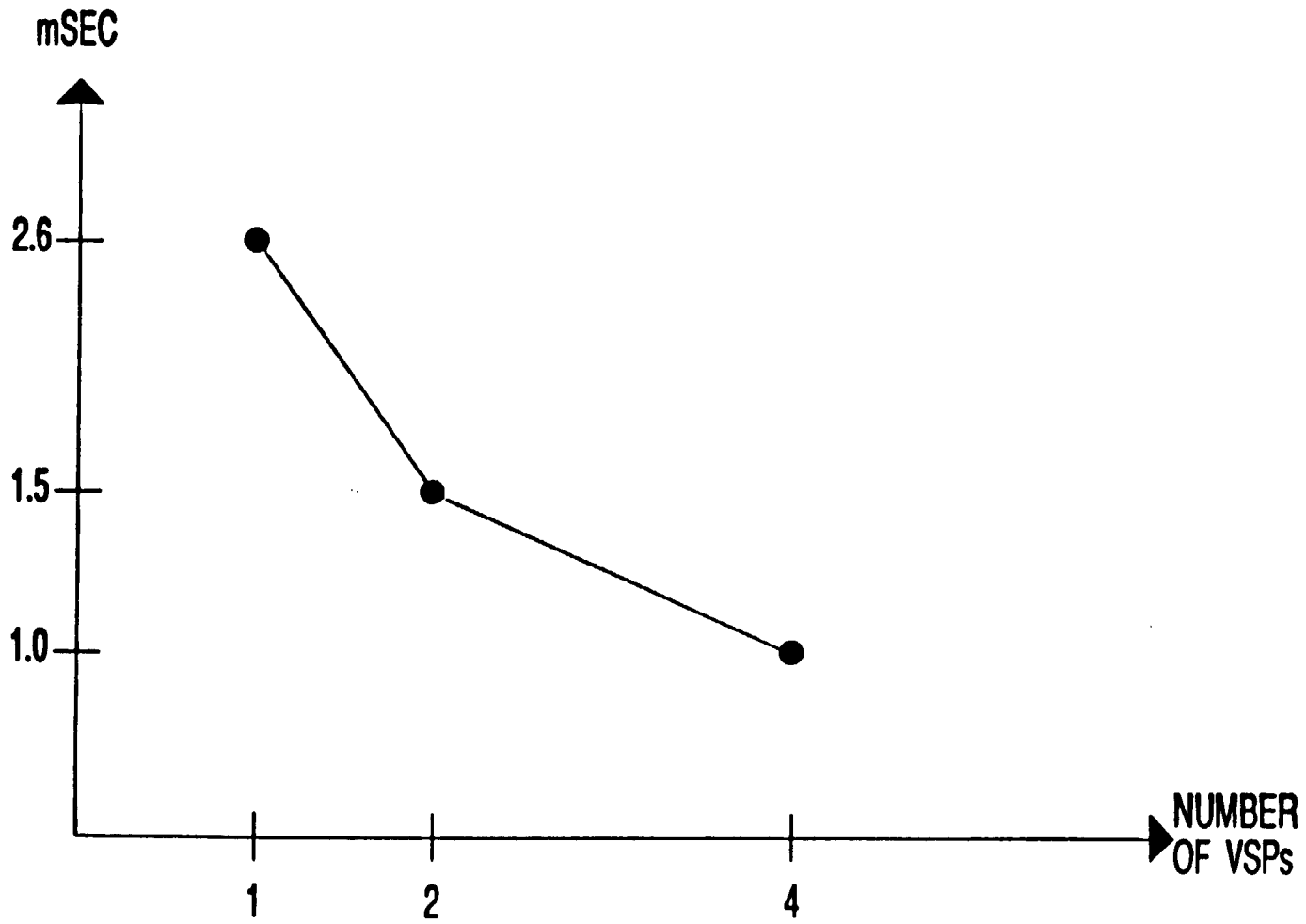


IMAGE COMPRESSION TECHNIQUES

- Binary Images (FAX, half-tone)
  - -- run-length encoding [error-free code]

- Monochrome and Color Images
  - -- image fidelity vs. compression ratio [error-inducing codes]
  - -- methods vary in quality and complexity
  - -- hybrids of several methods typically produce best quality

FILTER PERFORMANCE SUMMARY

Number of taps	4	8	16	32	64
TIME DOMAIN					
ICLKs/pt	17	21	29	45	77
Throughput (kHz)	580	475	345	222	130
FREQUENCY DOMAIN					
ICLKs/pt	21	21	23	26	39
Throughput (kHz)	470	470	430	380	250

- Can use slow memory (<145 ns)

VSP BASICS

- Larger FFTs
- Time-Domain Convolution
- Fast Convolution
- Hardware Interface
- **Multi-VSP Configuration**

ZORAN

IMAGE TRANSFORM COMPRESSION / DECOMPRESSION

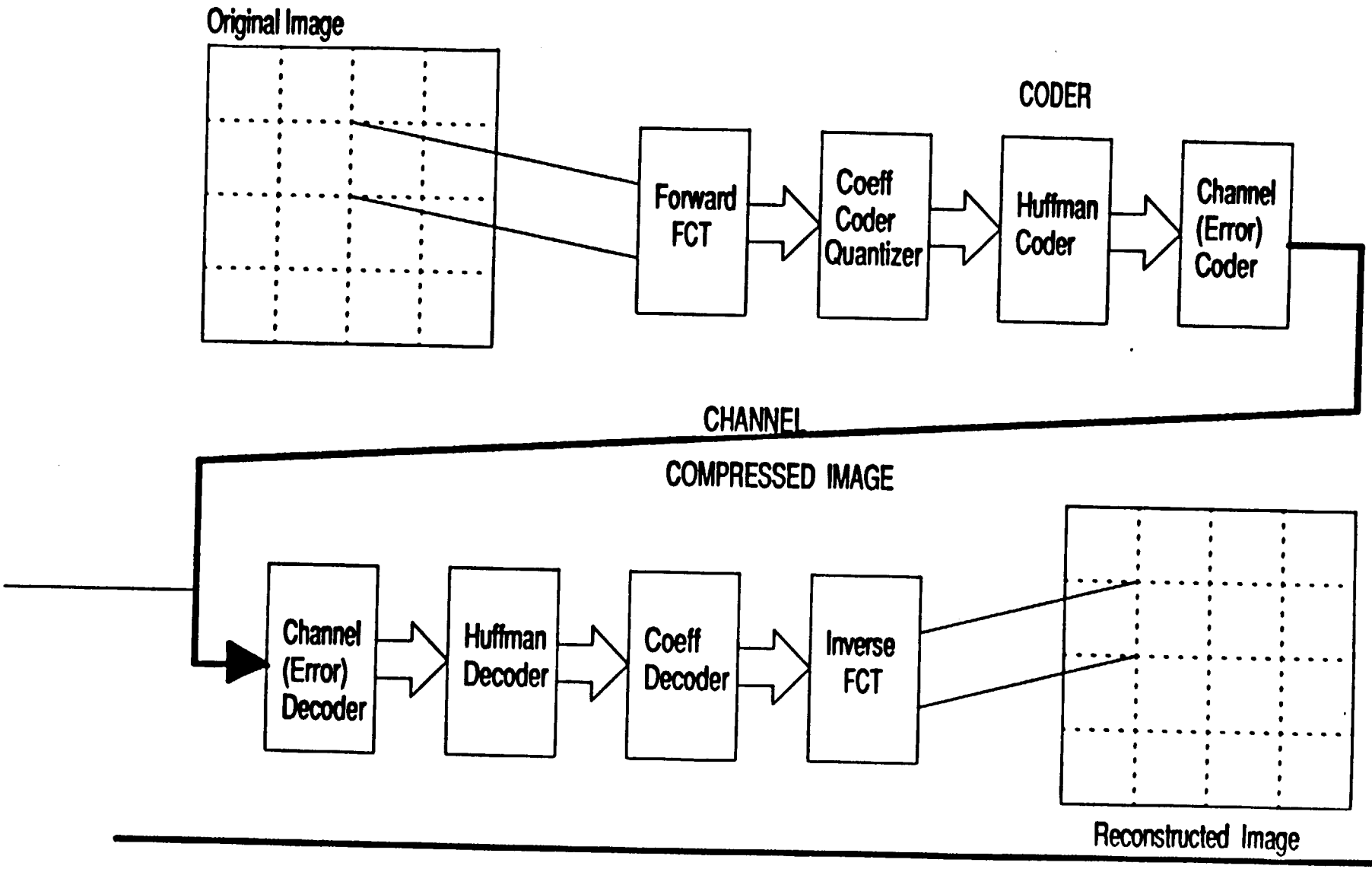


IMAGE COMPRESSION PROCESSOR
ZR36010

- Features

  - -- High-performance 16-bit integer processor
  - -- 32 complex words of internal RAM
  - -- FFT butterfly coefficient table for up to 16 x 16 FCT blocks
  - -- Concurrent I/O and arithmetic operations
  - -- Internal instruction queue
  - -- 16-bit demultiplexed data and address buses
  - -- Multiple processors yield linear throughput increase
  - -- Low-power double-layer metal CMOS (<300mW)
  - -- 48-pin plastic DIP package

COMPRESSION TECHNIQUES FOR MONOCHROME AND COLOR IMAGES

(1) PCM: basic digitization technique { 5 to 8 bits/pixel }

(2) DPCM: code PCM pixel differences, predictive { 1.5 to 3 bits/pixel }

(3) Interpolative: throw away pixels, estimate on reconstruction { 1 to 2 bits/pixel }

(4) Transform (zonal): change image basis functions to compact energy into few coefficients { 0.5 to 2 bits/pixel }

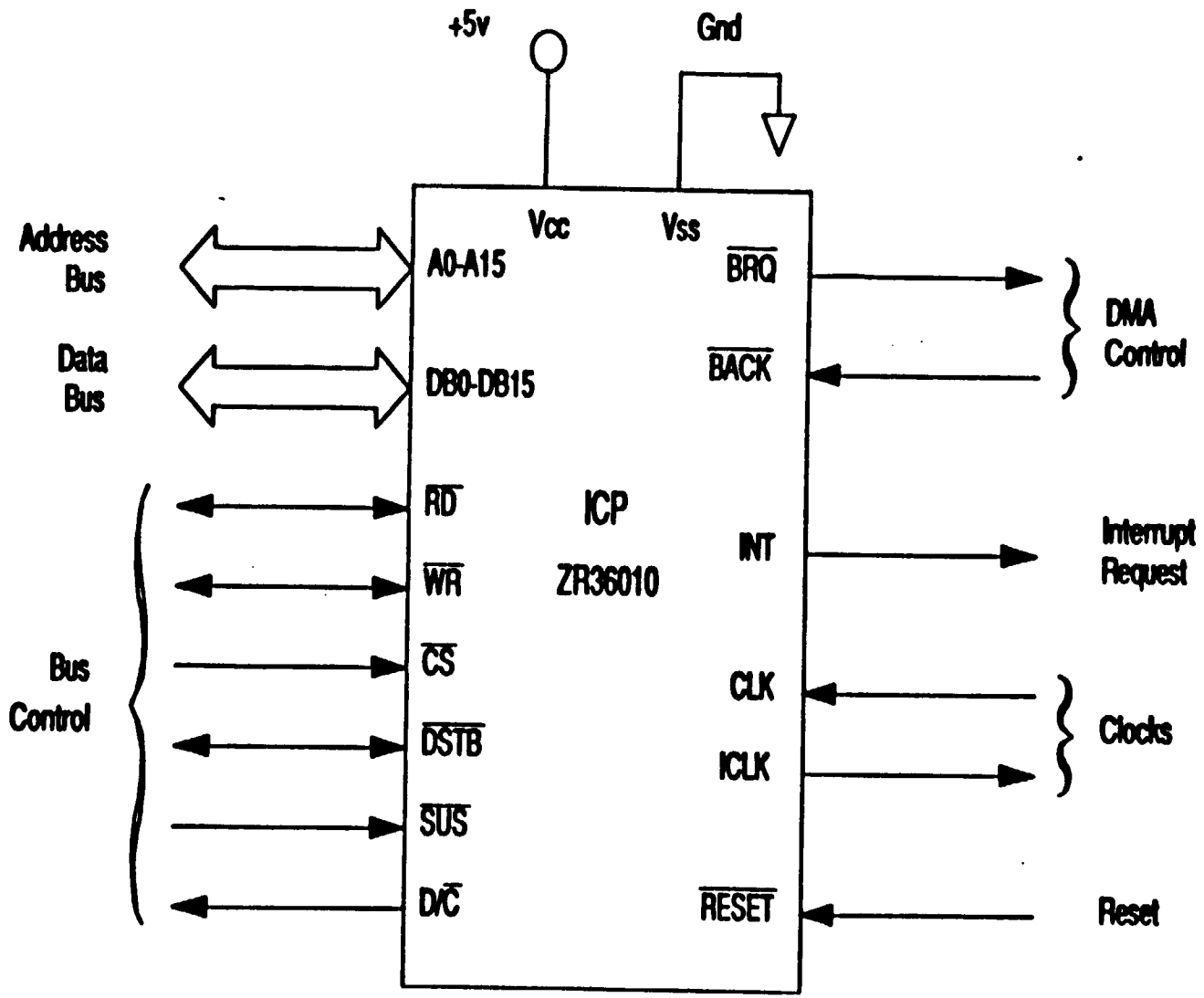
(5) Vector Quantization: build code book based on vector centroids { 1 to 3 bits/pixel }

(6) Hybrid Transform-DPCM: code difference of transform coefficients { 0.2 to 1.0 bits/pixel }

TRANSFORM CODING

- Advantages
  - Highest quality for low bit rates
  - Robust to channel errors
  - Information compaction (images can be 'browsed')
- Disadvantage
  - Computation intensive

Image Compression Processor - ZR36010



ZR36010 IMAGE COMPRESSION PROCESSOR PERFORMANCE

Sub-Block FCT Time

Sub-Block Size	
8 x 8	16 x 16
200 µs	750 µs

Single Frame FCT Time

	Sub-Block Size	
Frame Size	8 x 8	16 x 16
256 x 256	205 ms	192 ms
512 x 512	819 ms	768 ms
1024 x 1024	3277 ms	3072 ms

STILL-FRAME IMAGE DECOMPRESSION SYSTEM

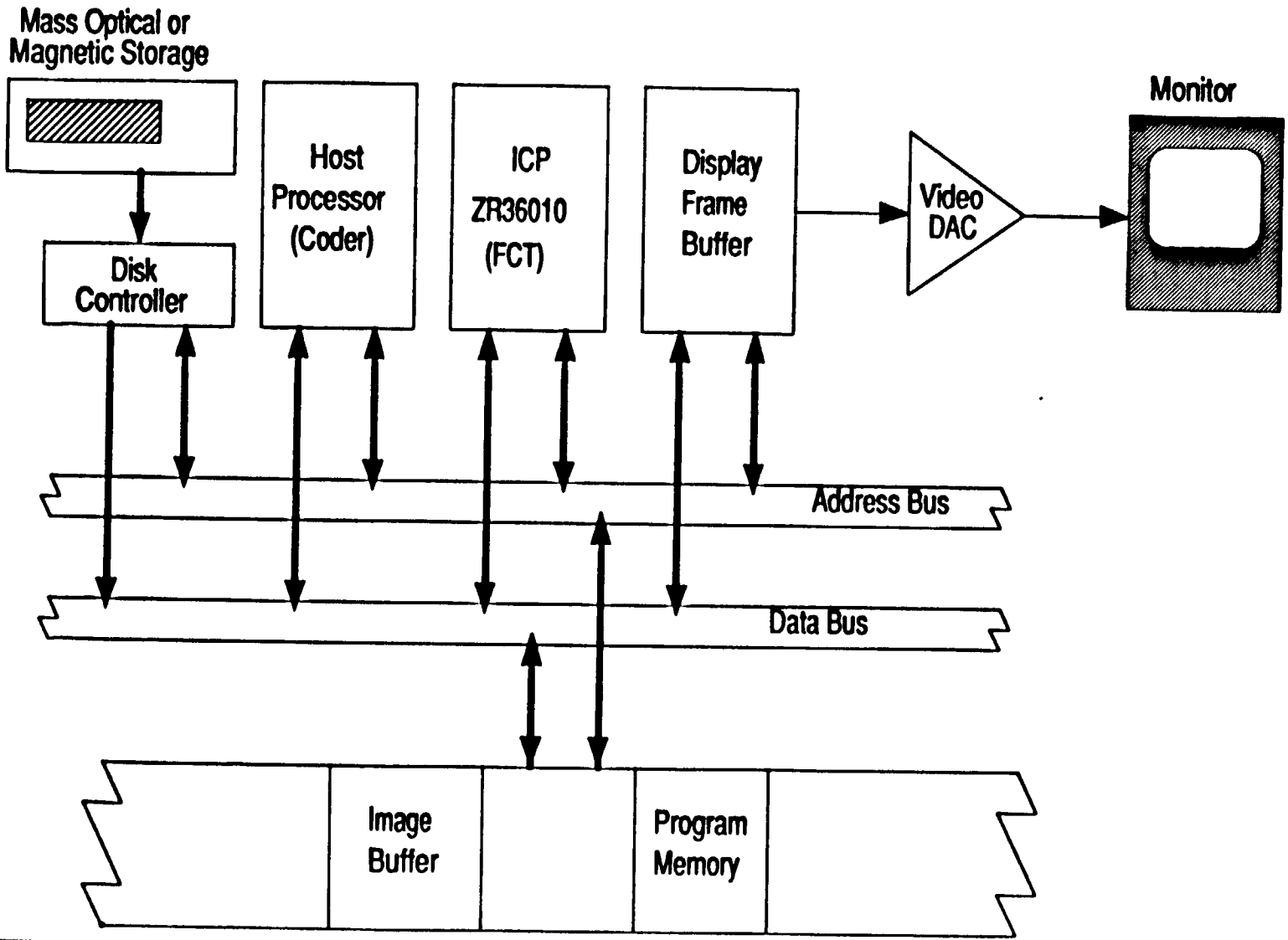


IMAGE COMPRESSION PROCESSOR
ZR36010

Hardware and Software Tools

- Hardware Evaluation Board
- Full-Speed Algorithm Verification/Development
- Fully-Integrated Debugger
- 16K Words of Data/Program Memory
- Hardware and Software Breakpoints
- Single-Step Mode
- IBM-PC/AT Host
- ICP Assembler

PRECEDING PAGE BLANK NOT FILMED

IMAGE COMPRESSION PROCESSOR
ZR36010

FCT Software Library

- Modified-Makhoul's FCT Algorithm
- Improved FFT-based FCT Algorithm
- Forward and Inverse FCT Source Programs for:
  - -- 8 x 8 blocks
  - -- 16 x 16 blocks
  - -- 32 x 32* blocks

* 32 x 32 FCT programs will only work with VSP (ZR34161)

VSP-325 KEY ARITHMETIC INSTRUCTIONS

- IIR (INFINITE IMPULSE RESPONSE)
  - -- Each Instruction:
    - * Cascades up to 16 second order sections
    - * Allows up to 1024 input samples
- FIR (FINITE IMPULSE RESPONSE)
  - -- Each Instruction:
    - * Uses up to 64 taps
    - * Allows up to 1024 samples
    - * Decimates by up to 64

32ND ORDER IIR FILTER

```
LDR     SC:1, SAR:filt_out

LD      NMPT:64, EMA:filt_coeff;

IIR     NMPT:64, REPEAT:1024, EMA:filt_in;
```

- Coefficients loaded to internal memory
- Filters 1024 input samples beginning at "filt_in"
- Stores results to "filt_out"

VSP-325 KEY ARITHMETIC INSTRUCTIONS

- FFT (FAST FOURIER TRANSORM)
  - -- Up to 64-pt: one instruction
  - -- Up to 1K-pt: internal CLUT (16 lines)
  - -- Up to 1M-pt: direct addressing (33 lines)
  - -- Up to 1K x 1K 2-D: direct addressing
  - -- More than one FFT per instruction when <64 points
  - -- 'Pruned' or 'Sparse' FFTs
  - -- Real/Complex FFTs

- DFT (DISCRETE FOURIER TRANSFORM)
  - -- Direct form implementation
  - -- Up to 1024-pt in one instruction

64-PT POWER SPECTRUM

LD	NMPT:64, IDF:3, EMA:in_addr;
FFT	NMBT:64, FPS:32, LPS:1, RBA:0;
MGSQ	NMPT:64;
ST	NMPT:64, ODF:3, RV:1, EMA:out_addr;

VSP-325 KEY ARITHMETIC INSTRUCTIONS

- FFT (Fast Fourier Transform)
- DFT (Discrete Fourier Transform)
- MAG (Magnitude)
- IIR (Infinite Impulse-Response Digital Filter)
- FIR (Finite Impulse-Response Digital Filter)
- POLY (Polynomial Expansion)
- MTX (Matrix Multiply)
- DEMO (Demodulate)
- MULT (Vector Multiply)
- ADD (Vector Add)
- SUB (Vector Subtract)

VSP-325 KEY INSTRUCTIONS

- LD (Vector Load)
- ST (Vector Store)
- MAX (Vector Maximum)
- THR (Vector Threshold)
- INTFP (Integer-to-Floating-Point Conversion)
- FPINT (Floating-Point-to-Integer Conversion)
- JMP (Unconditional Jump)
- JMPC (Conditional Jump)
- CALL (Call Subroutine)
- RET (Return from Subroutine)
- LOOP (Program Loop)

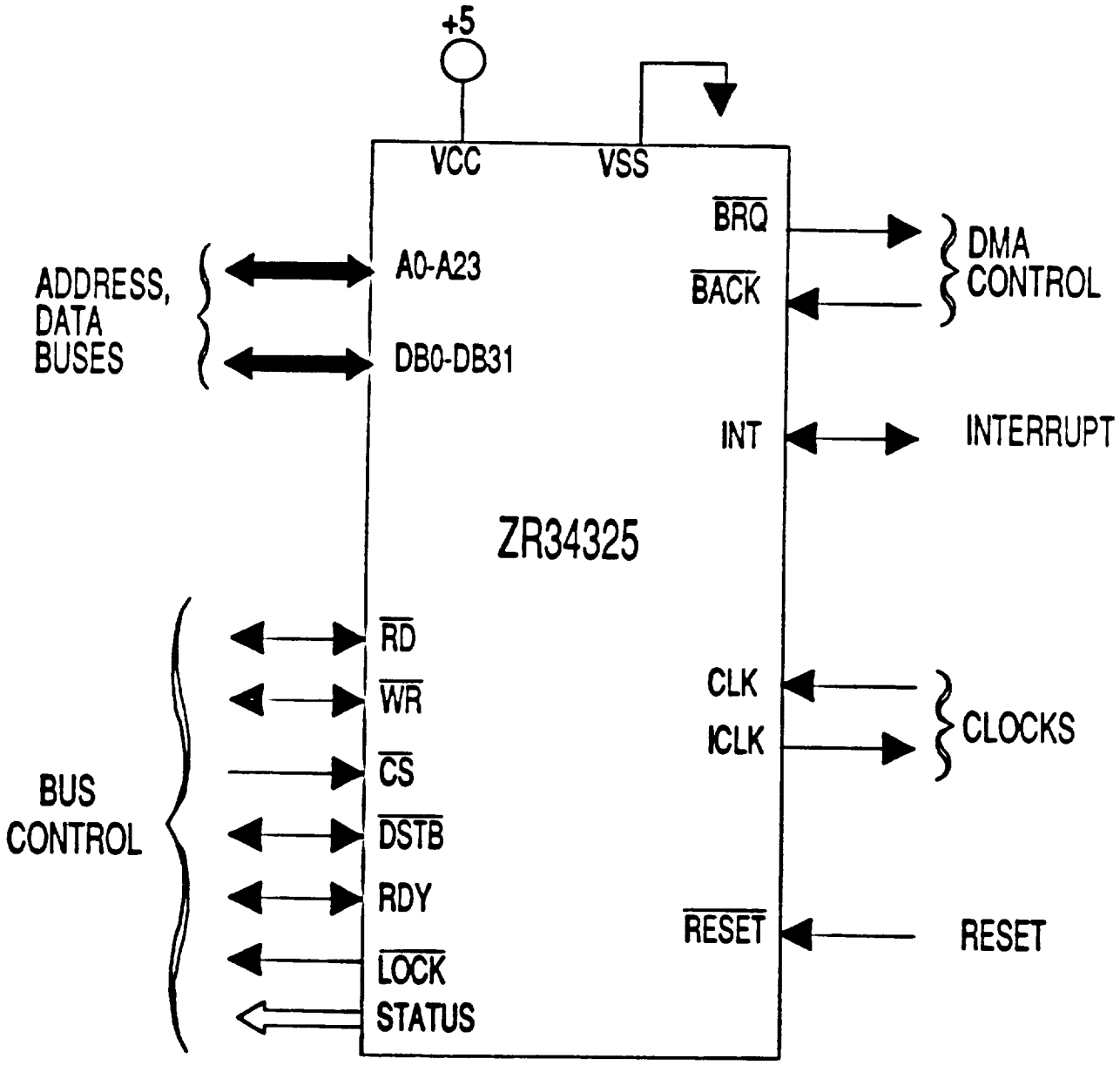
VSP-325 INTERNAL MEMORY

- RAM
  - -- 64 complex words organized as either
    - * 64 complex words
    - * 2 x 32 complex words
- COEFFICIENT LOOK-UP TABLE (CLUT)
  - -- 1024 complex 32-bit fl. pt. coefficients for:
    - * FFTs
    - * Modulation/Demodulation
- INSTRUCTION FIFO
  - -- Stores 4 instructions internally
  - -- Allows internal program looping

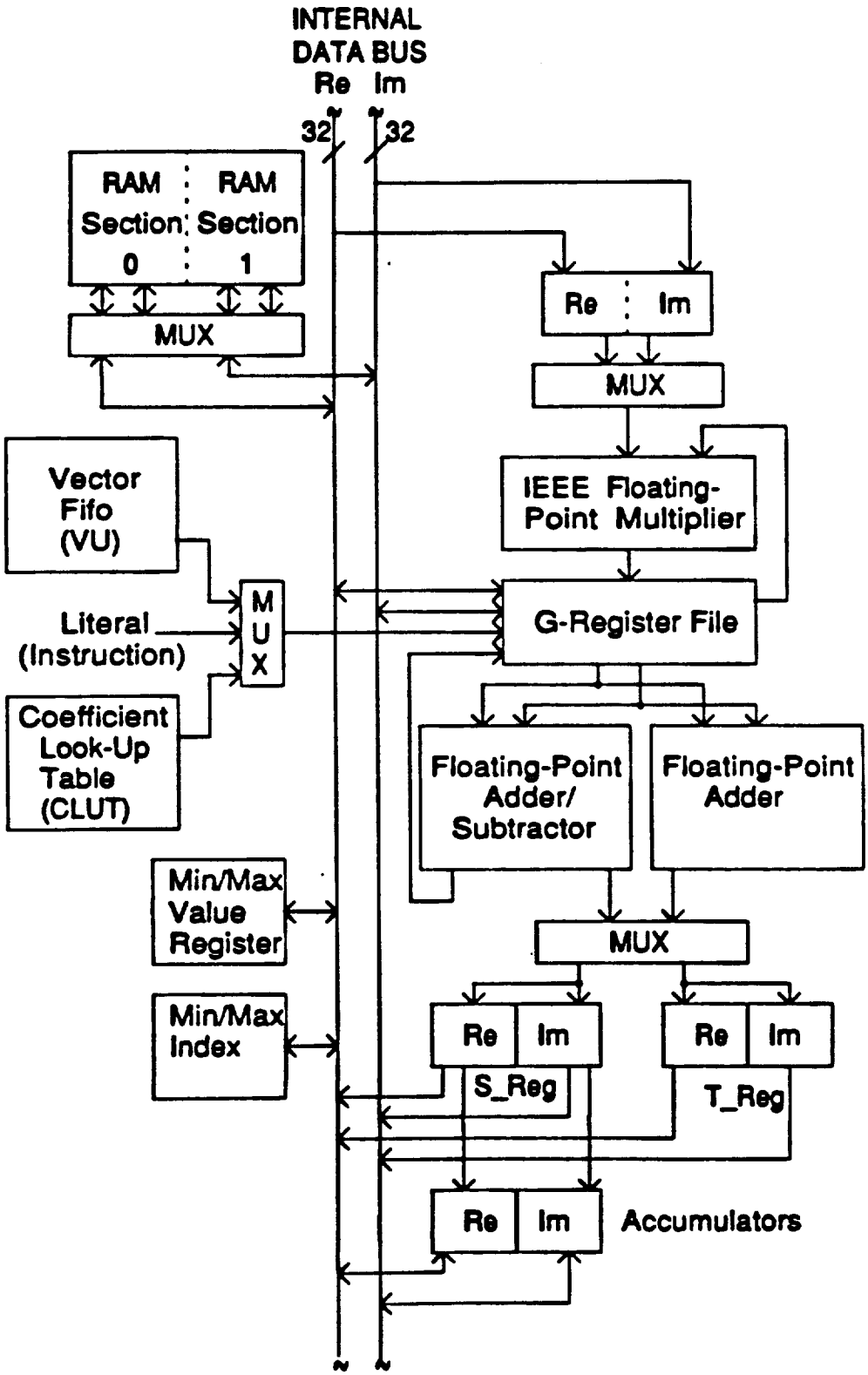
VSP-325 INSTRUCTION SET

- 52 VECTOR-ORIENTED INSTRUCTIONS
- FOUR CATEGORIES
  - Data movement
  - Registers
  - Arithmetic
  - Control
- UP TO 64K POINTS PER INSTRUCTION WITH 'REPEAT'
- INSTRUCTION LOOPING
  - 16 lines for 1K-pt complex FFT

VSP-325 LOGICAL PINOUT
84-PIN PGA



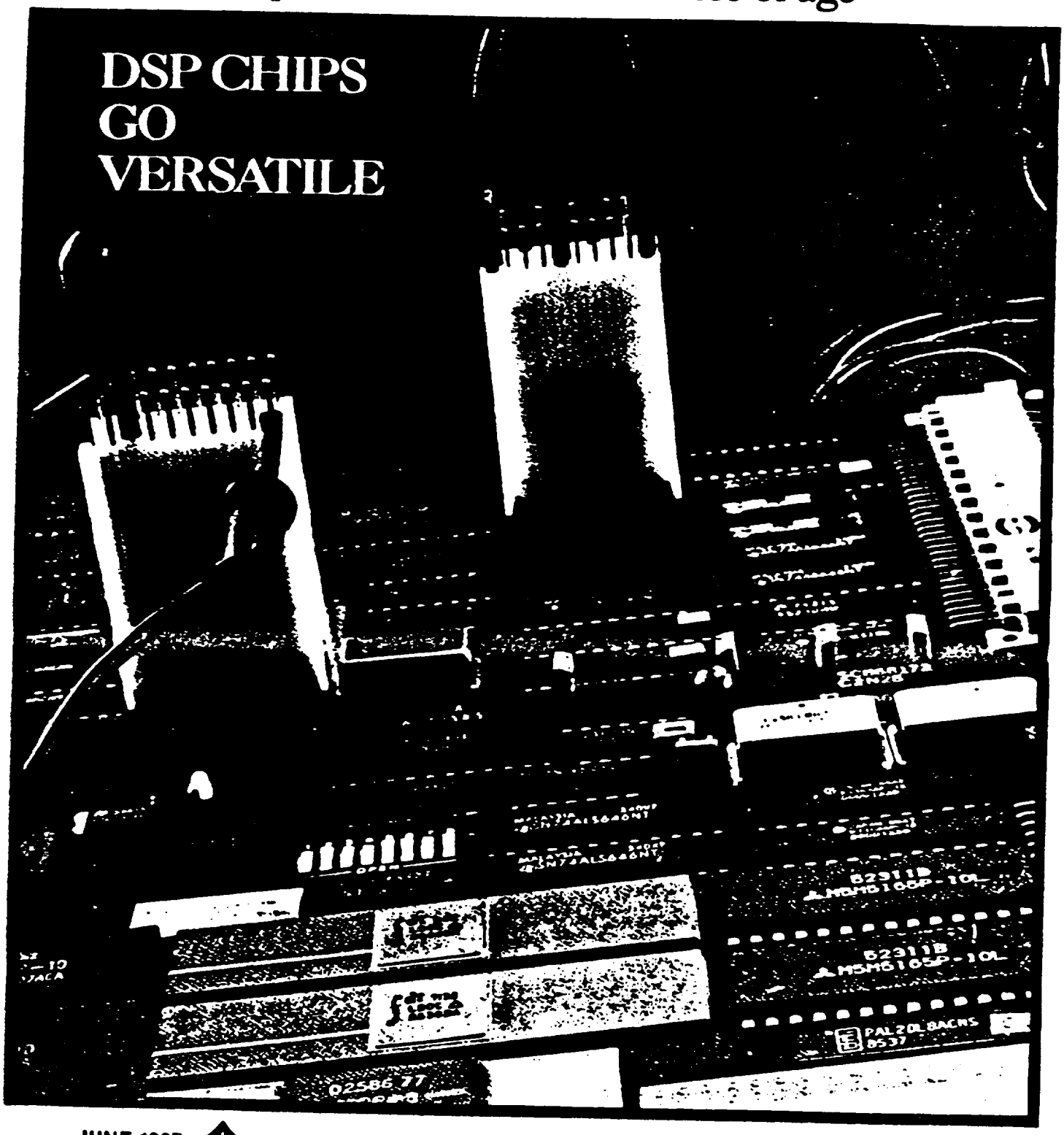
VSP-325 EXECUTION UNIT



IEEE
SPECTRUM

New oscilloscopes: myths and pitfalls
Human factors in advanced military systems
A parallel architecture comes of age

DSP CHIPS GO VERSATILE



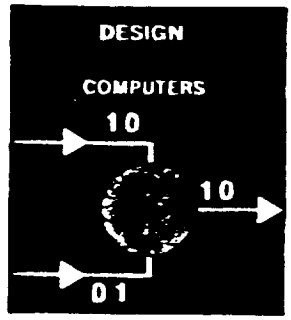
JUNE 1987 THE INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS, INC.

ORIGINAL PAGE IS OF POOR QUALITY

PRECEDING PAGE BLANK NOT FILMED

The versatility of digital signal processing chips

Uses for these architecturally modified microprocessors go all the way from simple digital filtering to echo canceling



As more and more long-distance telephone calls are routed through satellites, echoes in conversations are becoming particularly troublesome. Conventional echo suppressors do not work very well on satellite circuits, which have long propagation delays. What is needed is a sophisticated, adaptive approach to echo cancellation. But until recently such adaptive filters were too large and too expensive for widespread use. Each occupied a cabinet full of circuitry and cost tens of thousands of dollars.

Today, however, thanks to a special type of microprocessor that executes digital signal processing (DSP), an adaptive echo suppressor can fit onto one small circuit board.

Eliminating echoes is just one of many applications for DSP circuits. They are proving their worth also in processing data in speech recognition, radar, medical imaging, and speech compression, to cite a few areas.

What exactly is a DSP chip? Basically it is a microprocessor whose architecture is optimized to process sampled data at high rates. It performs such operations as accumulating the sum of multiple products much faster than an ordinary microprocessor can. Its architecture is designed to exploit the repetitive nature of signal processing by pipelining the data flow for extra speed.

Today the engineer who wants to take advantage of DSP chips can choose from an ever-growing variety of devices [see table]. But the choice requires care. The processor's throughput, address space, arithmetic precision, and benchmark performance must be considered. Engineering support and the quality of hardware and software development tools available for the chip are also important.

Each step in the evaluation narrows the engineer's choice of DSP chips. In some cases only one choice may remain. If there are several, the engineer may turn to the oldest criterion of differentiation: overall system cost.

As a starting point for selection, DSP circuits can be grouped by function: general-purpose chips and those that are application-specific. The application-specific chips are designed to perform one function more accurately, faster, or more cost-effectively than their general-purpose counterparts. Typical examples are digital filters and Fourier transform chips. Some application-specific chips are programmable, but only within the confines of the chip's function; the coefficients of a filter, for example, can be programmed.

DSP single-chip circuits can also be categorized by precision and by arithmetic types. Fixed-point DSP chips are available with 16, 24, and 32 bits of precision. DSP chips have recently been introduced with full floating-point arithmetic capabilities. These newcomers—operating at instruction cycle times that match those of their fixed-point counterparts, but with greater precision—allow the systems designer to jump directly from a software floating-point simulation of the system into a DSP implementation. In this manner the often time-consuming and difficult step of conversion to fixed-point arithmetic is bypassed. Floating-point capabilities open up a host of new applications, mostly in infinite-impulse response filters, where even fixed-point processing, combined with judicious scaling, cannot attain the required precision.

A key element on a typical DSP circuit [Fig. 1] is a fast array multiplier and accumulator that allows a multiply-accumulate operation to be executed in a single clock cycle, as opposed to about 25 clock cycles in a typical microprocessor. The chips also typically carry small, but very fast, caches and memories to accommodate program and data pipelines.

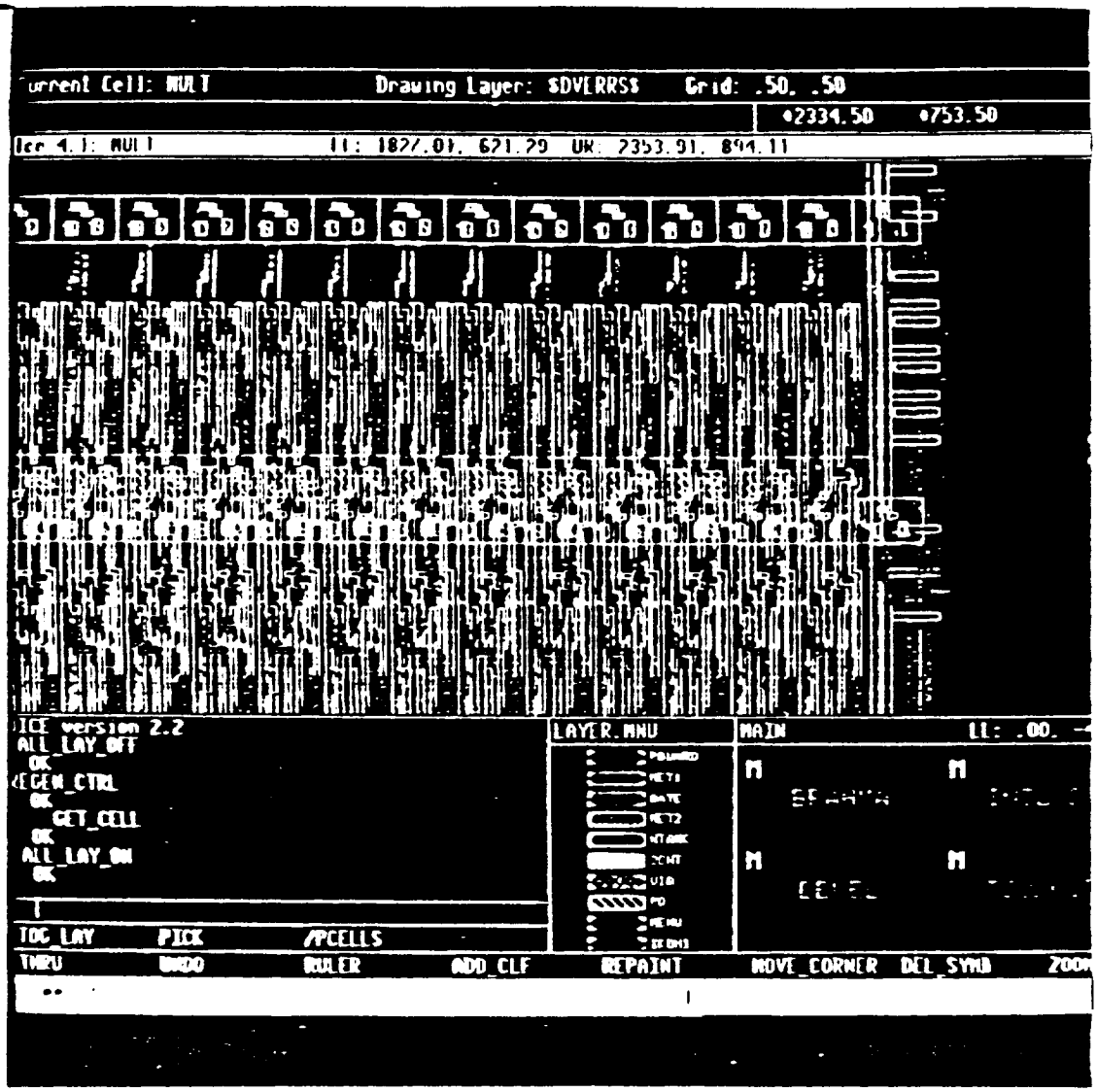
In some applications, speed is less important than the accuracy in representing data and coefficients that characterize a digital filter. With recursive, or infinite-impulse response, digital filters, for example—filters in which the outputs are fed back into the computation of the next outputs—round-off or truncation errors can build up and hurt the performance of the system with the DSP filter. Sometimes the degradation introduced by fixed-point arithmetic can be reduced with block floating-point techniques, which extend the dynamic range of the fixed-point representation by normalizing the data at appropriate stages in the algorithm. The resulting exponent or shift-factor is saved and restored at a later point.

Amnon Aliphas *DSP Associates*
Joel A. Feldman *Kurzweil Applied Intelligence*

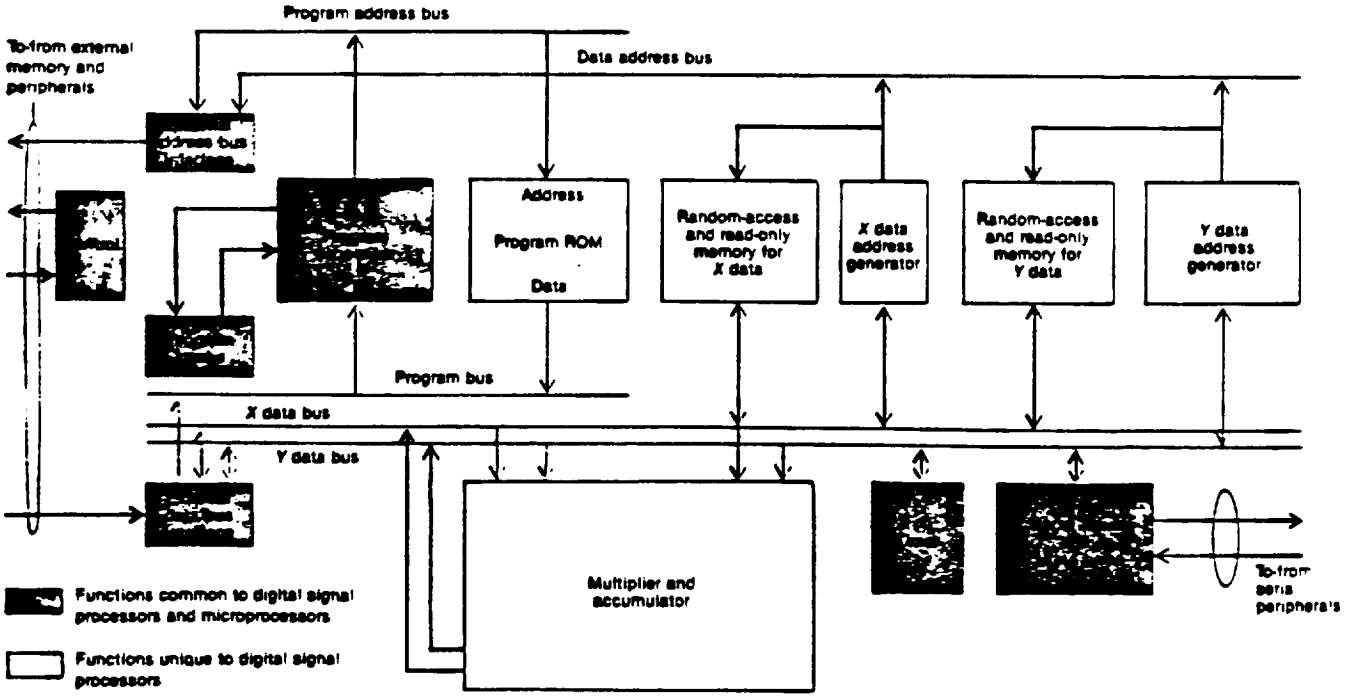
Defining terms

Arithmetic-and-logic unit: hardware that performs data operations such as addition, subtraction, and logical AND and OR.
Barrel shifter: hardware that allows arbitrary shifting of data.
Bit reversal: an addressing technique in which the order of bits in an address is reversed during a computation (such as a fast Fourier transform).
Benchmark: a specification that tells how well a product does a specified task; ideally, a benchmark simplifies a comparison of competing products by presenting a single piece of data that tells which product is better for a given application.
Cache: a small, fast memory positioned between a larger, slower memory and a processor; the cache's control maintains blocks of data adjacent to recently accessed data or instructions from the larger memory, thereby making access faster than with the larger memory alone.
Multiplier-accumulator: hardware dedicated to multiplying numbers and adding the results; the very fast units are typically able to multiply two numbers and add the product to previously accumulated results in a single clock period.
Pipelining: starting the execution of a new task before a preceding one has been completed.
Round-off and truncation noise: in digital signal processing, noise introduced when the word length of the computation exceeds that of the DSP chip and the extra bits are discarded.

0018-9235/87/0600-0040$01.00 © 1987 IEEE



High speed is characteristic of digital signal processing integrated circuits like the TMS320C30, under development at Texas Instruments Inc. in Houston. The design of this third-generation chip, as seen on an engineering workstation screen, includes a multiplier, along with shift registers [blue, top] that connect the multipliers' cells to a data bus.



[1] The features that most strongly distinguish this generic digital signal processing circuit from a general-purpose microprocessor are its multiplier-accumulator and multiple buses and memories. The multiplier-accumulator can perform a complete multiplication and addition in a single clock cycle, with the other circuitry making sure that it does not waste time waiting for operands.

The DSP chips that most efficiently implement block floating-point techniques have barrel shifters and circuits that detect the presence of exponents. Saturation arithmetic hardware is another way to get around the limitations of fixed-point arithmetic. The hardware automatically clamps a number that would otherwise overflow. While this technique adds an undesirable nonlinearity to the processing, it is necessary when the dynamic range of the signals is not well controlled.

Checking architectural features

Of the various architectural features that distinguish DSP chips, one of the most important is the multiplier-accumulator. More than any other element, it distinguishes the DSP circuit from the general-purpose microprocessor.

A typical fixed-point multiplier-accumulator performs a 16-by-16-bit multiplication, and the 32-bit product is added to a 32-bit accumulator register in a single instruction cycle. An alternative design uses separate elements for multiplication and addition, as in the μpd7720A and μpd77230 digital signal processors, made by NEC Electronics Inc., in Mountain View, Calif. This approach allows the system designer to fine-tune the processor for a specific application by controlling the software pipelining of the multiplication and addition operations. In this way a multiplication step in one computation and an addition step in another can be done at the same time. Of course, this also complicates the programming. In either architecture—a single multiplier-accumulator or separate elements—a DSP chip can perform an entire multiply-accumulate operation in a single clock cycle.

The numbers are impressive. A 16-by-16-bit multiplication on the WEDSP16, made by AT&T Bell Laboratories, Murray Hill, N.J., requires just 60 nanoseconds, or a single clock cycle. By contrast, serially executed multiplication by the MC68020 general-purpose microprocessor, made by Motorola Inc., Phoenix, Ariz., takes 1500 ns, or 25 clock cycles of this particular microprocessor.

Multiple buses and memories also affect throughput. A typical digital signal processor has two data memories and two data buses. With them it can deliver two operands required for a single-cycle execution of the multiply-accumulate function. Unlike general-purpose microprocessors, which store both instructions and data in the same memory, most DSP processors employ the so-called Harvard architecture, with separate program and data memories so that instruction and data can be fetched simultaneously. Many DSP chips go a step further, employing a modified Harvard architecture to allow storage of data in the program memory. In this way, static data like filter coefficients can be stored in the cheaper, slower program memory and then be moved to the faster but smaller data memory when needed.

There are a variety of ways to enhance the system's throughput by increasing the rate of data access (memory bandwidth). This is usually accomplished through multiple memories and corresponding buses. Several DSP chips have dual internal data memories, as well as access to an external data memory. Typically, the internal memories contain 128 to 512 16-bit words—adequate for many DSP tasks. Other DSP circuits have mechanisms for fetching instructions from an internal program RAM, thereby freeing an external bus for an additional data access. In the ADSP2100 from Analog Devices Inc., Norwood, Mass., for example, this is achieved with a 16-word program cache that allows two external data accesses in a single cycle.

Another DSP chip, the TMS320C25 from Texas Instruments Inc. in Houston, has a repeat facility, by which a single instruction can be repeated a specified number of times with only a single fetch from the external program memory. Again, when code is running internally, an external bus that is otherwise dedicated to accessing external code is free to access data. The design of the developmental TMS320C30 from Texas Instruments calls for a 64-word program cache. It also includes a direct memory access arrangement on the integrated circuit, which transfers blocks of data from an external memory into the DSP chip's memory in parallel with the operation of the DSP chip's arithmetic-and-logic unit.

For many of these architectural features, the disadvantage is that the DSP programmer, unlike the programmer of a general-purpose microprocessor, must give careful attention to the size and location of the data arrays and code loops.

Another approach to boosting throughput draws on external program and data memories that are fast enough to supply instructions and operands in a single cycle. However, this requires static RAMs with a minimum capacity of 16 384 to 65 536 bits and minimum cycle time of 25 to 70 ns. By way of comparison, general-purpose microprocessors typically use dynamic RAMs with capacities of 262 144 bits to 1 megabit and external program and data memory cycle times of 250 to 300 ns. While the 10-megahertz 68000 microprocessor from Motorola requires eight to 14 cycles (800 to 1400 ns) to access its memory, most DSP chips can access external memories within their cycle time—typically 100 to 200 ns. The tradeoff is that these fast static RAMs are six to nine times costlier than dynamic RAMs on a bit-for-bit basis.

Some DSP applications like image processing require data memories that can handle more than 64 000 words of address space. At least one DSP circuit under development—the TMS320C30 of Texas Instruments—is expected to store and access up to 16 million 32-bit words and program words.

Most state-of-the-art DSP chips provide an acknowledge signal after addressing. The processor waits for this signal before proceeding. By taking advantage of this, the system designer can trade off system performance and cost by using slower, less expensive memories. For instance, to run at full speed, the TMS320C25 requires fast, 25-ns static RAMs. If instead a wait state is inserted, slower, 70-ns static RAMs can be used, thereby reducing the memory cost by as much as 50 percent. In some cases, the use of internal program cache and data memories can compensate for the loss of performance due to these slow external memories. Also, the acknowledge line permits memories and peripherals with different access times to be mixed within the same address space.

A DSP chip architecture is also typified by separate arithmetic-and-logic units for data arithmetic and for address arithmetic. Since many of the instructions executed in DSP algorithms employ the data arithmetic-and-logic unit, the throughput rises substantially when a second unit is added to perform address calculations. When the address and data calculations are split between arithmetic-and-logic units, each unit's architecture can be optimized for the task at hand. For example, arithmetic-and-logic units for data contain multiply-accumulate structures and features such as saturation arithmetic and rounding, while the architecture of address arithmetic-and-logic units is tailored for such tasks as indexing and automatic address modification, as well as modulo arithmetic, which allows the address generator to automatically recognize the beginning and end of a data block within memory. Address arithmetic-and-logic units sometimes perform bit-reversal, which is used by the fast Fourier transform algorithm.

Development of a DSP algorithm generally starts with a floating-point simulation of satisfactory performance. Then a fixed-point implementation is sought—one that is efficient and has no ill effects on the algorithm's performance. This step is usually time-consuming and difficult.

For this reason, many systems designers welcome the appearance of general-purpose DSP chips that can perform full floating-point calculations. Among them are the 24-bit MSM 6992, made by Oki Electric Industry Co., Tokyo; the 32-bit μpd77230, from NEC Electronics; and the 32-bit WEDSP32 from AT&T Bell Laboratories. The latest member of the Texas Instruments TMS320 DSP family—the TMS320C30—is also based on a 32-bit floating-point format.

Although these processors require large die sizes and high transistor counts, they achieve instruction cycle times as fast as some of their fixed-point counterparts. Most also have fixed-point arithmetic-and-logic units that are especially useful for address

A sampling of digital signal processing integrated circuits[1]

Company	Product[2]	Package[3] number of pins	Power dissipation W	Instruction cycle time, ns	Internal memory size[4] RAM bits	Internal memory size[4] ROM bits	External memory size, bits	Word length[5] bits	Multiply-accumulate parameters: Accumulation precision, bits	Multiply-accumulate parameters: Time per operation,[6] ns
Analog Devices Inc., Norwood, Mass.	ADSP2100	PGA, 100	0.6	125	16 × 24 C	None	16K × 24 P 32K × 16 D	16	40	125
AT&T Bell Laboratories, Murray Hill, N.J.	WEDSP16	PLCC, 84; CLCC, 84	0.4	60	512 × 16 D,P	2K × 16 D,P	64K × 16 P	16	36	60
	WEDSP32	DIP, 40; PGA, 100	2.6	244	1K × 32 D,P	512 × 32 P	14K × 32 D,P	8 E, 24 M	8 E, 32 M	244
Fujitsu Ltd., Tokyo, Japan	MB8764	PGA, 88; PLCC, 84	0.3	100	256 × 16 D	1K × 24 P	1K × 16 P,D	16	26	100
Inmos Corp., Colorado Springs, Colo.	IMSA100	PGA, 84	<1.5	NA	NA	NA	NA	16	36	400
Motorola Inc., Phoenix, Ariz.	DSP56000	PGA, 88; PLCC, 88	NA	98	512 × 24 D	512 × 24 D, 2K × 24 P	64K × 24 P 128K × 24 D	24	56	98
	DSP56200	DIP, 28	NA	NA	256 × 24 D 256 × 16 D	NA	NA	16 D, 24 C	40	98
NEC Electronics Inc., Mountain View, Calif.	μpd77X20X	DIP, 28	0.6	250	128 × 16 D	512 × 23 P, 510 × 23 D (EPROM)	None	16	16	500
	μpd77230	PGA, 68	1.7	150	1K × 32 D	1K × 32 D, 2K × 32 P	4K × 32 P 4K × 32 D	8 E, 24 M	8 E, 47 M	150
Oki Electric Industry Co. Ltd., Tokyo, Japan	MSM6992	PGA, 132	0.4	100	1K × 32 P	1K × 32 P	64K × 32 D 64K × 32 P	6 E, 16 M	8 E, 16 M	100
Signetics Corp., Sunnyvale, Calif.	PCB5010	PLCC, 68	0.5	125	256 × 16 D 32 × 40 P	512 × 16 D 992 × 40 P	64K × 16 D,P	16	40	125
	PCB5011	PGA, 144	0.5	125	256 × 16 D	None	512 × 16 D 64K × 16 D,P	16	40	125
Thomson-CSF Components Corp., Canoga Park, Calif.	TS68930	DIP, 48	1.5	160	256 × 16 D	512 × 16 D 1K × 32 P	4K × 16 D	16, 32	32	160 R 360 C
	TS68931	PGA, 84; CLCC, 84	1.8	160	256 × 16 D	512 × 16 D	4K × 16 D, 64K × 32 P	16, 32	32	160 R, 360 C
Texas Instruments Inc., Houston, Texas	TMS320C1X	DIP, 40; PLCC, 44	0.2	160; 200	256 × 16 D	4K × 16 P (EPROM)	4K × 16 P	16	32	320, 400
	TMS32020	PGA, 68	1.2	200	288 × 16 D 256 × 16 D,P	None	64K × 16 D 64K × 16 P	16	32	200
	TMS320C25	PLCC, 68; PGA, 68	0.5	100	288 × 16 D 256 × 16 D,P	4K × 16 P	64K × 16 D 64K × 16 P	16	32	100
	TMS320C30	QFP, 100; PGA, 176	NA	60	2K × 32 D,P 64 × 32 C	4K × 32 D,P	16M × 32 D,P	8 E, 24 M	8 E, 32 M	60
Zoran Corp., Santa Clara, Calif.	ZR33881	LCC, 68	1.0	NA	NA	NA	NA	8	25	50
	ZR34161	DIP, 48	0.5	100	256 × 17 D	256 × 17 D	64K × 16 D,P	16	25	100

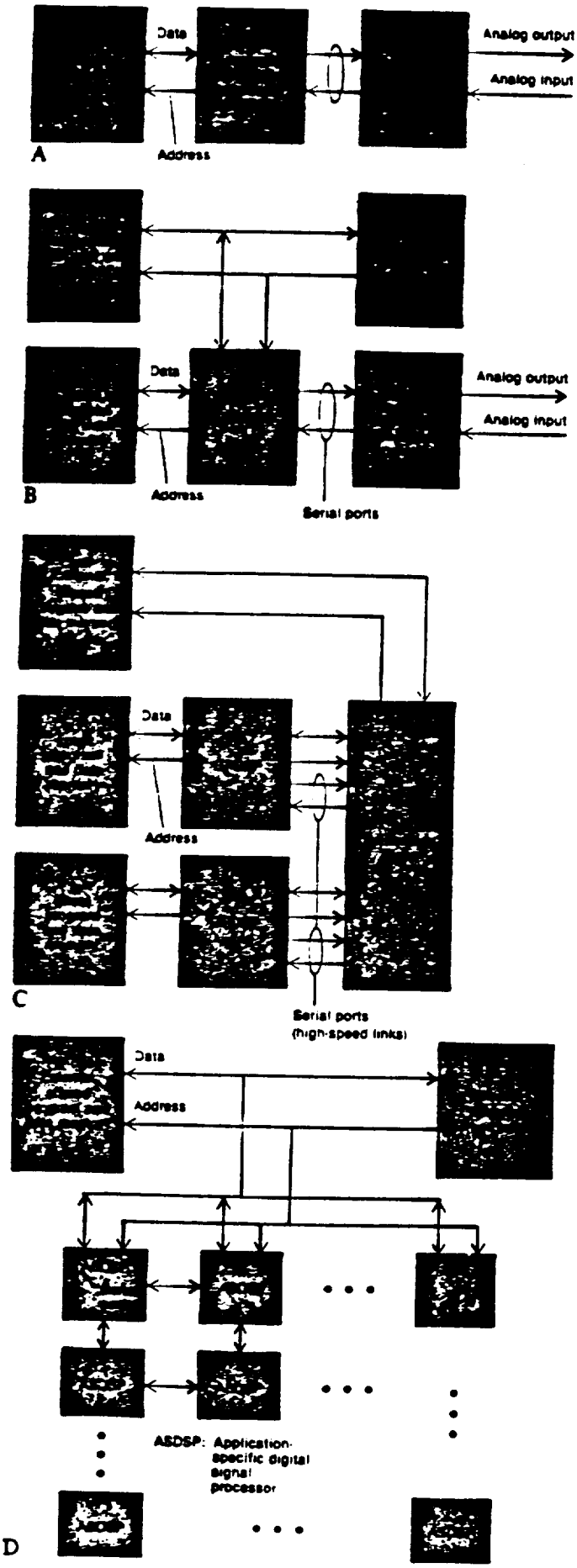
1. Based on an extensive table available from DSP Associates, 18 Peregrine Rd., Newton, Mass. 02159; except for the Motorola DSP56000 and the Texas Instruments TMS320C30, all ICs are currently available; NA—data not available or not applicable
2. Except for the digital filters IMSA100, DSP56200, and ZR33881 and the vector signal processor ZR34161, all the ICs are general-purpose digital signal processors; most devices handle integer-type data; the WEDSP32, μpd77230, and MSM 6992 are floating point devices; the TMS320C30 is designed to operate in either an integer or a floating point mode, and the ZR34161 is a block floating-point device; μpd77X20X represents three devices: the μpd7720A, 77C20, and 77P20; TMS 320C1X represents six devices: the TMS320C10, C15, E15, C17, E17, and TMS32011
3. Legend: PGA—pin-grid array; PLCC—plastic, leadless chip carrier; CLCC—ceramic, leadless chip carrier; DIP—dual in-line package; QFP—quad flat package; LCC—leadless chip carrier
4. Legend: D—data; P—program; C—cache, EPROM—electrically programmable read-only memory; K—1024 bits; M—1 048 576 bits
5. Legend: E—exponent; M—mantissa; C—coefficient
6. Legend: R—real numbers; C—complex numbers; for DSP56200, the time is per filter tap

calculation in parallel with floating-point data calculations. No standard floating-point format is used by these processors; however, most have instructions for format conversion to and from the IEEE 32-bit standard.

Architectural variations appear in DSP circuits that are targeted for high-volume, low-cost applications such as speech synthesis in consumer products. For instance, many DSP circuits are available in versions that have an internal program and data ROM in place of an external RAM. The advantages are a lower pin count and more economical packaging. Conceptually similar to general-purpose microcontrollers, such high-volume DSP chips reduce the system cost by achieving higher levels of functional integration. Toward this end, they generally include special interfaces to serial devices, like analog-to-digital converters, as well as such special-function units as timers.

Single- and multiprocessor operation

DSP circuits can be integrated into a system in three major configurations: stand-alone, slave, and multiprocessor. The most economical configuration is usually the stand-alone, found most



often in high-volume, low-cost applications [Fig. 2A]. Since the DSP chip does indeed stand alone in this case, a fairly sophisticated, programmable unit is required.

In the slave configuration, the DSP chip acts as a peripheral to a general-purpose microprocessor [Fig. 2B]. In this case the more time-consuming and repetitive processing is left to the DSP chip, while the control and communications tasks are taken care of by the general-purpose processor. This architecture is particularly well-suited for application-specific DSP circuits. Typical applications include speech recognition and multichannel processing.

The highest DSP performance is achieved in the multiprocessing configuration. Such an arrangement would be appropriate for a phased-array radar system, in which the tasks are distributed among multiple processors.

For general-purpose DSP chips, multiprocessing is usually handled by at least two processors that send and receive control information over communication links [Fig. 2C]. The processors also share data from a single memory by means of access arbitration logic.

In a typical multiprocessing configuration of application-specific DSP chips, several dozen processors in one- or two-dimensional arrays [Fig. 2D] perform very repetitive tasks, like filtering, convolution, and correlation.

Beware the benchmarks

In comparing the benchmark performance figures supplied by various DSP manufacturers, particular care is advisable. Key questions must be answered before making such a comparison. For example, how well did the programmers in the different companies optimize the benchmark code? Does the system design call for an on- or off-chip memory? A product offered by Company A may output the fast Fourier transform data in bit-reversed order, while Company B's chip may not. A timing comparison between the two would therefore be meaningless.

The engineer who will use the chips should examine the benchmark software in detail. The extra effort up front may well prevent some nasty surprises later.

A benchmark describes the performance of a chip under very specific conditions, and if there is even a slight shift in those conditions, the performance may change drastically. For example, if a chip has a 128-word internal data memory, it may turn in an impressive performance when benchmarked with a 64-complex point fast Fourier transform. A 128-complex point transform, however, would require that external memory be addressed, and performance may suffer.

In choosing a DSP chip, the availability of engineering support is a critical factor for engineers whose primary expertise is not digital signal processing. If the engineer who uses the DSP

[2] DSP chips are employed in three main configurations: stand-alone, slave, and multiprocessor. The stand-alone configuration [A], which uses primarily the serial ports of the DSP chip is typically found in low-cost applications, such as speech synthesis for consumer products. In the slave configuration, the DSP chip operates under the control of a "host" general-purpose microprocessor [B]. The slave setup is typical in applications that require a fair amount of support processing—speech recognition, for instance. For the highest level of performance, as in high-throughput filtering, convolution, and correlation, DSP chips are configured as multiprocessors. For example, a small number of DSP chips may communicate control information over high-speed serial links [C], to minimize the circuit's pin count. The multiprocessing approach may also involve dozens of processors set up in an array [D].

ORIGINAL PAGE IS OF POOR QUALITY

chip lacks such expertise, the chip manufacturer should be able to deliver it. Some manufacturers publish application notes along with DSP routine libraries for specific applications such as fast Fourier transforms and finite-duration impulse response filters.

In narrowing down the choices, the typical engineer should also ask what development tools the chip maker offers. As with engineering support, development tools are often overlooked, while technical performance and price are mistakenly given the lion's share of attention. Development tools can play a major role in speeding a product to market by reducing the design time, simplifying the debugging process, and helping to train the engineer who is unfamiliar with DSP chips.

On the hardware side, such tools include development boards. By working with a fully functional development board, the engineer can examine the DSP chip in operation, learn about timing, and observe the processor's interaction with the board's peripheral circuitry.

At the point of hardware-software integration, the availability of in-circuit emulation is just as important for DSP chip debugging as it is for general-purpose microprocessors. Most DSP manufacturers offer in-circuit emulation systems at prices ranging from $5000 to $15 000. Hardware development support for some of the more established DSP circuits like the Texas Instruments TMS32010 is also available from third-party sources such as Hewlett-Packard Co. in Corvalis, Ore. and Tektronix Inc. in Beaverton, Ore.

Software development tools include assemblers for coding algorithms, and simulators for checking that the algorithms work correctly before they are committed to the target system. High-level language compilers are becoming available for some DSP chips. C-compilers are available from third-party vendors for the Texas Instruments TMS32010 and the TMS32020 and the Analog Devices ADSP-2100, and Texas Instruments is also developing a C-compiler for its upcoming TMS320C30. Assemblers deal with device-specific instruction sets; developing code with high-level compilers, however, paves the way for transportable DSP chip application software. The use of high-level languages for portions of code that are not time-critical can shorten the development time, enhance the software reliability, and simplify the debugging process.

The pickings may be lean

The selection of a DSP circuit may be complicated in that the product is available from only a single manufacturer. Very few secondary manufacturers are making DSP chips under license from major semiconductor manufacturers, and so far the U.S. Department of Defense has certified only a limited number of devices under military specifications.

Telecommunications will continue to be the dominant applications area, given the widening acceptance of the Integrated Services Digital Network (ISDN). In other areas, computer-aided design and engineering, numeric processing, speech and image processing, instrumentation, and digital audio and television will also contribute to the increased use of DSP chips.

Certain architectural features of DSP circuits are likely to change. While today's DSP chips typically address 64 000 words of external data or program memory, this capacity is bound to expand to meet demands in areas such as image processing. The combination of faster cycle times and larger external memories leads to the requirement of external memory with a higher bandwidth in next-generation DSP-based systems. In a new generation of CMOS dynamic RAMs, a static column mode of operation allows access to data within a 512-word block at the speed of the more expensive static RAMs—at only one-sixth the price.

Floating-point capability in DSP chips will bypass the often difficult step of converting floating-point simulations to fixed-point implementations, making way for algorithms that were once practical only in research and development or in areas where relatively high costs could be tolerated.

As DSP circuits become more powerful, applications will become more complex, demanding higher-level languages. Higher levels of integration are likely to result in the availability of earlier DSP chips as "standard cells" for semicustom IC design. High-volume users will then be able to reduce systems costs further by developing a semicustom IC that is based on a "core" DSP chip, with digital and analog functions of their own choosing. Meanwhile, advances in general-purpose microprocessors—faster multiplication, for example—will spur a migration of these units into the DSP area.

To probe further

Articles on digital signal processors from Texas Instruments (TMS320C25), Motorola (DSP56000), Analog Devices (ADSP-2100) and NEC (μpd77230) can be found in the December 1986 digital signal processing issue of *IEEE Micro* magazine. Other recommended articles are: "Computer Architecture for Digital Signal Processing," by J. Allen, *Proceedings of the IEEE*, Vol. 73, No. 5, pp. 852-873, and "New Applications of Digital Signal Processing in Communications," by M. Bellanger, *IEEE ASSP Magazine*, July 1986, pp. 6-11.

A good reference is the book *Signal Processor Chips*, by David Quarmby, Prentice-Hall, Englewood Cliffs, N.J., 1984.

Additional information may be found in notes from seminars entitled "Digital Signal Processors," available from DSP Associates, 18 Peregrine Rd., Newton, Mass. 02159.

Applications of digital signal processors along with tutorial information are the subject of applications notes available from most manufacturers.

Representation of numbers in 32-bit arithmetic is addressed in ANSI/IEEE Standard 754, 1985, entitled "IEEE standard for binary floating-point arithmetic."

Echo canceling with a digital signal processing chip is addressed in the article "Digital voice echo canceller with a TMS32020," by David G. Messerschmitt, David Hedberg, Christopher Cole, Amine Hanoui, and Peter Winship, pp. 415-57 of the book *Digital Signal Processing Applications with the TMS320 Family*, available from Texas Instruments Inc. in Houston, Texas.

Adaptive filtering is discussed in *Adaptive Filters* by M.L. Honig and D.G. Messerschmitt, Kluwer Academic Publishers, Boston, Mass., 1984.

Adaptive digital signal processing is addressed in *Adaptive Signal Processing*, by Bernard Widrow and Samuel D. Stearns, Prentice Hall, Englewood Cliffs, N.J., 1985.

Echo canceling techniques are addressed in the paper "An adaptive echo canceller," by M. Sondhi, *Bell Systems Technical Journal*, Vol. 46, March 1967, pp. 497-511.

About the authors

Amnon Aliphas (M) is president and founder of DSP Associates in Newton, Mass. a company that develops digital signal processor-based systems and presents seminars on related topics. From 1983 to 1986, he was senior scientist for signal processing at Kurzweil Applied Intelligence in Waltham, Mass. While there, he worked on the development of speech recognition systems with large vocabularies and the architecture of the KSC2408 digital filter chip—a key component of the KVS3000, a 1000-word recognition system. He holds a B.S. degree from the Universidad Nacional Autonama de Mexico in Mexico City (1976) and M.S. and Ph.D. degrees in electrical engineering from Stanford University (1978 and 1981, respectively). He is an adjunct professor at Boston University in Massachusetts.

Joel A. Feldman (M) is a senior design engineer at Kurzweil Applied Intelligence. Since joining the company in 1984, he has been involved in hardware and algorithmic development for speech-recognition systems with large vocabularies. From 1980 to 1984, he was a technical staff member of the Speech Systems Technology Group at the Lincoln Laboratory of the Massachusetts Institute of Technology. He holds B.S. and M.S. degrees in electrical engineering from the Massachusetts Institute of Technology. ◆

ORIGINAL PAGE IS
OF POOR QUALITY