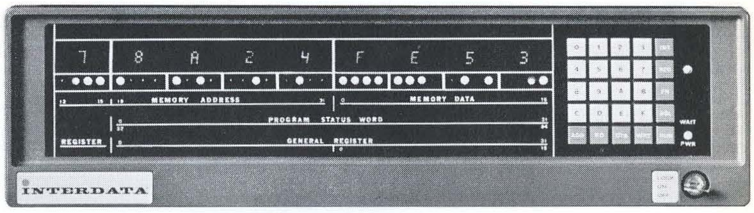
INTERDATA®

Product Bulletin

Model 7/32 Processor



STANDARD FEATURES INCLUDE:

- **Memory expansion to one million bytes of 750 ns core.**
- **Direct addressing to sixteen million bytes.**
- **Sixteen — 32 bit hardware general registers.**
- **Sixteen — 32 bit hardware I/O registers.**
- **Up to 1024 I/O interrupts with automatic hardware vectoring.**
- **Up to 1024 auto driver channels.**
- **242 instructions defining byte, halfword and fullword manipulations.**
- **Upwards compatible with current Models 70, 74, 80, 85 and the 7/16.**
- **Powerful memory protect feature.**
- **Lowest cost for performance.**
- **Powerful software modules available.**

A 32 BIT MINICOMPUTER?

Most people think that minicomputers must be 16 bits or less to qualify for the title; that in order to enjoy the cheap price you have to be prepared to suffer the pains of 16 bit architectures. Pains such as limited addressing (64K or less); 16 bit logical and arithmetic operations, primitive instruction sets, meager register complements, and inferior input/output schemes are tolerated because everyone knows you can't get something for nothing — low prices and low performance go together. Right?

Right Yesterday. Wrong Today. INTERDATA announces the Model 7/32, a true 32 bit computer for a true minicomputer price. Sixteen 32 bit general registers, full 32 bit logical and arithmetic manipulations, 242 instructions, a separate stack of sixteen 32 bit I/O registers, a powerful automatic character stuffing and code conversion I/O feature, hardware interrupt vectoring, memory expansion to one million bytes, direct addressing capability to sixteen million bytes, double indexing — doesn't sound like a typical minicomputer does it?

The Interdata 7/32 offers 32 general registers, each 32 bits wide, in two stacks of sixteen. Stack selection is controlled by a bit in the program status word. The dual stack organization offers fast and simple context switching (eg from user program to operating system I/O interrupt handling) without the necessity of storing and restoring general registers as most minicomputers are forced to do. Within the user stack, sixteen registers are available for general purpose use including index registers, accumulators, temporary storage of results, etc. This large number of GP registers allows simple, flexible programming. The I/O stack is further subdivided into two sets of eight: one for internal interrupts, and one for I/O interrupts. Upon detection of an interrupt, some of these registers are used to store the old program status word and other information such as device status or the address of a supervisor call parameter block. Other registers are available for general use by the interrupt driver.

In order to enhance the powerful addressing capability of the Interdata 7/32, 156 instructions are defined which include 16 and 32 bit arithmetic and logical operations, list processing, floating point, cyclic redundancy checking, and bit and byte manipulation instructions. Double indexing is also allowed, along with a multitude of branch instructions that utilize two separate condition codes: one set based upon the true 32 bit Register Operation, and one set controlled as if the operation had been 16 bits only. In addition, 86 extended branch mnemonics are defined to bring the total number of instructions to 242.

Up to 1024 auto driver channels provide fast automatic character input/output operations, including automatic conversion from one character code to another. Each character is transferred into or out of memory without any effect upon the running program except a small amount of stolen time, typically about 15 us. Additional operations that may be performed include cyclic redundancy or longitudinal redundancy checking, dual buffering and trapping to special subroutines upon receipt of special characters. This lessens the driver software needed; reducing the operating system overhead and improving I/O communication rates.

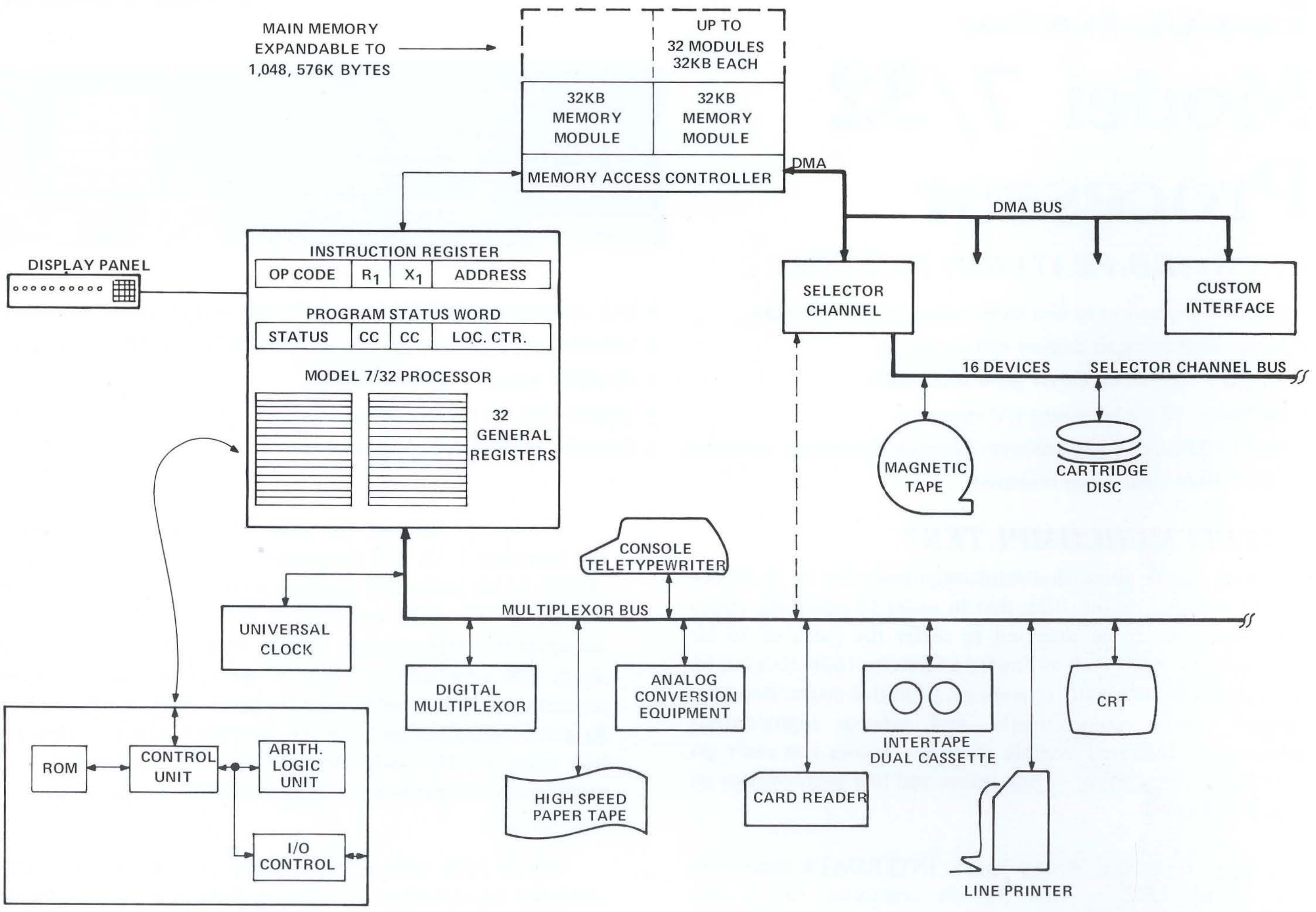
The 7/32 offers up to 1024 interrupt levels, each automatically vectored by hardware to a unique service routine (software driver or auto driver channel); relieving the programmer of the task of determining which device generated the interrupt and what its status is.

The optional memory access controller utilizes sixteen 32 bit hardware registers to allow segmentation, relocation and memory protection of user programs. This feature goes way beyond simple write protection, allowing seven different memory protect states: (1) no access (2) no protection (3) read & write (4) read & execute — no write (5) read only (6) non presence (7) trap after first write. The memory access controller is particularly useful for operating systems that embody a dynamic relocation foreground/background philosophy.

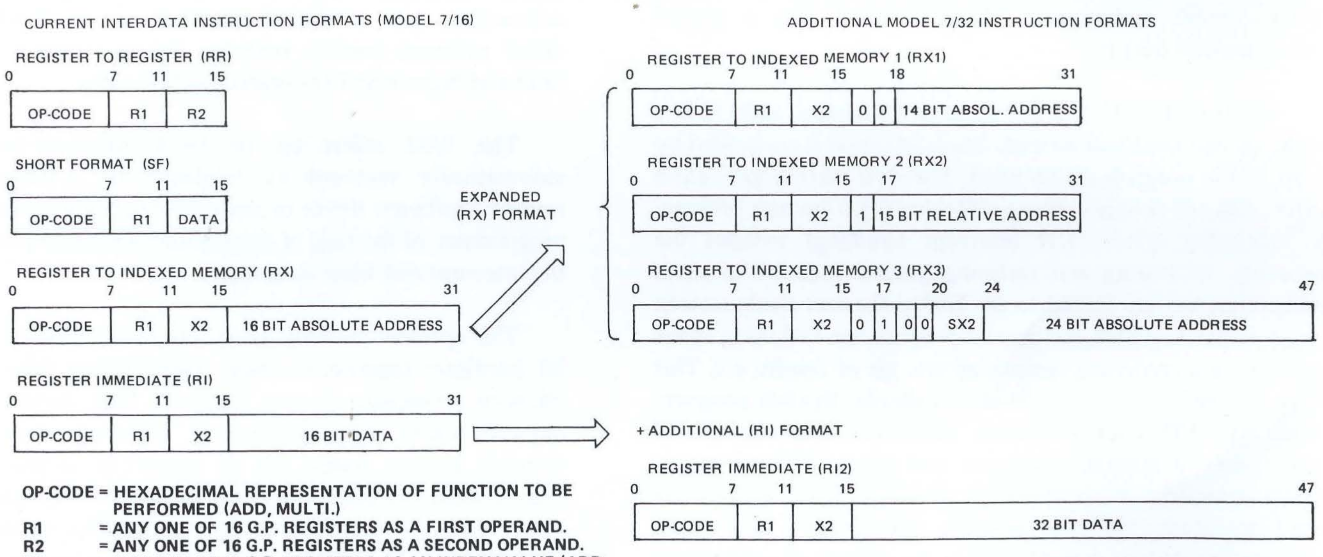
The information contained herein is intended to be a general product description and should not be utilized as an explicit specification for such product.

© INTERDATA, INC., 1973 ALL RIGHTS RESERVED

7/32 PROCESSOR BLOCK DIAGRAM



INTERDATA INSTRUCTION FORMATS


CURRENT INTERDATA INSTRUCTION FORMATS (MODEL 7/16)
REGISTER TO REGISTER (RR)
0 7 11 15
OP-CODE R1 R2
SHORT FORMAT (SF)
0 7 11 15
OP-CODE R1 DATA
REGISTER TO INDEXED MEMORY (RX)
0 7 11 15 31
OP-CODE R1 X2 16 BIT ABSOLUTE ADDRESS
REGISTER IMMEDIATE (RI)
0 7 11 15 31
OP-CODE R1 X2 16 BIT DATA
EXPANDED (RX) FORMAT
ADDITIONAL MODEL 7/32 INSTRUCTION FORMATS
REGISTER TO INDEXED MEMORY 1 (RX1)
0 7 11 15 18 31
OP-CODE R1 X2 0 0 14 BIT ABSOL. ADDRESS
REGISTER TO INDEXED MEMORY 2 (RX2)
0 7 11 15 17 31
OP-CODE R1 X2 1 15 BIT RELATIVE ADDRESS
REGISTER TO INDEXED MEMORY 3 (RX3)
0 7 11 15 17 20 24 47
OP-CODE R1 X2 0 1 0 0 SX2 24 BIT ABSOLUTE ADDRESS
+ ADDITIONAL (RI) FORMAT
REGISTER IMMEDIATE (RI2)
0 7 11 15 47
OP-CODE R1 X2 32 BIT DATA

OP-CODE	**= HEXADECIMAL REPRESENTATION OF FUNCTION TO BE PERFORMED (ADD, MULTI.)**
R1	**= ANY ONE OF 16 G.P. REGISTERS AS A FIRST OPERAND.**
R2	**= ANY ONE OF 16 G.P. REGISTERS AS A SECOND OPERAND.**
X2	**= ANY ONE OF 15 G.P. REGISTERS AS AN INDEX VALUE (ADD TO APPARENT ADDRESS FIELD TO OBTAIN TRUE VALUE OF ADDRESS).**
SX2	**= ANY ONE OF 15 G.P. REGISTERS AS A SECOND INDEX VALUE (TO BE ADDED TO SUM OF ADDRESS + FIRST INDEX VALUE).**

A'RELATIVE" ADDRESS IS CALCULATED BY ADDING THE CURRENT LOCATION (ADDRESS) OF THE INSTRUCTION BEING DECODED TO THE SUM OF THE APPARENT ADDRESS AND THE INDEX VALUE.

CONSIDER TOTAL SYSTEM COST

Where have you been spending most of your minicomputer dollars lately? Software and systems costs have been going up and up, while minicomputer vendors have been concentrating on bringing out faster, better, and cheaper hardware boxes. This has been a boon to users because cheaper, better mini's have made many applications economical for computer use that were ridiculously expensive to tackle a few years ago. On the other hand, it has also become painful since the software environments of many of the more powerful minis have become considerably more complicated and difficult to work with. All other minicomputers have an absolute addressing limitation forced by the number of bits used to reference memory. The highest address heretofore available was 64K (2^{16}=65,536). Some manufacturers offer a non simple segmentation and relocation scheme to offset this inherent limitation of 16 bits; however, the relief is marginally effective since only 64K is available at one time, and extensive software overhead is required to make it work.

INTERDATA has long recognized the demand for more powerful, cheaper solutions to extended memory minicomputer applications. The trick has been to come up with a way to help users combat the burgeoning costs of software and systems analysis. In other words, how could we help the not so poor ($) programmer, and still maintain a compatable family of computers?

The answer is the INTERDATA Model 7/32 – as powerful as most 32 bit machines, but priced to be competitive with other 16 bit minicomputers. Now the programmer need not be limited by the severe addressing restrictions minicomputers have been forcing on him; instead he can concentrate all of his efforts tackling the natural constraints of the problem or process he is trying to computerize.

The INTERDATA Model 7/32 has no practical limit in addressing power since it will address memory to sixteen million bytes. Although the current maximum physical memory size is one million bytes, the software architecture will allow significantly larger memories in the future as memory technologies continue to produce lower cost, higher performance memories.

ALL IN THE FAMILY

One of the most attractive features of the 7/32 is the fact that it represents the high end of an upwards compatable family of minicomputers which fill the mini, midi, and maxi performance ranges.

The 7/32 is upward compatible with the 7/16 and current 16 bit INTERDATA Models 70, 74, 80, and 85 by virtue of a halfword mode capability (controlled by a bit in the PSW). In this mode the machine executes all non-privileged instructions exactly the same way they are executed on the 16 bit processors. This capability allows mixing programs written for the 16 bit INTERDATA Models with other programs written around the powerful 32 bit architecture. Since the 8KB, 16KB and 32KB core memory modules and all peripheral interface boards will run in either the 7/16 or the 7/32, and further, since the 7/16 and the 7/32 are plug-in replacements for each other, it is a fairly easy task for a user to upgrade from an INTERDATA Model 7/16 to an INTERDATA Model 7/32.

SOFTWARE
NOT JUST ANOTHER PRETTY FACE

Of course, introducing new hardware boxes is a favorite tactic in the minicomputer business. Even though the architecture of the 7/32 goes a long way toward relieving the programmer's burden, we recognize that hardware by itself is not nearly enough. Accordingly, we designed and will deliver a powerful family of software to go with it.

1) A common assembly language (CAL) offers a comprehensive optimizing assembler that allows source programs to be written in a language common to both INTERDATA 16 and 32 bit machines. The user identifies the target machine at assembly time.

2) A serial tasking operating system (OS/32 ST) offers the user freedom from the tedious details of communicating with peripherals, building file structures, handling interrupts, keeping time, etc. In addition, OS/32 ST provides a powerful interpretive debugging capability to ease the program development effort.

3) A FORTRAN V compiler offers the user a high level language considerably more powerful and useful than FORTRAN IV. Among other things the FORTRAN V compiler offers Purdue/ISA extensions and run time tracing, as well as the capability of writing programs of any size up to one million bytes.

4) A Basic interpreter offers Dartmouth Basic plus additional features such as matrix manipulations, string manipulations, boolean operations, formatted printing, multidevice I/O and file handling capabilities.

5) A text editor (EDIT) allows easy editing of source programs on a line or character basis, utilizing an interactive terminal or a Batched input stream.

GROWING WITHOUT PAINS

The Model 7/32 is the forerunner of a series of products designed to satisfy users with requirements for larger, more powerful equipments in the minicomputer price range. Since the price of the 7/32 is so low, users may apply it to applications with requirements well below its capabilities, secure in the knowledge that system expansion is relatively painless given the large memory capacity, the powerful operating system management, and the on line language processing and debugging facilities. In addition, users with current 16 bit INTERDATA machines will be able to "trade up" to the 7/32 for a modest sum, reintroduce their programs to the 32 bit operating system and begin adding new programs written around the 32 bit capabilities of the 7/32.

THINK SMALL (PRICE, THAT IS)

The INTERDATA 7/32 not only offers the user unparalled horsepower, but it does so at an extremely low price – under \$10,000 for a 32KB CPU. That's less than the major 16 bit minicomputers that offer extended memory. It's certainly a whale of a lot less than any other 32 bit computer ever built (consider a cheap 370/135 which can be purchased, stripped, for only \$300,000).

THINK INTERDATA

As any knowledgeable computer user can attest, the costs of providing software have been skyrocketing over the years while hardware costs have been plummeting. INTERDATA's ultimate design objective with the 7/32 is to throw as much of the systems burden on the hardware as is possible in order to simplify and reduce the software effort. The net result to the user will be a total systems cost savings far beyond what has been possible with 16 bit minicomputers. If you have an extended memory minicomputer application problem to solve, can you afford to tackle it with anything other than the INTERDATA Model 7/32?

PROCESSOR OPTIONS

Memory Access Controller
Floating Point Hardware
Memory Parity Check & Generate
Power Fail/Auto Restart
DMA Buffer
Loader Storage Unit Controller
Universal Clock
Selector Channel
Turnkey Console
Binary Display Console
Hexidecimal Display Console
Memory Bus Buffer (required for every additional eight memory modules)

SPECIFICATIONS

Technology

Processor – T^2L – MSI and LSI
ROM – BiPolar LSI (60 ns Access Time)

Processor

Instruction Word Length – 16, 32, 48 Bits
Data Word Length – 8, 16, 32 bits
General Registers – 32 BiPolar hardware registers (32 bits each) separated into two stacks of 16; 30 usable as INDEX registers
Floating Point Registers – 8 registers (32 bits each-in main memory)
Direct Addressing – up to 16, 777,216 bytes
Arithmetic – two's complement (Fixed Point) – Sign/magnitude (floating point)
Instruction Repertoire – 225 instructions including multiply/divide, and list processing. Floating point instruction set is optional. (17 additional)
Processor Input/Output Modes – Programmed transfers, 26 – 150K 8 or 16 bit transfers
– Block transfer up to 360K bytes/sec.
– Auto driver channel, 50K bytes/sec.
– DMA Transfer – 2.6M bytes/sec.
Priority Interrupts – Identification of up to 1024 hardware levels with automatic device identification and vectoring.
**Interrupt Overhead Time – 6.5 μs*
Normal Interrupt Latency Time – 4.0 μs
Hardware I/O Timeout – 14 μs timeout on all micro processor operations to the I/O system, thereby ensuring that the processor will not lock-up should a module stall or otherwise fail to respont.
**Overhead time to allow servicing interrupts, including device identification, status recognition, and PSW swapping.*

MAIN MEMORY

Word Length – 16 bits (17 with parity option)
Organization – 8KB, 16KB and two types of 32KB modules available on single 15" plug in boards
Cycle Time – 8KB – 1000 nanoseconds
16KB – 1000 nanoseconds
32KB – 750 and 1000 nanoseconds
Maximum Memory Size – 1,048,576 Bytes

ENVIRONMENTAL

Temperature – 0-50°C
Humidity – 0-90% (non condensing)
Vibration – 0-55 CPS at 1.25 G's
Storage Temperature – –55°C to +85°C

PACKAGING

19" RETMA chassis, 14" high, 28" deep
15" x 15" printed circuit boards with ¼" aluminum stiffeners arranged horizontally
printed circuit backpanel
Pin to dual contact receptacle connections with locating pins
dual chassis with 16-15" board slots
processor occupies 3 boards, memory modules, 1 board

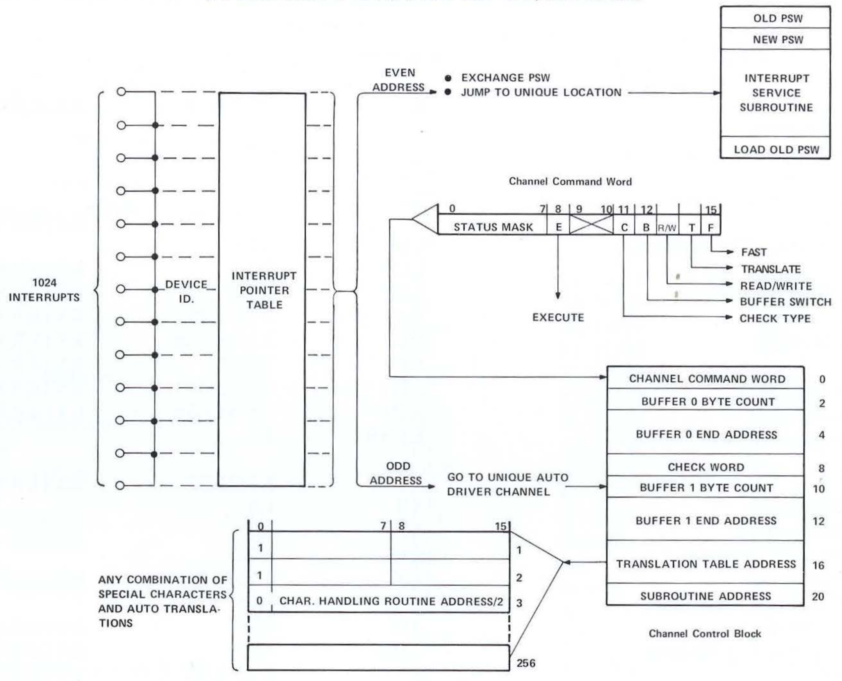
Power 115 or 230 VAC ± 10%, 50 or 60 Hertz, 16 amps max. (at 115 volts)

OPTIONAL DISPLAY CONSOLE

36 Binary Indicating LED's
9 Hexadecimal Display Matrices
16 Hexadecimal Character Keys
9 Function Select Keys
4 Function Indicating LED's
1-3 Position Keylock Switch

INTERDATA PRODUCT NUMBER: M73-023

INTERRUPT DRIVER INPUT/OUTPUT



Each 7/32 interrupt is automatically vectored by the hardware to either a unique service subroutine or to a unique auto driver channel. An auto driver channel permits variable length data records to be efficiently converted from one character code to another and transferred to and from any interrupt driven device on the multiplexer bus without disturbing the running program.

Each INTERDATA Auto Driver Channel has associated with it a Channel Control Block and a Translation Table in memory which specify the operation of that channel. The Channel Command Word is decoded as follows:

EXECUTE BIT (E): If reset, the channel does not perform any I/O operation and control is transferred to a subroutine whose address is specified by the last halfword of the CCB.

FAST BIT (F): If set, the channel performs the I/O transfer in the fast mode wherein only Buffer 0 is used and the redundancy check operations are not performed.

READ/WRITEBIT (R/W): If reset, a byte or a halfword is input from the device, otherwise, a byte or a halfword is output to the device.

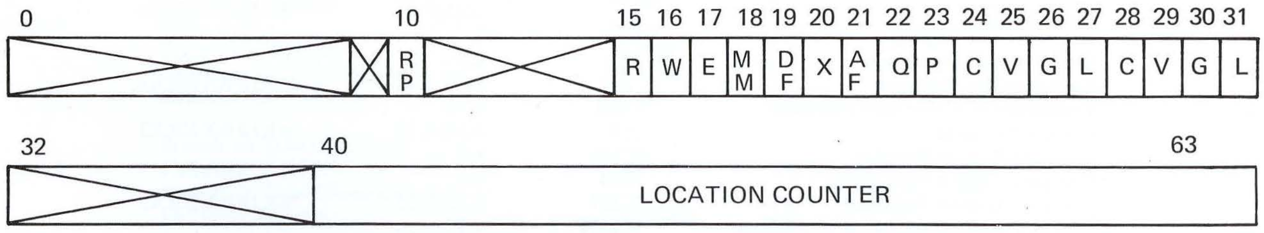
TRANSLATE BIT (T): If set, the character is automatically translated to the value contained in the translation table.

BUFFER SWITCH BIT (B): Specifies the buffer to be used for the transfer. If set, Buffer 1 is used, otherwise, Buffer 0 is used. A buffer may be any contiguous block of memory.

CHECK TYPE BIT (C): If reset, the channel performs a Longitudinal Redundancy Check (LRC). Otherwise, a Cyclic Redundancy Check is performed.

STATUS MASK: The channel performs an AND operation between the Status Mask and the device status. If the result is non-zero, no I/O operation is performed and control is transferred to the subroutine whose address is specified in the CCB. If the result is zero, the channel operation continues.

PROGRAM STATUS WORD (PSW)



Bit	Meaning
0:9	Reserved – Must be Zero
10	Relocation/Protection Enable
11-14	Reserved – Must be zero
15	Register Set Number
16	Wait State
17	External Interrupt Enable
18	Machine malfunction Interrupt Enable
19	Fixed Point Divide Fault Interrupt Enable
20	Extended Mode
21	Floating Point Arithmetic Fault Interrupt Enable
22	System Queue Service Interrupt Enable
23	Potect Mode
24-27	Fullword Condition Code
28-31	Halfword Condition Code
32-39	Reserved–Must be zero
40-63	Location Counter

INSTRUCTION REPERTOIRE

			Execution Time	
Type	*Instruction*	*Mnemonic*	*In μsec*	*Comments*
LOAD AND STORE INSTRUCTIONS	Load	L	3.25/3.75	RX1&RX2/RX3
	Load Register	*LR	1.0	
	Load Immediate	LI	2.5	
	Load Immediate Short	*LIS	1.0	
	Load Complement Short	*LCS	1.25	
	Load Halfword	*LH	2.75/3.25	RX1&RX2/RX3
	Load Halfword Immediate	*LHI	1.75	
	Load Halfword and Set	LHS	3/3.5	RX1&RX2/RX3
	Load Halfword Logical	LHL	2.75/3.25	RX1&RX2/RX3
	Load Address	LA	2.25/2.75	RX1&RX2/RX3
	Load Multiple	*LM	3.5/4+1.5n	RX1&RX2/RX3
	Store	ST	3/3.75	RX1&RX2/RX3
	Store Halfword	*STH	2.5/3.0	RX1&RX2/RX3
	Store Multiple	*STM	3.5/4+1.5n	RX1&RX2/RX3
	Exchange Halfword Register	EXHR	1.0	
FIXED POINT ARITHMETIC INSTRUCTIONS	Add	A	3.25/3.75	RX1&RX2/RX3
	Add Register	*AR	1.0	
	Add Immediate	AI	2.5	
	Add Immediate Short	*AIS	1.25	
	Add Halfword	*AH	2.75/3.25	RX1&RX2/RX3
	Add Halfword Immediate	*AHI	1.75	
	Add to Memory	AM	4/4.5	RX1&RX2/RX3
	Add Halfword to Memory	*AHM	3.5/4	RX1&RX2/RX3
	Subtract	S	3.25/3.75	RX1&RX2/RX3
	Subtract Register	*SR	1.0	
	Subtract Immediate	SI	2.5	
	Subtract Immediate Short	*SIS	1.25	
	Subtract Halfword	*SH	2.75/3.25	RX1&RX2/RX3
	Subtract Halfword Immediate	*SHI	1.75	
	Compare	C	5.75/6.25	RX1&RX2/RX3
	Compare Register	*CR	3.5	
	Compare Immediate	CI	5.00	
	Compare Halfword	*CH	5.25/5.75	RX1&RX2/RX3
	Compare Halfword Immediate	*CHI	4.25	
	Multiply	M	16/16.5	
	Multiply Register	MR	15	
	Multiply Halfword	*MH	6.0/6.5	RX1&RX2/RX3
	Multiply Halfword Register	*MHR	5.0	
	Divide	D	100	
	Divide Register	DR	100	
	Divide Halfword	*DH	12.5/13	RX1&RX2/RX3
	Divide Halfword Register	*DHR	11	
FIXED POINT LOGICAL INSTRUCTIONS	AND	N	3.25/3.75	RX1&RX2/RX3
	AND Register	*NR	1.0	
	AND Immediate	NI	2.5	
	AND Halfword	*NH	2.75/3.25	RX1&RX2/RX3
	AND Halfword Immediate	*NHI	1.75	
	OR	O	3.25/3.75	RX1&RX2/RX3
	OR Register	*OR	1.0	
	OR Immediate	OI	2.5	
	OR Halfword	*OH	2.75/3.25	RX1&RX2/RX3
	OR Halfword Immediate	*OHI	1.75	
	Exclusive OR	X	3.25/3.75	RX1&RX2/RX3
	Exclusive OR Register	*XR	1.0	
	Exclusive OR Immediate	XI	2.5	
	Exclusive OR Halfword	XH	2.75/3.25	RX1&RX2/RX3
	Exclusive OR Halfword Immediate	*XHI	1.75	
	Compare Logical	CL	3.25/3.75	RX1&RX2/RX3
	Compare Logical Register	*CLR	1.0	
	Compare Logical Immediate	CLI	2.5	
	Compare Logical Halfword	*CLH	2.75/3.25	RX1&RX2/RX3
	Compare Logical Halfword Immediate	*CLHI	1.75	
	Test Immediate	TI	2.5	
	Test Halfword Immediate	*THI	1.75	
SHIFT INSTRUCTIONS	Shift Right Logical	*SRL	3.0+.25(n−1)	n=no. of shifts
	Shift Right Fullword Logical Short	SRWLS	2.5+.25(n−1)	n=no. of shifts
	Shift Right Halfword Logical	*SRHL	2.75+.25(n−1)	n=no. of shifts
	Shift Right Logical Short	*SRLS	2.0+.25(n−1)	n=no. of shifts
	Shift Left Logical	*SLL	3.0+.25(n−1)	n=no. of shifts
	Shift Left Fullword Logical Short	SLWLS	2.5+.25(n−1)	n=no. of shifts
	Shift Left Halfword Logical	*SLHL	2.75+.25(n−1)	n=no. of shifts
	Shift Left Logical Short	*SLLS	2.0+.25(n−1)	n=no. of shifts
	Shift Right Arithmetic	*SRA	3.25+.25(n−1)	n=no. of shifts

**Compatible with 7/16 in Halfword Mode* *+Privileged Instruction*

INSTRUCTION REPERTOIRE

Type	Instruction	Mnemonic	Execution Time In μsec	Comments
SHIFT INSTRUCTIONS (Continued)	Shift Right Halfword Arithmetic	*SRHA	3.0+.25(n–1)	n=no. of shifts
	Shift Left Arithmetic	*SLA	3.5+.25(n–1)	n=no. of shifts
	Shift Left Halfword Arithmetic	*SLHA	3.25+.25(n–1)	n=no. of shifts
	Rotate Right Logical	*RRL	3.0+1.0n	n=no. of shifts
	Rotate Left Logical	*RLL	3.0+1.0n	n=no. of shifts
FLOATING POINT INSTRUCTIONS	Add	*AE	15/19/25	Min/Avg/Max
	Add Register	*AER	14/18/24	Min/Avg/Max
	Subtract	*SE	15/20/28	Min/Avg/Max
	Subtract Register	*SER	14/19/27	Min/Avg/Max
	Compare	*CE	11/13/10	+,+/–,–/+,–
	Compare Register	*CER	10/12/9	+,+/–,–/+,–
	Multiply	*ME	32/33/34	Min/Avg/Max
	Multiply Register	*MER	31/32/33	Min/Avg/Max
	Divide	*DE	55/55/57	Min/Avg/Max
	Divide Register	*DER	54/54/56	Min/Avg/Max
	Load	*LE	8/12/20	Min/Avg/Max
	Load Register	*LER	8/12/20	Min/Avg/Max
	Load Multiple	LME	4+3.07n	n= no. of regs.
	Store	*STE	6.0	
	Store Multiple	STME	4+3.0n	n= no. of regs.
	Fix	FXR	9	RR
	Float	FLR	12	RR
STATUS AND CONTROL INSTRUCTIONS	Load Program Status Word	+LPSW	4.75/5.25	RX1&RX2/RX3
	Load Program Status Word Register	+LPSWR	2	
	Exchange Program Status Register	+EPSR	1.5	
	Supervisor Call	SVC	5.5/6.0	RX1&RX2/RX3
LIST HANDLING INSTRUCTIONS	Add to Top of List	*ATL	5/13/13	OVF/MORM/WRAP
	Add to Bottom of List	*ABL	5/11/11	OVF/MORM/WRAP
	Remove From Top of List	*RTL	4/13/13	EMPTY/NORM/WRAP
	Remove From Bottom of List	*RBL	4/11.5/12	EMPTY/NORM/WRAP
INPUT/OUTPUT INSTRUCTIONS	Autoload	+AL	6/6.5+4n	RX1&RX2/RX3
	Simulate Interrupt	+SINT		
	Read Data	+RD	4/4.5	RX1&RX2/RX3
	Read Data Register	+RDR	2.0	
	Read Halfword	+RH	5/5.5	RX1&RX2/RX3
	Read Halfword Register	+RHR	2.75	
	Read Block	+RB	5.5+2.75n	
	Read Block Register	+RBR	5+2.75n	
	Write Data	+WD	4.0/4.5	RX1&RX2/RX3
	Write Data Register	+WDR	2.25	
	Write Halfword	+WH	4.5/5	RX1&RX2/RX3
	Write Halfword Register	+WHR	3.25	
	Write Block	+WB	5.5+2.75n	
	Write Block Register	+WBR	5+2.75n	
	Sense Status	+SS	4/4.5	RX1&RX2/RX3
	Sense Status Register	+SSR	2.25	
	Output Command	+OC	4.0/4.5	RX1&RX2/RX3
	Output Command Register	+OCR	2.25	
BYTE HANDLING INSTRUCTIONS	Load Byte	*LB	2.75/3.25	RX1&RX2/RX3
	Load Byte Register	*LBR	1.0	
	Store Byte	*STB	3.25/4	RX1&RX2/RX3
	Store Byte Register	*STBR	1.5	
	Exchange Byte Register	*EXBR	1.0	
	Compare Logical Byte	*CLB	3.25/3.75	RX1&RX2/RX3
BRANCH INSTRUCTIONS	Branch on True Condition Fullword	BTCW	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NO BR
	Branch on True Condition Fullword Register	BTCWR	1.5	
	Branch on True Condition Forward Fullword Short	BTFWS	2.0/1.5	
	Branch on True Condition Backword Fullword Short	BTBWS	2.0/1.5	BB/NO BR
	Branch on False Condition Fullword	BFCW	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NO BR
	Branch on False Condition Fullword Register	BFCWR	1.5	
	Branch on False Condition Forward Fullword Short	BFFWS	2.0/1.5	BR/NO BR
	Branch on False Condition Backward Fullword Short	BFBWS	2.0/1.5	BR/NO BR
	Branch on True Condition	*BTC	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NO BR
	Branch on True Condition Register	*BTCR	1.5	

**Compatible with 7/16 in Halfword Mode* *+Privileged Instruction*

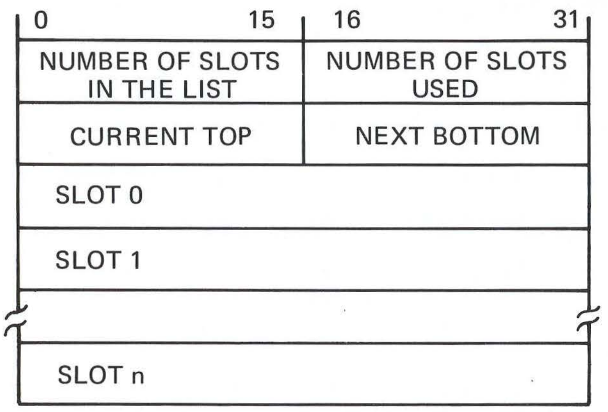
INSTRUCTION REPERTOIRE

Type	*Instruction*	*Mnemonic*	*Execution Time In µsec*	*Comments*
BRANCH INSTRUCTIONS (Continued)	Branch on True Condition Forward Short	*BTFS	2.0/1.5	BR/NO BR
	Branch on True Condition Backward Short	*BTBS	2.0/1.5	BR/NO BR
	Branch on False Condition	*BFC	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NO
	Branch on False Condition Register	*BFCR	1.5	BR
	Branch on False Condition Forward Short	*BFFS	2.0/1.5	BR/NO BR
	Branch on False Condition Backward Short	*BFBS	2.0/1.5	BR/NO BR
	Branch and Link	*BAL	2.0/2.5	
	Branch and Link Register	*BALR	1.75	
	Branch on Index High	*BH	5.5	
	Branch on Index Low or Equal	*BXLE	5.5	
COMMUNI-CATIONS INSTRUCTIONS	Cyclic Redundancy Check 12	CRC12	13.5	AVG.
	Cyclic Redundancy Check 16	CRC16	15.75	AVG.
	Translate	TLATE	4.5/5.25	Special CHAR./VALID CHAR.
	Load Byte High Speed Indirect	LBHI	7.75/8.75	NORM./TERMINATE
	Store Byte High Speed Indirect	STBHI	8.0/7.75	NORM./TERMINATE
BIT MANIPULATION INSTRUCTIONS	Test Bit	TBT	6.75/7.25	RXI&RX2/RX3 Avg.
	SET BIT	SBT	7/7.5	RX1&RX2/RX3 Avg.
	Reset Bit	RBT	7.25/7.75	BX1&RX2/RX3 Avg.
	Complement Bit	CBT	7/7.5	BX1&RX2/RX3 Avg.
EXTENDED BRANCH MNEMONICS	Branch on Carry	*BC	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Carry Register	*BCR	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on No Carry	*BNC	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on No Carry Register	*BNCR	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Equal	*BE	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Equal Register	*BER	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Not Equal	*BNE	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Not Equal Register	*BNER	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Low	*BL	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Low Register	*BLR	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Not Low	*BNL	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Not Low Register	*BNLR	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Minus	*BM	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Minus Register	*BMR	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Not Minus	*BNM	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Not Minus Register	*BNMR	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Plus	*BP	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Plus Register	*BPR	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Not Plus	*BNP	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Not Plus Register	*BNPR	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Overflow	*BO	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Overflow Register	*BOR	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch Unconditional	*B	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch Unconditional Register	*BR	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Zero	*BZ	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Zero Register	*BZR	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Not Zero	*BNZ	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Not Zero Register	*BNZR	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	No Operation	*NOP	2.0	
	No Operation Register	*NOPR	1.5	
	Branch on Carry Short	*BCS	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Not Carry Short	*BNCS	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Equal Short	*BES	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Not Equal Short	*BNES	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Low Short	*BLS	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Not Low Short	*BNLS	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Minus Short	*BMS	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Not Minus Short	*BMNS	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Plus Short	*BPS	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Not Plus Short	*BNPS	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Overflow Short	*BOS	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR

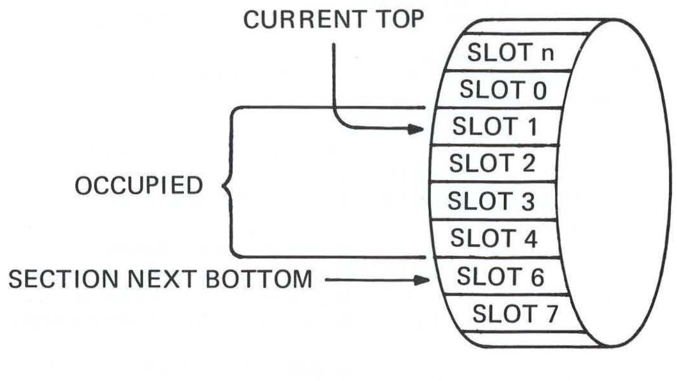
INSTRUCTION REPERTOIRE

Type	*Instruction*	*Mnemonic*	*Execution Time In µsec*	*Comments*
	Branch Unconditional Short	*B	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Zero Short	*BS	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Not Zero Short	*BNZS	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Carry Fullword	BCW	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Carry Fullword Register	BCWR	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on No Carry Fullword	BNCW	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on No Carry Fullword Register	BNCWR	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Equal Fullword	BEW	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Equal Fullword Register	BEWR	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Not Equal Fullword	BNEW	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Not Equal Fullword Register	BNEWR	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Low Fullword	BLW	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Low Fullword Register	BLWR	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Not Low Fullword	BNLW	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Not Low Fullword Register	BNLWR	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Minus Fullword	BMW	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Minus Fullword Register	BMWR	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Not Minus Fullword	BNMW	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Not Minus Fullword Register	BNMWR	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Plus Fullword	BPW	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Plus Fullword Register	BRWR	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Not Plus Fullword	BNPW	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Not Plus Fullword Register	BNPWR	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Overflow Fullword	BOW	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Overflow Fullword Register	BOWR	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Unconditional Fullword	BW	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Unconditional Fullword Register	BWR	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Zero Fullword	BZW	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Zero Fullword Register	BZWR	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Not Zero Fullword	BNZW	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Not Zero Fullword Register	BNZWR	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Carry Fullword Short	BCWS	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Not Carry Fullword Short	BNCWS	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Equal Fullword Short	BEWS	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Not Equal Fullword Short	BNEWS	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Low Fullword Short	BLWS	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Not Low Fullword Short	BNLWS	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Minus Fullword Short	BMWS	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Not Minus Fullword Short	BNMWS	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Plus Fullword Short	BPWS	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Not Plus Fullword Short	BNPWS	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Overflow Fullword Short	BOWS	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch Unconditional Fullword Short	BWS	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Zero Fullword Short	BZWS	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR
	Branch on Not Zero Fullword Short	BNZWS	2.0/2.5/2.0	RX1&RX2BR/RX3BR/NOBR

NOTE: Branch instructions with a Fullword Designation utilize the Fullword Condition Code All other Branch instructions use the Halfword Condition Code.



List Parameters

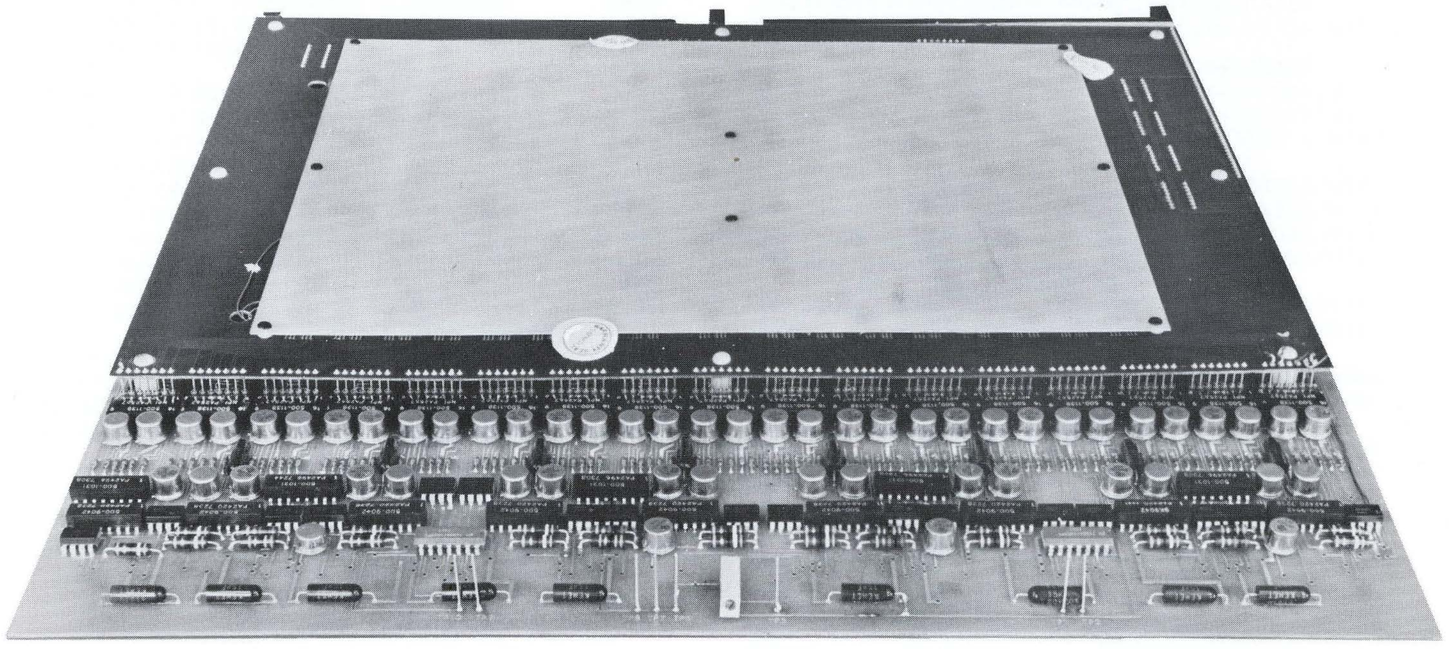


Circular List

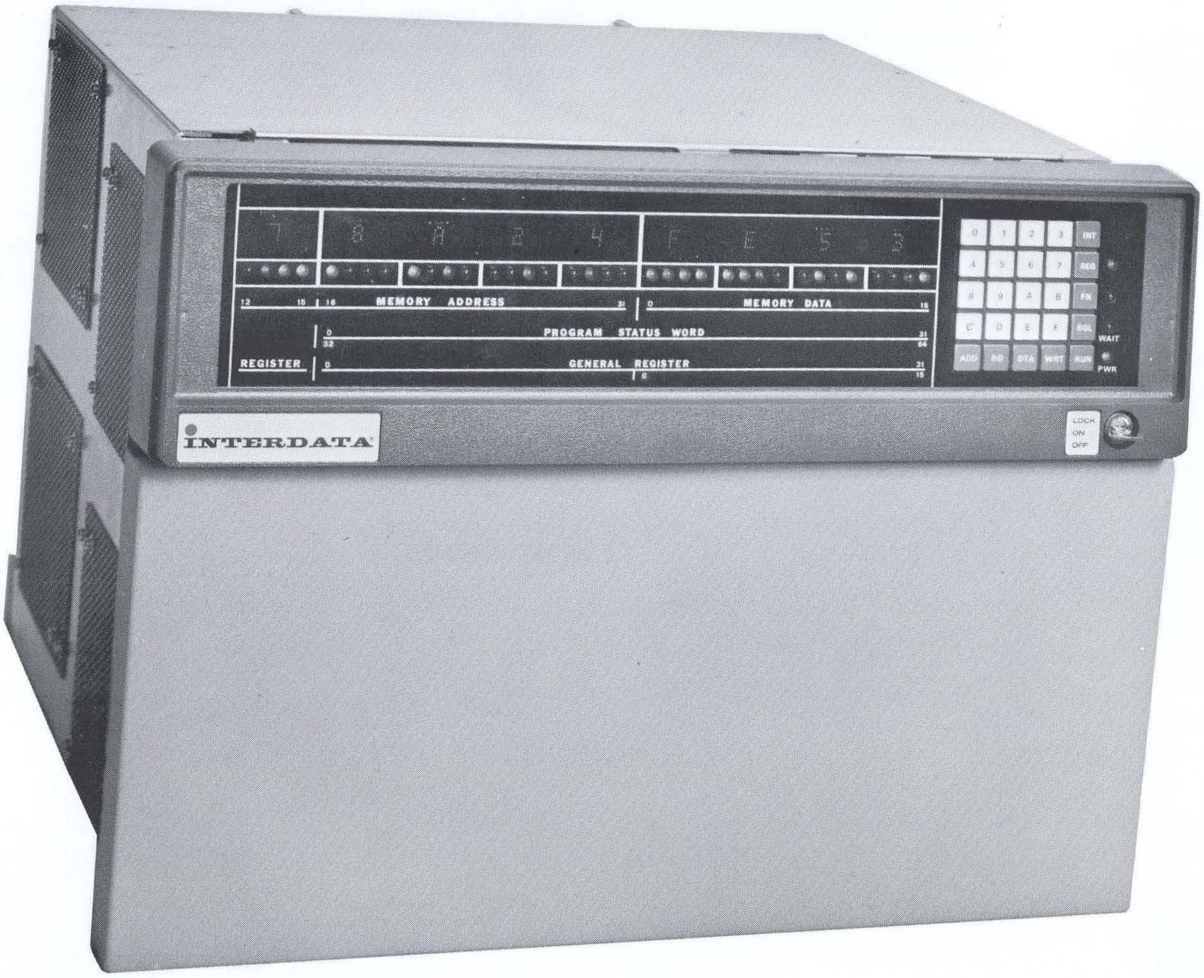
The List Processing instructions manipulate a circular list defined as follows: The first four halfwords contain the list parameters. Immediately following the parameter block is the list itself. The first fullword in the list is designated Slot 0. The remaining slots are designated 1, 2, 3, etc. up to a maximum of 65,535.

The first parameter halfword indicates the number of slots (fullwords) in the entire list. The second parameter halfword indicates the current number of slots being used. When this halfword equals zero, the list is empty; when this halfword equals the number of slots in the list, the list is full. Once initialized, this halfword is maintained automatically. It is incremented when elements are added to the list and decremented when elements are removed.

The third and fourth halfwords of the list parameters specify the current top of the list and the next bottom of the list respectively. These pointers are also updated automatically.



INTERDATA single board 32KB memory module. Both high speed (750 nanosecond) and standard (1000 nanosecond) version are available.



MEMORY ADDRESS
MEMORY DATA
PROGRAM STATUS WORD
REGISTER
GENERAL REGISTER
INT
REG
FN
SGL
ADD
RD
DTA
WRT
RUN
WAIT
PWR
INTERDATA
LOCK
ON
OFF

Sales and Service Offices:

Corporate Offices
2 Crescent Place
Oceanport, New Jersey 07757
(201) 229-4040

New York
121 Monmouth Parkway
West Long Branch
New Jersey 07764
(201) 229-4040

Boston
60 Hickory Drive
Waltham, Mass. 02154
(617) 890-0557

Washington
1800 North Kent Street
Suite 813
Rosslyn, Virginia 22209
(703) 525-4806

Philadelphia
Box K
Paoli, Pa. 19301
(215) 436-5579

Orlando
7200 Lake Ellenor Drive
Suite 142
Orlando, Fla. 32809
(305) 851-6962

Chicago
605 East Algonquin Road
Arlington Heights, Ill. 60005
(312) 437-5120

Detroit
20100 Civic Center Drive, Suite 213
Southfield, Michigan 48076
(313) 356-5515

Dayton
Financial South Office Park
5335 Far Hills Avenue
Kettering, Ohio 45429
(513) 434-4193

Kansas City
Clover Leaf Building No. 1
6811 West 63rd Street, Suite 204
Shawnee Mission, Kansas 66202
(913) 384-1606

Houston
6620 Harwin Drive
Houston, Texas 77036
(713) 783-3060

Dallas
300 N. Central Expressway
Richardson, Texas 75080
(214) 238-9656

Denver
1660 South Albion
Suite 225, Writers' Towers
Denver, Colorado 80222
(303) 758-0474

Los Angeles
888 No. Sepulveda Blvd., Suite 666
El Segundo, Calif. 90245
(213) 640-0451

Phoenix
1801 So. Jen Tilly Lane, Suite C-6
Tempe, Arizona 85281
(602) 968-2477

San Diego
7841 Balboa Avenue
San Diego, Calif. 92111
(714) 565-0602

San Francisco
1032 Elwell Court, Suite 240
Palo Alto, Calif. 94303
(415) 969-1180

Seattle
400 108th Ave., N.E., Suite 607
Bellevue, Washington 98004
(206) 455-0680

Toronto
6471 Northam Drive
Mississauga, Ontario
(416) 677-8990

Tokyo
Kyokuto Boeki Kaisha, Ltd.
C.P.O. Box 330
Tokyo, Japan
(270) 7711

Sydney
92 Chandos Street
St. Leonards
Sydney, Australia 2065
439-8400

London
Arundel Road
Uxbridge, Middlesex, England
Uxbridge 52441

Munich
8032 Grafelfing/Munich
Waldstr 3A
West Germany
0811-8543887

INTERDATA 2 Crescent Place, Oceanport, New Jersey 07757 (201) 229-4040 Printed in U.S.A. 79173I-1