VARIAN 75 SYSTEM SUPPLEMENT

varian data machines a varian subsidiary
2722 michelson drive p o box e irvine california 92664
1975 printed in USA

98 A 9906 220

SEPTEMBER 1975

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

SECTION 1 INTRODUCTION

This supplement, when used in conjunction with the Varian 74 System Handbook (document number 98A 9906 210), describes the Varian 75 Computer System.

The Varian 75 (figure 1-1) is a general-purpose, microprogrammed computer system for scientific, industrial, and data-communication applications. While maintaining full compatibility with all V70 series software and peripherals, the V75 features a significantly expanded instruction set. In addition to the assembly-language instructions available with other V70 series computers, the V75 has 27 instructions that permit a programmer to have access to 8 general-purpose registers and to operate on 8-bit, 16-bit, and 32-bit operands.

All V75 systems contain a 65,536-word memory with dual-port or single-port access and expansion capability to 262,144 words. Figure 1-2 shows the layout of components in the equipment cabinet.

Specifications for the V75 system are listed in table 1-1.

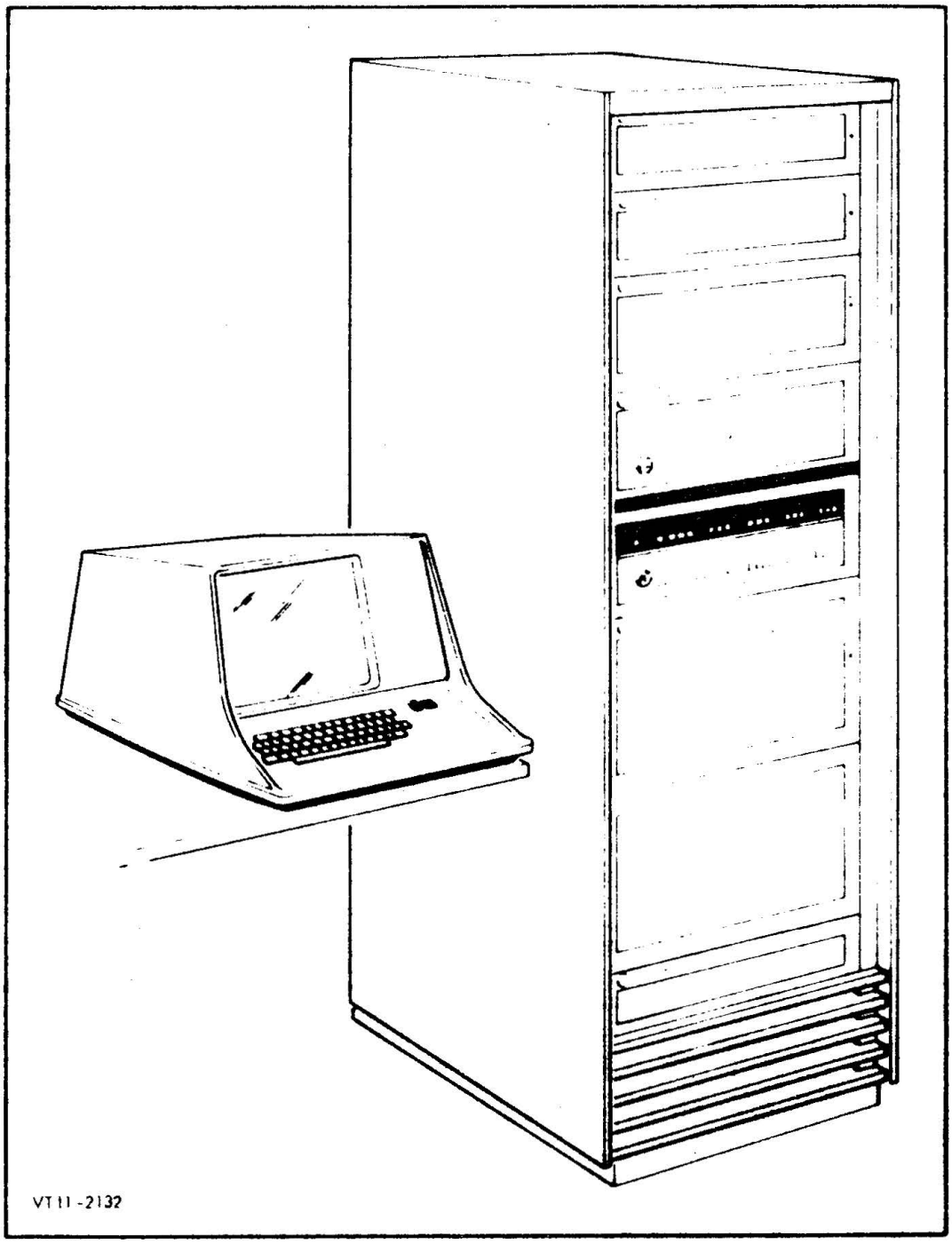
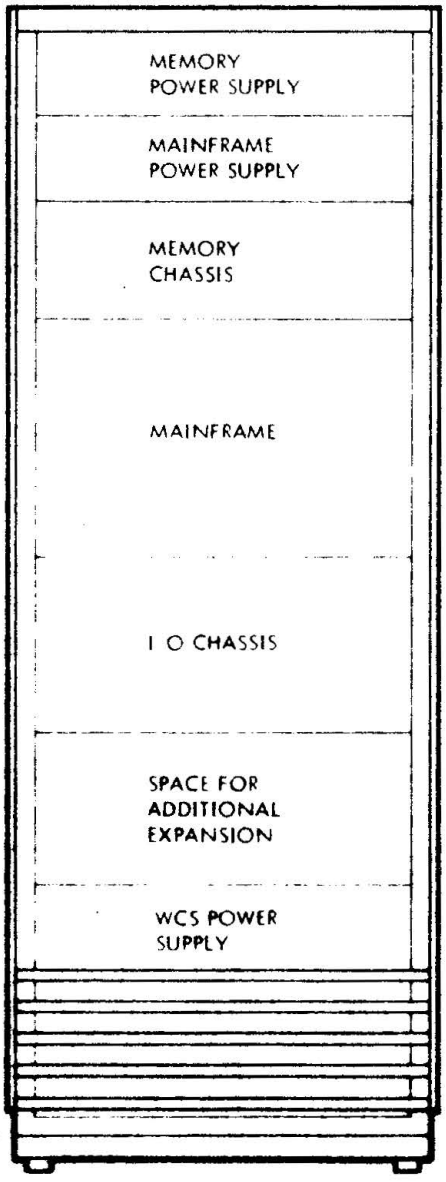


Figure 1-1. Varian 75 Computer System



NOTES:

1. THE I/O POWER SUPPLY IS INSTALLED IN THE REAR OF THE CABINET
2. MODELS 75-1000 AND 75-1050 DO NOT CONTAIN THE MEMORY POWER SUPPLY AND MEMORY CHASSIS.

VT11-3428

Figure 1-2. Equipment Cabinet Layout

Table 1-1. Specifications

Type	General-purpose microprogrammed digital computer.
Memory	
Semiconductor	Dual-port, 330-nanosecond cycle time, 16-bit word, and optional byte parity.
Core	Two 16-bit word core memories with optional byte parity are available: Dual-port with 450-nanosecond cycle time (660 nanoseconds without interleaving). Single-port with 800-nanosecond cycle time (990 nanseconds without interleaving).
Word length	8, 16, or 32 bits.
Registers	24 registers. 8 16-bit registers are addressable by the V75 instructions (7 of these registers can be used as index registers and 4 can be used as 2 double-word registers). 16 16-bit registers can be used for microprogramming (8 of these are special purpose).
Arithmetic	Binary twos complement
I/O Transfer Rates (Maximum rates are given for non-interleaved operation)	
Semiconductor Memory	DMA: 969,000 words per second. DMA (620 compatible): 372,900 words per second. PMA: 1,102,000 words per second (writing). 1,010,000 words per second (reading).
Core Memory (450/660 ns)	DMA: 897,800 words per second. DMA (620 compatible): 361,800 words per second. PMA: 1,010,000 words per second (writing). 932,000 words per second (reading).
Core Memory (800/990 ns)	DMA: 835,876 words per second. DMA (620 compatible): 352,112 words per second. PMA: 713,000 words per second (writing). 673,000 words per second (reading).
Special High-Speed I/O	A direct memory interface (DMI) that provides transfers up to 2,970,000 words per second on one or more buses for non-standard devices.
Instructions	187 standard, can be extended with writable control store. Floating point processor option provides an additional 14 instructions.

Table 1-1. Specifications *(continued)*

Addressing Modes	Byte addressing, word addressing, and double-word addressing, preindexed direct or indirect to 32,768 words using any of the 7 index registers. Direct to 2,048 words. Relative to P, X or B register to 512 words. Preindexing with X or B register. Multilevel indirect to 32,768 words. Indirect indexed. Immediate. Post indexing with X or B register. Extended mode to 32,768 words. Memory-map addressing to 262,144 words.
Standard features	Power failure/restart Real time clock Multiply/divide I/O bus with DMA Automatic bootstrap loaders (paper tape, rotating memory, and Teletype) Keyboard-CRT terminal Memory map Writable control store Core memory interleaving Priority memory access Programmers control panel 65,536 words of main memory Equipment cabinet Power for approximately 10 peripheral controllers I/O chassis with 19 I/O slots
Options	Byte parity Block transfer controller Priority interrupt module Buffer interface controller Floating point processor Additional writable control stores, up to a maximum of 3.
Logic levels	Internal (positive logic): High = +2.4 to 5.0V dc Low = 0 to +0.5V dc I/O bus (negative logic): High = +2.8 to +3.6V dc Low = 0 to +0.5V dc

Table 1-1. Specifications *(continued)*

Software	Language processors Macro assembler FORTRAN IV RPG II RPG IV BASIC COBOL Operating systems BEST real time MOS batch VORTEX and VORTEX II multi-tasking Data base management TOTAL Interactive Time-sharing subsystem (TSS) Microprogramming support Assembler Simulator Utility for loading and debugging Data communications VTAM Application software HASP/RJE Message switching All of the above software operated with VORTEX II.
Dimensions	Equipment cabinet is (outside dimensions) 77 inches high, 26 inches wide, and 36 inches deep (195.6 by 66 by 91.4 cm). The table-top keyboard-CRT terminal is approximately 15 inches high, 17 inches wide, and 27 inches deep (38.1 by 43.2 by 68.6 cm).
Input voltage	105 to 125V ac or 210 to 250V ac, at 50 or 60 Hz.
Input current	With 115V ac, the maximum ac current requirements for individual system components in the equipment cabinet are: 12 amperes ac for mainframe power supply, 12 amperes ac for memory power supply, 6 amperes ac for I/O power supply, 4 amperes ac for WCS power supply, and 3 am-

Table 1-1. Specifications *(continued)*

	peres ac for cabinet fans. The keyboard-CRT terminal requires 1.5 amperes ac.
Operational environment	0 to 50 degrees C (32 to 122 degrees F), 0 to 90 percent relative humidity without condensation. For keyboard-CRT terminal, operating temperature is 0 to 40 degrees C (32 to 104 degrees F).

SECTION 2 REGISTER USAGE

The Varian 75 computer system contains eight registers which are available to the programmer. Table 2-1 lists the nomenclature and functional descriptions of the V75 programming registers along with corresponding registers of the other V70 series computers.

Table 2-1. Assembly-Programming Registers

V75 Nomenclature	V75 Function	V70 Series Nomenclature	V70 Series Function
R0	Byte or word accumulator, or most-significant half of double-precision register R0-R1	A	Accumulator
R1	Word accumulator, index register, or least-significant half of double-precision register R0-R1	B	Accumulator or index register
R2	General-purpose register	X	Index register
R3	General-purpose register		
R4	General-purpose register or most-significant half of double precision register R4-R5		
R5	General-purpose register or least-significant half of double precision register R4-R5		
R6	General-purpose register		
R7	General-purpose register		

In the V75 system, contents of the eight programming registers can be displayed or altered by using the appropriate controls and indicators on the control panel. The binary codes for selecting the eight registers (using the REG SELECT switches) are listed in table 2-2. Refer to section 11 of the V74 System Handbook for V75 control-panel operation since the controls and indicators for the V74 and V75 computers are the same.

Table 2-2. Binary Codes for V75 Register Selection

REG SELECT Switches				Selected Register
8	4	2	1	
0	0	0	0	R0
0	0	0	1	R1
0	0	1	0	R2
0	0	1	1	R3
0	1	0	0	R4
0	1	0	1	R5
0	1	1	0	R6
0	1	1	1	R7
1	0	0	0*	
1	0	0	1*	
1	0	1	0*	
1	0	1	1*	
1	1	0	0*	
1	1	0	1*	
1	1	1	0*	
1	1	1	1*	

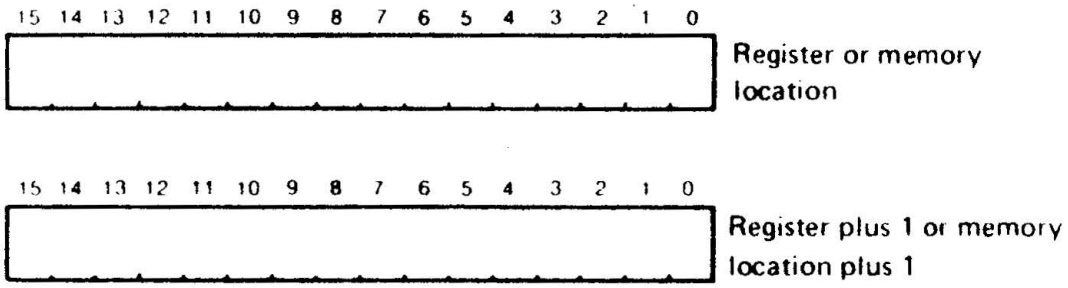
*These codes select registers that are used for WCS microprogramming. With two exceptions the contents of these registers can be displayed and altered using the control panel; however, alteration from the control panel should be done only for maintenance purposes or special applications. The register selected with the binary code of 1000 alway contains the contents of the instruction register. The registers selected with binary codes of 1011 and 1100 always contain all zeros and all ones, respectively; the contents of these two registers can not be altered from the control panel.

SECTION 3 DATA FORMATS

The V75 instructions operate on three additional data formats: double-precision nonarithmetic data, double-precision arithmetic data, and byte data.

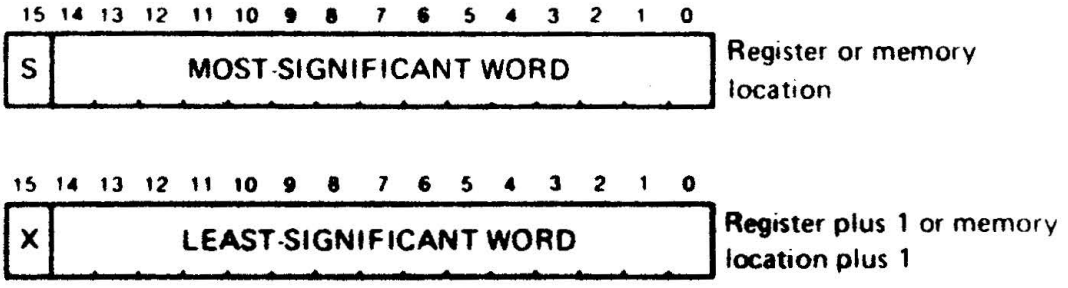
3.1 DOUBLE-PRECISION NONARITHMETIC DATA

The format for double-precision nonarithmetic data consists of a 32-bit unsigned operand stored in consecutive memory locations or registers as illustrated below:



3.2 DOUBLE-PRECISION ARITHMETIC DATA

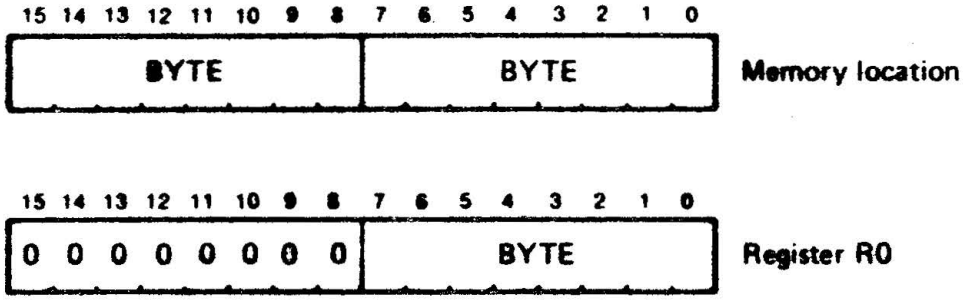
The format for double-precision arithmetic data consists of a 32-bit twos complement integer stored in consecutive memory locations or registers as illustrated below:



Note: Bit 15 of the least-significant word is not used.

3.3 BYTE DATA

The byte data format consists of 8-bit nonarithmetic characters stored as two bytes of a memory word in consecutive memory locations or as a single byte in the right half of register R0 (A) as illustrated below:



SECTION 4 ADDRESSING

In addition to the addressing modes for other V70 series computers, the V75 includes byte addressing. Byte addressing is used in conjunction with the two V75 byte instructions.

As shown in figures 4-1 and 4-2, byte addressing consists of an indexed mode and an indexed indirect mode. The base address word is summed with the contents of the index register (shifted arithmetically one bit to the right) to form the effective word address of the byte operand. The least-significant bit (bit 0) of the index register, referred to as the byte pointer (BP), determines the position of the byte operand. When the byte pointer is zero, the left byte (bits 8 through 15) is specified; when it is one, the right byte (bits 0 through 7) is specified. Indexed or indexed indirect addressing is specified by bit 15 (I bit) of the address. When the I bit is zero, indexed addressing is specified; when it is one, indexed indirect addressing is specified. If indexed indirect addressing is specified, postindexing is used. One level of indirect addressing is the maximum for byte instructions. For the other V75 instructions (non-byte instructions), up to four addressing modes are possible. These four modes are direct, indexed, indirect, and preindexed indirect.

All four addressing modes are used with the 2-word register-to-memory and double-precision instructions. The first word of both the register-to-memory and double-precision instructions contains the instruction field and the index register field (RX). The second word contains the indirect bit (I bit) and the address (ADDR). Each of the four addressing modes and the corresponding RX field and I-bit values for these instructions are listed in table 4-1.

Only direct or indirect addressing is used with the V75 jump-if instructions. The remaining V75 instructions have no addressing modes.

Table 4-1. Addressing Modes

Mode	RX Field	I Bit
Direct	0	0
Indexed	1 thru 7	0
Indirect	0	1
Preindexed indirect	1 thru 7	1

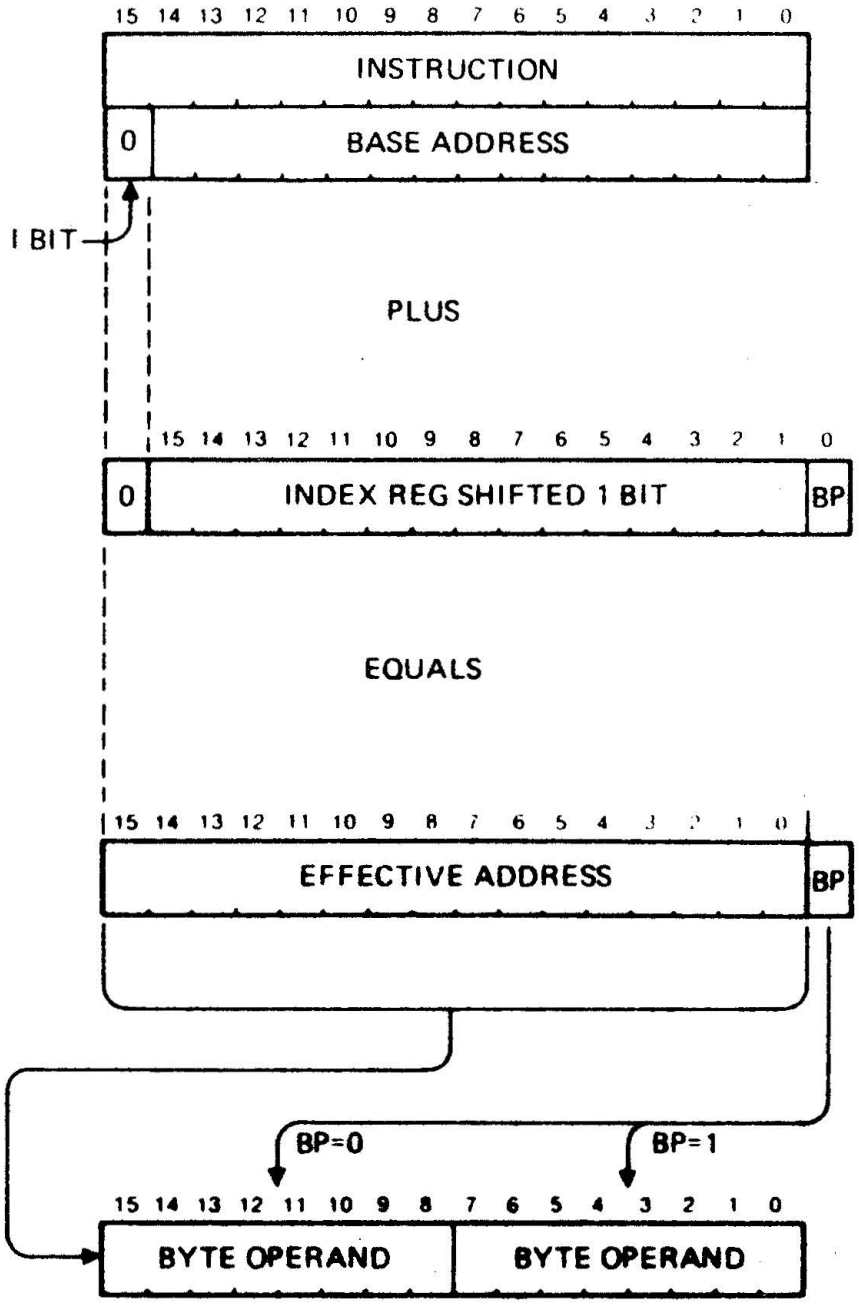


Figure 4-1. Byte Addressing, Indexed Mode

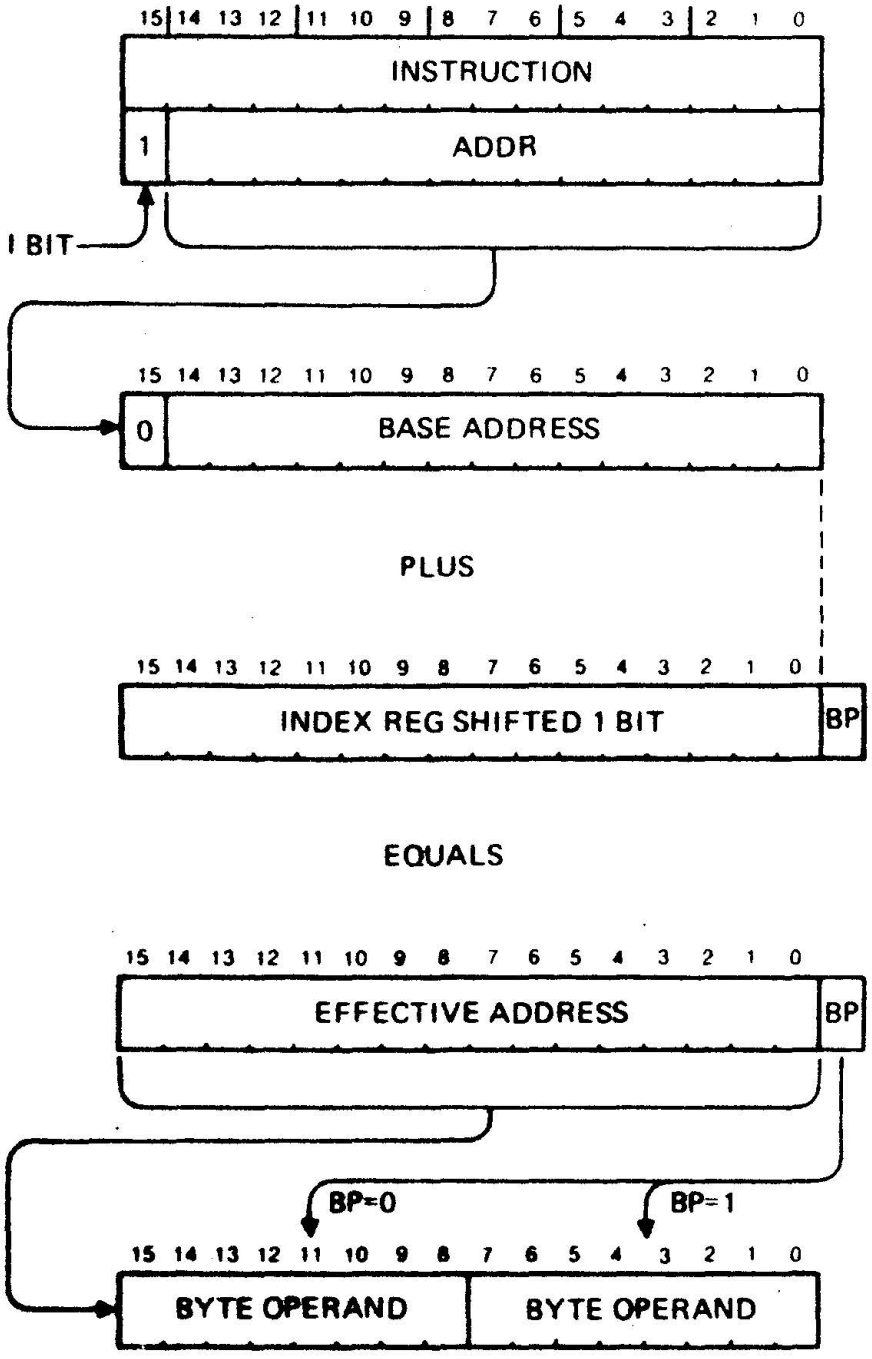


Figure 4-2. Byte Addressing, Indexed Indirect Mode

SECTION 5 INSTRUCTIONS

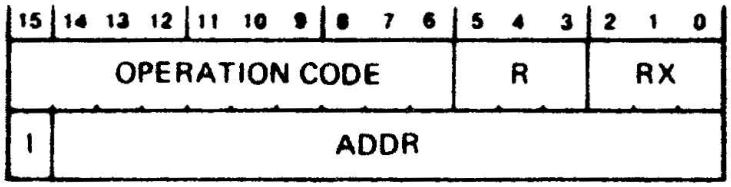
This section contains an explanation of new instruction formats and detailed specifications and execution times for each of the V75 instructions.

5.1 INSTRUCTION FORMATS

Each of the V75 instructions follow one of the seven instruction formats listed below. The first five are double-word formats and the last two are single-word formats.

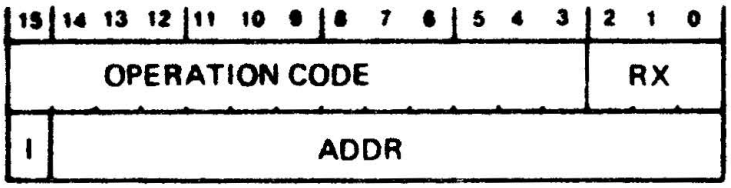
a. Register to Memory
b. Byte
c. Jump If
d. Double Precision
e. Immediate
f. Register to Register
g. Single Register

For **register-to-memory instructions**, the format is:



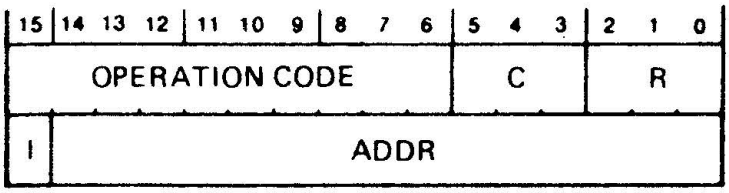
RX field (bits 0 through 2) specifies the index register.
R field (bits 3 through 5) specifies the source or destination register.
ADDR (second word) specifies the base address (see Section 4 for addressing modes).
I (second word) specified direct or indirect addressing (when I is one, indirect addressing is specified).

For **byte instructions**, the format is:



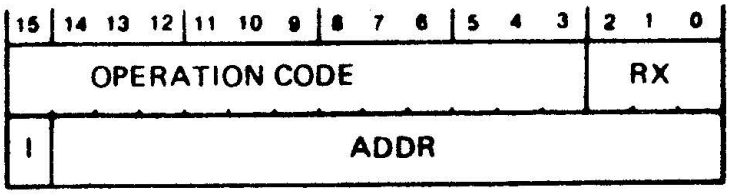
RX field (bits 0 through 2) specifies the index register.
ADDR (second word) specifies the base address (see Section 4 for addressing modes).
I (second word) specifies indexed or indexed indirect addressing (when I is one, indexed indirect addressing is specified).

For **jump-if instructions**, the format is:



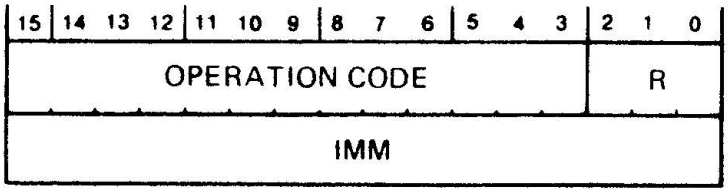
R field (bits 0 through 2) specifies the register to be tested.
C field (bits 3 through 5) specifies the test condition.
ADDR (second word) specifies the jump address (see Section 4 for addressing modes).
I (second word) specifies direct or indirect addressing (when I is one, indirect addressing is specified).

For **double-precision instructions**, the format is:



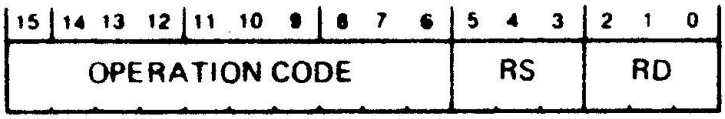
RX field (bits 0 through 2) specifies the index register.
ADDR (second word) specifies the base address (see Section 4 for addressing modes).
I (second word) specifies direct or indirect addressing (when I is one, indirect addressing is specified).

For **immediate instructions**, the format is:



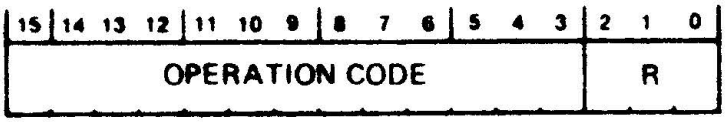
R field (bits 0 through 2) specifies the destination register.
IMM (second word) specifies the 16-bit immediate operand.

For **register-to-register instructions**, the format is:



RD field (bits 0 through 2) specifies the source register.
RS field (bits 3 through 5) specifies the destination register.

For **single register instructions**, the format is:



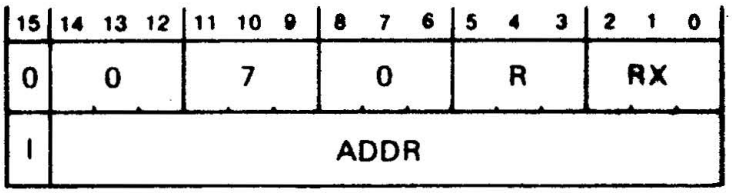
R field (bits 0 through 2) specifies the operand register.

5.2 INSTRUCTION SPECIFICATIONS

The instruction specifications for the V75 instructions are grouped according to the seven instruction formats listed in paragraph 5.1. The specification for each instruction includes: mnemonic, name, word diagram, format, type of addressing, and description.

5.2.1 Register-To-Memory Instructions

LD **Load**

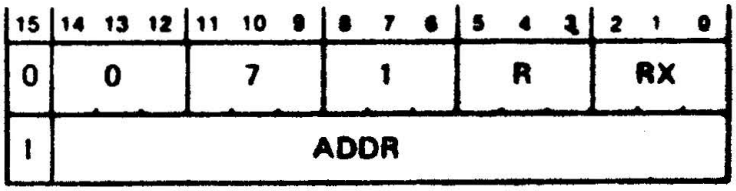


Format: Register-to-Memory
Addressing: Direct, Indexed, Indirect, Preindexed Indirect
Description: The contents of the effective memory address replace the contents of the register specified by the R field.

eg.

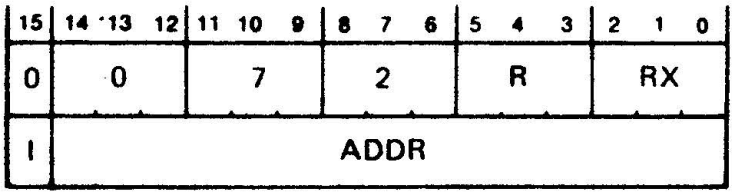
LD, R OFFSET, RX

ST **Store**



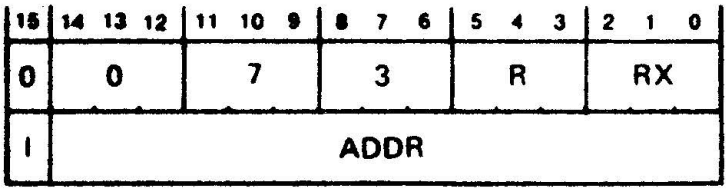
Format: Register to Memory
Addressing: Direct, Indexed, Indirect, Preindexed Indirect
Description: The contents of the register specified by the R field replace the contents of the effective memory address.

AD — Add



Format: Register to Memory
Addressing: Direct, Indexed, Indirect, Preindexed Indirect
Description: Contents of the register specified by the R field are added to the effective memory address. The twos complement of the sum replaces the contents of the register specified by the R field. If both operands have the same sign and the result has the opposite sign, the overflow flag is set.

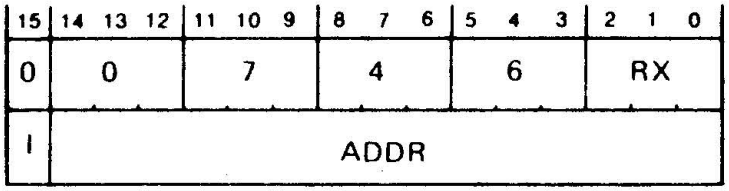
SB — Subtract



Format: Register to Memory
Addressing: Direct, Indexed, Indirect, Preindexed Indirect
Description: Contents of the effective memory address are subtracted from the contents of the register specified by the R field. The twos complement of the difference replaces the contents of the register specified by the R field. If the operands have opposite signs and the sign of the result equals the sign of the contents of the effective memory address, the overflow flag is set.

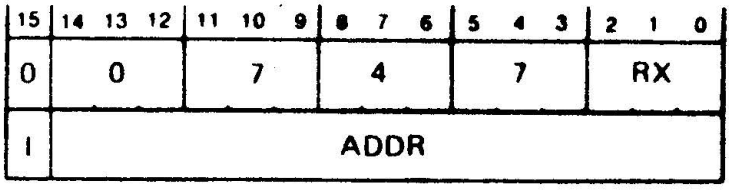
5.2.2 Byte Instructions

LBT **Load Byte**



Format: Byte
Addressing: Byte
Description: The contents of the effective byte address replaces the contents of the right byte of register R0. The contents of the left byte of register R0 are replaced with zeros.

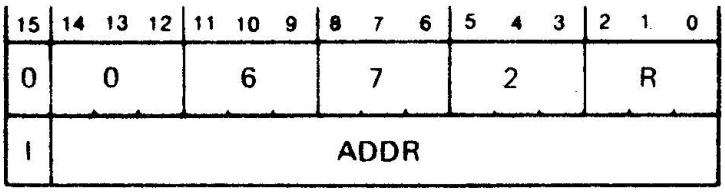
SBT **Store Byte**



Format: Byte
Addressing: Byte
Description: The contents of the right byte of register R0 replace the contents of the effective byte address.

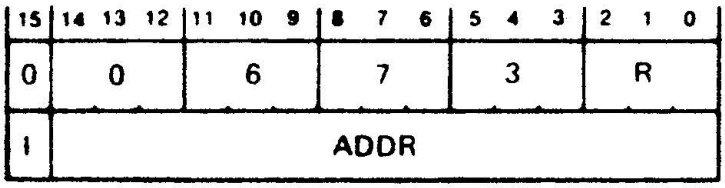
5.2.3 Jump-If Instructions

JZ — **Jump If Register Zero**



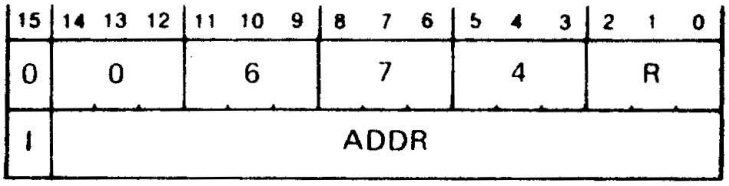
Format: Jump If
Addressing: Direct, Indirect
Description: If the register specified by the R field contains zero, the instruction at the effective jump address (ADDR) is executed. If the register (R) does not contain zero, the next instruction in sequence is executed. Contents of the register (R) are unaltered.

JNZ — **Jump If Register Not Zero**



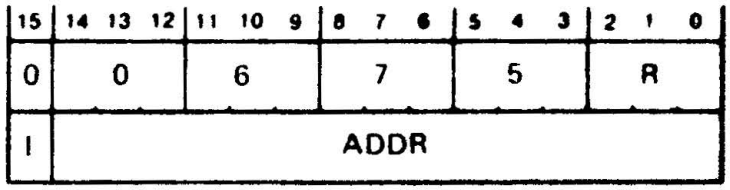
Format: Jump If
Addressing: Direct, Indirect
Description: If the register specified by the R field contains a value that is not zero, the instruction at the effective jump address (ADDR) is executed. If the register (R) contains zero, the next instruction in sequence is executed. Contents of the register (R) are unaltered.

JN — Jump If Register Negative



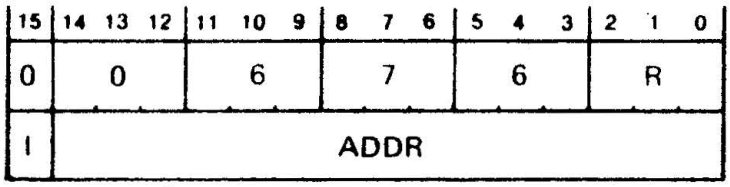
Format: Jump If
Addressing: Direct, Indirect
Description: If the register specified by the R field contains a negative value, the instruction at the effective jump address (ADDR) is executed. If the register (R) contains a positive value (including zero), the next instruction in sequence is executed. Contents of the register (R) are unaltered.

JP — Jump If Register Positive



Format: Jump If
Addressing: Direct, Indirect
Description: If the register specified by the R field contains a positive value (including zero), the instruction at the effective jump address (ADDR) is executed. If the register (R) contains a negative value, the next instruction in sequence is executed. Contents of the register (R) are unaltered.

JDZ — Jump If Double-Precision Register Zero

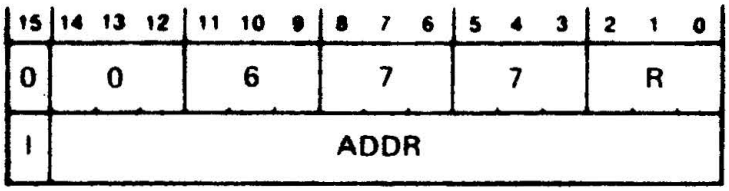


Format: Jump If

Addressing: Direct, Indirect

Description: If the double-precision register specified by the R field contains zero, the instruction at the effective jump address (ADDR) is executed. If the value of the R field is 0, double-precision register R0-R1 is specified; if the value is 4, double-precision register R4-R5 is specified. If the double-precision register (R) does not contain zero, the next instruction in sequence is executed. Contents of the double-precision register (R) are unaltered.

JDNZ — Jump If Double-Precision Register Not Zero



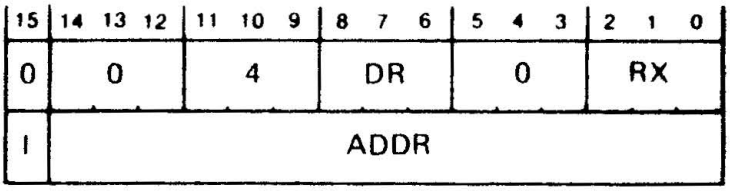
Format: Jump If

Addressing: Direct, Indirect

Description: If the double-precision register specified by the R field contains a value that is not zero, the instruction at the effective jump address (ADDR) is executed. If the value of the R field is 0, double-precision register R0-R1 is specified; if the value is 4, double-precision register R4-R5 is specified. If the double-precision register (R) contains zero, the next instruction in sequence is executed. Contents of the double-precision register (R) are unaltered.

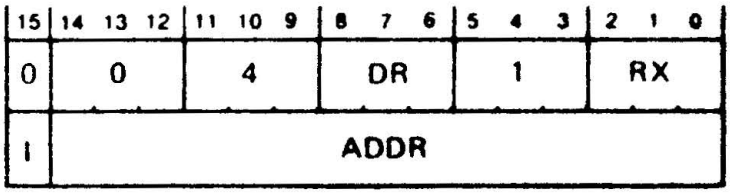
5.2.4 Double-Precision Instructions

DLD — **Double Load**



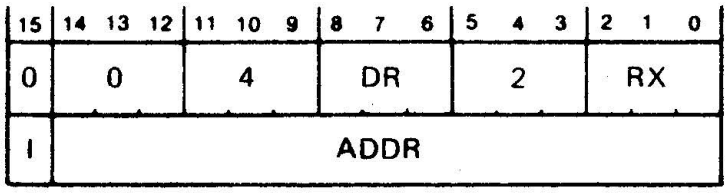
Format: Double Precision
Addressing: Direct, Indexed, Indirect, Preindexed Indirect
Description: Contents of the double-precision effective address replace the contents of the double-precision register specified by the DR field. If the value of the DR field is 6, double-precision register R0-R1 is specified; if the value is 7, double-precision register R4-R5 is specified.

DST — **Double Store**



Format: Double Precision
Addressing: Direct, Indexed, Indirect, Preindexed Indirect
Description: Contents of the double-precision register specified by the DR field replace the contents of the double-precision effective address. If the value of the DR field is 6, double-precision register R0-R1 is specified; if the value is 7, double-precision register R4-R5 is specified.

DADD — Double Add

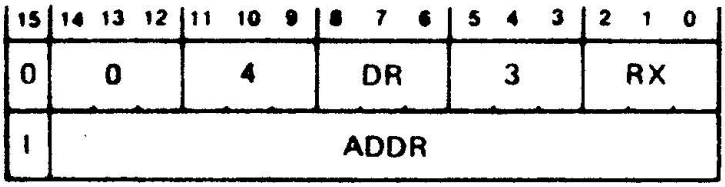


Format: Double Precision

Addressing: Direct, Indexed, Indirect, Preindexed Indirect

Description: Contents of the double-precision register specified by the DR field are added to the double-precision effective memory address. The twos complement of the sum replaces the contents of the double-precision register specified by the DR field. If the value of the DR field is 6, double-precision register R0-R1 is specified; if the value is 7, double-precision register R4-R5 is specified. If both double-precision operands have the same sign and the result has the opposite sign, the overflow flag is set.

DSUB — Double Subtract

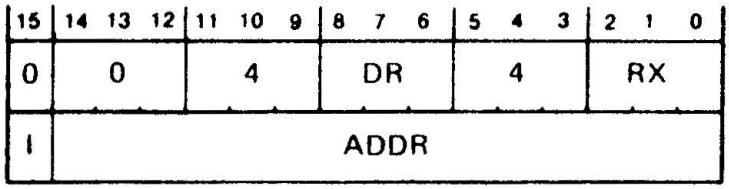


Format: Double Precision

Addressing: Direct, Indexed, Indirect, Preindexed Indirect

Description: Contents of the double-precision effective memory address are subtracted from the contents of the double-precision register specified by the DR field. The twos complement of the difference replaces the contents of the double-precision register specified by the DR field. If the value of the DR field is 6, double-precision register R0-R1 is specified; if the value is 7, double-precision register R4-R5 is specified. If the double-precision operands have opposite signs and the sign of the result does not equal the sign of the original contents of the specified double-precision register, the overflow flag is set.

DAN — Double And

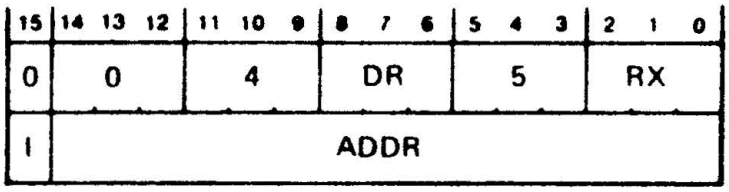


Format: Double Precision

Addressing: Direct, Indexed, Indirect, Preindexed Indirect

Description: A bit by bit logical AND function is formed between corresponding bits of the double-precision register specified by the DR field and the double-precision effective address. The logical results replace the contents of the specified double-precision register. If the value of the DR field is 6, double-precision register R0-R1 is specified; if the value is 7, double-precision register R4-R5 is specified.

DOR — Double Or

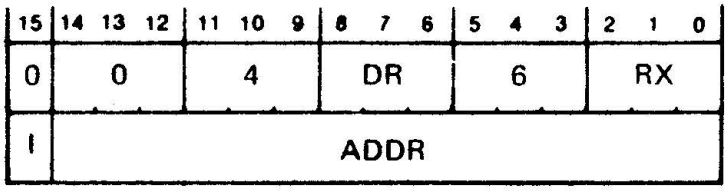


Format: Double Precision

Addressing: Direct, Indexed, Indirect, Preindexed Indirect

Description: A bit by bit logical OR function is formed between corresponding bits of the double-precision register specified by the DR field and the double-precision effective address. The logical results replace the contents of the specified double-precision register. If the value of the DR field is 6, double-precision register R0-R1 is specified; if the value is 7, double-precision register R4-R5 is specified.

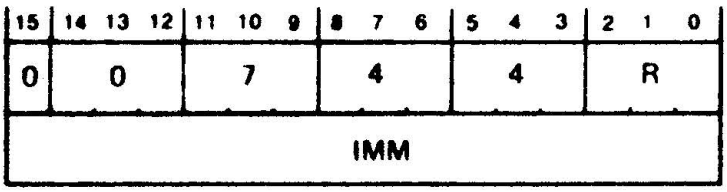
DER **Double Exclusive Or**



Format: Double Precision
Addressing: Direct, Indexed, Indirect, Preindexed Indirect
Description: A bit by bit logical EXCLUSIVE-OR function is formed between corresponding bits of the double-precision register specified by the DR field and the double-precision effective address. The logical results replace the contents of the specified double-precision register. If the value of the DR field is 6, double-precision register R0-R1 is specified; if the value is 7, double-precision register R4-R5 is specified.

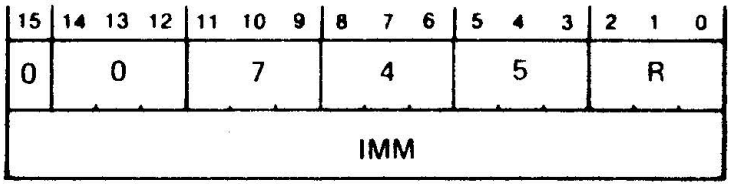
5.2.5 Immediate Instructions

LDI **Load Immediate**



Format: Immediate
Addressing: None
Description: The immediate operand (IMM) replaces the contents of the register specified by the R field.

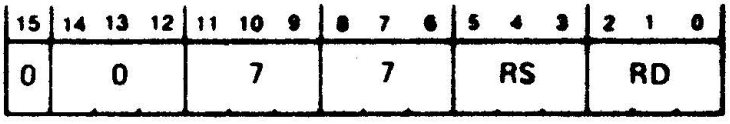
ADI **Add Immediate**



Format: Immediate
Addressing: None
Description: Contents of the register specified by the R field are added to the immediate operand (IMM). The twos complement of the sum replaces the contents of the register specified by the R field. If the operands have the same sign and the result has an opposite sign, the overflow flag is set.

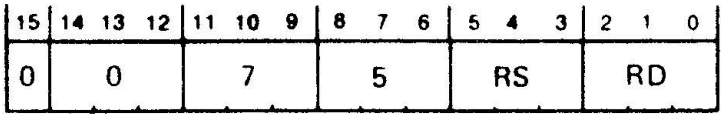
5.2.6 Register-To-Register Instructions

T **Transfer**



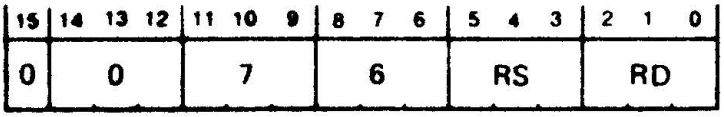
Format: Register to Register
Addressing: None
Description: The contents of the source register specified by the RS field replace the contents of the destination register specified by the RD field.

ADR — Add Register



Format: Register to Register
Addressing: None
Description: Contents of the source register specified by the RS field are added to the contents of the destination register specified by the RD field. The twos complement of the sum replaces the contents of the specified destination register. If both operands have the same sign and the result has the opposite sign, the overflow flag is set.

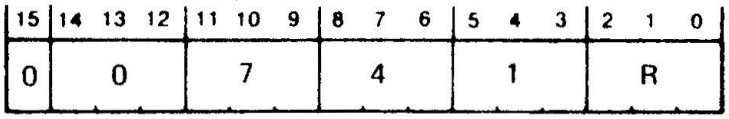
SBR — Subtract Register



Format: Register to Register
Addressing: None
Description: Contents of the source register specified by the RS field are subtracted from the contents of the destination register specified by the RD field. The twos complement of the difference replaces the contents of the specified destination register. If the operands have opposite signs and the sign of the result equals the sign of the contents of the specified source register, the overflow flag is set.

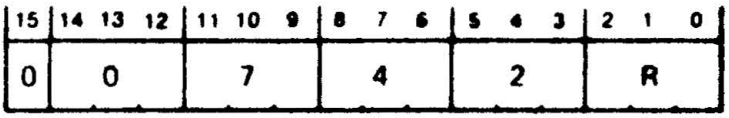
5.2.7 Single Register Instructions

INC — **Increment**



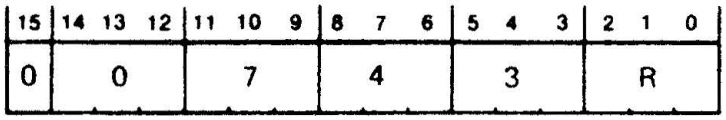
Format: Single Register
Addressing: None
Description: Contents of the register specified by the R field are incremented by 1. The twos complement of the incremented value replaces the contents of the specified register (R). If the specified register (R) contains an original value of 077777, the resulting value of the register becomes 10000 and the overflow flag is set.

DEC — **Decrement**



Format: Single Register
Addressing: None
Description: Contents of the register specified by the R field are decremented by 1. The twos complement of the decremented value replaces the contents of the specified register (R). If the specified register (R) contains an original value of 100000, the resulting value of the register becomes 0777777 and the overflow flag is set.

COM — Complement



Format: Single Register
Addressing: None
Description: The ones complement (logical inversion) of the contents of the register specified by the R field replaces the original contents of the specified register (R).

5.3 INSTRUCTION EXECUTION TIMES

Execution times for the V75 instructions are listed in table 5-1.

Table 5-1. Execution Times for V75 Instructions

		Execution Times In Nanoseconds		
		330 ns Mem.	660 ns Mem.	990 ns Mem.
Register-to-Memory Instructions				
LD	Load	1485	1980	2970
ST	Store	1815	2310	3960
AD	Add	1485	1980	2970
SB	Subtract	1485	1980	2970
Byte Instructions				
LBT	Load Byte	1815	2475	3135
SBT	Store Byte	1815	2475	3135
Jump-If Instructions				
JZ	Jump if Register Zero	990*	1320*	1980*
		1115**	1485**	1980**
JNZ	Jump if Register Not Zero	990*	1320*	1980*
		1115**	1485**	1980**
JN	Jump if Register Negative	990*	1320*	1980*
		1115**	1485**	1980**
JP	Jump if Register Positive	990*	1320*	1980*
		1115**	1485**	1980**

(continued)

Table 5-1. Execution Times for V75 Instructions

(continued)

		Execution Times In Nanoseconds		
		330 ns Mem.	660 ns Mem.	990 ns Mem.
JDZ	Jump if Double-Precision	1320*	1815*	2145*
	Register Zero	1155**	1485**	1980**
JDNZ	Jump if Double-Precision	990*	1320*	1980*
	Register Not Zero	1485**	1815**	2310**
		*Times are for conditions met.		
		**Times are for conditions not met.		
Double-Precision Instructions				
DLD	Double Load	2475	3135	4290
DST	Double Store	2805	3960	5940
DADD	Double Add	2640	3300	4455
DSUB	Double Subtract	2805	3465	4620
DAN	Double AND	2475	3135	4290
DOR	Double OR	2475	3135	4290
DER	Double Exclusive OR	2475	3135	4290
Register-to-Register Instructions				
T	Transfer	660	660	990
ADR	Add Register	660	660	990
SBR	Subtract Register	660	660	990
Single Register Instructions				
INC	Increment Register	990	990	990
DEC	Decrement Register	990	990	990
COM	Complement Register	990	990	990
Immediate Instructions				
LDI	Load Immediate	1320	1650	1980
ADI	Add Immediate	1320	1650	1980

SECTION 6 DAS ASSEMBLER

All V75 instructions are recognized by the V70 series assembler language (DAS). It should be noted that from the earliest Varian 620 software, the assembler syntax uses the convention that the X register is index register 1 and the B register is index register 2. However, the V70 emulation microprograms use hardware register R1 for the B register and hardware register R2 for the X register (section 2). The VORTEX DASMR assembler resolves this by mapping references to register R1 into references to hardware register R2 and vice versa. Thus, for V70 series instructions, references to the X register generate instructions referencing hardware register R2 (X register). Since the programmer is usually indifferent to the hardware register number assigned the X and B registers (except possibly a diagnostic programmer), this should cause no programming problems. If a diagnostic programmer does want to reference a particular hardware register, the register designation in his assembly statements should be written as follows:

a. To reference register R0 (A), write 0.
b. To reference register R1 (B), write 2.
c. To reference register R2 (X), write 1.
d. To reference registers R3 through R7, write 3 through 7, respectively.

The remainder of this section lists the V75 instruction mnemonics recognized by the DAS assembler and provides examples of assembly statements.

6.1 REGISTER-TO-MEMORY INSTRUCTIONS

Assembler mnemonics for the register-to-memory instructions are:

AD
LD
SB
ST

Example:

```
LD,0        0300,3
```

In the above example, register R0 is loaded with the contents of the memory address specified by the sum of 0300 and contents of register R3. Thus if register R3 contains 0200, the operand for this instruction is in memory address 0500.

6.2 BYTE INSTRUCTIONS

Assembler mnemonics for the byte instructions are:

LBT
SBT

Example:

```
SBT     0200,3
```

In the above example, contents of the right byte of register R0 are stored at the address specified by the sum of 0200 and contents of register R3 (shifted right one bit). Thus if register R3 contains 041, the operand is stored in the right byte at address 0220.

6.3 JUMP-IF INSTRUCTIONS

Assembler mnemonics for the jump-if instructions are:

JDNZ	JNZ
JDZ	JP
JN	JZ

Example:

```
JZ,3    ADDR
```

In the above example, the program jumps to the symbolic address ADDR if register R3 contains zero. If register R3 does not contain zero, the next instruction in sequence is executed.

6.4 DOUBLE-PRECISION INSTRUCTIONS

Assembler mnemonics for the double-precision instructions are:

DADD	DOR
DAN	DST
DER	DSUB
DLD	

Example:

```
DST,4   0200
```

In the above example, contents of double-precision register R4-R5 are stored at the two consecutive memory locations starting at address 0200.

6.5 IMMEDIATE INSTRUCTIONS

Assembler mnemonics for the immediate instructions are:

ADI
LDI

Example:

```
ADI,5   0642
```

In the above example, the immediate operand value of 0642 is added to the contents of register R5.

6.6 REGISTER-TO-REGISTER INSTRUCTIONS

Assembler mnemonics for the register-to-register instructions are:

ADR
SBR
T

Example:

T,3,4

In the above example, contents of register R3 are transferred to register R4.

6.7 SINGLE REGISTER INSTRUCTIONS

Assembler mnemonics for the single register instructions are:

COM
DEC
INC

Example:

INC,3

In the above example, contents of register R3 are incremented by 1.

INDEX OF INSTRUCTIONS