INSTRUCTION SET

The instructions that are implemented by hardware are described in detail on the pages that follow.

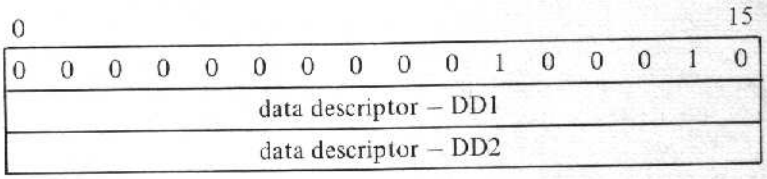
ACM

Instruction:
Alphanumeric compare

Restriction:
Traps (TV#5) on Models 23, 33, 43, and 53

Type:
Alphanumeric

Format:



Description:

The character string specified by the first operand is compared with the character string specified by the second operand. The comparison proceeds character-by-character from left to right. When a mismatch occurs, the binary values of the two unlike characters determine whether the first character string is greater or less than the second, and the G or L bit of the CIP indicator register is set accordingly.

If the lengths of the strings are unequal, the shorter string is unconditionally extended to the right with the fill character specified by the second data descriptor. Note that the extension takes place internally in the CIP and not in main memory.

If the lengths of both strings are zero, the strings are considered to be equal.

Both operands may specify IMOs.

If the value of the byte length specified by the first data descriptor is zero, the length is contained in the right byte of register R4 and can be from 0 through 255 bytes. If the value of the byte length specified in the first data descriptor is not zero, that value, which can be from 1 through 31, is the length.

If the value of the byte length specified by the second data descriptor is zero, then register R5 contains the fill character (in the left byte) and the length (in the right byte). When escape to register R5 occurs, the length can be from 1 through 255 characters. If the value of the byte length specified in the second data descriptor is not zero, that value is the length and the fill character is an ASCII blank (20 hexadecimal). In this case, the length can be from 1 through 31 bytes.

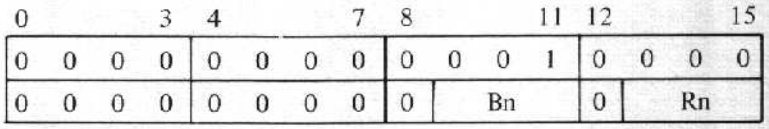
ACQ

Instruction:
Acquire stack space

Restriction:
Traps (TV#5) on Models 23 and 33

Type:
Generic

Format:



Description:
Acquires a portion of the remaining available stack space. The amount of space in words to be acquired is specified by Rn, and Bn is loaded with the leftmost address of the newly acquired space. CW is updated to the new stack length.

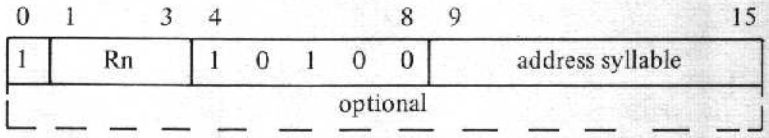
The contents of the I Register are unaffected. Trap vector #10 will occur if MW would be exceeded by the ACQ.

ADD

Instruction:
Add to R register

Type:
Double Operand

Format:



Description:
Adds the word at the effective address to the word contained in the designated R register. The carry (C) and overflow (OV) indicators will be set if carry and/or overflow occurs respectively; otherwise they will be cleared to zero. The address syllable can specify a memory location, an immediate operand, or another R register.

Some examples of addition are:

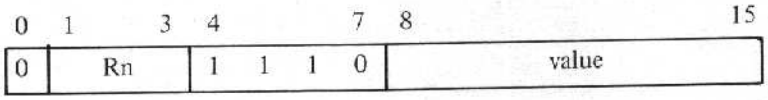
R Register (Before)	Effective Address (Before & After)	R Register (After)	C (After)	OV (After)
$+(32,766)_{10}$	+1	+32,767	0	0
$+(32,767)_{10}$	+1	-32,767	0	1
+1	-2	-1	0	0
-1	+2	+1	1	0
$(FFFF)_{16}$	+2	0001	1	0

ADV

Instruction:
Add value to R register

Type:
Short Value Immediate

Format:



Description:
Adds the value contained in the instruction (with the sign extended) to the designated R register. Overflow (OV) and carry (C) indicators are set/cleared according to the results of the addition.

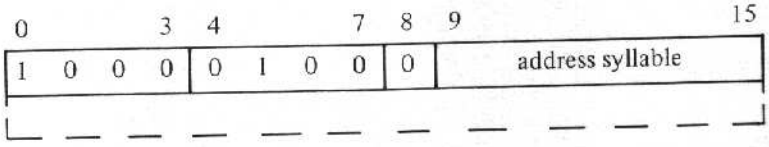
AID

Instruction:
Add double-word integer

Restriction:
Traps (TV#5) on Models 23 and 33

Type:
Single Operand

Format:



Description:
Loads the sum of the double-word integer contained in R6 and R7 and the contents of the double-word location specified by the address syllable into R6 and R7.

R6(0) is considered the high-order bit.

R7(15) is considered the low-order bit.

The contents of the I register are affected as follows:

- o If carry, C is set to 1; otherwise 0.
- o If overflow, OV is set to 1; otherwise 0.

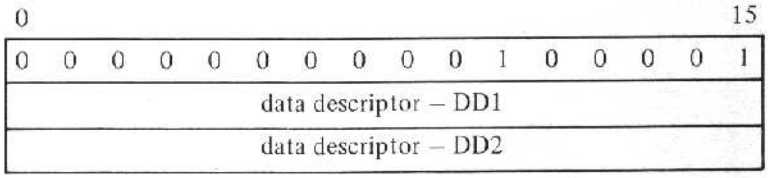
ALR

Instruction:
Alphanumeric Move

Restriction:
Traps (TV#5) on Models 23, 33, 43, and 53

Type:
Alphanumeric

Format:



Description:
The character string is moved from the address specified by the first operand (sending field) to the address specified by the second operand. If the length of the receiving field is zero, the TR-bit (truncation bit) of the CIP indicator register is set to 1, and the instruction is aborted. Trap 28, truncation, may then be generated.

The length of the first operand is contained in the first data descriptor (DD1) and can have a value from 1 through 31 bytes. If this length is zero, then the length is contained in the right byte of R4 and can be from 0 to 255 bytes.

The length of the second operand is contained in the second data descriptor (DD2) and can have a value from 1 through 31 bytes. If this length is zero, then R5 contains the length in the right byte and the fill character in the left byte. In this case, the length can be 0 through 255 bytes. If the length specified in the data descriptor is non-zero, then the fill character is an ASCII blank (20 hexadecimal). Fill control is specified by DD2.

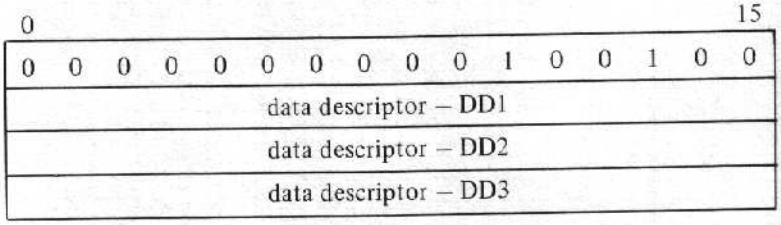
AME

Instruction:
Alphanumeric move and edit

Restriction:
Traps (TV#5) on Models 23, 33, 43, and 53

Type:
Edit

Format:



Description:

The character string in the sending field specified by the first data descriptor (DD1) is edited in accordance with the micro operations in the field specified by the third data descriptor (DD3), and moved to the receiving field specified by the second data descriptor (DD2).

The number of edited characters stored in the receiving field can be either more or less than the number in the sending field depending upon whether the micro operations require characters to be inserted in the receiving field or characters to be skipped from the sending field.

The instruction terminates normally when the receiving field is filled. Normal termination occurs even though the sending field or the string of micro operations have not been exhausted.

An illegal specification trap (Trap 26) is generated if either the sending field or the string of micro operations is exhausted before the receiving field is filled.

NOTE: Refer to Section 6 of the *GCOS 6 Assembly Language Reference* manual, Order No. CB07 for an explanation concerning micro edit operations which are required by the AME and DME edit move instructions.

AND

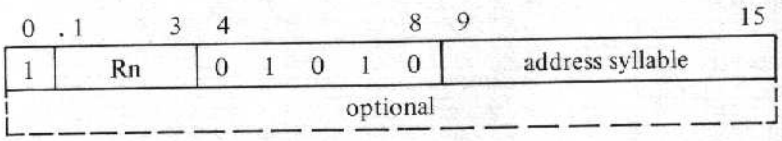
Instruction:

AND with R register

Type:

Double Operand

Format:



Description:

Logically ANDs the word at the effective address to the word contained in the designated R register. No indicators are affected. The address syllable can specify a memory address, an immediate operand, or another R register.

The following chart illustrates the result of logically ANDing bits:

First operand bit	0	0	1	1
Second operand bit	1	0	1	0
Result	0	0	1	0

Examples:

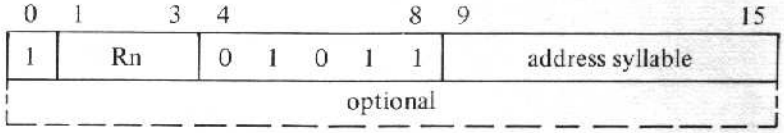
R Register (before)	Effective Address (before & after)	R Register (after)
$(ABCD)_{16}$	$(00FF)_{16}$	$(00CD)_{16}$
$(ABCD)_{16}$	$(7777)_{16}$	$(2345)_{16}$

ANH

Instruction:
AND half-word (byte) with R register

Type:
Double Operand

Format:



Description:
Logically ANDs the byte at the effective address (with its sign extended) to the word contained in the designated R register. No indicators are affected. The address syllable can specify a memory address, an immediate operand, or another R register, with the byte being positioned as follows:

– memory location, not indexed	– left byte
– memory location, indexed	
index value even	– left byte
index value odd	– right byte
– immediate operand	– left byte
– R register	– right byte

The following chart illustrates the result of logically ANDing bits:

First operand bit	0	0	1	1
Second operand bit	1	0	1	0
Result	0	0	1	0

Examples:

R Register (before)	Effective Address (before & after)	R Register (after)
$(ABCD)_{16}$	$(FF)_{16}$	$(ABCD)_{16}$
$(ABCD)_{16}$	$(77)_{16}$	$(0045)_{16}$

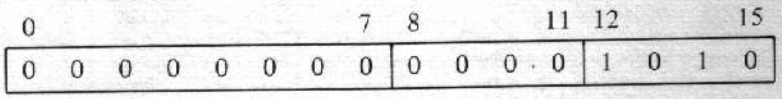
ASD

Instruction:
Activate segment descriptor

Restriction:
Traps (TV#5) on models without the Memory Management Unit option

Type:
Generic

Format:



Description:
Other than via a level change, this privileged instruction is the only way MMU segment descriptor tables can be changed.

The ASD instruction loads one segment descriptor into the MMU. Base register B5 specifies the segment, and the descriptor is contained in general registers R6 and R7. The new segment descriptor is effective immediately.

In using this instruction, the programmer must remember that the memory tables associated with each level are not updated by an ASD or by a level change. Therefore, if the new descriptor is to remain valid after an interrupt, it must also be loaded into the MMU image associated with the current level. Normally, to avoid anomalies resulting from interrupts between the two loads, the memory image should be loaded first, and it should be loaded by an (uninterruptible) SDI instruction.

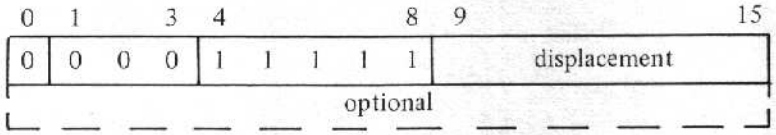
The contents of the I-Register are unaffected. If S(P) $\neq$ 1x, then trap vector #13 will occur.

B

Instruction:
Branch unconditionally

Type:
Branch on Indicator

Format:



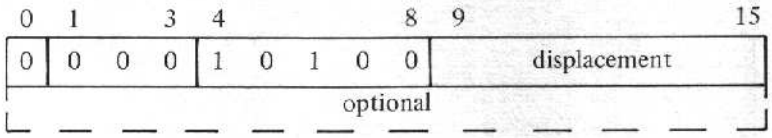
Description:
Branches to the specified location unconditionally. If the J bit in the mode control (M1) register is set, control then traps to trap vector #2.

BAG

Instruction:
Branch if algebraically greater than

Type:
Branch on Indicator

Format:



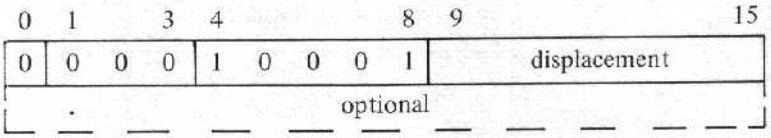
Description:
Branches to the specified location if either the greater than indicator (G) or the unlike signs indicator (U), but not both, is equal to one. If the J bit in the mode control (M1) register is set, and if the branch would have taken place, this instruction traps to trap vector #2.

BAGE

Instruction:
Branch if algebraically greater than or equal to

Type:
Branch on Indicator

Format:



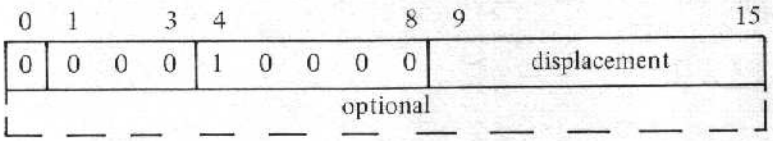
Description:
Branches to the specified location if both the less than indicator (L) and the unlike signs indicator (U), or neither is equal to one. If the J bit in the mode control (M1) register is set, and if the branch would have taken place, this instruction traps to trap vector #2.

BAL

Instruction:
Branch if algebraically less than

Type:
Branch on Indicator

Format:



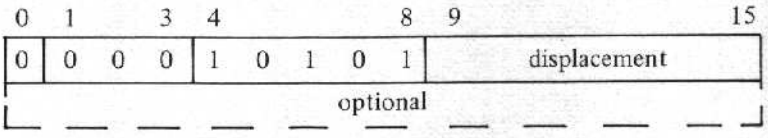
Description:
Branches to the specified location if either the less than indicator (L) or the unlike signs indicator (U), but not both, is equal to one. If the J bit in the mode control (M1) register is set, and if the branch would have taken place, this instruction traps to trap vector #2.

BALE

Instruction:
Branch if algebraically less than or equal to

Type:
Branch on Indicator

Format:



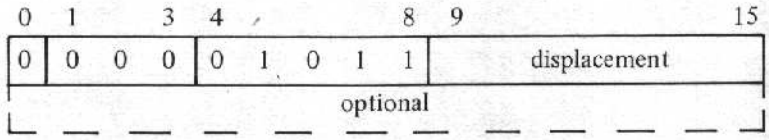
Description:
Branches to the specified location if both the greater than indicator (G) and the unlike signs indicator (U), or neither is equal to one. If the J bit in the mode control (M1) register is set, and if the branch would have taken place, this instruction traps to trap vector #2.

BBF

Instruction:
Branch if bit-test indicator false

Type:
Branch on Indicator

Format:



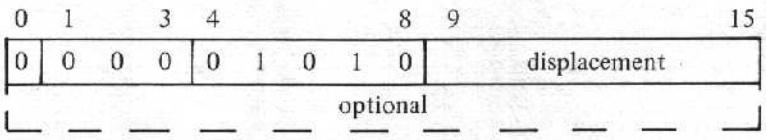
Description:
Branches to the specified location if the bit-test indicator (B) is equal to zero. If the J bit in the mode control (M1) register is set, and if the branch would have taken place, this instruction traps to trap vector #2.

BBT

Instruction:
Branch if bit-test indicator true

Type:
Branch on Indicator

Format:



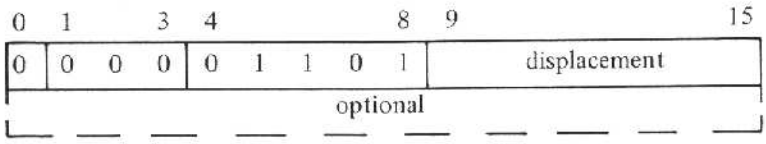
Description:
Branches to the specified location if the bit-test indicator (B) is equal to one. If the J bit in the mode control (M1) register is set, and if the branch would have taken place, this instruction traps to trap vector #2.

BCF

Instruction:
Branch if carry false

Type:
Branch on Indicator

Format:



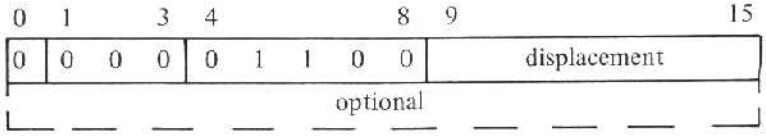
Description:
Branches to the specified location if the carry indicator (C) is equal to zero. If the J bit in the mode control (M1) register is set, and if the branch would have taken place, this instruction traps to trap vector #2.

BCT

Instruction:
Branch if carry true

Type:
Branch on Indicator

Format:



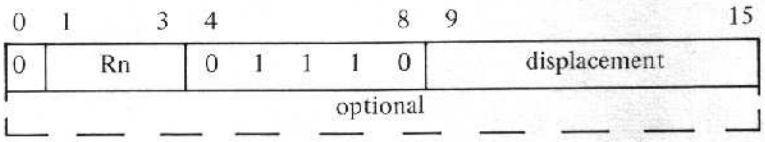
Description:
Branches to the specified location if the carry indicator (C) is equal to one. If the J bit in the mode control (M1) register is set, and if the branch would have taken place, this instruction traps to trap vector #2.

BDEC

Instruction:
Branch and decrement

Type:
Branch on Register

Format:



Description:

Subtracts one from the contents of the designated R register and then branches to the specified location if the result is not equal to −1 (i.e., branches unless the register initially contained zero). If the J bit in the mode control (M1) register is set, and if the branch would have taken place, this instruction traps to trap vector #2.

BE

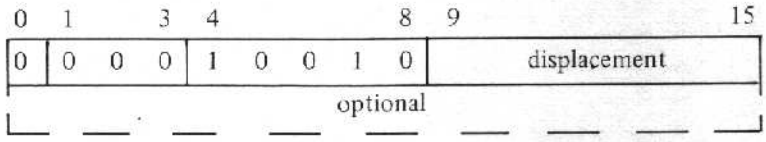
Instruction:

Branch if equal

Type:

Branch on Indicator

Format:



Description:

Branches to the specified location if neither the greater than indicator (G) nor the less than indicator (L) is equal to one. If the J bit in the mode control (M1) register is set, and if the branch would have taken place, this instruction traps to trap vector #2.

BEVN

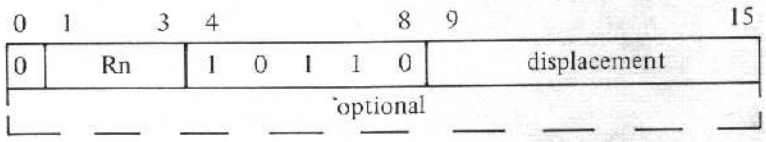
Instruction:

Branch if R register even

Type:

Branch on Register

Format:



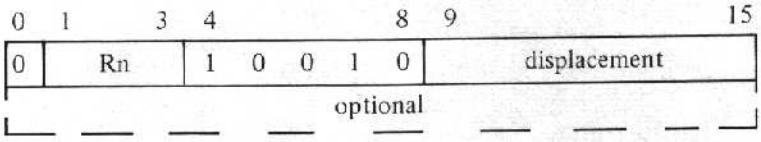
Description:
Branches to the specified location if bit 15 of the designated R register is equal to zero. If the J bit in the mode control (M1) register is set, and if the branch would have taken place, this instruction traps to trap vector #2.

BEZ

Instruction:
Branch if R register equal to zero

Type:
Branch on Register

Format:



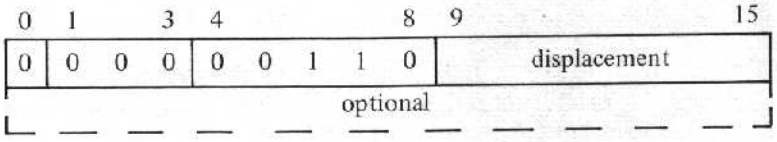
Description:
Branches to the specified location if the contents of the designated R register are equal to zero. If the J bit in the mode control (M1) register is set, and if the branch would have taken place, this instruction traps to trap vector #2.

BG

Instruction:
Branch if greater than

Type:
Branch on Indicator

Format:



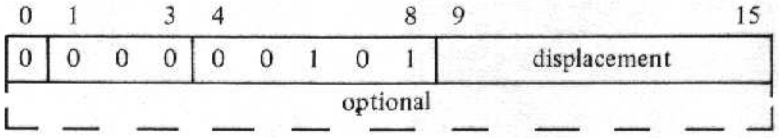
Description:
Branches to the specified location if the greater than indicator (G) is equal to one. If the J bit in the mode control (M1) register is set, and if the branch would have taken place, this instruction traps to trap vector #2.

BGE

Instruction:
Branch if greater than or equal to

Type:
Branch on Indicator

Format:



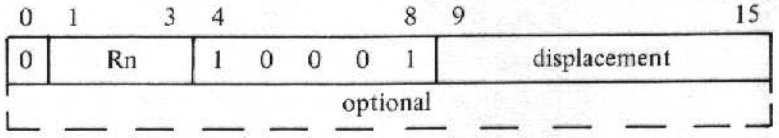
Description:
Branches to the specified location if the less than indicator (L) is equal to zero. If the J bit in the mode control (M1) register is set, and if the branch would have taken place, this instruction traps to trap vector #2.

BGEZ

Instruction:
Branch if R register greater than or equal to zero

Type:
Branch on Register

Format:



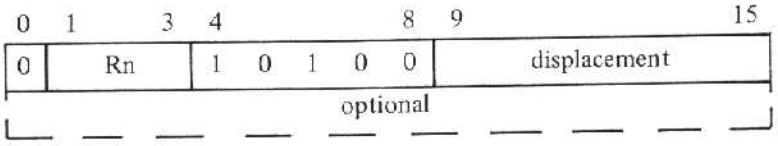
Description:
Branches to the specified location if bit zero of the designated R register is a zero. If the J bit in the mode control (M1) register is set, and if the branch would have taken place, this instruction traps to trap vector #2.

BGZ

Instruction:
Branch if R register greater than 0

Type:
Branch on Register

Format:



Description:

Branches to the specified location if bit zero of the designated R register is equal to zero and bits 1–15 are not equal to zero. If the J bit in the mode control (M1) register is set, and if the branch would have taken place, this instruction traps to trap vector #2.

BINC

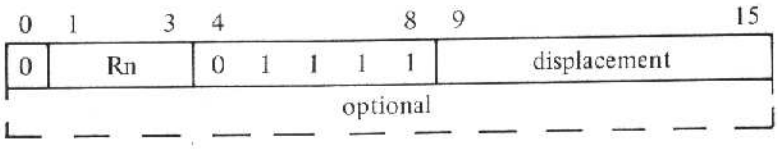
Instruction:

Branch and increment

Type:

Branch on Register

Format:



Description:

Adds one to the contents of the designated R register and then branches to the specified location if the result is not equal to zero (i.e., branches unless the register initially contains –1). If the J bit in the mode control (M1) register is set, and if the branch would have taken place, this instruction traps to trap vector #2.

BIOF

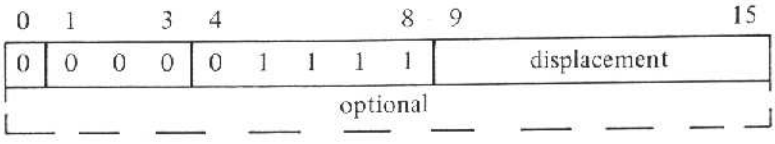
Instruction:

Branch if I/O indicator false

Type:

Branch on Indicator

Format:



Description:

Branches to the specified location if the I/O indicator (I) is equal to zero. If the J bit in the mode control (M1) register is set, and if the branch would have taken place, this instruction traps to trap vector #2.

BIOT

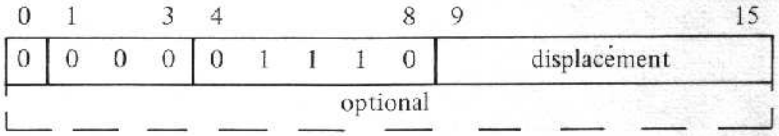
Instruction:

Branch if I/O indicator true

Type:

Branch on Indicator

Format:



Description:

Branches to the specified location if the I/O indicator (I) is equal to one. If the J bit in the mode control (M1) register is set, and if the branch would have taken place, this instruction traps to trap vector #2.

BL

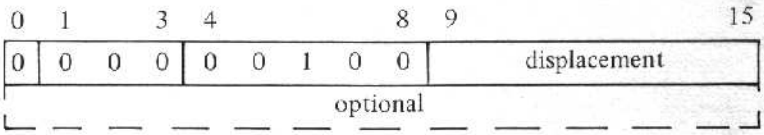
Instruction:

Branch if less than

Type:

Branch on Indicator

Format:



Description:

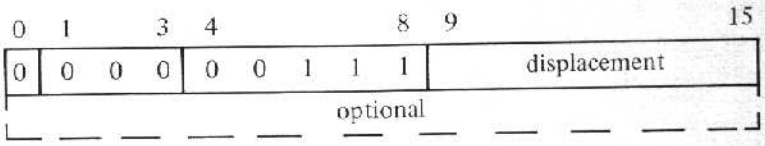
Branches to the specified location if the less than indicator (L) is equal to one. If the J bit in the mode control (M1) register is set, and if the branch would have taken place, this instruction traps to trap vector #2.

BLE

Instruction:
Branch if less than or equal to

Type:
Branch on Indicator

Format:



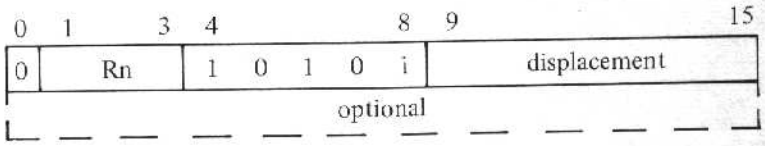
Description:
Branches to the specified location if the greater than indicator (G) is equal to zero. If the J bit in the mode control (M1) register is set, and if the branch would have taken place, this instruction traps to trap vector #2.

BLEZ

Instruction:
Branch if R register less than or equal to zero

Type:
Branch on Register

Format:



Description:
Branches to the specified location if bit zero of the designated R register is equal to one, or if the contents are equal to zero. If the J bit in the mode control (M1) register is set, and if the branch would have taken place, this instruction traps to trap vector #2.

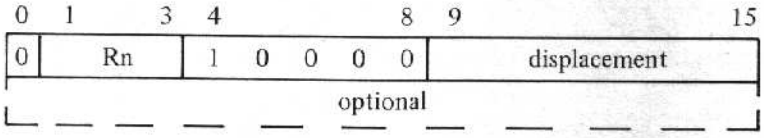
BLZ

Instruction:
Branch if R register less than zero

Type:
Branch on Register

BLZ/BNE/BNEZ

Format:



Description:

Branches to the specified location if bit zero of the designated R register is a one. If the J bit in the mode control (M1) register is equal to one, and if the branch would have taken place, this instruction traps to trap vector #2.

BNE

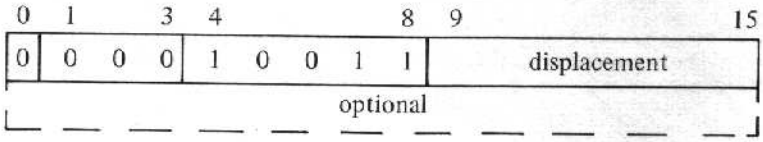
Instruction:

Branch if not equal

Type:

Branch on Indicator

Format:



Description:

Branches to the specified location if either the greater than indicator (G) or the less than indicator (L) is equal to one. If the J bit in the mode control (M1) register is set, and if the branch would have taken place, this instruction traps to trap vector #2.

BNEZ

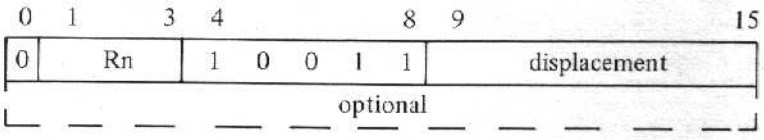
Instruction:

Branch if R register not equal to zero

Type:

Branch on Register

Format:



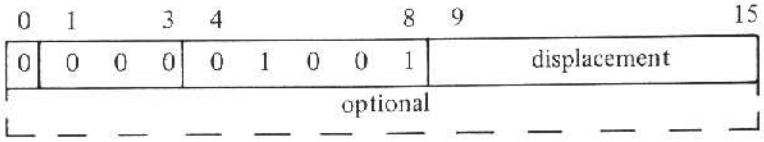
Description:
Branches to the specified location if the contents of the designated R register are not equal to zero. If the J bit in the mode control (M1) register is set, and if the branch would have taken place, this instruction traps to trap vector #2.

BNOV

Instruction:
Branch if no overflow

Type:
Branch on Indicator

Format:



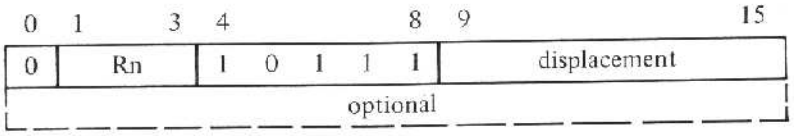
Description:
Branches to the specified location if the overflow indicator (OV) is equal to zero. If the J bit in the mode control (M1) register is set, and if the branch would have taken place, this instruction traps to trap vector #2.

BODD

Instruction:
Branch if R register odd

Type:
Branch on Register

Format:



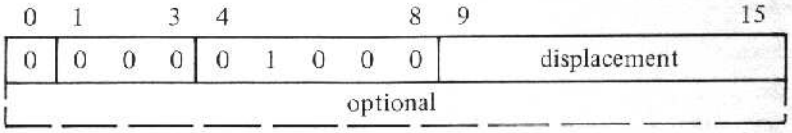
Description:
Branches to the specified location if bit 15 of the designated R register is equal to one. If the J bit in the mode control (M1) register is set, and if the branch would have taken place, this instruction traps to trap vector #2.

BOV

Instruction:
Branch if overflow

Type:
Branch on Indicator

Format:



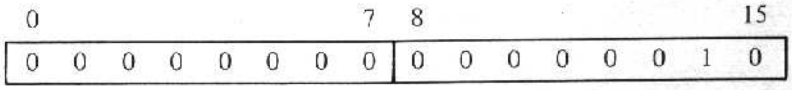
Description:
Branches to the specified location if the overflow indicator (OV) is equal to one. If the J bit in the mode control (M1) register is set, and if the branch would have taken place, this instruction traps to trap vector #2.

BRK

Instruction:
Breakpoint trap

Type:
Generic

Format:



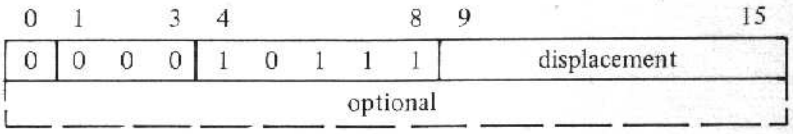
Description:
Causes a trap to trap vector #2; this instruction is used for debugging.

BSE

Instruction:
Branch if signs equal

Type:
Branch on Indicator

Format:



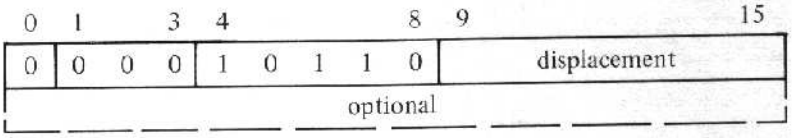
Description:
Branches to the specified location if the unlike signs indicator (U) is equal to zero. If the J bit in the mode control (M1) register is set, and if the branch would have taken place, this instruction traps to trap vector #2.

BSU

Instruction:
Branch if signs unlike

Type:
Branch on Indicator

Format:



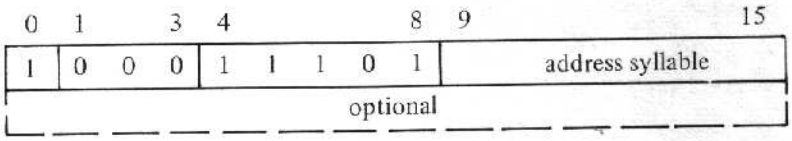
Description:
Branches to the specified location if the unlike signs indicator (U) is equal to one. If the J bit in the mode control (M1) register is set, and if the branch would have taken place, this instruction traps to trap vector #2.

CAD

Instruction:
Add carry bit

Type:
Single Operand

Format:



Description:
Adds the contents of the C bit in the I register to the contents of the location specified in the address syllable. The address syllable can specify a memory location, an immediate operand, or one of the R registers.

The contents of the I register are affected as follows:

- o If a carry occurs during the operation, the C bit is set to 1; otherwise, it is set to 0.
- o If the result is more than 16 bits long, the OV bit is set to 1; otherwise, it is set to 0. This cannot cause an overlap trap.

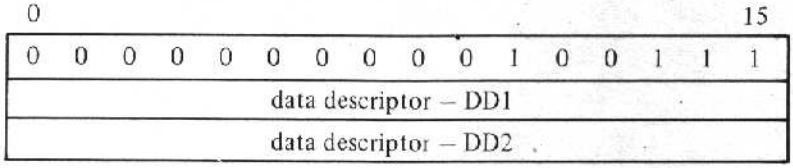
CBD

Instruction:
Convert binary to decimal

Restriction:
Traps (TV#5) on Models 23, 33, 43, and 53

Type:
Numeric

Format:



Description:
The signed binary value specified by the first operand is converted to the decimal data type specified by the second operand and stored, right justified, at the address specified by the second operand. The stored decimal integer can be 16 or 32 bits in length as specified in the data descriptor.

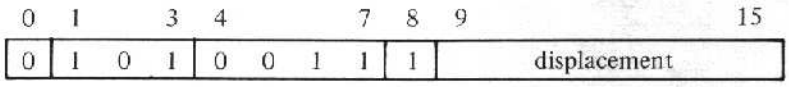
CBE

Instruction:
Branch if equal

Restriction:
Traps (TV#5) on Models 23, 33, 43, and 53

Type:
Commercial Branch

Format:



Description:
Branches to the location specified by the operand if the G-bit and the L-bit of the CIP indicator register are both 0.

If the J-bit in the M1 register contains a binary 1, the trace procedure is entered via trap vector #2. Upon completion, the trace procedure automatically branches to the address specified by the operand. In this case, or if the J-bit contains a binary 0, the instruction sequence starting at the location specified by the operand is executed.

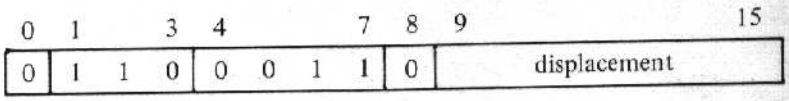
CBG

Instruction:
Branch if greater

Restriction:
Traps (TV#5) on Models 23, 33, 43, and 53

Type:
Commercial Branch

Format:



Description:
Branches to the location specified by the operand if the G-bit of the CIP indicator register is 1.

If the J-bit in the M1 register contains a binary 1, the trace procedure is entered via trap vector #2. Upon completion, the trace procedure automatically branches to the address specified by the operand. In this case, or if the J-bit contains a binary 0, the instruction sequence starting at the location specified by the operand is executed.

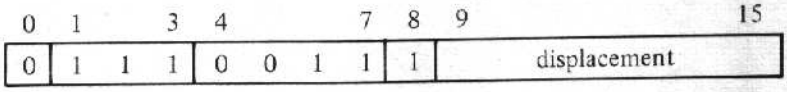
CBGE

Instruction:
Branch if greater than or equal

Restriction:
Traps (TV#5) on Models 23, 33, 43, and 53

Type:
Commercial Branch

Format:



Description:
Branches to the location specified by the operand if the L-bit of the CIP indicator register is 0.

If the J-bit in the M1 register contains a binary 1, the trace procedure is entered via trap vector #2. Upon completion, the trace procedure automatically branches to the address specified by the operand. In this case, or if the J-bit contains a binary 0, the instruction sequence starting at the location specified by the operand is executed.

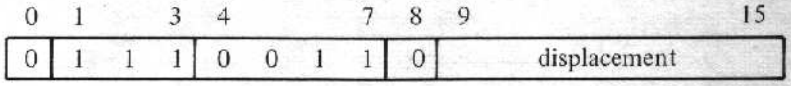
CBL

Instruction:
Branch if less

Restriction:
Traps (TV#5) on Models 23, 33, 43, and 53

Type:
Commercial Branch

Format:



Description:
Branches to the location specified by the operand if the L-bit of the CIP indicator register is 1.

If the J-bit in the M1 register contains a binary 1, the trace procedure is entered via trap vector #2. Upon completion, the trace procedure automatically branches to the address specified by the operand. In this case, or if the J-bit contains a binary 0, the instruction sequence starting at the location specified by the operand is executed.

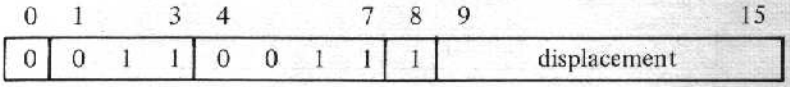
CBLE

Instruction:
Branch if less than or equal

Restriction:
Traps (TV#5) on Models 23, 33, 43, and 53

Type:
Commercial Branch

Format:



Description:
Branches to the location specified by the operand if the G-bit of the CIP indicator register is 0.

If the J bit in the M1 register contains a binary 1, the trace procedure is entered via trap vector #2. Upon completion, the trace procedure automatically branches to the address specified by the operand. In this case, or if the J-bit contains a binary 0, the instruction sequence starting at the location specified by the operand is executed.

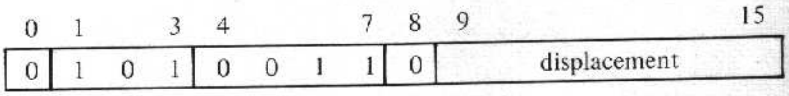
CBNE

Instruction:
Branch if not equal

Restriction:
Traps (TV#5) on Models 23, 33, 43, and 53

Type:
Commercial Branch

Format:



Description:
Branches to the location specified by the operand if either the G-bit or the L-bit of the CIP indicator register is 1.

If the J-bit in the M1 register contains a binary 1, the trace procedure is entered via trap vector #2. Upon completion, the trace procedure automatically branches to the address specified by the operand. In this case, or if the J-bit contains a binary 0, the instruction sequence starting at the location specified by the operand is executed.

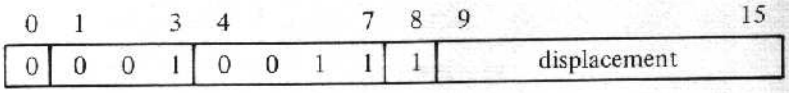
CBNOV

Instruction:
Branch on no overflow

Restriction:
Traps (TV#5) on Models 23, 33, 43, and 53

Type:
Commercial Branch

Format:



Description:

Branches to the location specified by the operand if the OV-bit of the CIP indicator register is 0.

If the J-bit in the M1 register contains a binary 1, the trace procedure is entered via trap vector #2. Upon completion, the trace procedure automatically branches to the address specified by the operand. In this case, or if the J-bit contains a binary 0, the instruction sequence starting at the location specified by the operand is executed.

CBNSF

Instruction:

Branch on no sign fault

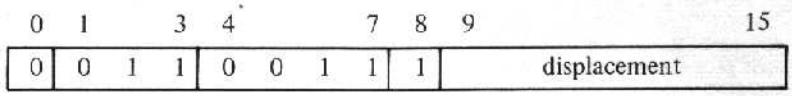
Restriction:

Traps (TV#5) on Models 23, 33, 43, and 53

Type:

Commercial Branch

Format:



Description:

Branches to the location specified by the operand if the SF-bit of the CIP indicator register is 0.

If the J-bit in the M1 register contains a binary 1, the trace procedure is entered via trap vector #2. Upon completion, the trace procedure automatically branches to the address specified by the operand. In this case, or if the J-bit contains a binary 0, the instruction sequence starting at the location specified by the operand is executed.

CBNTR

Instruction:

Branch on no truncation

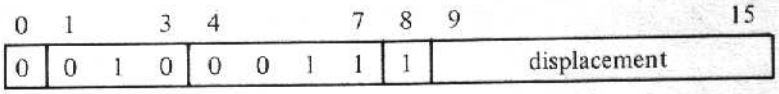
Restriction:

Traps (TV#5) on Models 23, 33, 43, and 53

Type:

Commercial Branch

Format:



Description:

Branches to the location specified by the operand if the TR-bit of the CIP indicator register is 0.

If the J-bit in the M1 register contains a binary 1, the trace procedure is entered via trap vector #2. Upon completion, the trace procedure automatically branches to the address specified by the operand. In this case, or if the J-bit contains a binary 0, the instruction sequence starting at the location specified by the operand is executed.

CBOV

Instruction:

Branch on overflow

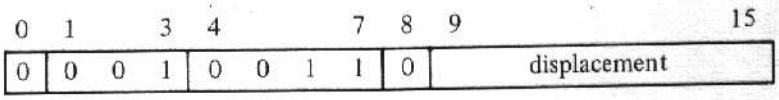
Restriction:

Traps (TV#5) on Models 23, 33, 43, and 53

Type:

Commercial Branch

Format:



Description:

Branches to the location specified by the operand if the OV-bit of the CIP indicator register is 1.

If the J-bit in the M1 register contains a binary 1, the trace procedure is entered via trap vector #2. Upon completion, the trace procedure automatically branches to the address specified by the operand. In this case, or if the J-bit contains a binary 0, the instruction sequence starting at the location specified by the operand is executed.

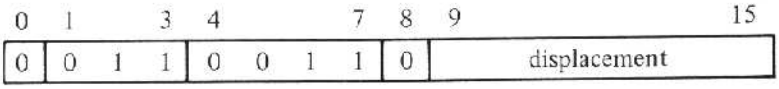
CBSF

Instruction:
Branch on sign fault

Restriction:
Traps (TV#5) on Models 23, 33, 43, and 53

Type:
Commercial Branch

Format:



Description:
Branches to the location specified by the operand if the SF-bit of the CIP indicator register is 1.

If the J-bit in the M1 register contains a binary 1, the trace procedure is entered via trap vector #2. Upon completion, the trace procedure automatically branches to the address specified by the operand. In this case, or if the J-bit contains a binary 0, the instruction sequence starting at the location specified by the operand is executed.

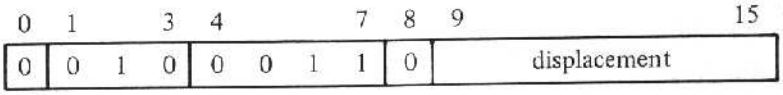
CBTR

Instruction:
Branch on truncation

Restriction:
Traps (TV#5) on Models 23, 33, 43, and 53

Type:
Commercial Branch

Format:



Description:
Branches to the location specified by the operand if the TR-bit of the CIP indicator register is 1.

If the J-bit in the M1 register contains a binary 1, the trace procedure is entered via trap vector #2. Upon completion, the trace procedure automatically branches to the address specified by the operand. In this case, or if the J-bit contains a binary 0, the instruction sequence starting at the location specified by the operand is executed.

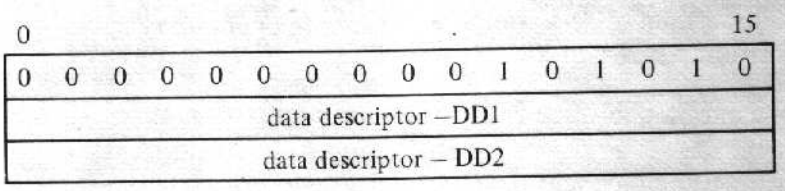
CDB

Instruction:
Convert decimal to binary

Restriction:
Traps (TV#5) on Models 23, 33, 43, and 53

Type:
Numeric

Format:



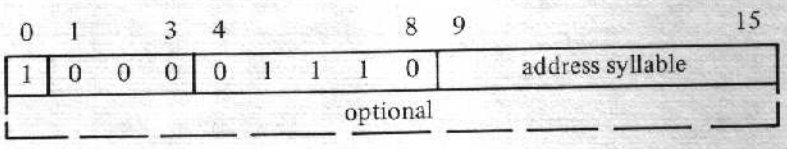
Description:
The decimal value specified by the first operand is converted to binary and stored in two's complement notation at the address specified by the second operand. The stored binary integer can be 16 or 32 bits in length as specified in the data descriptor.

CL

Instruction:
Clear

Type:
Single Operand

Format:



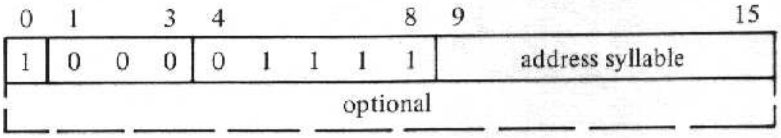
Description:
Stores zeros in the location specified in the address syllable. The address syllable can specify a memory location, an immediate operand, or one of the R registers.

CLH

Instruction:
Clear half-word

Type:
Single Operand

Format:



Description:
Stores zeros in the half-word location specified in the address syllable. The address syllable can specify a memory location, an immediate operand, or another R register, with the byte being positioned as follows:

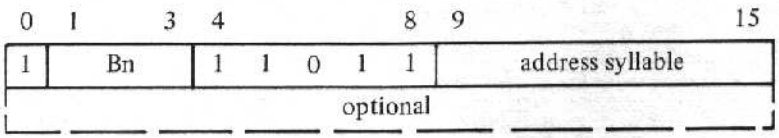
- memory location, not indexed — left byte
- memory location, indexed
  - index value even — left byte
  - index value odd — right byte
- immediate operand — left byte
- R register — right byte

CMB

Instruction:
Compare to B register

Type:
Double Operand

Format:



Description:

Compares the word in the designated B register with the word(s) in the effective address and sets the G and L indicators according to the results of the comparison. The address syllable can specify a memory location, an immediate operand, or another B register.

The greater than (G) and less than (L) indicators are set or cleared depending on the unsigned contents of the two operands. The setting of the unlike signs (U) indicator is not defined.

CMH

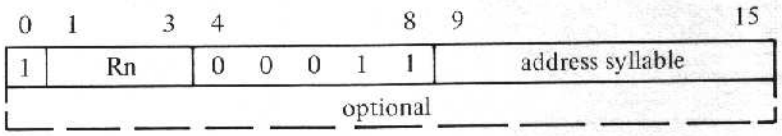
Instruction:

Compare half-word (byte) to R register

Type:

Double Operand

Format:



Description:

Compares the word in the designated R register with the byte (sign extended) in the effective address and sets the G, L, and U indicators according to the results of the comparison. The address syllable can specify a memory location, an immediate operand, or another R register, with the byte being positioned as follows:

– memory location, not indexed	– left byte
– memory location, indexed	
index value even	– left byte
index value odd	– right byte
– immediate operand	– left byte
– R register	– right byte

The greater than (G) and less than (L) indicators are set or cleared depending on the 16-bit unsigned contents of the two operands. The unlike signs (U) indicator is cleared if bit zero of both operands are the same, set if different.

Example: R register contains $(00FF)_{16}$
Effective address contains $(FF)_{16}$
CMH sets L and U, clears G

CMN

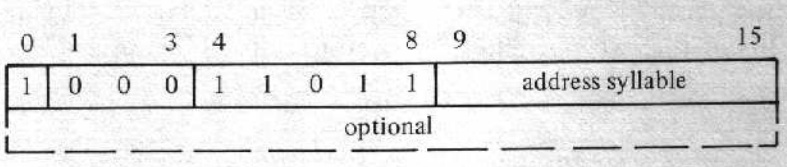
Instruction:

Compare address to null

CMN/CMR

Type:
Single Operand

Format:



Description:
Compares the contents of the location or B register specified by the address syllable to a null address (the address 0).

The contents of the I register are affected as follows:

- o The G bit is set to 0 if the specified address is equal to null; otherwise, it is set to 1.
- o The L bit is set to 0.
- o The U bit is affected, but its value is undefined.

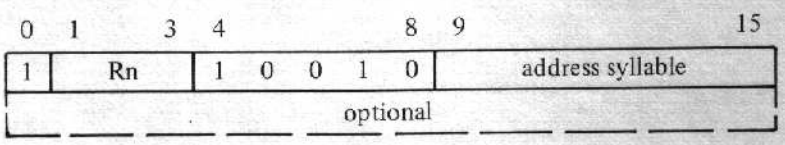
The address syllable can specify a memory location, an immediate operand, or one of the B registers.

CMR

Instruction:
Compare to R register

Type:
Double Operand

Format:



Description:
Compares the word in the designated R register with the word in the effective address and sets the G, L, and U indicators according to the results of the comparison. The address syllable can specify a memory location, an immediate operand, or another R register.

The greater than (G) and less than (L) indicators are set or cleared depending on the 16-bit unsigned contents of the two operands. The unlike signs (U) indicator is cleared if bit zero of both operands are the same, set if different.

Note that this instruction can be used to do either a logical (alphabetical) comparison or an algebraic comparison. The branch instruction that follows performs either an algebraic test or a logical test.

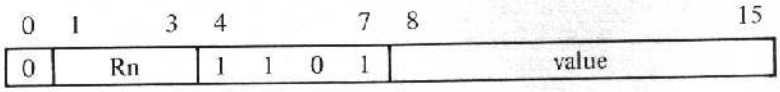
Example: R register contains $(FFFF)_{16}$
Effective address contains $(7777)_{16}$
CMR sets G and U, clears L
BG (Branch if greater) branches
BAG (Branch if algebraically greater) does not

CMV

Instruction:
Compare value to R register

Type:
Short Value Immediate

Format:



Description:
Compares the word in the designated R register with the byte (with its sign extended) contained in the value field of the instruction and sets/clears the G (greater than), L (less than), and U (unlike signs) indicators according to the results of the comparison.

Examples:

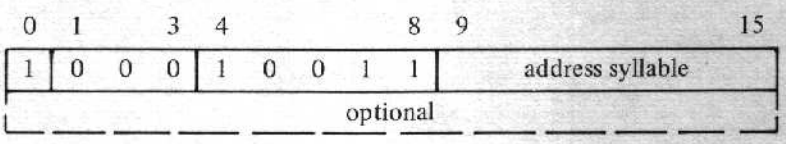
Contents of R register	Value Field in Instruction	G	L	U
$(00AB)_{16}$	$(AB)_{16}$	0	1	1
$(FFAB)_{16}$	$(AB)_{16}$	0	0	0
$(FFAB)_{16}$	$(AA)_{16}$	1	0	0
$(-7)_{10}$	$(-7)_{10}$	0	0	0
$(-7)_{10}$	$(+7)_{10}$	1	0	1

CMZ

Instruction:
Compare to zero

Type:
Single Operand

Format:



Description:

Compares the contents of the location specified in the address syllable to 0. The address syllable can specify a memory location, an immediate operand, or one of the R registers.

The contents of the I register are affected as follows:

- o If the contents of the specified location do not equal 0, the G bit is set to 1; otherwise, it is set to 0.
- o The L bit is set to 0.
- o If the first bit of the specified location equals 1, the U-bit is set to 1; otherwise, it is set to 0.

CNFG

Instruction:

Configure

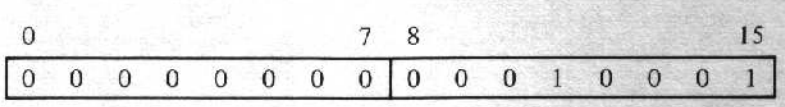
Restriction:

Traps (TV#5) on models without Scientific Instruction Processor/Commercial Instruction Processor.

Type:

Generic

Format:



Description:

Causes the CP to perform an input/output operation and a scan. The effect of the input/output operation is equivalent to that produced by the execution of the following instruction: IO = $R7, = $R6

Register R6 contains the command word (CH,F) that specifies a channel number and a function code. (See the IO instruction, for format of the command word.) The function code must designate an output function. The command word and the word contained in Register R7 are sent to the addressed IO channel.

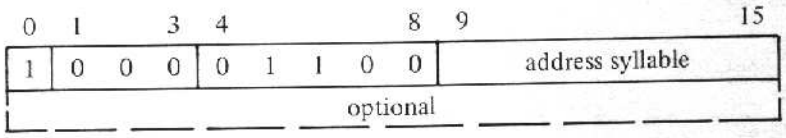
A predetermined list of channel numbers is scanned to determine the presence or absence of an SIP/CIP. The results of the scan are used to update internal CP firmware/hardware flags that direct instruction execution; i.e., trap or execute by an SIP/CIP. An identical scan is performed automatically when the system is powered up or initialized.

CPL

Instruction:
Complement

Type:
Single Operand

Format:



Description:
Ones complements the contents of the location specified in the address syllable. No indicators are affected. The address syllable can specify a memory location, an immediate operand, or one of the R registers.

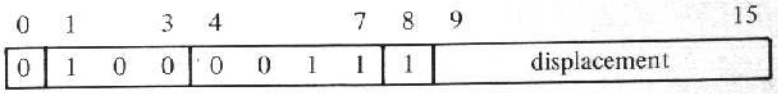
CSNCB

Instruction:
Synchronize and Branch

Restriction:
Traps (TV#5) on Models 23, 33, 43, and 53

Type:
Commercial Branch

Format:



Description:
Branches to the location specified by the operand after the previous CIP instruction has been completed. Used to synchronize the CPU/CIP in those cases where data being processed by the asynchronous CIP is to be utilized by the CPU instructions.

If the J-bit in the M1 register contains a binary 1, the trace procedure is entered via trap vector #2. Upon completion, the trace procedure automatically branches to the address specified by the operand. In this case, or if the J-bit contains a binary 0, the instruction sequence starting at the location specified by the operand is executed.

CSYNC/DAD

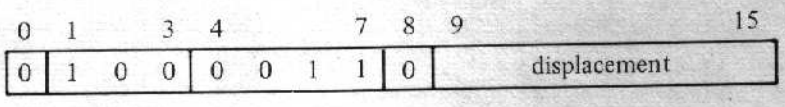
CSYNC

Instruction:
Synchronize

Restriction:
Traps (TV#5) on Models 23, 33, 43, and 53

Type:
Commercial Branch

Format:



Description:
Prevents the CP from going to the next instruction until the previous CIP instruction has been completed. Performs no operation. Used to synchronize the CPU/CIP in those cases where data being processed by the asynchronous CIP is to be utilized by the CP instructions.

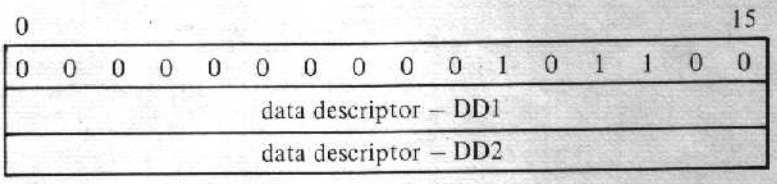
DAD

Instruction:
Decimal add

Restriction:
Traps (TV#5) on Models 23, 33, 43, and 53

Type:
Numeric

Format:



Description:
Algebraically adds the decimal value at the address specified by the first operand to the decimal value at the address specified by the second operand and stores the result at the address of the second operand.

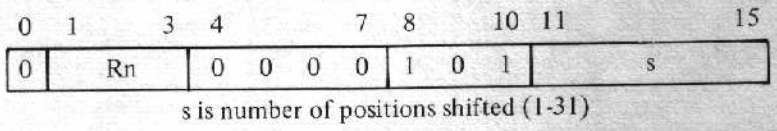
Whenever Trap 23 or Trap 24 occurs, the preservation of the original operands cannot be guaranteed. If any other trap occurs, the operands remain unchanged.

DAL

Instruction:
Double-shift arithmetic-left

Type:
Shift Long

Format:



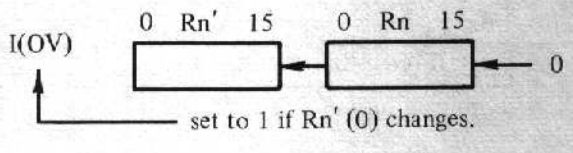
s is number of positions shifted (1-31)

Description:
Shifts the contents of the even-odd R register pair (i.e., R2 and R3, R4 and R5, R6 and R7), identified in the Rn field (which must be odd), left the number of bit positions specified by the s field. The bit positions vacated by the shift are filled with binary 0s. If the s field contains 0, the shift distance is obtained from bits 11 through 15 of general register R1.

The contents of the I register are affected as follows:

- o If the contents of bit 0 in the even-numbered R register changes, the OV bit is set to 1; otherwise, it is set to 0.

Pictorial Representation:

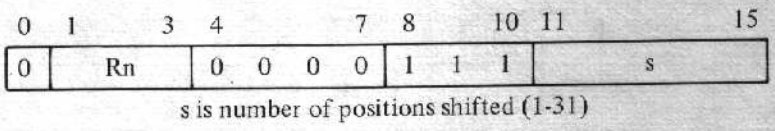


DAR

Instruction:
Double-shift arithmetic-right

Type:
Shift Long

Format:



s is number of positions shifted (1-31)

DAR/DCL

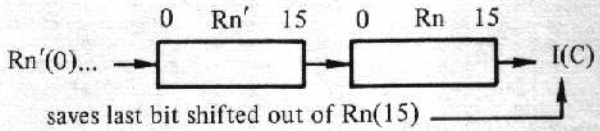
Description:

Shifts the contents of the even-odd R register pair (i.e., R2 and R3, R4 and R5, R6 and R7), identified in the Rn field (which must be odd), right the number of bit positions specified by the s field. The bit positions vacated by the shift are filled with the sign value originally contained in bit 0. If the s field contains 0, the shift distance is obtained from bits 11 through 15 of general register R1.

The contents of the I register are affected as follows:

- o C-bit contains the last binary digit shifted out of the odd-numbered R register.

Pictorial Representation:



DCL

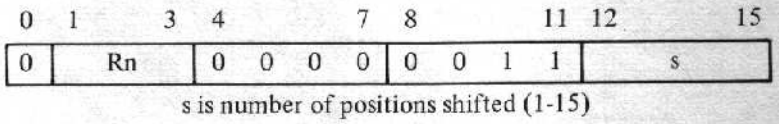
Instruction:

Double-shift closed-left

Type:

Shift Short

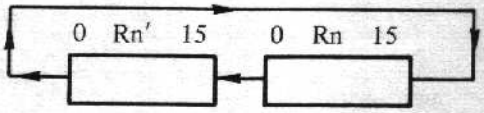
Format:



Description:

Shifts the contents of the even-odd R register pair (i.e., R2 and R3, R4 and R5, R6 and R7), identified in the Rn field (which must be odd), left the number of bit positions specified by the s field. The bits shifted out of the even-numbered R register are placed in the bit positions of the odd-numbered R register vacated as the bits are shifting left. If the s field contains 0, the shift distance is obtained from bits 12 through 15 of general register R1.

Pictorial Representation:



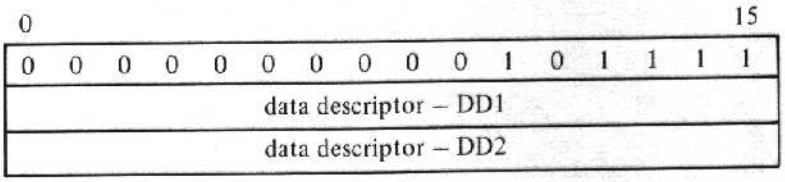
DCM

Instruction:
Decimal compare

Restriction:
Traps (TV#5) on Models 23, 33, 43, and 53

Type:
Numeric

Format:



Description:
The decimal value specified by the first operand is compared algebraically with the decimal value specified by the second operand.

NOTE: In zoned decimal operation, the zone bits are ignored.

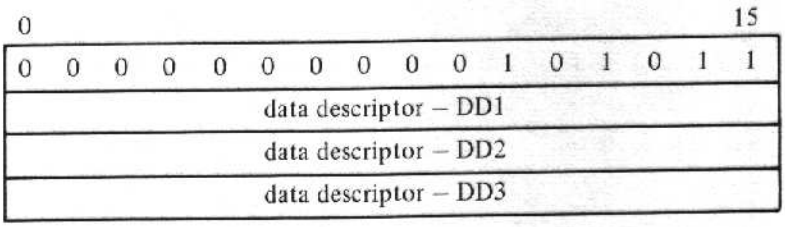
DDV

Instruction:
Decimal divide

Restriction:
Traps (TV#5) on Models 23, 33, 43, and 53

Type:
Numeric

Format:



Description:
Divides the decimal value (the dividend) at the address specified by the second operand by the decimal value (the divisor) at the address specified by the first operand. Places the quotient at the address specified by the third operand. Places the remainder at the address specified by the second operand. If the absolute value of the divisor is greater than that of the dividend, the quotient is zero and the dividend and the divisor remain unchanged. If the divisor has a value of zero, a divide by zero trap (trap vector #25) is unconditionally generated.

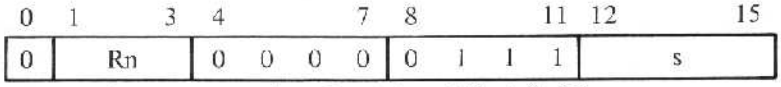
- o If the sign of DD1 is the same as that of DD2, the sign of the quotient is +.
- o If the sign of DD1 is not the same as that of DD2, the sign of the quotient is −.
- o The sign of the remainder is always the same as that of the dividend (DD2) unless the remainder is zero.

DCR

Instruction:
Double-shift closed-right

Type:
Shift Short

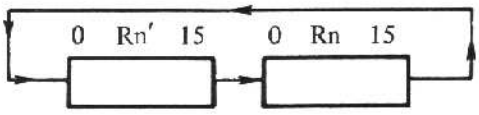
Format:



s is number of positions shifted (1-15)

Description:
Shifts the contents of the even-odd R register pair (i.e., R2 and R3, R4 and R5, R6 and R7), identified in the Rn field (which must be odd), right the number of bit positions specified by the s field. The bits shifted out of the odd-numbered R register are placed in the bit positions of the even-numbered R register vacated as the bits are shifting right. If the s field contains 0, the shift distance is obtained from bits 12 through 15 of general register R1.

Pictorial Representation:

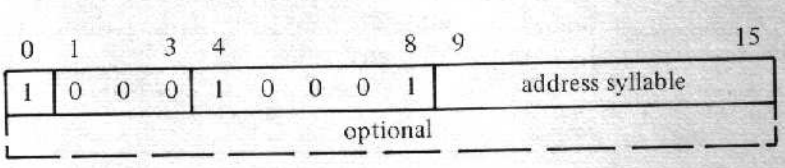


DEC

Instruction:
Decrement

Type:
Single Operand

Format:



Description:

Decrements by 1 the contents of the location or register specified in the address syllable, then copies bit 0 of the addressed word or register into I(B).

This instruction operates in read modify write (RMW) mode, which prevents any other processor in a multiprocessor environment from similarly accessing the location being modified until the modification is completed. The address syllable can specify a memory location, an immediate operand, or one of the R registers.

The contents of the I register are affected as follows:

- o If the decrementation causes a carry to occur, the C-bit is set to 1; otherwise, it is set to 0.
- o If the value being decremented was $-32{,}768$ (-2^{15}), I(OV) is set to 1; otherwise, I(OV) is cleared to 0. However, this can never cause an overflow trap.
- o I(B) is set as described above.

DIV

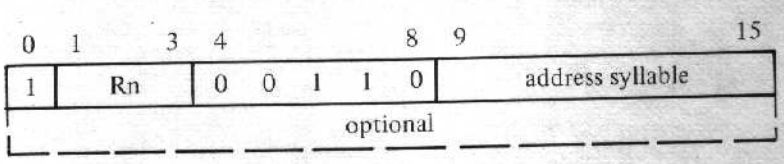
Instruction:

Divide R register

Type:

Double Operand

Format:



Description:

If the designated R register is R1 to R6, its contents are divided by the word at the effective address and the remainder is lost; if the designated R register is R7, the double-precision value in R6 and R7 is divided by the word at the effective address with the quotient being developed in R7 and the remainder in R6. The address syllable can specify a memory location, an immediate operand, or another R register.

The contents of the I register are affected as follows:

1. I(OV) is set to 1 if
   a. The divisor = 0
   b. The dividend is -2^{15} times the divisor and the divisor is positive.
   c. The quotient is greater than $2^{15} - 1$ or less than -2^{15}.

   Otherwise I(OV) is cleared to 0.

   Divide operations that cause I(OV) to be set terminate with all operands unchanged.

2. The carry indicator (C) is set only if the remainder is not 0 *and* if the remainder is not stored in R6 (i.e., when the first operand does not specify R7). If R7 were specified as the first operand, the remainder would be stored in R6 and the carry bit would be unchanged whether or not the remainder was zero.
3. The carry indicator (C) is cleared if the remainder is zero and is not stored in R6.

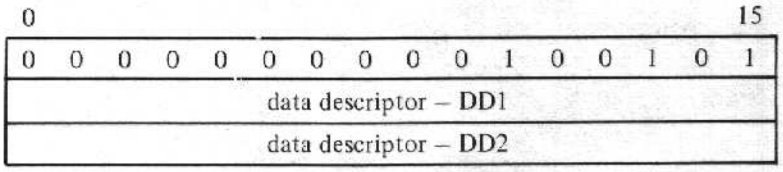
DMC

Instruction:
Decimal move and convert

Restriction:
Traps (TV#5) on Models 23, 33, 43, and 53

Type:
Numeric

Format:



Description:
The decimal value of the data type specified by the first operand is converted to the data type specified by the second operand and stored, right justified, at the address specified by the second operand.

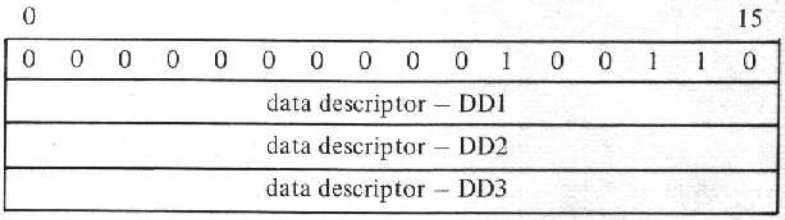
DME

Instruction:
Decimal move and edit

Restriction:
Traps (TV#5) on Models 23, 33, 43, and 53

Type:
Edit

Format:



Description:

The decimal digits in the sending field specified by the first data descriptor (DD1) are edited in accordance with the micro operations in the field specified by the third data descriptor (DD3), and moved to the receiving field specified by the second data descriptor. If the sending field contains packed decimals they are converted to unpacked decimals before they are stored in the receiving field.

The number of edited characters stored in the receiving field can be either more or less than the number of digits in the sending field. The receiving field may have more characters when micro operations specify one or more characters are to be inserted. The receiving field may have less characters when a micro operation specifies that one or more digits are to be skipped.

The instruction terminates normally when the receiving field is filled. Normal termination occurs even though the sending field or the string of micro operations has not been exhausted.

An illegal specification trap (Trap 26) is generated if either the sending field or the string of micro operations is exhausted before the receiving field is filled.

NOTE: Refer to Section 6 of the *GCOS 6 Assembly Language Reference* manual, Order No. CB07 for an explanation concerning micro edit operations which are required by the AME and DME edit move instructions.

DML

Instruction:

Decimal multiply

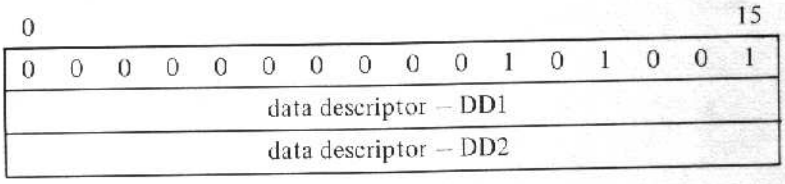
Restriction:

Traps (TV#5) on Models 23, 33, 43, and 53

Type:

Numeric

Format:



Description:

Multiplies the decimal value at the address specified by the first operand by the decimal value at the address specified by the second operand and stores the result, right justified, at the address of the second operand.

DOL

Instruction:

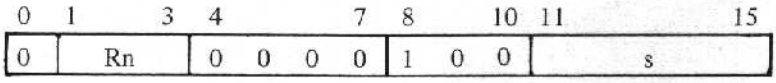
Double-shift open-left

Type:

Shift Long

DOL/DOR

Format:



s is number of positions shifted (1-31)

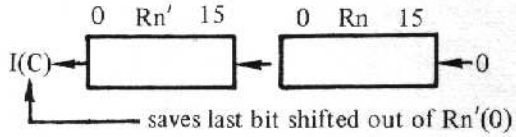
Description:

Shifts the contents of the even-odd R register pair (i.e., R2 and R3, R4 and R5, R6 and R7), identified in the Rn field (which must be odd), left the number of bit positions specified by the s field. The bit positions vacated by the shift are filled with binary 0s. If the s field contains 0, the shift distance is obtained from bits 11 through 15 of general register R1.

The contents of the I register are affected as follows:

- C-bit contains the last binary digit shifted out of the even-numbered R register.

Pictorial Representation:



DOR

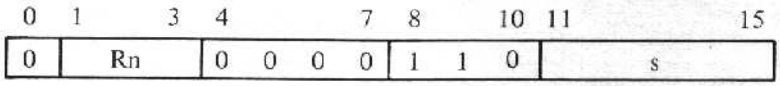
Instruction:

Double-shift open-right

Type:

Shift Long

Format:



s is number of positions shifted (1-31)

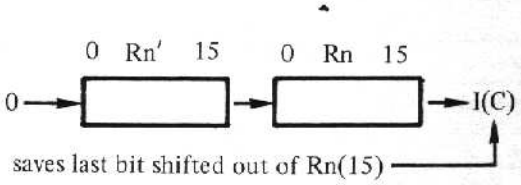
Description:

Shifts the contents of the even-odd R register pair (i.e., R2 and R3, R4 and R5, R6 and R7), identified in the Rn field (which must be odd), right the number of bit positions specified by the s field. The bit positions vacated by the shift are filled with binary 0s. If the s field contains 0, the shift distance is obtained from bits 11 through 15 of general register R1.

The contents of the I register are affected as follows:

- C-bit contains the last binary digit shifted out of the odd-numbered R register.

Pictorial Representation:



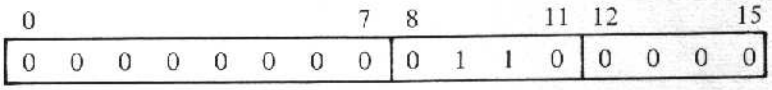
DQA

Instruction:
Dequeue on address

Restriction:
Traps (TV#5) on Models 23 and 33

Type:
Generic

Format:



Description:
Unlinks a frame whose priority word address exactly matches B1. B2 points to the lock word.

The contents of the I register are affected as follows:

- o If the operation is completed, C is set to 1; otherwise 0.
- o If the frame was found and unlinked, L is set to 0; otherwise 1.
- o G is set to 0.

DQH

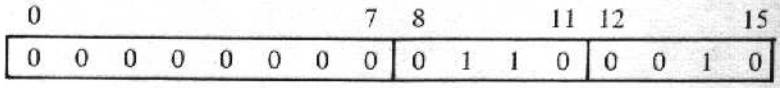
Instruction:
Dequeue from head

Restriction:
Traps (TV#5) on Models 23 and 33

Type:
Generic

DQH/DSB

Format:



Description:
Unlinks the first frame from the list whose priority value equals or is numerically greater than the contents of R5. B2 is used to point to the lock word, and B1 is returned containing a pointer to the priority word of the unlinked frame. I(G) and I(L) indicate the hit conditions.

The contents of the I Register are affected as follows:

o If the operation is completed, C is set to 1; otherwise 0.

G	L	Condition
0	0	Unlinked frame was first whose priority equalled [R5].
1	0	Unlinked frame was first whose priority exceeded [R5].
0	1	No frame was unlinked, B1 is unchanged, no priority found equal to or greater than [R5].

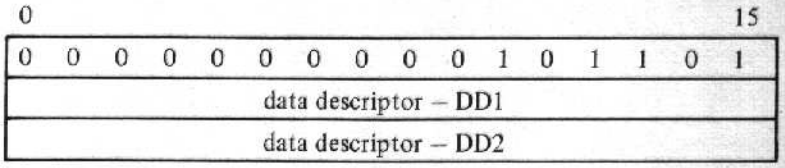
DSB

Instruction:
Decimal subtract

Restriction:
Traps (TV#5) on Models 23, 33, 43, and 53

Type:
Numeric

Format:



Description:
Algebraically subtracts the decimal value at the address specified by the first operand from the decimal value at the address specified by the second operand and stores the result at the address of the second operand.

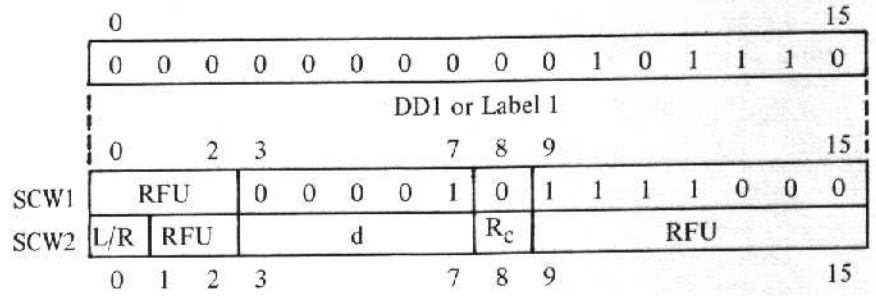
DSH

Instruction:
Decimal shift

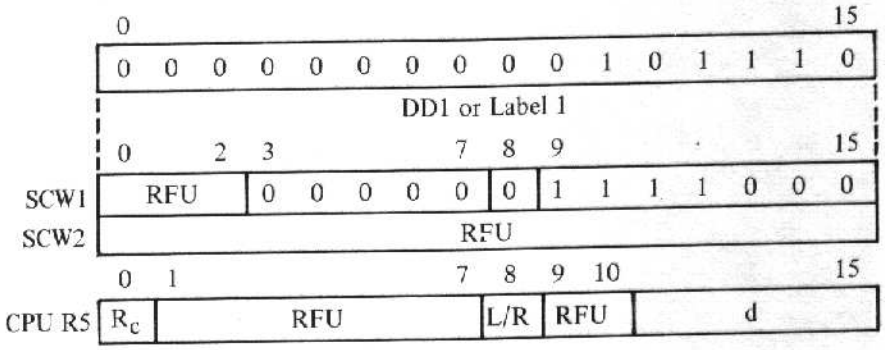
Restriction:
Traps (TV#5) on Models 23, 33, 43, and 53

Type:
Numeric

Format:



a. With shift control information in line



b. With shift control information in CPU register R5

Description:
The decimal value designated by the first operand is shifted the distance (number of digits) and in the direction specified by the shift control words. The digit positions that are vacated are zero filled. Rounding may be specified for a right shift. If rounding is specified and the value of the last digit shifted out is from 5 through 9, the absolute value of the operand is increased by one; e.g., +12 becomes +13; -12 becomes -13.

The decimal shift instructions use two shift control words, SCW1 and SCW2.

The second operand, if present, specifies the value of SCW2, and the assembler generates 0178 (hexadecimal) as the value of SCW1. The shift control information is taken from SCW2.

If the second operand is omitted, the Assembler generates 0078 (hexadecimal) and 0000 (hexadecimal) as the values of SCW1 and SCW2, respectively, and the shift control information is taken from register R5. The sign character, if any, is not affected by the shift operation (i.e., it does not get shifted).

Note that the formats of SCW2 and R5 are reversed; i.e., the information that is assigned to the first byte of SCW2 is assigned to the second byte of R5, and that assigned to the second byte of SCW2 is assigned to the first byte of R5.

Shift control information is specified by SCW2 or by register R5 as follows:

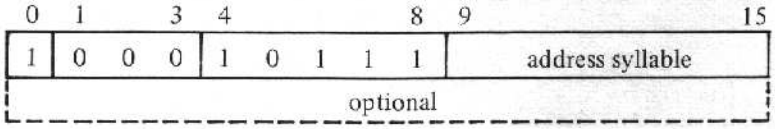
- o Direction of shift: bit 0 of SCW2; bit 8 of R5
  bit = 0 left shift
  bit = 1 right shift
- o Distance of shift in digits: bits 3 through 7 of SCW2; bits 11 through 15 of R5
  range = 0 through 31
- o Rounding: bit 8 of SCW2; bit 0 of R5
  bit = 0 do not round
  bit = 1 round after shifting right (ignored for left shift)

ENT

Instruction:
Enter

Type:
Single Operand

Format:



Description:
Puts the processor into the unprivileged (user) mode and then jumps to the location specified by the effective address. If the J bit in the mode control (M1) register is on, this instruction traps to trap vector #2.

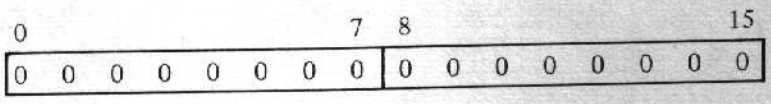
No indicators are affected. The address syllable must specify a memory location, not a register or an immediate operand.

HLT

Instruction:
Halt

Type:
Generic

Format:



Description:

Stops program execution. HLT state is indicated on the control panel. All interrupts will be honored. The P field in the S register must equal 1x (i.e., the central processor must be in the privileged state) for this instruction to be executed. If not, the unprivileged use of a privileged operation is signified by a trap to trap vector #13.

INC

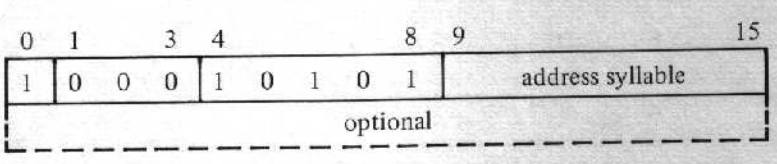
Instruction:

Increment

Type:

Single Operand

Format:



Description:

Copies bit 0 of the contents of the location or register specified in the address syllable into I(B), then increments by 1 the contents of the location or register.

This instruction operates in read modify write (RMW) mode, which prevents any other processor in a multiprocessor environment from similarly accessing the location being modified until the modification is completed. The address syllable can specify a memory location, an immediate operand, or an R register.

The contents of the I register are affected as follows:

- o If the incrementation causes a carry to occur, the C bit is set to 1; otherwise, it is set to 0.
- o If the value being incremented was 32,767, I(OV) is set to 1; otherwise, it is cleared to 0. However, this can never cause an overflow trap.
- o I(B) is set as described above.

IO

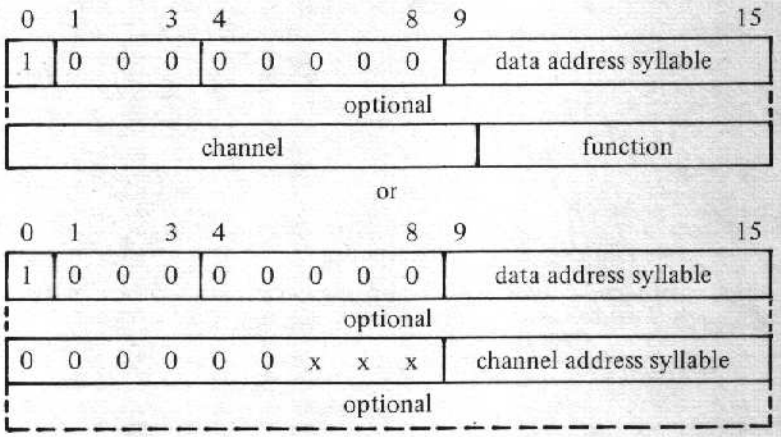
Instruction:

Input/output (word)

Type:

Input/Output

Formats:



Description:

A privileged instruction that can be executed only if the P field in the S register equals 1x; otherwise the instruction traps through trap vector #13.

This instruction initiates an I/O command bus cycle that either outputs a word to a channel or requests the input of a word from a channel. If the channel accepts the I/O command, the input/output (I) indicator is set to a one; if it does not accept the command, the I bit is cleared to zero.

The location of the word to be output or of where the word is to be input is described by the data address syllable in the instruction. This address syllable is identical to those in single- and double-operand instructions and can specify a memory location, an immediate operand, or an R register. If a memory location is specified on an output, for example, the data word is first read from the memory to the CP and then output from the CP to the channel. The memory does not send or receive the data directly to or from the channel.

The channel number is a 10-bit value contained in the same word with a 6-bit function code. This word is contained directly in the instruction, or, if the channel number field (bits 0–5) in the instruction is zero, in a location pointed to by the channel address syllable.[2] If the latter is utilized, it is of the general address syllable form and can specify a memory location or an R register.

The channel number can specify an output channel (if it is odd) or an input channel (if it is even). This has nothing to do with whether the I/O instruction initiates an output command or an input command. (It is possible to output to input channels or input from output channels, as well as output to output and input from input.)

The direction of the command is determined by the function code, which also specifies the type of data to be input or output. The function code normally addresses a set up, control or status register in the channel. However, it could specify a data buffer and thus be used to implement a programmed input/output operation. (The latter technique is not used by standard I/O channels, which all work on a DMA basis of I/O.)

Function codes are thus dependent on the design of a particular channel controller. However, the following codes have been assigned and are used by standard channels:

[2] On processor to processor transfers, a CAS must be used as the first six bits of a processor channel number are always zero.

Input codes:
- 02 – Input interrupt control word
- 06 – Input task word
- 0C – Input range
- 10 – Input configuration word A
- 12 – Input configuration word B
- 18 – Input status word #1
- 1A – Input status word #2
- 26 – Input device ID

Output codes:
- 01 – Output control word
- 03 – Output interrupt control word
- 07 – Output task
- 09 – Output address (see note)
- 0D – Output range (see note)
- 11 – Output configuration word A
- 13 – Output configuration word B

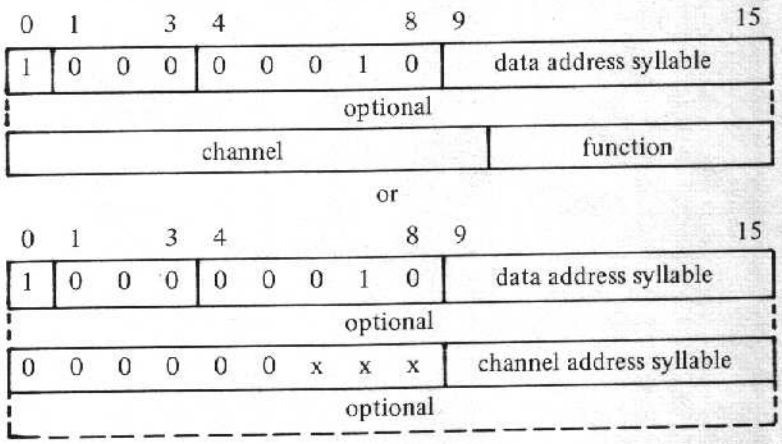
NOTE: The I/O instruction should not use codes 09 and 0D as the address and range should be output with an IOLD instruction.

IOH

Instruction:
Input/output half-word

Type:
Input/Output

Formats:



Description:
A privileged instruction that is identical to the I/O instruction except that it transfers a byte rather than a word. The data address syllable generates a reference to (1) the right byte of an R register or (2) to the left *or* right byte of a word in main memory. If the data address

syllable identifies an unindexed memory location, the left byte of that memory location is the target of the input/output operation; if the data address syllable identifies an indexed memory location, the index value is applied (as a byte displacement) to the left byte of that memory location to identify the target byte (which may be either the left *or* right byte of a memory location, depending on whether the index value is even or odd).

In cases where a byte is written into the right half of an R register or into a memory location (input operation), the other byte of the same word is not modified.

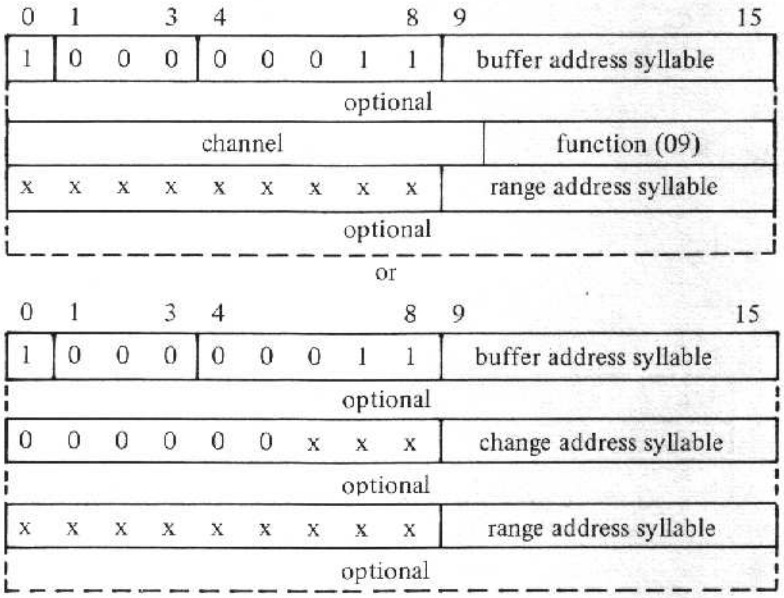
This instruction is designed for programmed I/O operations. It is not currently used by Honeywell implemented controllers.

IOLD

Instruction:
Input/output load

Type:
Input/Output

Formats:



Description:

A privileged instruction that is used to set up a device for DMA operation by outputting an address and a range to it. It is similar in function to two consecutive I/O instructions in that it initiates two I/O output commands, one with a function code of $(09)_{16}$ and one with a function code of $(0D)_{16}$.

The main difference is that it outputs a 24-bit byte address to the DMA address register. The address register is 24 bits in length as opposed to the 16-bit length of the other registers in the channel. To generate a 24-bit word, the IOLD outputs the effective address of the buffer as specified by the "buffer" address syllable, not the contents of the location pointed to by it.

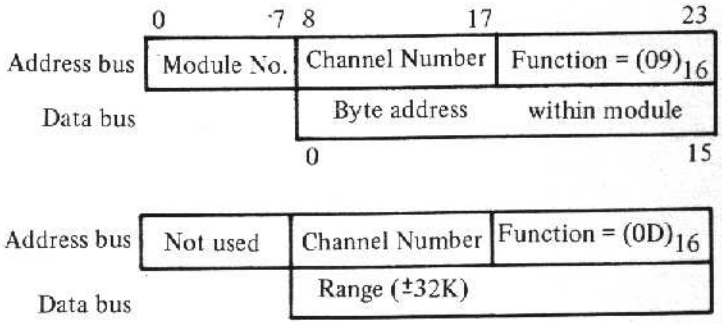
This address is split when put on the bus, with 16 least significant bits being put on the data lines and the high-order bits in the "module" field of the address bus.

The buffer address syllable must define a memory location. If it is not indexed the effective address will define the left byte in a word; if it is indexed it can specify either side.

The channel can either be defined within the instruction or can be pointed to by the channel address syllable.

After the first output command, 04 is automatically added to the function code and another output command is initiated. This command outputs the contents (a 16-bit word) of the location pointed to by the range address syllable. A memory location, immediate operand, or R register form of addressing may be used. The range itself is a signed 16-bit value which defines the number of bytes to be transferred.

The formats of the I/O output command bus cycles are as follows:

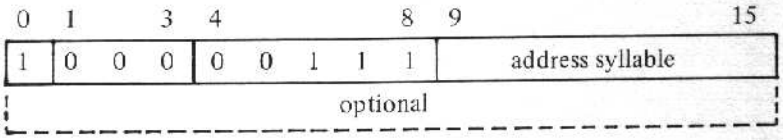


JMP

Instruction:
Jump

Type:
Single Operand

Format:



Description:
Jumps to the location specified in the address syllable. If the J bit in the mode control (M1) register is set, control then traps to trap vector #2.

No indicators are affected. The address syllable must specify a memory address, not a register or an immediate operand.

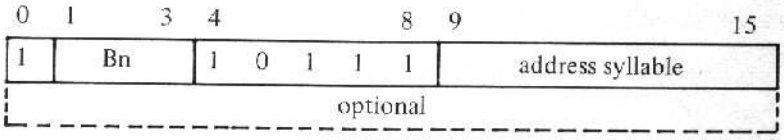
LAB

Instruction:
Load effective address into B register

Type:
Double Operand

LAB/LB

Format:



Description:

Loads the effective address specified by the address syllable into the designated B register. Note that this is the address itself, not its contents. No indicators are affected. The address syllable can specify a memory address or an immediate operand, but not a register.

Example: Assume the following:

Register R3 contains $(0007)_{16}$
Register B5 contains $(B010)_{16}$
Memory location $(B010)_{16}$ contains $(FA03)_{16}$

The instruction LAB SB7, *B5.SR3 (indirect through B5 indexed by R3) will load B7 with $(FA0A)_{16}$.

LB

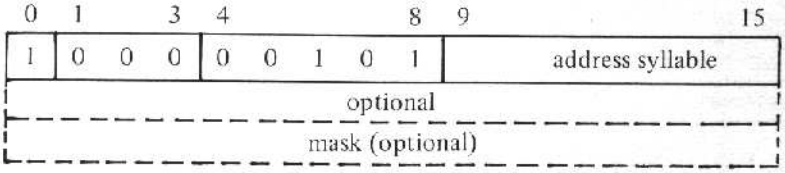
Instruction:

Load bit

Type:

Single Operand

Format:



Description:

Loads the B-bit indicator in the I-register with either a single bit or with the result of a test on a group of bits in a word. If any are on, the B bit is set to 1; if none are on, to zero.

To address a single bit, the address syllable may specify an indexed memory address. The index value counts bits relative to bit zero of the effective address.

To test a group of bits in a word, a mask word following the instruction is utilized. Bit positions in the word at the effective address that correspond to one bits in the mask are tested. In this form the address syllable can specify a memory location (unindexed), an immediate operand, or an R register.

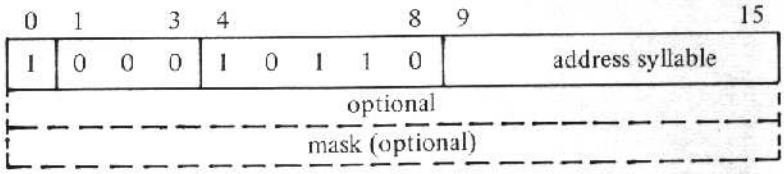
For example, to test bit 13 in a word, indexed addressing with an index value of 13 could be used (or −3, +29, etc.) or unindexed addressing and a mask value of 0004. To test bits 13, 14, and 15, a mask with a value of 0007 should be used.

LBC

Instruction:
Load bit and complement

Type:
Single Operand

Format:



Description:
Loads the B-bit indicator in the I-register with either a single bit or with the result of a test on a group of bits in a word. If any are on, the B bit is set to 1; if none are on, to zero. The single bit loaded or each of the bits tested is then complemented.

To address a single bit, the address syllable may specify an indexed memory address. The index value counts bits relative to bit zero of the effective address.

To test a group of bits in a word, a mask word following the instruction is utilized. Bit positions in the word at the effective address that correspond to one bits in the mask are tested. In this form the address syllable can specify a memory location (unindexed), an immediate operand, or an R register.

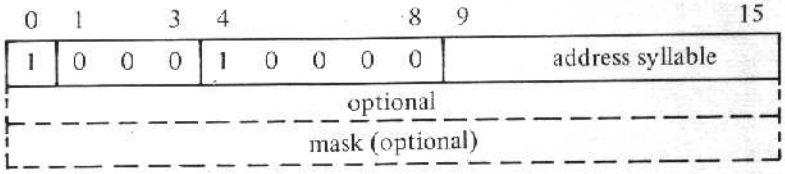
This instruction operates in read modify write (RMW) mode, which prevents any other processor in a multiprocessor environment from similarly accessing the location being modified until the modification is completed.

LBF

Instruction:
Load bit and set false

Type:
Single Operand

Format:



Description:
Loads the B-bit indicator in the I-register with either a single bit or with the result of a test on a group of bits in a word. If any are on, the B bit is set to 1; if none are on, to zero. The single bit loaded or each of the bits tested is then set to zero.

To address a single bit, the address syllable may specify an indexed memory address. The index value counts bits relative to bit zero of the effective address.

To test a group of bits in a word, a mask word following the instruction is utilized. Bit positions in the word at the effective address that correspond to one bits in the mask are tested. In this form the address syllable can specify a memory location (unindexed), an immediate operand, or an R register.

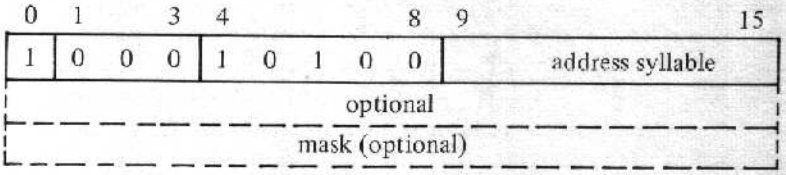
This instruction operates in read modify write (RMW) mode, which prevents any other processor in a multiprocessor environment from similarly accessing the location being modified until the modification is completed.

LBS

Instruction:
Load bit and swap

Type:
Single Operand

Format:



Description:

Loads the B-bit indicator in the I-register with either a single bit or with the result of a test on a group of bits in a word. If any are on, the B bit is set to 1; if none are on, to zero. The single bit loaded or each of the bits tested is then set to the previous contents of the B bit.

To address a single bit, the address syllable may specify an indexed memory address. The index value counts bits relative to bit zero of the effective address.

To test a group of bits in a word, a mask word following the instruction is utilized. Bit positions in the word at the effective address that correspond to one bits in the mask are tested. In this form the address syllable can specify a memory location (unindexed), an immediate operand, or an R register.

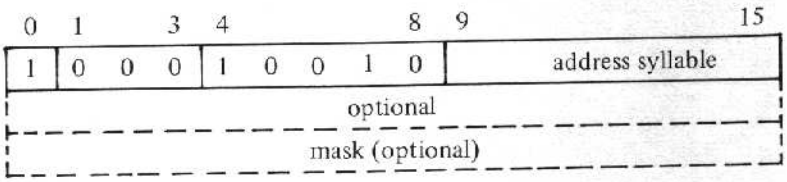
This instruction operates in read modify write (RMW) mode, which prevents any other processor in a multiprocessor environment from similarly accessing the location being modified until the modification is completed.

LBT

Instruction:
Load bit and set true

Type:
Single Operand

Format:



Description:

Loads the B-bit indicator in the I-register with either a single bit or with the result of a test on a group of bits in a word. If any are on, the B bit is set to 1; if none are on, to zero. The single bit loaded or each of the bits tested is then set to one.

To address a single bit, the address syllable may specify an indexed memory address. The index value counts bits relative to bit zero of the effective address.

To test a group of bits in a word, a mask word following the instruction is utilized. Bit positions in the word at the effective address that correspond to one bits in the mask are tested. In this form the address syllable can specify a memory location (unindexed), an immediate operand, or an R register.

This instruction operates in read modify write (RMW) mode, which prevents any other processor in a multiprocessor environment from similarly accessing the location being modified until the modification is completed.

LDB

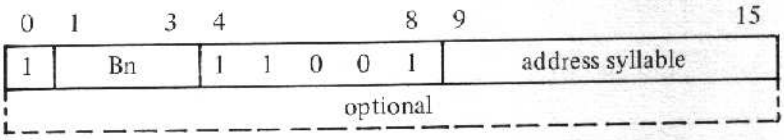
Instruction:

Load B register

Type:

Double Operand

Format:



Description:

Loads the designated B register with the pointer contained at the effective address. No indicators are affected. The address syllable can specify a memory location, an immediate operand, or another B register.

LDH

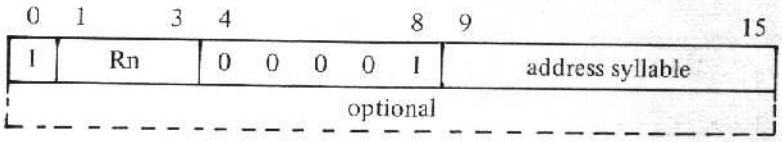
Instruction:

Load half-word (byte) into R register

Type:

Double Operand

Format:



Description:

Loads the byte contained at the effective address into the right half of the designated R register, with the sign being extended into the left half, bits 0–7. No indicators are affected.

The effective address can specify a memory location, an immediate operand, or another R register, with the byte being positioned as follows:

– memory location, not indexed	– left byte
– memory location, indexed	
index value even	– left byte
index value odd	– right byte
– immediate operand	– left byte
– R register	– right byte

Example: Assume that:

memory location 1000 contains $(6789)_{16}$
register R1 contains 0
register R2 contains 1
register B1 contains $(1000)_{16}$

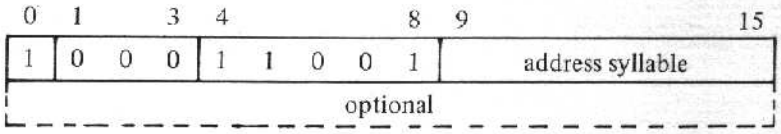
then if LDH $R5, $B1.$R1 is executed: R5 will contain $(0067)_{16}$
but if LDH $R5, $B1.$R2 is executed: R5 will contain $(FF89)_{16}$

LDI

Instruction:
Load double-word integer

Type:
Single Operand

Format:



Description:

Loads the word contained at the effective address into R6 and the following word into R7. No indicators are affected. The address syllable can specify a memory location, an immediate operand, or an R register as follows:

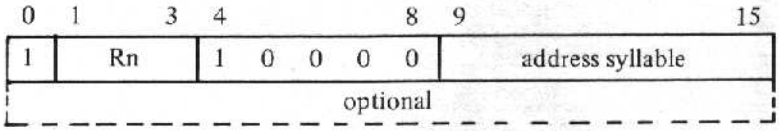
- o *Memory location:* The left-hand word of a double-word pair will be addressed. Auto-increment and auto-decrement modes of addressing will add (subtract) two from the base register. A value in the index register will count double-words; it is shifted one place to the left before being applied. Auto-indexing will cause one to be added (subtracted) from the index register.
- o *Immediate operand:* This will be a double-word operand and thus the instruction will be three words in length.
- o *Register operand:* This form of addressing must address the right-hand register of a double-word pair, i.e., either R3, which loads the contents of R2 and R3 into R6 and R7, or R5, which selects R4 and R5.

LDR

Instruction:
Load R register

Type:
Double Operand

Format:



Description:
Loads the designated R register with the word contained at the effective address. No indicators are affected. The address syllable can specify a memory location, an immediate operand, or another R register.

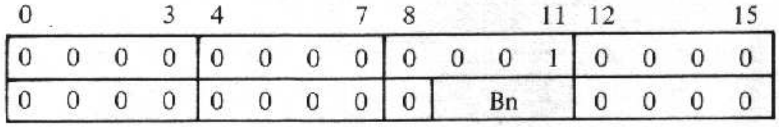
LDT

Instruction:
Load stack address register

Restriction:
Traps (TV#5) on Models 23 and 33

Type:
Generic

Format:



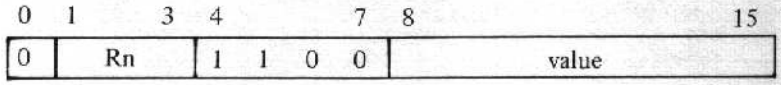
Description:
Loads the contents of Bn into the Stack Address Register (T). The contents of the I-register are unaffected.

LDV

Instruction:
Load value into R register

Type:
Short Value Immediate

Format:



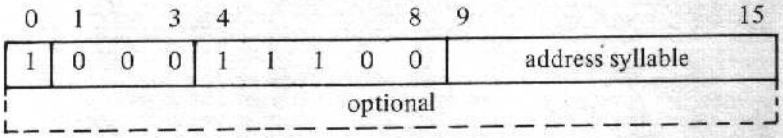
Description:
Loads the byte contained in the value field of the instruction into the right half of the designated R register with the sign being extended into the left half. No indicators are affected.

LEV

Instruction:
Level change

Type:
Single Operand

Format:



Description:
Sets or clears level activity bits according to the contents of the location indicated by the address syllable.

The address syllable can specify a memory location, an individual operand, or an R register.

The P bit in the S register must be set to 1 or the ring field in the S register must be 1x, as appropriate, (i.e., the central processor must be in the privileged state) for this instruction to be executed. If the P bit is not set to 1, the unprivileged use of a privileged operation is signified by a trap to trap vector #13.

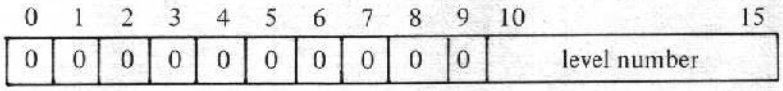
The contents of the S register are affected as follows:

- o Bits 10 through 15 of the S register are set to indicate the priority level at which processing continues after execution of the LEV instruction.

The LEV instruction, when used on a Model 43 or larger with the MMU, performs the same level management functions performed without the MMU. However, when an MMU is present and a level change takes place, the MMU segment descriptor registers are optionally loaded. In SAF mode, the 16 descriptors of segments 0.00 through 0.15 are loaded; in LAF mode, all 31 descriptors are loaded.

The following bit configurations in the indicated location produce the actions described.

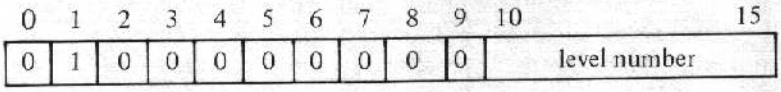
Schedule Interrupt Level, Scan, and Dispatch



The level activity bit for the designated level is set. The level activity bits are scanned and the highest active level ascertained. The context of the current level is saved (unless the current level is the highest active level). The context of the highest active level is restored (again, unless the current level is the highest active level).

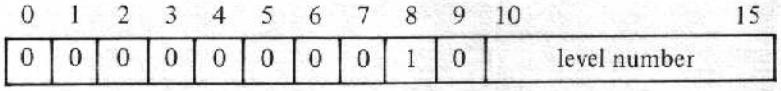
NOTE: Context save/restore is also skipped if the current level and the highest active level, though different, share a common save area.

Schedule Interrupt Level, Defer Interrupt



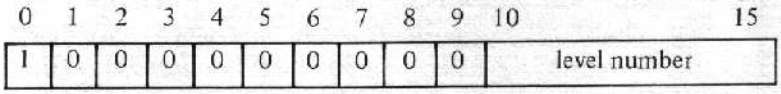
The level activity bit for the designated level is set. Execution continues at the current level.

Inhibit



The level activity bit for the designated level is set. The interrupt vector for the designated level is set equal to the interrupt vector for the current level. Execution of the current task continues at the designated level.

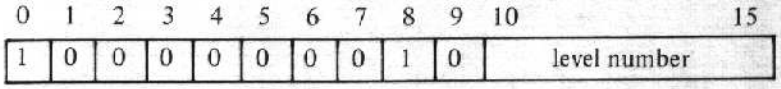
Schedule Interrupt Level, Suspend, Scan, and Dispatch



The level activity bit for the designated level is set. The level activity bit for the current level is then cleared. The level activity bits are scanned and the highest level ascertained. The context of the current level is saved. The context of the highest active level is restored.

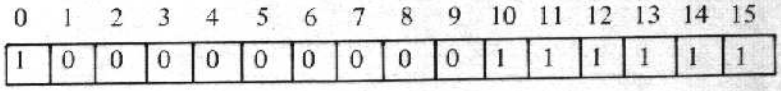
NOTE: Context save/restore is skipped if the current level and the highest active level share a common save area.

Suspend, Inhibit



The level activity bit for the designated level is set. The level activity bit for the current level is then cleared. The interrupt vector for the designated level is set equal to the interrupt vector for the current level. Execution of the task continues at the designated level.

Suspend, Scan, and Dispatch



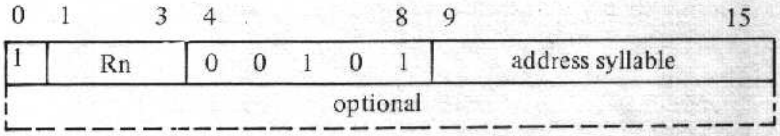
Ends execution at the current level. The level activity bit for the current level is cleared. The level activity bits are scanned and the highest active level ascertained. The context of the current level is saved. The context of the highest active level is restored (unless the current level and the highest active level share a common save area).

LLH

Instruction:
Load logical half-word (byte) into R register

Type:
Double Operand

Format:



Description:
Loads the byte contained at the effective address into the right half of the designated R register, with the left half, bits 0–7 being cleared to zero. No indicators are affected.

The effective address can specify a memory location, an immediate operand, or another R register, with the byte being positioned as follows:

– memory location, not indexed	– left byte
– memory location, indexed	
index value even	– left byte
index value odd	– right byte
– immediate operand	– left byte
– R register	– right byte

Example: Assume that:

memory location 1000 contains $(6789)_{16}$
register R1 contains 0
register R2 contains 1
register B1 contains $(1000)_{16}$

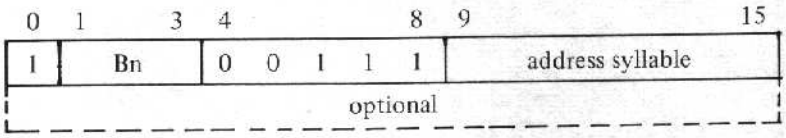
then if LLH $R5, $B1.$R1 is executed: R5 will contain $(0067)_{16}$
but if LLH $R5, $B1.$R2 is executed: R5 will contain $(0089)_{16}$

LNJ

Instruction:
Load B register and jump (link jump)

Type:
Double Operand

Format:



Description:
Loads the address of the next sequential instruction into the designated B register and jumps to the location specified by the effective address. If the J bit in the M1 register is on, this instruction traps to trap vector #2.

No indicators are affected. The address syllable must specify a memory location, not a register or an immediate operand.

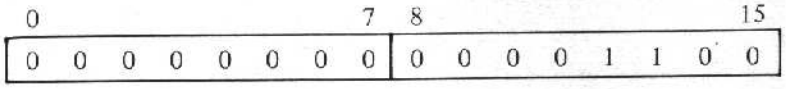
LRDB

Instruction:
Load remote descriptor base

Restriction:
Traps (TV#5) on models without the Memory Management Unit option

Type:
Generic

Format:



Description:
Loads the contents of register B3 into the Remote Descriptor Base Register.

NOTE: Refer to Appendix H of the *GCOS 6 Assembly Language Reference* manual, Order No. CB07 for an explanation concerning the use of Remote Descriptors and Data Descriptors.

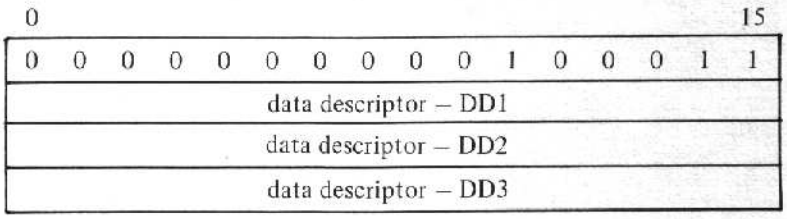
MAT

Instruction:
Alphanumeric move and translate

Restriction:
Traps (TV#5) on Models 23, 33, 43, and 53

Type:
Alphanumeric

Format:



Description:
The character string in the sending field (specified by the first data descriptor) is translated and moved to the receiving field (specified by the second data descriptor). The third data descriptor specifies a 256-byte translation table. Each character in the sending field is used as a displacement from the base of the table and the corresponding character from the table is stored in the receiving field.

If the byte length specified by the first data descriptor is zero, the length is contained in the right byte of register R4 and can be from 0 through 255 bytes. If the byte length specified in the first data descriptor is not zero, the value, which can be from 1 through 31, is the length.

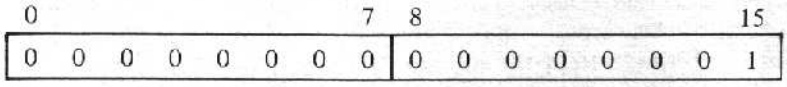
If the byte length specified in the second data descriptor is not zero, the value, which can be from 1 through 31, is the length. If the byte length specified by the second data descriptor is zero, then R5 contains the length on the right byte, and a fill character on the left byte. In this case, the length can be 0 to 255 bytes. If the length in the data descriptor is non-zero, then the fill character is an ASCII blank (20_{16}). Fill characters, if fill is specified by data descriptors, are not translated. If the length of the receiving field specified by register R5 is zero, the instruction is aborted and the truncation bit (TR bit) of the CIP indicator register is set to 1.

MCL

Instruction:
Monitor call

Type:
Generic

Format:



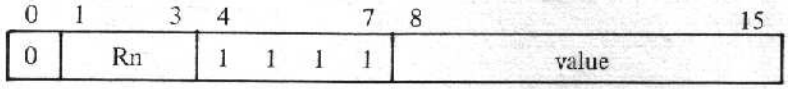
Description:
Calls monitor by a trap to trap vector #1.

MLV

Instruction:
Multiply R-register by value

Type:
Short Value Immediate

Format:



Description:
Multiplies the value in the designated R-register by the value (with sign extended) contained in the instruction. A single precision 16-bit product is formed, with overflow bit set if necessary, unless the designated register is R7, in which case a double-precision, 32-bit signed product is developed in R6 and R7.

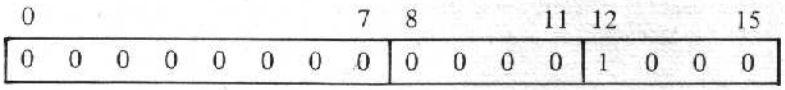
MMM

Instruction:
Memory to memory move

Restriction:
Traps (TV#5) on Models 23 and 33

Type:
Generic

Format:



Description:
Moves "n" bytes of memory from one location to another. The address of the first byte to be moved is identified by [B2] + [R2], where R2 contains a signed byte displacement from [B2]. The first byte location to be moved to is identified by [B3] + [R3], where

R3 contains a signed byte displacement from [B3]. The number of bytes to be moved is contained in R6. This number is added to the contents of R2 and R3, and R6 is decremented to zero. This instruction is interruptable and intermediate results in R2, R3, and R6 may become visible. Attempts to move partially overlapped fields cause unspecified results.

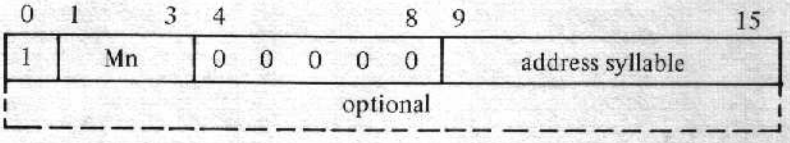
The contents of the I Register are unaffected. If R6(0) = 1, Trap 16 occurs. Trap 15 occurs if the move would exceed available memory or cause register overflow.

MTM

Instruction:
Modify/test M register

Type:
Double Operand

Format:



Description:
Modifies and/or tests the contents of the M register identified in the first operand with the contents (mask) of the location specified by the address syllable.

The address syllable can specify a memory location, an immediate operand, or an R register. If the CP does not contain the designated M register, the MTM traps to trap vector #5.

The mask is treated as two 8-bit fields; then, depending on the content of corresponding bits in the two fields (i.e., bit 1 in the first field and bit 1 in the second; bit 2 in the first field and bit 2 in the second; etc.), the corresponding bit in the M register (i.e., if bit 1 in the two mask fields, then bit 1 in the M register) is altered or tested as follows:

- o If bit n in the first mask field is 1, the corresponding bit in the M register is loaded with the contents of the corresponding bit from the second mask field (i.e., M register is modified).
- o If bit n in the first mask field is 0 and the same bit in the second mask field is 1, the corresponding bit in the M register is tested. If any of the bits so tested is 1, the B bit in the I register is set to 1; otherwise, it is set to 0 (i.e., M register is tested).
- o If bit n in the first mask field is 0 and the same bit in the second mask field is 0, the corresponding bit in the M register is neither modified nor tested.

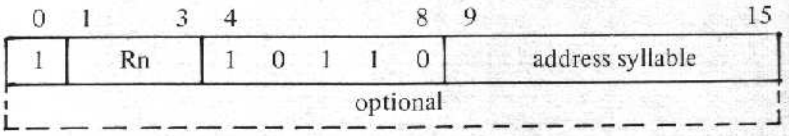
NOTE: The assembly language instructions LEV, SAVE, and STM store the contents of the M register in a form suitable for reloading by MTM.

MUL

Instruction:
Multiply R register

Type:
Double Operand

Format:



Description:

Multiplies the word in the designated R register by the word in the effective address. The product is a 16-bit value with the high order bits truncated unless R7 is specified, in which case a double-precision, 32-bit signed product is developed in R6 and R7. Single precision multiplication sets or clears the overflow (OV) indicator depending on the size of the product; double-precision multiplication cannot cause overflow and thus OV is cleared.

The address syllable can specify a memory location, an immediate operand, or another R register.

The contents of the I register are affected as follows:

- o If R7 is not specified and the product is more than $2^{15} - 1$ (32,767) or less than -2^{15} (−32,768), the OV bit is set to 1; otherwise, it is set to 0.

NEG

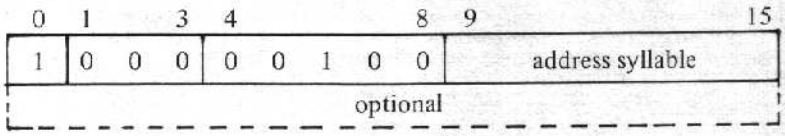
Instruction:

Negate

Type:

Single Operand

Format:



Description:

Twos complements the contents of the location specified in the address syllable.

The address syllable can specify a memory location, an immediate operand, or one of the R registers.

The contents of the I register are affected as follows:

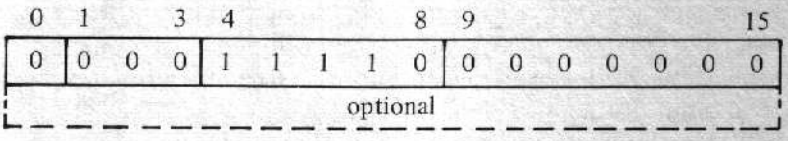
- o If a carry occurs (i.e., the value was zero) during the operation, the C bit is set to 1; Otherwise, it is set to 0.
- o If the value complemented was −32,768, the OV bit is set to 1; otherwise, it is set to 0. However, this cannot cause an overflow trap.

NOP

Instruction:
No operation

Type:
Branch on Indicator

Format:



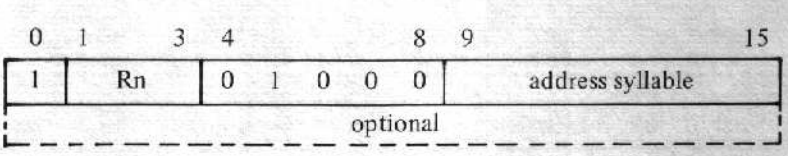
Description:
Performs no operation. In effect, it is the opposite of an unconditional branch (B) instruction.

OR

Instruction:
Inclusive OR with R register

Type:
Double Operand

Format:



Description:
Logically ORs the word at the effective address to the word contained in the designated R register. No indicators are affected. The address syllable can specify a memory address, an immediate operand, or another R register.

The following chart illustrates the result of inclusively ORing bits:

First operand bit	0	0	1	1
Second operand bit	1	0	1	0
Result	1	0	1	1

Examples:

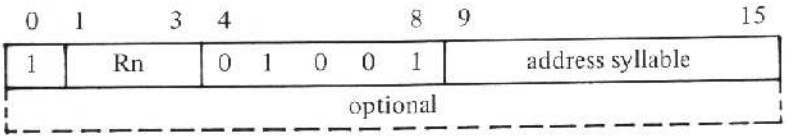
R register (before)	Effective Address	R register (after)
$(0000)_{16}$	$(0007)_{16}$	$(0007)_{16}$
$(FFF0)_{16}$	$(0007)_{16}$	$(FFF7)_{16}$
$(FFF3)_{16}$	$(0007)_{16}$	$(FFF7)_{16}$

ORH

Instruction:
Inclusive OR or half-word (byte)

Type:
Double Operand

Format:



Description:
Logically ORs the byte at the effective address (with its sign extended) to the word contained in the designated R register. No indicators are affected. The address syllable can specify a memory address, an immediate operand, or another R register, with the byte being positioned as follows:

- memory location, not indexed — left byte
- memory location, indexed
  - index value even — left byte
  - index value odd — right byte
- immediate operand — left byte
- R register — right byte

The following chart illustrates the results of inclusively ORing bits:

First operand bit	0	0	1	1
Second operand bit	1	0	1	0
Result	1	0	1	1

Examples:

R register (before)	Effective Address	R register (after)
$(0000)_{16}$	$(07)_{16}$	$(0007)_{16}$
$(FFF0)_{16}$	$(07)_{16}$	$(FFF7)_{16}$
$(FFF3)_{16}$	$(07)_{16}$	$(FFF7)_{16}$
$(0007)_{16}$	$(87)_{16}$	$(FF87)_{16}$

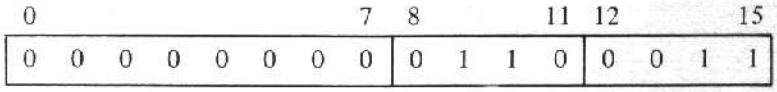
QOH

Instruction:
Queue on head

Restriction:
Traps (TV#5) on Models 23 and 33

Type:
Generic

Format:



Description:
Links a new frame into the list before the first frame that has the same priority number, before the first frame that has a numerically higher priority number, or as the last frame if no equal or greater priority is found. B2 is used to point to the lock word of the list, and R5 contains the priority to be assigned to the new frame. The priority word will be loaded with the contents of R5. B1 points to the priority word of the frame to be added. The pointer following the priority word will be loaded by the CP with the correct frame pointer.

The contents of the I register are affected as follows:

- o If the operation is completed, C is set to 1; otherwise 0.

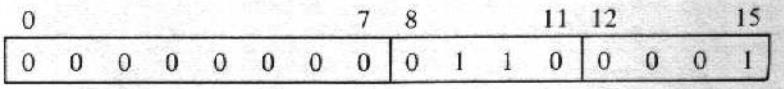
QOT

Instruction:
Queue on tail

Restriction:
Traps (TV#5) on Models 23 and 33

Type:
Generic

Format:



Description:

Links a new frame into the list after the last frame that has the same priority number, before the first frame that has a numerically higher priority number, or as the last frame if no equal or greater priority is found. B1, B2, and R5 are used as QOH.

The contents of the I register are affected as follows:

- o If the operation is completed, C is set to 1; otherwise 0.

RLQ

Instruction:

Relinquish stack space

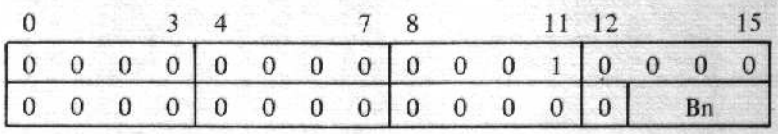
Restriction:

Traps (TV#5) on Models 23 and 33

Type:

Generic

Format:



Description:

Converts a consumed stack frame into available space by updating the current length in words (CW) in the stack header. Bn is loaded with the leftmost address of the frame at the top of the stack. The contents of the I register are unaffected.

Trap 16 will occur if CW is already zero, or if RLQ would cause CW to go negative.

Trap 9 will occur if CW is reduced to zero.

RSC

Instruction:

Rescan configuration

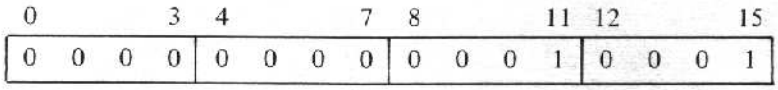
Restriction:

Traps (TV#5) on Models 23 and 33

Type:

Generic

Format:



Description:

A privileged instruction that causes the CP to transmit the contents of R7 to the channel number (and with the function code) specified in R6, and then to scan a predetermined list of channel numbers to determine the presence/absence of optional processors. The results of the scan are used to update internal CP firmware/hardware flags that direct instruction execution (e.g., trap or execution by optional processor).

An identical scan is performed automatically on system power up or initialize. The contents of the I register are unaffected.

RSTR

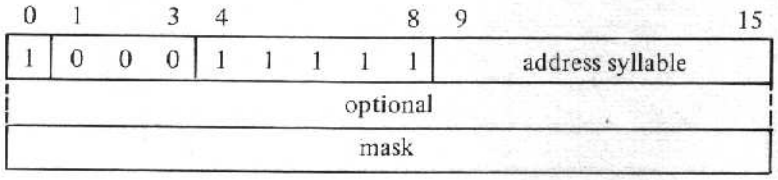
Instruction:

Restore context

Type:

Single Operand

Format:

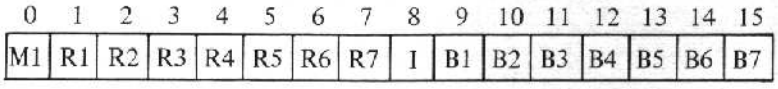


Description:

Restores the registers specified in the mask starting from the location specified in the address syllable.

The address syllable must specify a memory location. The mask specifies which registers are to be restored. If the mask is all zeros, the contents of R1 are used as the mask.

Depending on which bits in the specified mask are set to 1, the registers that can be restored are as follows:



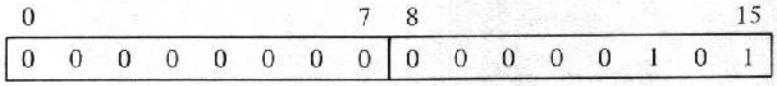
This mask should be the same as the one used to save the registers (see the SAVE instruction).

RTCF

Instruction:
Real-time clock off

Type:
Generic

Format:



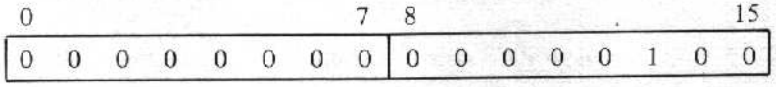
Description:
Disables real-time clock. The P field in the S register must equal 1x (i.e., the central processor must be in the privileged state) for this instruction to be executed. If not, the unprivileged use of a privileged operation is signified by a trap to trap vector #13.

RTCN

Instruction:
Real-time clock on

Type:
Generic

Format:



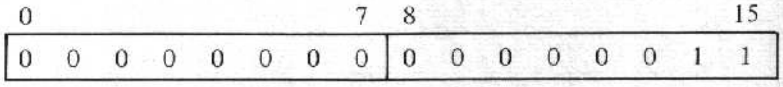
Description:
Starts the real-time clock. The P field in the S register must equal 1x (i.e., the central processor must be in the privileged state) for this instruction to be executed. If not, the unprivileged use of a privileged operation is signified by a trap to trap vector #13.

RTT

Instruction:
Return from trap

Type:
Generic

Format:



RTT/SAD/SAL

Description:
Restores the registers that were saved in the trap save area when the trap was entered. Returns the trap save area block to the trap save area memory pool. Returns control to the next instruction to be executed (determined by the pointer in the trap save area).

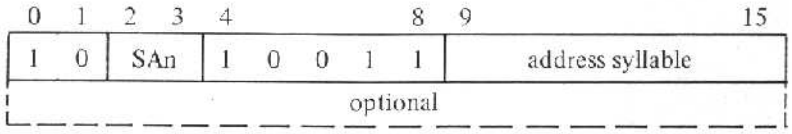
SAD

Instruction:
Scientific add

Restriction:
Traps (TV#5) on models without Scientific Instruction Processor

Type:
Double Operand

Format:



Description:
Adds the floating-point or integer value in the location, scientific accumulator, or R register identified in the address syllable to the contents of the scientific accumulator specified in the first operand. The result is saved in the scientific accumulator, SAn. If exponent underflow occurs and the EUM in M5 is 0, then SAn is set to 0; otherwise, the exponent of SAn is 128 too large.

This instruction uses the optional Scientific Instruction Processor (SIP). A Floating-Point Simulator is available to allow this instruction to be executed on systems that do not include an SIP.

Scientific indicator settings are as follows:

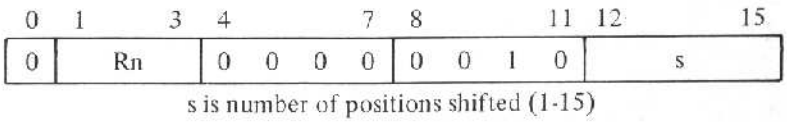
- EU – set to 1 on exponent underflow; otherwise, set to 0.

SAL

Instruction:
Single-shift arithmetic-left

Type:
Shift Short

Format:



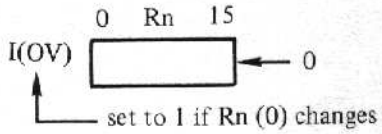
Description:

Shifts the contents of the R register, identified in the Rn field, left the number of bit positions specified in the s field. The bit positions vacated by the shift are filled with binary 0s. If the s field contains 0, the shift distance is obtained from bits 12 through 15 of general register R1.

The contents of the I register are affected as follows:

- o If the contents of bit 0 in the R register changes, the OV bit is set to 1; otherwise, it is set to 0.

Pictorial Representation:



SAR

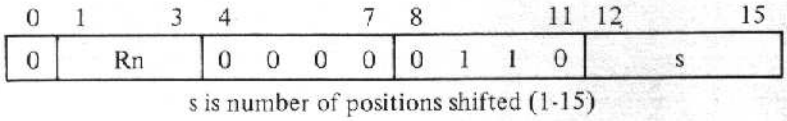
Instruction:

Single-shift arithmetic-right

Type:

Shift Short

Format:



s is number of positions shifted (1-15)

Description:

Shifts the contents of the R register, identified in the Rn field, right the number of bit positions specified in the s field. The bit positions vacated by the shift are filled with the sign value originally contained in bit 0.

If the s field contains 0, the shift distance is obtained from bits 12 through 15 of general register R1.

The contents of the I register are affected as follows:

- o C bit contains the last binary digit shifted out of the R register.

SAVE

Instruction:

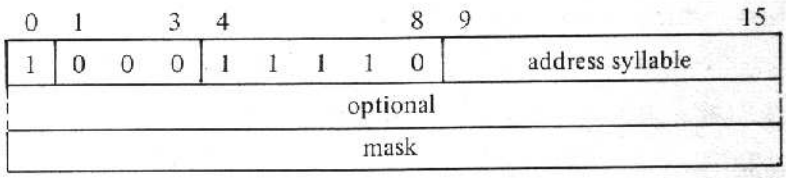
Save context

Type:

Single Operand

SAVE/SBE

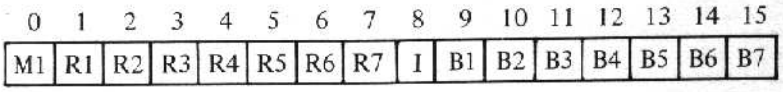
Format:



Description:

Saves the registers specified in the mask starting at the location specified in the address syllable.

The address syllable must specify a memory address. The mask specifies which registers are to be saved. Each bit in the mask represents a particular register which can be saved, as shown below:



If a mask bit is set to 1, the corresponding register is saved. If a mask bit is 0, the corresponding register is not saved. If the mask is all zeros, the contents of R1 are used as the mask.

The registers are saved in reverse order. For example, if the mask specified X'CA01' (which, when translated into binary is 1100 1010 0000 0001), indicating that registers M1, R1, R4, R6, and B7 are to be saved, the context save area will contain the registers starting with B7 and ending with M1.

SBE

Instruction:

Scientific branch if equal

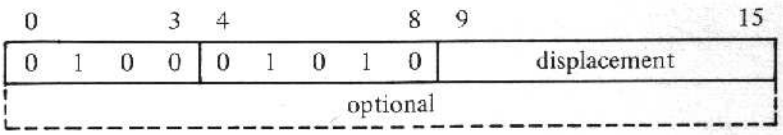
Restriction:

Traps (TV#5) on models without Scientific Instruction Processor

Type:

Scientific Indicator Branch

Format:



Description:

Branches to the specified location if the result of the most recent scientific comparison sets the SL- and SG-bits of the SI register to 0.

If the J-bit in the mode control (M1) register is set, and if the branch would have taken place, this instruction traps to trap vector #2.

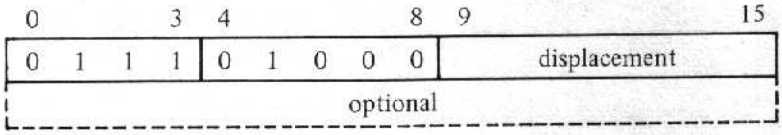
SBEU

Instruction:
Scientific branch if exponent underflow

Restriction:
Traps (TV#5) on models without Scientific Instruction Processor

Type:
Scientific Indicator Branch

Format:



Description:
Branches to the specified location if the EU-bit of the SI register is equal to 1.
If the J-bit in the mode control (M1) register is set, and if the branch would have taken place, this instruction traps to trap vector #2.

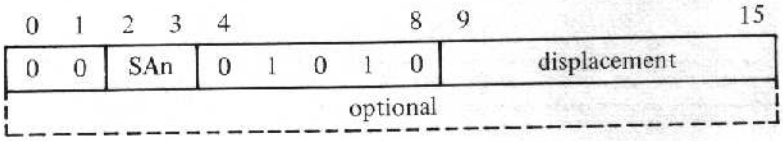
SBEZ

Instruction:
Scientific branch if SA equal to zero

Restriction:
Traps (TV#5) on models without Scientific Instruction Processor

Type:
Scientific Accumulator Branch

Format:



Description:
Branches to the specified location if the designated scientific accumulator contains a floating-point value algebraically equal to zero.
If the J-bit in the mode control (M1) register is set, and if the branch would have taken place, this instruction traps to trap vector #2.

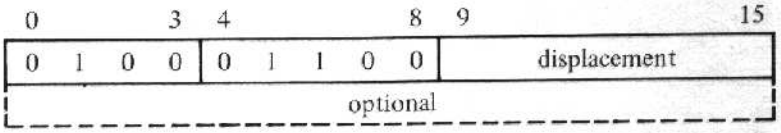
SBG

Instruction:
Scientific branch if greater than

Restriction:
Traps (TV#5) on models without Scientific Instruction Processor

Type:
Scientific Indicator Branch

Format:



Description:
Branches to the specified location if the result of the most recent scientific comparison sets the SG-bit of the SI register to 1.

If the J-bit in the mode control (M1) register is set, and if the branch would have taken place, this instruction traps to trap vector #2.

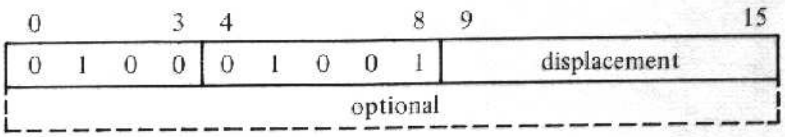
SBGE

Instruction:
Scientific branch if greater than or equal

Restriction:
Traps (TV#5) on models without Scientific Instruction Processor

Type:
Scientific Indicator Branch

Format:



Description:
Branches to the specified location if the result of the most recent scientific comparison sets the SL-bit of the SI register to 0.

If the J-bit in the mode control (M1) register is set, and if the branch would have taken place, this instruction traps to trap vector #2.

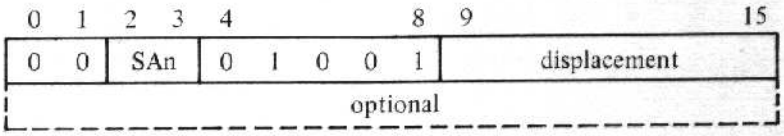
SBGEZ

Instruction:
Scientific branch if SA greater than or equal to zero

Restriction:
Traps (TV#5) on models without Scientific Instruction Processor

Type:
Scientific Accumulator Branch

Format:



Description:
Branches to the specified location if the designated scientific accumulator contains a non-negative floating-point value.

If the J-bit in the mode control (M1) register is set, and if the branch would have taken place, this instruction traps to trap vector #2.

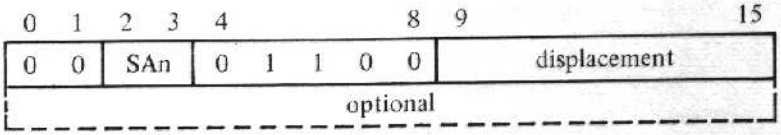
SBGZ

Instruction:
Scientific branch if SA greater than zero

Restriction:
Traps (TV#5) on models without Scientific Instruction Processor

Type:
Scientific Accumulator Branch

Format:



Description:
Branches to the specified location if the designated scientific accumulator contains a positive floating-point value.

If the J-bit in the mode control (M1) register is set, and if the branch would have taken place, this instruction traps to trap vector #2.

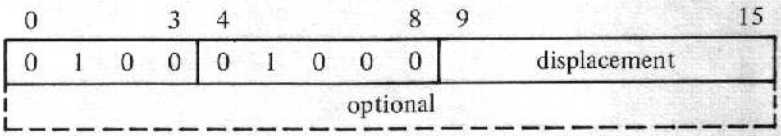
SBL

Instruction:
Scientific branch if less than

Restriction:
Traps (TV#5) on models without Scientific Instruction Processor

Type:
Scientific Indicator Branch

Format:



Description:
Branches to the specified location if the result of the most recent scientific comparison sets the SL-bit of the SI register to 1.

If the J-bit in the mode control (M1) register is set, and if the branch would have taken place, this instruction traps to trap vector #2.

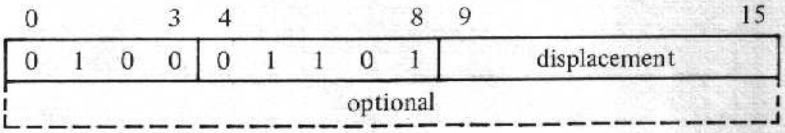
SBLE

Instruction:
Scientific branch if less than or equal

Restriction:
Traps (TV#5) on models without Scientific Instruction Processor

Type:
Scientific Indicator Branch

Format:



Description:
Branches to the specified location if the result of the most recent scientific comparison sets the SG-bit of the SI register to 0.

If the J-bit in the mode control (M1) register is set, and if the branch would have taken place, this instruction traps to trap vector #2.

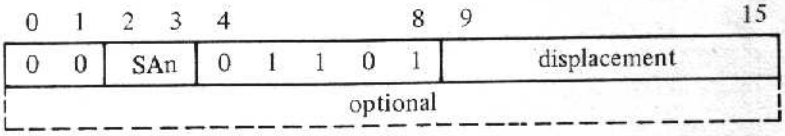
SBLEZ

Instruction:
Scientific branch if SA less than or equal to zero

Restriction:
Traps (TV#5) on models without Scientific Instruction Processor

Type:
Scientific Accumulator Branch

Format:



Description:
Branches to the specified location if the designated scientific accumulator identified in the first operand contains a floating-point value algebraically equal to or less than 0.

If the J-bit in the mode control (M1) register is set, and if the branch would have taken place, this instruction traps to trap vector #2.

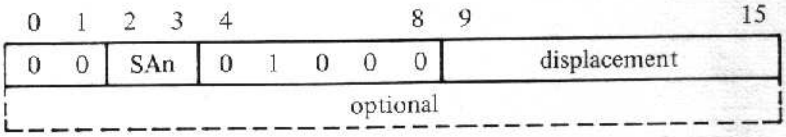
SBLZ

Instruction:
Scientific branch if SA less than zero

Restriction:
Traps (TV#5) on models without Scientific Instruction Processor

Type:
Scientific Accumulator Branch

Format:



Description:
Branches to the specified location if the designated scientific accumulator identified in the first operand contains a negative floating-point value.

If the J-bit in the mode control (M1) register is set, and if the branch would have taken place, this instruction traps to trap vector #2.

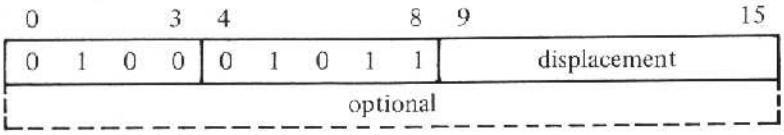
SBNE

Instruction:
Scientific branch if not equal

Restriction:
Traps (TV#5) on models without Scientific Instruction Processor.

Type:
Scientific Indicator Branch

Format:



Description:
Branches to the location in the operand if the result of the most recent scientific comparison sets either the SL- or SG-bit of the SI register to 1.

If the J-bit in the mode control (M1) register is set, and if the branch would have taken place, this instruction traps to trap vector #2.

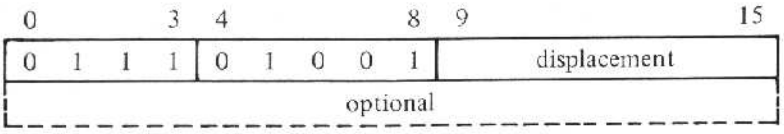
SBNEU

Instruction:
Scientific branch if no exponent underflow

Restriction:
Traps (TV#5) on models without Scientific Instruction Processor

Type:
Scientific Indicator Branch

Format:



Description:
Branches to the specified location if the EU-bit of the SI register is equal to 0.

If the J-bit in the mode control (M1) register is set, and if the branch would have taken place, this instruction traps to trap vector #2.

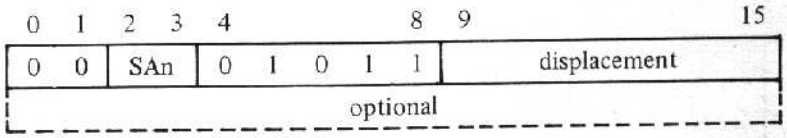
SBNEZ

Instruction:
Scientific branch if SA not equal to zero

Restriction:
Traps (TV#5) on models without Scientific Instruction Processor

Type:
Scientific Accumulator Branch

Format:



Description:
Branches to the specified location if the designated scientific accumulator identified in the first operand contains a floating-point value not algebraically equal to 0.

If the J-bit in the mode control (M1) register is set, and if the branch would have taken place, this instruction traps to trap vector #2.

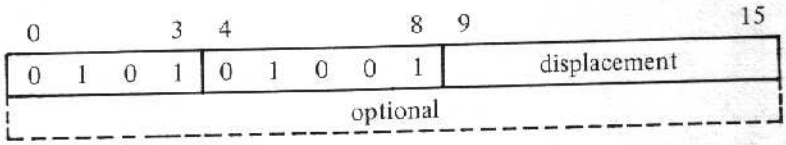
SBNPE

Instruction:
Scientific branch if no precision error

Restriction:
Traps (TV#5) on models without Scientific Instruction Processor

Type:
Scientific Indicator Branch

Format:



Description:
Branches to the specified location if the PE-bit of the SI register is equal to 0.

If the J-bit in the mode control (M1) register is set, and if the branch would have taken place, this instruction traps to trap vector #2.

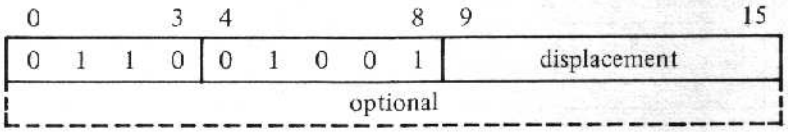
SBNSE

Instruction:
Scientific branch if no significance error

Restriction:
Traps (TV#5) on models without Scientific Instruction Processor

Type:
Scientific Indicator Branch

Format:



Description:
Branches to the specified location if the SE-bit of the SI register is equal to 0.
If the J-bit in the mode control (M1) register is set, and if the branch would have taken place, this instruction traps to trap vector #2.

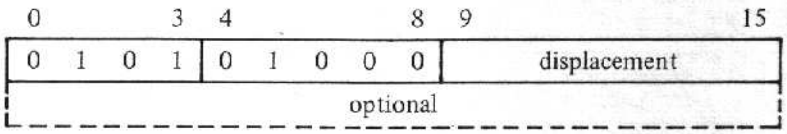
SBPE

Instruction:
Scientific branch if precision error

Restriction:
Traps (TV#5) on models without Scientific Instruction Processor

Type:
Scientific Indicator Branch

Format:



Description:
Branches to the specified location if the PE-bit of the SI register is equal to 1.
If the J-bit in the mode control (M1) register is set, and if the branch would have taken place, this instruction traps to trap vector #2.

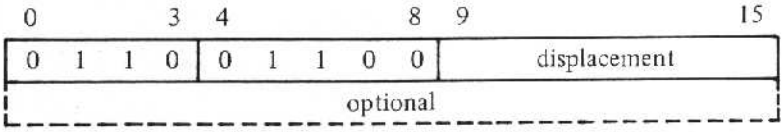
SBSE

Instruction:
Scientific branch if significance error

Restriction:
Traps (TV#5) on models without Scientific Instruction Processor

Type:
Scientific Indicator Branch

Format:



Description:
Branches to the specified location if the SE-bit of the SI register is equal to 1.

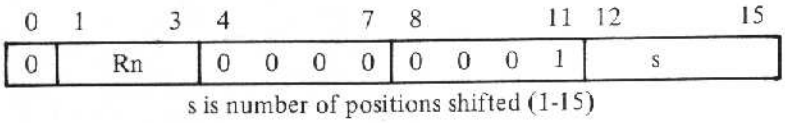
If the J-bit in the mode control (M1) register is set, and if the branch would have taken place, this instruction traps to trap vector #2.

SCL

Instruction:
Single-shift closed-left

Type:
Shift Short

Format:

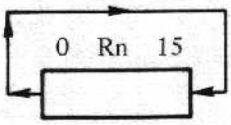


s is number of positions shifted (1-15)

Description:
Shifts the contents of the R register identified in the Rn field left the number of bit positions specified in the s field. The bits shifted out of the register are placed in the bit positions vacated by shifted bits as they are shifting.

If the s field contains 0, the shift distance is obtained from bits 12 through 15 of general register R1. No indicators are affected.

Pictorial Representation:



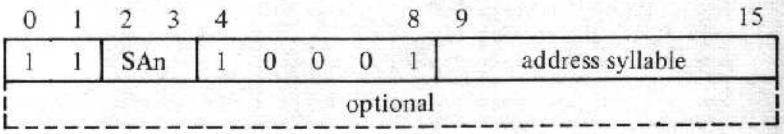
SCM

Instruction:
Scientific Compare

Restriction:
Traps (TV#3) on models without Scientific Instruction Processor.

Type:
Double Operand

Format:



Description:
Compares the contents of the scientific accumulator identified in the first operand to the floating-point or integer value in the location specified in the second operand.
Scientific indicator settings include the following:

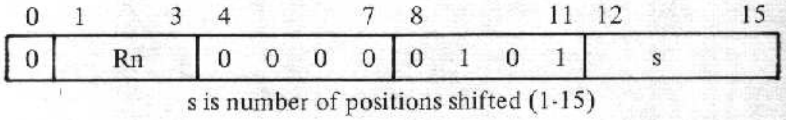
SG – Set to 1 if contents of the scientific accumulator are greater than the contents of the location; otherwise, set to 0.
SL – Set to 1 if contents of the scientific accumulator are less than the contents of the location; otherwise, set to 0.

SCR

Instruction:
Single-shift closed-right

Type:
Shift Short

Format:



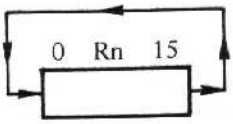
s is number of positions shifted (1-15)

Description:

Shifts the contents of the R register, identified in the Rn field, right the number of bit positions specified in the s field. The bits shifted out of the register are placed in the bit positions vacated by shifted bits as they are shifting.

If the s field contains 0, the shift distance is obtained from bits 12 through 15 of general register R1. No indicators are affected.

Pictorial Representation:



SCZD

Instruction:

Scientific compare to zero (2 words)

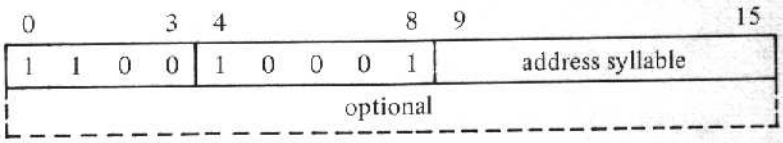
Restriction:

Traps (TV#3) on models without Scientific Instruction Processor

Type:

Single Operand

Format:



Description:

Compares the short-precision floating-point value in the specified location or scientific accumulator to 0.

Scientific indicator settings include the following:

SL – Set to 1 if contents of the location are less than 0; otherwise, set to 0.
SG – Set to 1 if the contents of the location are greater than 0; otherwise, set to 0.
PE – Set to 1 if nonzero bits are lost during right shift for sacaling before comparison; otherwise, set to 0.

SCZQ

Instruction:

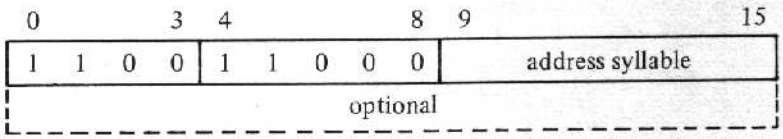
Scientific compare to zero (4 words)

SCZQ/SDI

Restriction:
Traps (TV#3) on models without Scientific Instruction Processor

Type:
Single Operand

Format:



Description:
Compares the floating-point value in the specified location or scientific accumulator to 0. Scientific indicator settings include the following:

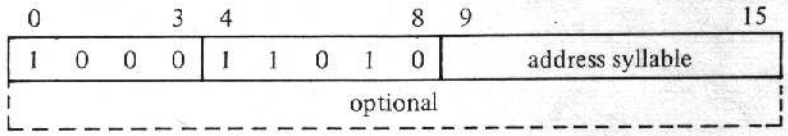
SL – Set to 1 if contents of the location are less than 0; otherwise, set to 0.
SG – Set to 1 if the contents of the location are greater than 0; otherwise, set to 0.

SDI

Instruction:
Store double-word integer

Type:
Single Operand

Format:



Description:
Stores the contents of R6 and R7 into the effective address and the following word. No indicators are affected. The address syllable can specify a memory location, an immediate operand, or another R register as follows:

- *Memory location:* the left-hand word of a double-word pair will be addressed. Auto-increment and auto-decrement modes of addressing will add (subtract) two from the base register. A value in the index register will count double words; it is shifted one place to the left before being applied. Auto-indexing will cause one to be added (subtracted) from the index register.
- *Immediate operand:* this will be a double-word operand and thus the instruction will be three words in length.
- *Register operand:* this form of addressing must address the right-hand register containing a double-word pair, i.e., either R3, which causes the contents of R6 and R7 to be stored in R2 and R3, or R5, which selects R4 and R5.

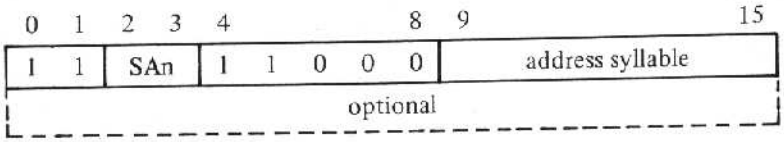
SDV

Instruction:
Scientific divide

Restriction:
Traps (TV#3) on models without Scientific Instruction Processor

Type:
Double Operand

Format:



Description:
Divides the contents of the scientific accumulator identified in the first operand by the contents of the location, scientific accumulator, or R register specified in the second operand. The result is saved in the scientific accumulator (except for the remainder, which is ignored).

If the second operand is =$R4, =$R5, or =$R6, the integer value contained in the specific R register is internally converted to floating-point format before it is divided into the SA register specified by the first operand.

Scientific indicator settings include the following:

EU – Set to 1 on exponent underflow; otherwise, set to 0.

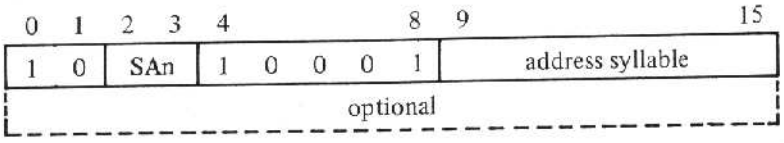
SLD

Instruction:
Scientific load

Restriction:
Traps (TV#3) on models without Scientific Instruction Processor

Type:
Double Operand

Format:



Description:
Loads the contents of the location, scientific accumulator, or R register identified in the second operand into the scientific accumulator identified in the first operand.

Scientific indicator settings include the following:

EU – Set to 1 on exponent underflow; otherwise, set to 0.

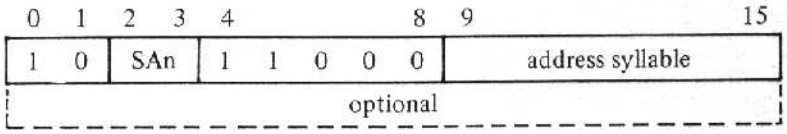
SML

Instruction:
Scientific multiply

Restriction:
Traps (TV#3) on models without Scientific Instruction Processor

Type:
Double Operand

Format:



Description:
Multiplies the contents of the scientific accumulator identified in the first operand by the contents of the location, scientific accumulator, or R register specified in the second operand. The result is saved in the scientific accumulator.

If the second operand is an R register, the integer value contained in the specific R register is internally converted to floating-point format before it is multiplied to the SA register specified by the first operand.

Scientific indicator settings include the following:

EU – Set to 1 on exponent underflow; otherwise, set to 0.

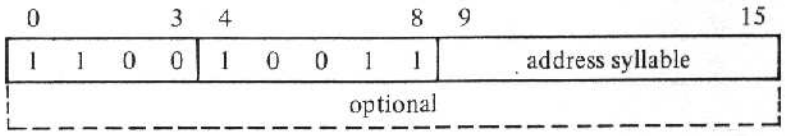
SNGD

Instruction:
Scientific negate (2 words)

Restriction:
Traps (TV#3) on models without Scientific Instruction Processor

Type:
Single Operand

Format:



Description:
Negates the short-precision floating-point number at the location or scientific accumulator specified in the operand.

If the address syllable specifies =$R4, =SR5, =SR6, or =$R7, then a scientific program error trap (TV#20) occurs.

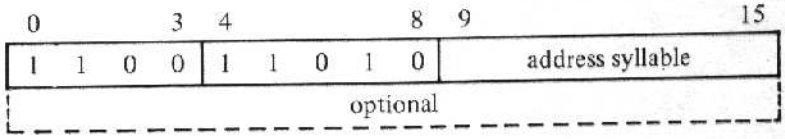
SNGQ

Instruction:
Scientific negate (4 words)

Restriction:
Traps (TV#3) on models without Scientific Instruction Processor

Type:
Single Operand

Format:



Description:
Negates the long-precision floating-point number at the location or scientific accumulator specified in the operand. If the address syllable specifies =$R4, =$R5, =$R6, or =$R7, then a scientific program error trap (TV#20) occurs.

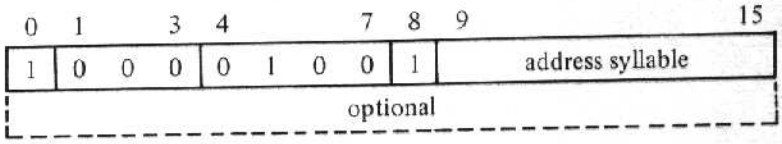
SID

Instruction:
Subtract double-word integer

Restriction:
Traps (TV#5) on Models 23 and 33

Type:
Double Operand

Format:



Description:
Loads the difference of the double-word integer contained in R6 and R7 and the contents of the double-word location specified by the address syllable in R6 and R7.
R6(0) is considered the high-order bit.
R7(15) is considered the low-order bit.
The contents of the I register are affected as follows:

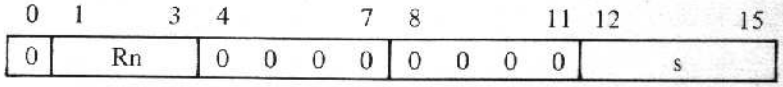
- o If carry C is set to 1; otherwise 0.
- o If overflow OV is set to 1; otherwise 0.

SOL

Instruction:
Single-shift open-left

Type:
Shift Short

Format:



s is number of positions shifted (1-15)

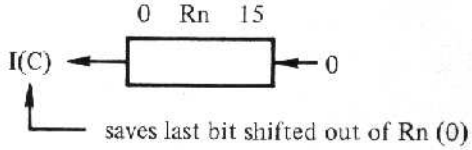
Description:
Shifts the contents of the R register, identified in the Rn field, left the number of bit positions specified in the s field. The bit positions vacated by the shift are filled with binary 0s.

If the s field contains 0, the shift distance is obtained from bits 12 through 15 of general register R1.

The contents of the I register are affected as follows:

- C bit contains the last binary digit shifted out of the R register.

Pictorial Representation:

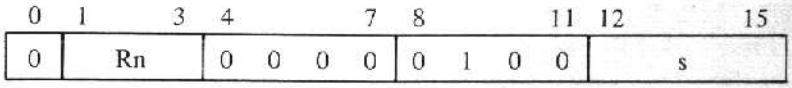


SOR

Instruction:
Single-shift open-right

Type:
Shift Short

Format:



s is number of positions shifted (1-15)

Description:

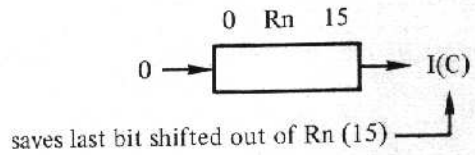
Shifts the contents of the R register, identified in the Rn field, right the number of bit positions specified in the s field. The bit positions vacated by the shift are filled with binary 0s.

If the s field contains 0, the shift distance is obtained from bits 12 through 15 of general register R1.

The contents of the I register are affected as follows:

- o C bit contains the last binary digit shifted out of the R register.

Pictorial Representation:



SRCH

Instruction:

Alphanumeric search

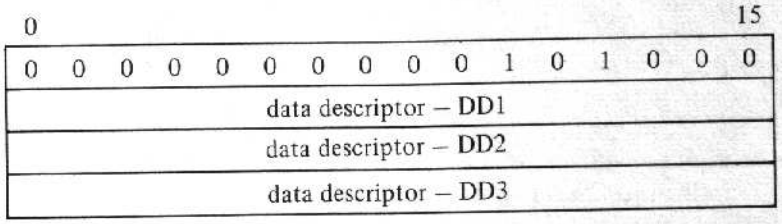
Restriction:

Traps (TV#5) on Models 23, 33, 43, and 53

Type:

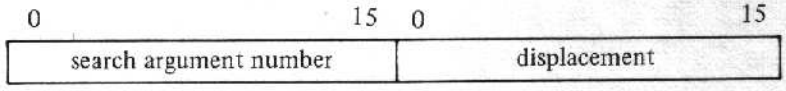
Alphanumeric

Format:



Description:

The character string or array of character strings defined by the third data descriptor (DD3) is searched to see if it contains any of the search arguments (one or more) in the search list defined by the first data descriptor (DD1). If a match is found, the G and L bits of the CIP indicator register are cleared to zero, and the displacement and search argument number are loaded into the receiving field defined by the second data descriptor (DD2). The receiving field must be four bytes long and word aligned, otherwise the results are unspecified. The displacement is the distance in bytes between the origin of the string (or array) to be searched and the position at which the first match occurs. The search argument number designates the one that causes the match. The first argument in the list is identified as 0, the second as 1, etc. The format of the receiving field is shown below.

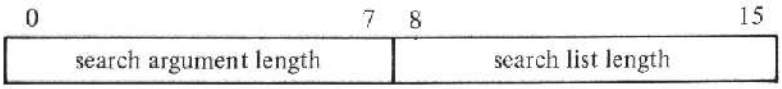


If a match is not found, the G-bit of the CIP indicator register is cleared to zero, the L-bit is set to one, and the receiving field is not changed.

The search argument list can contain one or more search arguments each consisting of one or more characters. If multiple arguments are specified, each must be the same length.

If the length field of DD1 is not equal to zero, the search argument list contains only one search argument whose length (1 to 31 bytes) is specified by the field.

If the length field of DD1 is equal to zero, the search argument list is specified by register R4. The format of register R4 is shown below.

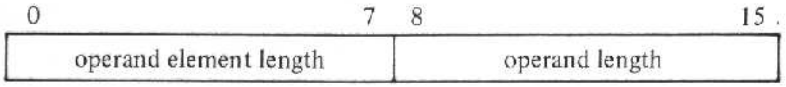


If the search argument length is equal to the search list length, the search list consists of only one argument.

If the ratio of the search list length to the search argument length is an integer, that integer designates the number of search arguments.

If the ratio of the search list length to the search argument length is not an integer, the ratio is truncated to the integer value and that integer designates the number of search arguments.

The character string (or array) to be searched is specified by DD3. If the length field of DD3 is not equal to zero, the operand is a character string whose length (1 through 31) is specified by the length field. If the length field is equal to zero, the operand to be searched is specified by register R6. The format of register R6 is shown below.



If the operand element length is one, the operand is a character string whose length (specified by the low order byte of R6) must be from 1 through 255. In this case the search argument length must be less than or equal to the operand length, otherwise a *not found* indication will result.

If the operand element length is not equal to one, it specifies the length of each element of an array. The length of the array is the largest multiple of the operand element length that is less than or equal to the operand length (specified by the low order byte of R6). In this case, if the search argument length is greater than the operand element length, each search will overflow into the next entry except when the last operand element is searched. Thus the last comparison will result in a mismatch.

SRDB

Instruction:

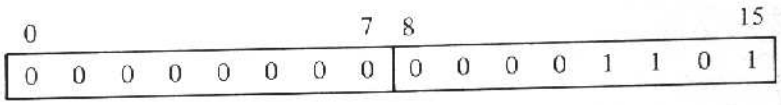
Store remote descriptor base

Restriction:

Traps (TV#5) on models without the Memory Management Unit option.

Type:
Generic

Format:



Description:
Stores the contents of the Remote Descriptor Base Register in register B3.

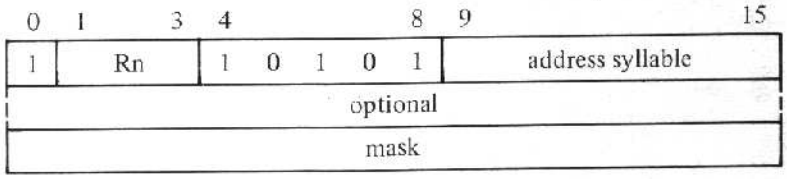
NOTE: Refer to Appendix H of the *GCOS 6 Assembly Language Reference* manual, Order No. CB07 for an explanation concerning the use of Remote Descriptors and Data Descriptors.

SRM

Instruction:
Store register masked

Type:
Double Operand

Format:



Description:
Stores contents of the designated register in the effective address under control of a 16-bit mask. Every bit (zero or one) in the register with a corresponding mask bit of one is stored; those corresponding to zero bits in the mask are not ransferred and the original bits in the effective address remain unchanged.

The mask is contained in the word following the instruction. If it is all zeros, the contents of R1 are used as a mask. If R1 also contains all zeros, the instruction does no operation.

No indicators are affected. The address syllable can specify a memory location, an immediate operand, or another R-register.

Example: Contents of:

Register	Mask	Effective Address (before)	Effective Address (after)
$(ABCD)_{16}$	$(00FF)_{16}$	$(1234)_{16}$	$(12CD)_{16}$
$(ABCD)_{16}$	$(1111)_{16}$	$(1234)_{16}$	$(0325)_{16}$

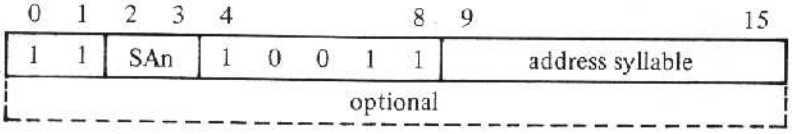
SSB

Instruction:
Scientific subtract

Restriction:
Traps (TV#3) on models without Scientific Instruction Processor

Type:
Double Operand

Format:



Description:
Subtracts the contents of the location, scientific accumulator, or R register identified in the second operand from the contents of the scientific accumulator specified in the first operand. The result is saved in the scientific accumulator.

If the second operand is an R register, the integer value contained in the specific R register is internally converted to floating-point format before it is subtracted from the SA register specified by the first operand.

Scientific indicator settings include the following:

EU – Set to 1 on exponent underflow; otherwise, set to 0.

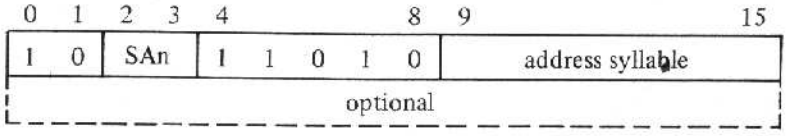
SST

Instruction:
Scientific store

Restriction:
Traps (TV#3) on models without Scientific Instruction Processor

Type:
Double Operand

Format:



Description:

Stores the contents of the scientific accumulator identified in the first operand in the location, scientific accumulator, or R register specified in the address syllable.

If the second operand is an R register, the floating-point value contained in the specific scientific accumulator is converted to integer format before it is stored into the specified R register.

Scientific indicator settings include the following:

EU – Set to 1 on exponent underflow; otherwise, set to 0.

SE – If the converted integer is too large to fit in a 16-bit (=$R4, =SR5, =$R6) or 32-bit (=SR7) field, the SE bit is set to one; if the converted integer fits, the SE bit is cleared to zero. If an R register is not specified (location =SSA1, =SSA2, or =SSA3), however, the SE bit is unchanged.

PE – Changed only when the floating-to-integer conversion takes place. If the floating-point value contains a non-zero fraction, PE is set to a one, otherwise, it is cleared to zero. If the SEM bit of M5 is zero (no SE trap or SE cleared), the preceding description of PE is correct; however, if SEM is set to one, and SE is set by this instruction, the PE is redefined to mean that if the fractional portion of the floating-point value is $\geq 1/2$ and M4 (round) = 1, then PE is set, otherwise, PE is cleared.

SEM affects the result stored in R4, R5, R6, or R7 if SE is set by this execution:

- o If SEM is zero the largest magnitude integer (i.e., 7FFF, 7FFFFFFF, 8000, or 80000000) of the same sign as the floating-point value is stored and no trap occurs.
- o If SEM is one, the floating-to-integer conversion is recomputed with the temporary assumption that truncation instead of round applies. The low-order 15 or 31 bits of this result plus the proper sign are then stored.

SSW

Instruction:

Scientific swap

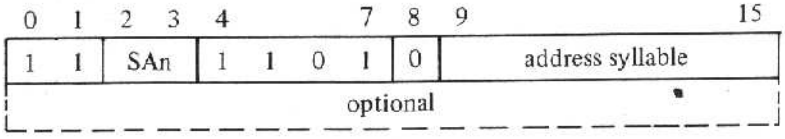
Restriction:

Traps (TV#3) on models without Scientific Instruction Processor

Type:

Double Operand

Format:



Description:

Swaps the contents of the scientific accumulator identified in the first operand with the contents of the location, scientific accumulator, or R register specified in the address syllable.

If an R register is specified as the second operand, the value specified by the first operand is internally converted to integer format, and the value specified by the second operand is internally converted to floating-point. These converted values are then interchanged.

If SEM is zero and the floating-point value converted to integer format cannot fit in the specified R register(s), then the largest magnitude of the proper sign is swapped instead (one of 7FFF, 7FFFFFFF, 8000, or 80000000).

If SEM is one, and the floating-point value converted to integer format cannot fit in the specified R register(s), then the instruction is cancelled (no changes in either SAn or R) and an SE trap (TV#21) occurs.

Scientific indicator settings include the following:

EU – Set to 1 on exponent underflow; otherwise set to 0.

SE – Set to 1 if result of floating-to-integer conversion is too large to fit into integer register; cleared to 0 if it fits the specified register; otherwise, unchanged if no floating-to-integer conversion.

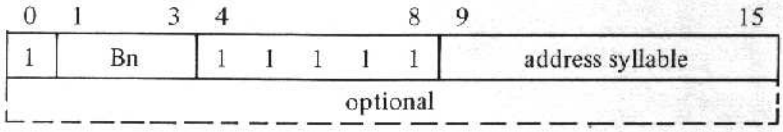
PE – Modified only if floating-to-integer conversion performed; set to 1 if floating-point value contains non-zero fraction; otherwise, cleared to 0.

STB

Instruction:
Store B register

Type:
Double Operand

Format:



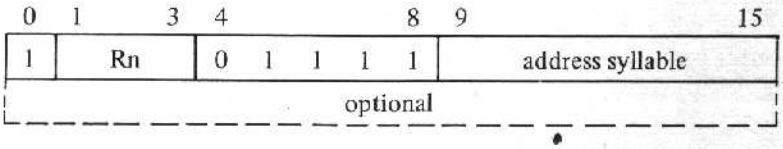
Description:
Stores the address contained in the designated B register into the location specified by the effective address. No indicators are affected. The address syllable can specify a memory location, an immediate operand, or another B register.

STH

Instruction:
Store R register half-word (byte)

Type:
Double Operand

Format:



Description:
Stores the right-hand byte of the designated R register into the half-word specified by the effective address. The other half is not changed. No indicators are affected.

The effective address can specify a memory location, an immediate operand, or another R register, with the byte being positioned as follows:

– memory location, not indexed	– left byte
– memory location, indexed	
index value even	– left byte
index value odd	– right byte
– immediate operand	– left byte
– R register	– right byte

Example: Assume that:

– register B1	contains 1000_{16}
– register R1	contains 0
– register R2	contains 1
– register R5	contains $(ABCD)_{16}$
– location 1000	contains $(1234)_{16}$

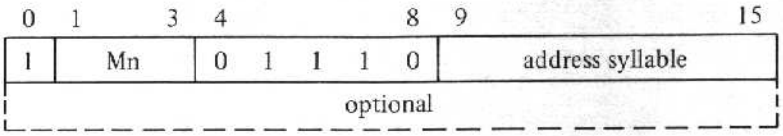
then if STH $R5, $B1.SR1 is executed: location 1000 will contain $(CD34)_{16}$
but if STH $R5, $B1.SR2 is executed: location 1000 will contain $(12CD)_{16}$

STM

Instruction:
Store M register

Type:
Double Operand

Format:



Description:
Stores the 8-bit M register identified in the first operand in the right half-word of the location specified in the address syllable; the left half-word of the location is filled with 1s.

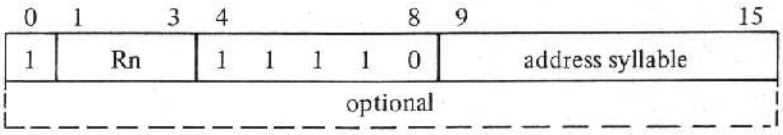
The address syllable can specify a memory location, an immediate operand, or one of the R registers. No indicators are affected. On Models 23 and 33, the register number must be 1; otherwise, traps to trap vector #5.

STR

Instruction:
Store R register

Type:
Double Operand

Format:



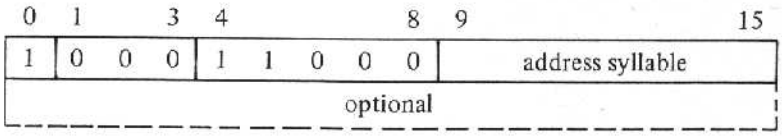
Description:
Stores the word contained in the designated R register into the location specified by the effective address. No indicators are affected. The address syllable can specify a memory location, an immediate operand, or another R register.

STS

Instruction:
Store S register

Type:
Single Operand

Format:



Description:
Stores the contents of the status register in the location specified by the effective address. No indicators are affected. The address syllable can specify a memory location, an immediate operand, or an R register.

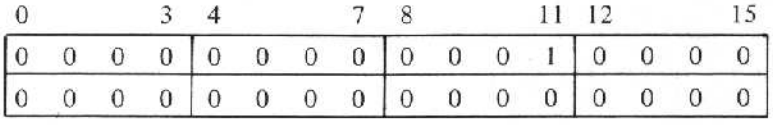
STT

Instruction:
Store stack register

Restriction:
Traps (TV#5) on Models 23 and 33

Type:
Generic

Format:



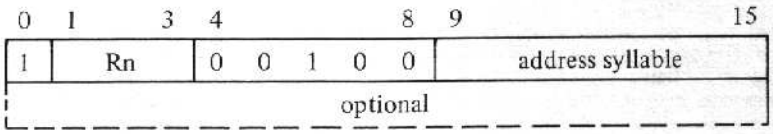
Description:
Stores the contents of the Stack Address Register (T) into B7. The contents of the I register are unaffected.

SUB

Instruction:
Subtract from R register

Type:
Double Operand

Format:



Description:
Subtracts the word at the effective address from the word contained in the designated R register. The carry (C) and overflow (OV) indicators are set if carry and/or overflow occurs respectively; otherwise, they are cleared to zero. The address syllable can specify a memory location, an immediate operand, or another R register.

Examples:

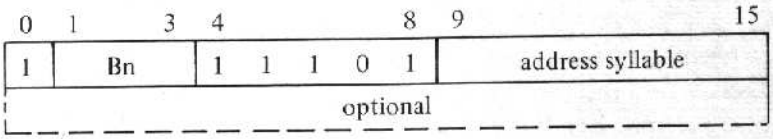
R Register (before)	Effective Address (before and after)	R Register (after)	C (after)	OV (after)
-32767	+1	-32768	0	0
-32768	+1	+32767	1	1
-1	-1	0	1	0

SWB

Instruction:
Swap B register

Type:
Double Operand

Format:



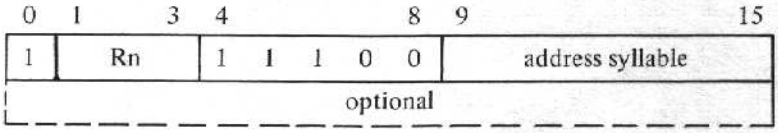
Description:
Swaps the address contained in the designated B register with the address at the effective address. No indicators are affected. The address syllable can specify a memory location, an immediate operand, or another B register.

SWR

Instruction:
Swap R register

Type:
Double Operand

Format:



Description:
Swaps the word contained in the designated R register with the word at the effective address. No indicators are affected. The address syllable can specify a memory location, an immediate operand, or another R register.

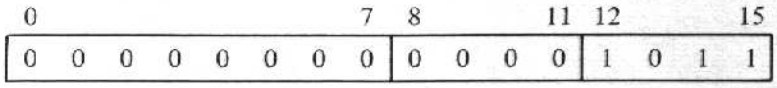
VLD

Instruction:
Validate address, range, and access rights

Restriction:
Traps (TV#5) on models without the Memory Management Unit option.

Type:
Generic

Format:



Description:
Determines whether or not certain data accesses are valid without actually attempting them. It accepts a virtual address, interpreted as the start of an area to be verified; a size, in bytes; and an effective ring number to be used in the validation. These items are loaded into base register B5, general register R3 and general register R5, respectively, before execution. The instruction returns a condition code in R3 as follows:

-1 – Invalid address. Segment does not exist or is not large enough for the data area specified.
0 – Read access permitted in given ring, write access forbidden.
+2 – Read access and write access both permitted in given ring.
-2 – Read access and write access both forbidden in given ring.

This instruction is useful, for example, in a system subroutine that executes at a high level of privilege and has access to critical data. When called on behalf of a user, the system subroutine can validate that user's right to access the data in question by using the VLD instruction.

The contents of the I register are unaffected.

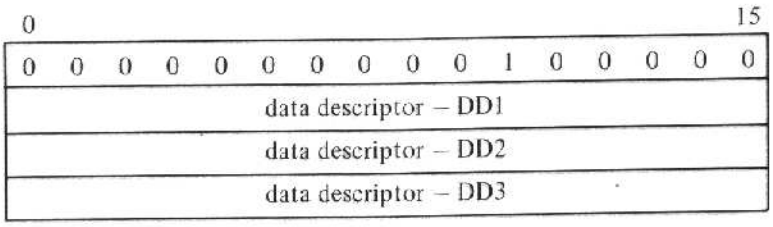
VRF

Instruction:
Alphanumeric verify

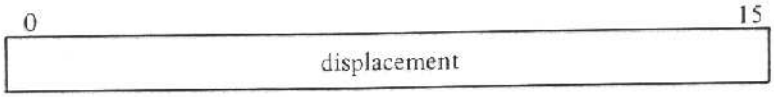
Restriction:
Traps (TV#5) on Models 23, 33, 43, and 53

Type:
Alphanumeric

Format:



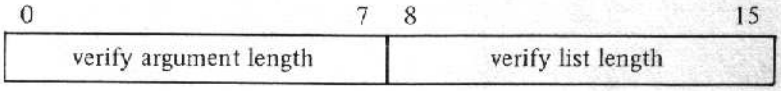
Description:
The character string or array of character strings defined by the third data descriptor (DD3) is examined. If at least one character of the string (or element of the array) does not match any one of the verify arguments, the G-bit of the CIP indicator register is cleared to zero, the L-bit is set to one, and the receiving field specified by the second data descriptor (DD2) is loaded with the displacement. The displacement is the distance in bytes between the origin of the string (or array) and the place where the first mismatch is found. The format of the receiving field is shown below.



If each of the characters of the string (or elements of the array) is equal to any one of the verify arguments, the G- and L-bits of the CIP indicator register are cleared to zero and the receiving field is not changed.

If the length field of DD1 is not equal to zero, the verify argument list contains only one search argument whose length (1 through 31 bytes) is specified by the field.

If the length field of DD1 is equal to zero, the verify argument list is specified by register R4. The format of register R4 is shown below.

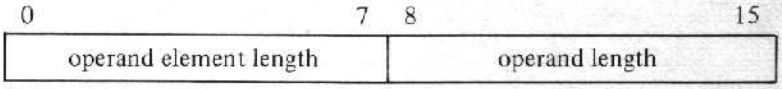


If the verify argument length is equal to the verify list length, the verify list consists of only one argument.

If the ratio of the verify list length to the verify argument length is an integer, that integer designates the number of verify arguments.

If the ratio of the verify list length to the verify argument length is not an integer, the ratio is truncated to the integer value and that integer designates the number of verify arguments.

The character string (or array) to be verified is specified by DD3. If the length field of DD3 is not equal to zero, the operand is a character string whose length (1 through 31) is specified by the length field. If the length field is equal to zero, the operand to be searched is specified by register R6. The format of register R6 is shown below.



If the operand element length is one, the operand is a character string whose length (specified by the low-order byte of R6) must be from 1 through 255. If the operand is a character string, the length of the verify argument must be one, otherwise, the results are unspecified.

If the operand element length is not equal to one, it specifies the length of each element of an array. The length of the array is the largest multiple of the operand element length that is less than or equal to the operand length (specified by the low-order byte of R6). In this case, if the verify argument length is greater than the operand element length, each verify will overflow into the next entry except when the last operand is verified. Thus the last comparison will result in a mismatch.

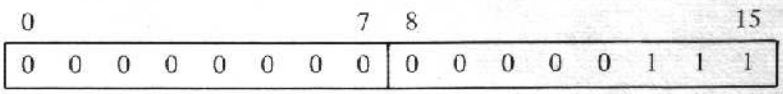
WDTF

Instruction:
Watchdog timer off

Restriction:
Traps (TV#5) on Model 33 without Watchdog Timer option.

Type:
Generic

Format:



Description:
Disables the watchdog timer. The P field in the S register must equal 1x (i.e., the central processor must be in the privileged state) for this instruction to be executed. If not, the unprivileged use of a privileged operation is signified by a trap to trap vector #13.

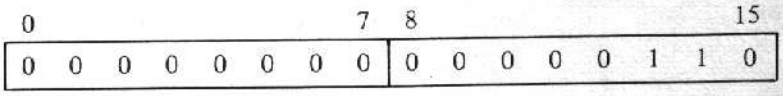
WDTN

Instruction:
Watchdog timer on

Restriction:
Traps (TV#5) on Model 33 without Watchdog Timer option

Type:
Generic

Format:



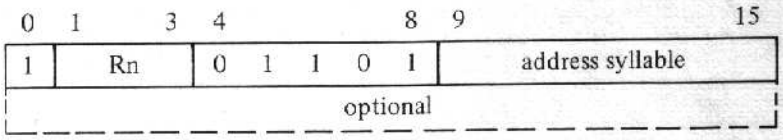
Description:
Enables watchdog timer. The P field in the S register must equal 1x (i.e., the central processor must be in the privileged state) for this instruction to be executed. If not, the unprivileged use of a privileged operation is signified by a trap to trap vector #13.

XOH

Instruction:
Half-word (byte) exclusive OR with R register

Type:
Double Operand

Format:



Description:
Logically "exclusive ORs" the byte at the effective address (with sign extended) to the word contained in the designated R register. No indicators are affected. The address syllable can specify a memory address, an immediate operand, or another R register, with the byte being positioned as follows:

- memory location, not indexed — left byte
- memory location, indexed
  - index value even — left byte
  - index value odd — right byte
- immediate operand — left byte
- R register — right byte

The following chart illustrates the result of exclusively ORing bits:

First operand bit	0	0	1	1
Second operand bit	1	0	1	0
Result	1	0	0	1

Examples:

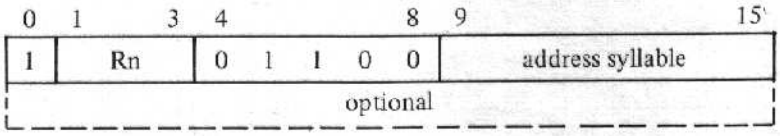
R Register (before)	Effective Address	R Register (after)
$(0000)_{16}$	$(07)_{16}$	$(0007)_{16}$
$(0007)_{16}$	$(07)_{16}$	$(0000)_{16}$
$(FFFF)_{16}$	$(88)_{16}$	$(0077)_{16}$

XOR

Instruction:
Exclusive OR with R register

Type:
Double Operand

Format:



Description:
Logically "exclusive ORs" the word at the effective address to the word contained in the designated R register. No indicators are affected. The address syllable can specify a memory address, an immediate operand, or another R register.

The following chart illustrates the result of exclusively ORing bits:

First operand bit	0	0	1	1
Second operand bit	1	0	1	0
Result	1	0	0	1

Examples:

R Register (before)	Effective Address	R Register (after)
$(0000)_{16}$	$(0007)_{16}$	$(0007)_{16}$
$(0007)_{16}$	$(0007)_{16}$	$(0000)_{16}$
$(FFFF)_{16}$	$(8888)_{16}$	$(7777)_{16}$