INTEGER ARITHMETIC

Instructions and facilities are included in the integer arithmetic class of operations which allow elementary arithmetic to be performed on single length and multiple length operands.

The integer arithmetic instructions have the following format:

op	i	j	k

where j and k designate the arithmetic registers containing the source operands and i specifies the result register(s).

The instructions are divided into three groups:

1. The single length integer arithmetic operations (add, subtract, multiply and divide) operate on 48-bit operands and yield a 48-bit result. In each operation the specified function is performed between two arithmetic registers A^j and A^k. The result replaces the 48-bit contents of arithmetic register A^i. The contents of A^j and A^k are not changed.

2. Mixed length integer arithmetic operations (multiply and divide) have a 96-bit operands or 96-bit result. The 96-bit operand occupies an even-odd pair of 48-bit arithmetic registers.

3. The "continued" integer arithmetic operations (high order continued add, high order continued subtract, low order continued add, and low order continued subtract) use an implied third operand. In each operation the specified function is performed between the single precision integer contents of arithmetic registers A^j and A^k and the 3-bit contents of Multiprecision Carry Register (MPC). The result replaces the contents of A^i and MPC. Registers A^j and A^k are not changed.

Register A^0 is specified to be a source of 0's. When A^0 is specified as a source operand, 48 or 96 0's will be provided depending on whether a single or mixed length instruction is used. If A^0 is specified as the result register, the result will be lost, and the only effect of the operation will be a possible change in the Multiprecision Carry Register or the exception register.

Number Representation

Integer operands are represented in two's-complement form. The formats are as follows:

Single Precision Integer

weight	-2^{47}	2^{46}		2^{47-m}		2^{1}	2^{0}
operand bit position	0	1		m		46	47

for $1 \leq m \leq 47$

Double Precision Integer

weight	-2^{94}	2^{93}		2^{94-m}		2^{48}	2^{47}	-2^{47}	2^{46}		2^{95-n}		2^{1}	2^{0}
operand bit position	0	1		m		46	47	48	49		n		94	95

for $1 \leq m \leq 47$
$49 \leq n \leq 95$

Multiple Precision Integer

Multi-length operands are a multiple of 48 bits in length and have the format:

r	r-1	…	2	1	n
0 47	0 47		0 47	0 47	m

$$I = \sum_{m,n} b_m^n \, w_{m,n}$$

where I is the value of the integer operand of multiplicity r; b_m^n is the binary value of the m^{th} bit of the n^{th} 48-bit register; and $w_{m,n}$ is the weighting factor of bit b_m^n as follows:

for $m = 0$ $\quad w_{0,n} = -2^{47n}$

for $m \neq 0$ $\quad w_{m,n} = 2^{47n-m}$

Note that the single and double precision integer formats can be considered special cases of the above format with r equal to 1 and 2, respectively.

Standard Form

In the multi-length format the magnitude of the weights of bit 0 of register n and bit 47 of register n+1 are the same, but the signs are different. Thus the combination 00 has the same value as the combination 11. A "standard form" is defined to circumvent this non-unique format. A number is defined to be in standard form if bit zero of every register, except the high order register, is equal to 0. Except for a few numbers near the negative limit of the representable range, all numbers have a standard form. For example, the number which has a 1 in bit zero of every word and 0's elsewhere has no standard form.

Multiprecision addition and multiplication can be performed on operands without requiring the operands to be in standard form. The operation will yield a result in standard form. The subtrahend in multiprecision subtraction must be in standard form. The requirement for multiprecision dividends and divisors to be in standard form is dependent on the programming algorithm to be used. The 96-bit product formed in mixed length multiply is in standard form and mixed length divide requires the 96-bit dividend to be in standard form.

Overflow

The integers which can be represented in 48 bits in two's-complement form range from -2^{47} to $2^{47}-1$. Wherever a single length add, subtract, or multiply; a mixed length multiply or divide; a high order continued add or subtract; or an arithmetic shift results in a number which cannot be represented in the satisfactory range of -2^{47} to $2^{47}-1$, an integer overflow condition exists and the appropriate exception bit is set to 1.

Add Integer AI

	i	j	k

The contents of register A^i are replaced by the low order 48 bits of the sum formed by the addition of the single precision integers in A^j and A^k.

Exception	Exception bit
result $> 2^{47}-1$	AO
result $< -2^{47}$	AO

Subtract Integer SI

	i	j	k

The contents of register A^i are replaced by the low order 48 bits of the difference formed by the subtraction of the single precision integer in A^k from the single precision integer in A^j.

Exception	Exception bit
result $> 2^{47}-1$	AO
result $< -2^{47}$	AO

Multiply Integer MI

	i	j	k

The contents of A^i are replaced by the low order 48 bits of the product of the single precision integers in A^j and A^k.

Exception	Exception bit
result $> 2^{47}-1$	MO
result $< -2^{47}$	MO

Multiply Integer, Mixed Length MMI

	i	j	k

The contents of register pair $A^{i,i+1}$ are replaced by the product in standard form of the single precision integers in A^j and A^k.

In the special case where A^j and A^k are both -2^{47} the product cannot be represented by a double precision integer; then $A^{i,i+1}$ are set to 0's and the MO exception bit is set to 1.

Exception	Exception bit
result $> 2^{94}-1$	MO
i odd	RS

Integer Divide Instructions

The integer divide instructions are performed as follows:

1. If the divisor is zero, the result is set to zero, the divide overflow exception bit (DO) is set to 1, and the remaining steps are omitted.
2. If the divide is a DMI and the dividend is not in standard form, the ILO exception bit is set to 1, and the operation proceeds as if the dividend was in standard form.
3. The dividend is divided by the divisor to form an exact quotient.
4. If the exact quotient is an integer, it forms the intermediate result.
5. If the exact quotient was not an integer, the integer part of the exact quotient forms the intermediate result. That is, the exact quotient is rounded toward zero.
6. If the intermediate result cannot be represented in 48 bits, the divide overflow exception bit is set to 1.
7. The result is set to the low order 48 bits of the intermediate result.

Divide Integer DI

	i	j	k

The contents of A^i are replaced by the single precision integer quotient formed by dividing the single precision integer dividend in A^j by the single precision integer divisor in A^k.

Exceptions	Exception bit
$A^k = 0$	DO
result $> 2^{47}-1$	DO

Divide Integer, Mixed Length DMI

	i	j	k

The contents of register A^i are replaced by the single precision integer quotient formed by dividing the double precision integer dividend in $A^{j,j+1}$ by the single precision integer divisor in A^k.

Exceptions	Exception bit
$A^k = 0$	DO
result $> 2^{47}-1$	DO
result $< -2^{47}$	DO
$A^{j+1}_0 = 1$	ILO
j odd	RS

Continued Add, Low Order ACL

	i	j	k

The contents of $A^i_{1,\ldots,47}$ are replaced by the low order 47 bits of the sum of the integers in A^j and A^k and the contents of the Multiprecision Carry Register MPC. A^i_0 is set to 0. The MPC is set so that:

$$2^{47} \times MPC_{new} + A^i = A^j + A^k + MPC_{old}$$

Exception	Exception bit
$MPC_{old} \neq -3,-2,-1,0,$ or $+1$	ILO

Continued Subtract, Low Order SCL

	i	j	k

The contents of $A^i_{1,\ldots,47}$ are replaced by the low order 47 bits formed by subtracting the integer in A^k from the integer in A^j and adding the contents of MPC to the difference. A^i_0 is set to 0. The MPC is set so that:

$$2^{47} \times MPC_{new} + A^i = A^j - A^k + MPC_{old}$$

Exception	Exception bit
$MPC_{old} \neq -3,-2,-1,$ or 0	ILO

<u>Continued Add, High Order</u> ACH

	i	j	k

The contents of A^i are replaced by the low order 48 bits of the sum formed by adding the integers in A^j and A^k and the contents of MPC. The MPC is then set to zero.

Except for the participation of MPC, this instruction is identical to AL.

Exceptions	Exception bit
result $< -2^{47}$	AO
result $> 2^{47}-1$	AO
$MPC_{old} \neq -3,-2,-1,0,$ or $+1$	ILO

<u>Continued Subtract, High Order</u> SCH

	i	j	k

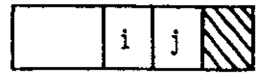
The contents of A^i are replaced by the low order 48 bits of the difference formed by subtracting the integer in A^k from the integer in A^j and adding the contents of MPC to the result. The MPC is then set to zero.

Except for the participation of MPC, this instruction is identical to SL.

Exceptions	Exception bit
result $< -2^{47}$	AO
result $> 2^{47}-1$	AO
$MPC_{old} \neq -3,-2,-1,$ or 0	ILO

Set Positive, Integer

SPI



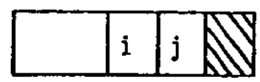
The contents of register A^i are replaced by the absolute value of the integer represented by the contents A^j.

If the integer in A^j is -2^{47}, an overflow exception condition exists. The integer add overflow exception bit AO is set to 1, and the contents of A^i are replaced by the value zero.

Exception	Exception bit
$A^j = -2^{47}$	AO

Set Negative, Integer

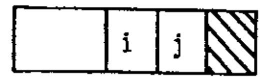
SNI



The contents of register A^i are replaced by the negative of the absolute value of the integer represented by the contents of A^j.

Exceptions: none

Convert to Normalized CVN

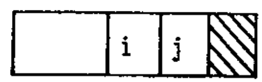


The single precision integer in A^j is converted to a normalized single precision floating point number; the floating point number replaces the contents of A^i.

If the conversion results in an intermediate fraction length greater than 36 bits, the fraction is truncated to 36 bits.

Exceptions: none

Convert to Integer CVI



The single precision floating point number in A^j is converted to a single precision integer which replaces the contents of A^i.

If the absolute value of the number in A^j is greater than $2^{48}-1$ or if it is u, the conversion will not be done correctly. In this case the AO exception bit is set to 1 and no meaning should be given to the result.

Exception	Exception bit
$\lvert A^j \rvert > 2^{48}-1$	AO
$A^j = u$	AO