Bipolar Integrated Technology, Inc.

B2130/B3130/B4130

Advance Information

Single Chip Floating Point Processors

Features

- Single chip solution for support of ANSI/IEEE STD. 754 single and double precision, and DEC® (F&G) formats
- 10 ns (worst case) Pipeline mode for maximum throughput
  200 MFLOPS peak (ECL)
  100 MFLOPS double precision multiply data rate (ECL)
  100 MFLOPS double precision ALU data rate (ECL)
  100 MIPS integer data rate (ECL)
- Six data formats
  64-bit floating point (DEC and IEEE)
  32-bit floating point (DEC and IEEE)
  64-bit integer (fixed point)
  32-bit integer (fixed point)
- Instruction set compatible with B2110A/B2120A (TTL) and B3110A/B3120A (ECL)
  Floating point instructions include:
  Multiply, divide, square root, add, subtract, absolute value, negate, min/max, compare
  Integer instructions include:
  Multiply, divide, add, subtract, boolean functions, absolute value and shift
  Conversion operations to/from all supported formats
- Three port, 64-bit I/O architecture
  Byte-Parity (odd or even) generation and checking
  Scan path through all registers
- Internal paths feed back any result to any operand
- Supports fully concurrent operations between ALU, MPY, and DIV/SQRT blocks
- Output register configurable for flowthrough operation
- Synchronous and asynchronous output enables for status flags and output port
- 395-pin Pin-grid-array package

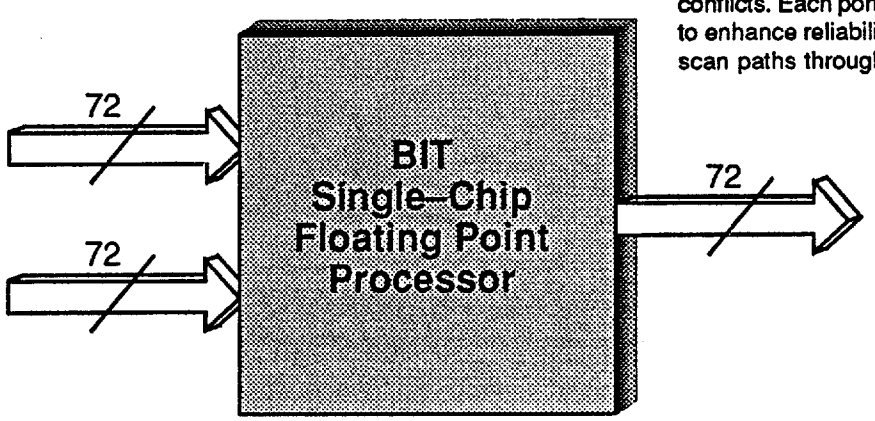
Description

The B2130, B3130 and B4130 comprise BIT's family of high performance single-chip floating point processors: The B2130 has a TTL compatible I/O, the B3130 a 10KH ECL compatible I/O, and the B4130 a 100K ECL compatible I/O. These floating point processors deliver ultra-high performance floating point and integer operations. Fabricated with BIT's high performance VLSI process, the Bx130 family provides single-chip solutions for high-end technical applications. All floating point operations can be either single or double precision and are compatible with the IEEE standard 754 or DEC F and G formats. The floating point instruction set includes add, subtract, multiply, divide, square-root, conversion operations, minimum, maximum, absolute value and compare. All four IEEE rounding modes are supported. The ALU and MPY elements which perform the computations can be configured as two-stage pipelines or flowthrough combinatorial blocks. Pipelined, the device can achieve up to 200 MFLOPS using internal feedback paths and concurrent operation of the elements. Flowthrough, the architecture provides 20ns latency for ALU and MPY operations, resulting in up to 100 MFLOPS for random scalar operations.

The Bx130 also supports a large repertoire of 32- and 64-bit integer functions. The ALU provides integer operations at the same performance level as it does floating point functions. Integer ALU functions include add and subtract (with and without carry/borrow), negate, absolute value, all 16 boolean functions, and shift. Shifts use the Y operand to define shift distance for the X operand. The MPY and DIV/SQRT blocks provide integer multiply and divide operations, respectively.

The three port, 64-bit (with 8-bit parity) architecture of the Bx130 provides maximum throughput. Input operands are clocked into edge-triggered registers, and the output result can either be registered, or configured to flowthrough to the output port. Individual clock enables are provided for both input and output registers. A 14-bit Flag output bus presents selected ALU and MPY/DIV/SQRT register flag sets.

Output enables can be either synchronous, asynchronous or both. The synchronous output enable option helps reduce bus conflicts. Each port uses byte-parity (odd or even) to enhance reliability. Built in test features include scan paths through all registers.



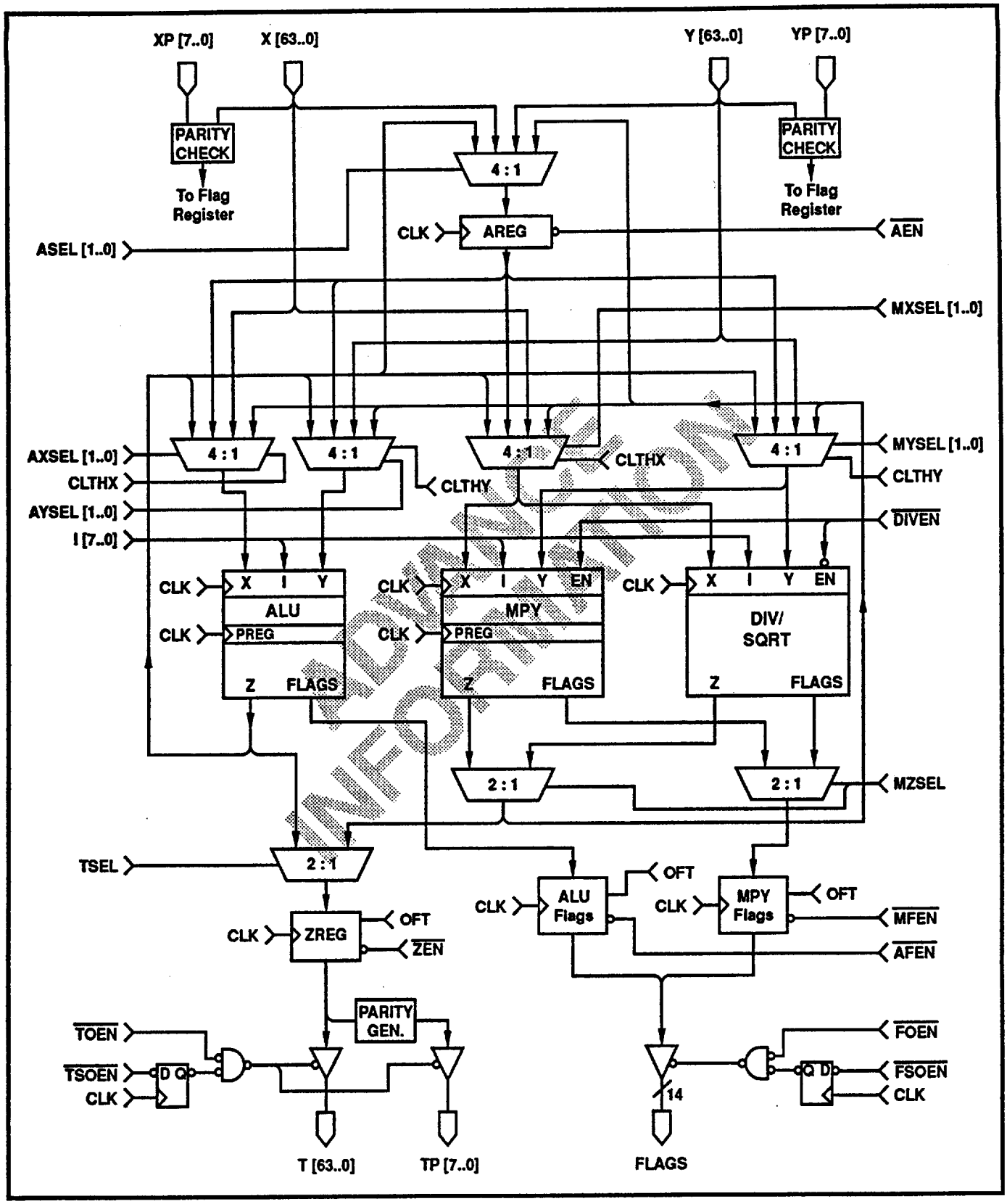


Figure 1 — B2130 / B3130 / B4130 Block Diagram

Single Chip
Floating Point Processors

B2130/B3130/B4130

T-49-12-05

FUNCTIONAL DESCRIPTION

The Bx130 floating point processor is a single chip solution for high-end technical computing. It supports both 32- and 64-bit operands and performs both floating point and integer computations. Functional blocks of the Bx130 include a floating point and integer ALU, a floating point and integer multiplier (MPY), a divide/square root element (DIV/SQRT), internal data path multiplexers, a temporary storage register, and internal status and control registers. The Bx130 provides two 64-bit input ports and one 64-bit output port for maximum data throughput capability. Registers are provided for input operands for each computing element.

COMPUTE ELEMENTS

The ALU is a fast, combinatorial circuit which performs addition, subtraction, conversion, shift, compare, and other operations. It can be configured as a two stage pipeline for maximum throughput. With the pipeline register disabled, the ALU produces results one operation delay after operands and instruction are clocked into the ALU input registers. Operand and opcode registers are individually controlled. Flags from the ALU are clocked into the ALU Flag register on the rising edge of CLK.

Similarly, the MPY element is a combinatorial circuit with comparable performance to the ALU. It's pipeline register, operand and opcode registers are all separately enabled from each other as well as from the ALU. A common hardware pin (PLFT) configures both the ALU and the MPY as either pipelined or flowthrough. The MPY performs all floating point multiplications and integer multiplications for 8-, 16- and 32-bit integers.

The DIV/SQRT block performs floating point and integer division, floating point square root operations and 64-bit integer multiplications. It is an internally clocked, synchronous state machine which executes IEEE compatible division and square root instructions. Input opcodes and operands are controlled using the MPY clock enables, logically AND'd with a separate divide enable signal (DIVEN*). Thus the DIV/SQRT can operate concurrently with the MPY. Results from the DIV/SQRT element are multiplexed with the results from the MPY element; one result from either the MPY or DIV/SQRT can be unloaded every cycle. Additionally, flags from the DIV/SQRT are multiplexed with the flags from the MPY prior to being clocked into the MPY flag register. Both multiplexers are controlled with a common select signal (MZSEL).

CLOCK ENABLES

Input operand and opcode register enables, the divide enable, and the pipeline register enables are sampled on the rising edge of CLK. In addition, the result register Z enable and the temporary register A enable are also sampled on the rising edge of CLK. When the pipeline registers and the Z register are in flowthrough mode, their associated enables are ignored.

OUTPUT ENABLES

Support of common external buses is provided with output enables for the flags and result. In addition, a synchronous version of these enable signals is included for better control of buffer collisions in TTL systems.

FEEDBACK PATHS

An internal feedback path is included from the output of the ALU to a set of four-input multiplexers which select input operands for the compute elements. This arrangement allows the ALU result to be used as either the X or Y input for any element. Similarly, the multiplexed result from either MPY or DIV/SQRT is fed back and can also be used by any input. The MPY and DIV/SQRT share X and Y multiplexers for their inputs.

The multiplexers are selected using a set of eight control signals, two per multiplexer. The ALU has a multiplexer for each X and Y input. The MPY and DIV/SQRT share their multiplexer pair. Each multiplexer selects one of four sources: ALU result, MPY/DIV/SQRT result, A-register content, or X/Y inputs, as appropriate. A-register contents are common to all multiplexers, while X input multiplexers select the X input port and Y input multiplexers select the Y input port.

If the result of a single precision instruction is fed back to be used as a double precision operand, the least significant 32 bits are undefined. If a double precision result is fed back and used as a single precision operand, only the most significant 32 bits are used.

CONCURRENT OPERATION

Feedback paths, independent registers for operands and opcode, a multiplexer for MPY and DIV/SQRT results, and the A-register allow a large degree of on chip concurrency. Multicycle divide or square root operations may be started by registering the operands, and multiplier functions can then be performed while waiting for the DIV/SQRT block to finish. The A-register can be used to hold intermediate results at the end of a multiply-add/subtract operation, as a scalar operand for the vector operation AX+Y, or for any other temporary value.

To support concurrent ALU and MPY floating point operations, the flag registers can be set to operate concurrently (both ALU and MPY flags updated each clock cycle) or independently. In concurrent mode (EC=1) the precision of the MPY function will follow the precision of the ALU. Additionally, independent register enables allow the inhibiting of flag register updates when required.

PARITY SUPPORT

Faults on the input data buses can be detected using the parity checking feature of the Bx130 and byte parity is generated for data outputs. Both odd and even byte (8-bit) parity is supported. A hardware pin indicates a parity error has occured and 5 bits in the ALU flag register indicate the port and byte location of the error.

Input parity is checked only on operands which are clocked into one or more of the operand registers; either the ALU, MPY, or DIV/SQRT, X or Y registers, or the A register. When parity is checked, all 8 bytes of all enabled registers are checked for correct parity, whether the registers are used or not used.

After a reset, the Bx130 defaults to no parity check. If parity checking is desired, the mode register must be written to enable parity check (EP) and to define the type of parity (odd or even) to be checked and generated (PM).

SCAN PATH

All on-chip registers can be accessed with a serial scan path. Operating in a first-in, first-out, serial shift fashion, scan path allows diagnostics to be performed with just four signals: scan mode, scan clock, scan data in, and scan data out.

SINGLE PRECISION OPERATION

The functional units of the Bx130 (the ALU, MPY, and DIV/SQRT) expect single precision operands (FLOATING POINT AND INTEGER) to be input on the high 32 bits of the input operand buses [63..32]. The control signals CLTHX and CLTHY (Copy Low To High, X/Y), control multiplexers located ahead of these inputs, and allow the low 32 bits of each operand to be copied to the high 32 bits. If a single precision operand is supplied to the Bx130 on X[31..0], then setting CLTHX=1 will duplicate this value onto bits [63..32] of the internal operand bus, where it can be used by one of the functional units. CLTHY performs the same function on the Y Port. Note that CLTHX (and Y) must be set to 0 while clocking in a double precision operand. The incoming instruction will NOT override this select signal.

Single precision results are output to both the high and low word of the T Port (T[63..32] and T[31..0]).

INTEGER DIVISION

Integer division and remainder operations are available in 8-, 16-, 32- and 64-bit operand widths, using signed (2's complement) integers only. The division operations returning quotients are:

8-bit = 8-bit / 8-bit (opcode 89H, BDIV),
16-bit = 16-bit / 16-bit (opcode 8DH, HDIV),
32-bit = 32-bit / 32-bit (opcode 83H, IDIV),
32-bit = 64-bit / 32-bit (opcode 84H, MIDIV), and
64-bit = 64-bit / 64-bit (opcode C3H, LIDIV).

The remainder operations are similar:

8-bit = 8-bit % 8-bit (opcode 8BH, BREM),
16-bit = 16-bit % 16-bit (opcode 8FH, HREM),
32-bit = 32-bit % 32-bit (opcode 87H, IREM),
32-bit = 64-bit % 32-bit (opcode 88H, MIREM), and
64-bit = 64-bit % 64-bit (opcode C7H, LIREM).

Performing an integer division or remainder operation, where the divisor (or modulus) is zero, returns zero and raises the divide by zero flag (DIVZ). An integer division that overflows will raise the overflow flag (OV) and return the least significant bits of the quotient. Integer remainder operations never overflow.

The sign of the remainder will be the same as the sign of the dividend, except when the remainder is zero. Thus either:

$$\text{dividend} \geq 0 \text{ AND } |\text{divisor}| > \text{remainder} \geq \text{zero}$$

or:

$$\text{dividend} < 0 \text{ AND zero} \geq \text{remainder} > -|\text{divisor}|$$

Furthermore, the quotient and remainder for the same pair of operands (dividend and divisor) will satisfy:

$$\text{dividend} = \text{quotient} \cdot \text{divisor} + \text{remainder}$$

WRAPPED NUMBERS

In their default operating mode, BIT floating point chips treat denormalized operands as zero and do not produce denormalized results. To process denormalized numbers, the wrapped underflow mode bit must be set. When set, the ALU will directly operate upon denormalized operands, and all of the functional units (ALU, MPY, and DIV/SQRT) will return "wrapped underflows" which can be converted to denormalized results.

If a normalized number and a denormalized number are multiplied, the denormalized and inexact flags will be raised and the value zero returned. The denormalized number can then be converted to a wrapped number by passing it through the ALU (opcode 2CH or 2DH). The ALU will raise its underflow flag and return a wrapped equivalent of the denormalized number. This wrapped number can be multiplied by the normalized operand by using one of the wrapped multiply instructions available on the MPY (opcodes 04H to 07H). If the result is too small to be normalized, the underflow flag will be raised and a wrapped underflow will be returned. If the underflow flag is not raised, then the result is a normalized number.

The ALU can be used to convert the wrapped underflow to a denormalized result and provide the correct IEEE underflow flag for the multiplication. In order to avoid multiple rounding errors, the inexact and rounded up flags from the multiplication must be passed to the ALU for this wrapped to denormalized number conversion.

No instruction for multiplying two wrapped numbers is provided because the product of two denormalized operands always underflows.

Wrapped numbers are a simple extension of IEEE normalized numbers. The value of an IEEE normalized number can be determined by:

$$X = (-1)^s \cdot 2^{e\text{-bias}} \cdot (1.f)$$

where "s", "e", and "f" are the sign, exponent, and fraction of the number, and the bias is either 127 (single precision) or 1023 (double precision). The value of a wrapped number is determined by the same formula as normalized numbers. A wrapped underflow with an exponent of zero (e=0) uses the same bias value as normalized IEEE numbers. A wrapped underflow with a non-zero exponent value (e > 0) uses a bias of either 383 (single precision) or 3071 (double precision). Wrapped overflows with an exponent of all ones (single precision, e=255 or double precision, e=2047) use the same bias value as normalized IEEE numbers, while other wrapped overflows have a bias of either −129 (single precision) or −1025 (double precision). Mapping the unbiased exponent to the biased exponent for single precision is shown in the following table:

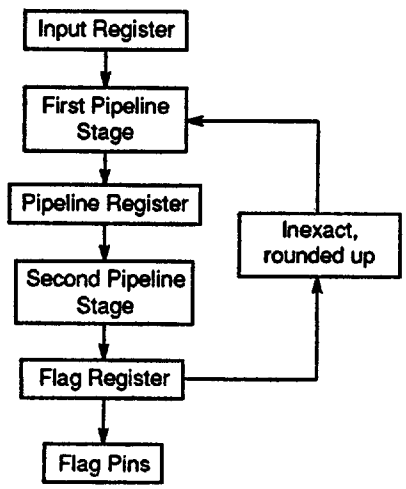
Single Chip Floating Point Processors B2130/B3130/B4130

T-49-12-05

unbiased exponent	biased exponent (hex)	type
383	FE	wrapped overflow (maximum)
382	FD	wrapped overflow
•	•	• •
•	•	• •
129	00	wrapped overflow
128	FF	wrapped overflow (minimum)
127	FE	IEEE normalized (maximum)
126	FD	IEEE normalized
•	•	• •
•	•	• •
–125	02	IEEE normalized
–126	01	IEEE normalized (minimum)
–127	00	wrapped underflow (maximum)
–128	FF	wrapped underflow
•	•	• •
•	•	• •
–381	02	wrapped underflow
–382	01	wrapped underflow (minimum)

WRAPPED TO DENORMALIZED CONVERSION

In order to convert a wrapped number to a correctly rounded denormalized number, the inexact and rounded up flags from the operation which generated the wrapped number are required. The state of these flags during the first stage of the pipeline is assumed to match the state when the wrapped result was generated, and determines the rounding of the result. This can be accomplished by immediately preceeding a wrapped to denorm conversion operation with an ALU flag register write operation which restores the inexact and rounded up flags to their state at the end of the operation which produced the wrapped result. Note that the ALU flag register write operation executes in the MPY block (see Status Register Operation, page 9). When OFT=1 (output flow through mode), a wrapped to denorm conversion will return unspecified results. The underflow flag returned by the wrapped to denorm conversion will be the underflow flag that applies to the combined floating point operation and denorm conversion. Tininess is detected after rounding, while a loss of accuracy is detected as an inexact result (ANSI/IEEE Standard 754, section 7.4, items (1) and (4)).



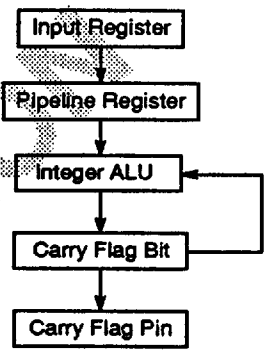
WRAPPED TO DENORM CONVERSION PIPELINING

To convert a wrapped number to a denorm, use the following instruction sequence:

```
          FREGAW    Wrapped_flags
Denorm = WDNM       Wrapped_number
```

ADD WITH CARRY

Add-with-carry type ALU instructions use the contents of the carry flag as of the beginning of the second stage of the pipeline. This allows multiple pipelined add-with-carry instructions to operate with each instruction using the carry out of the previous instruction as its carry input. Starting a write ALU flag register instruction in the multiplier at the same time that an add-with-carry instruction is started in the ALU will return unspecified results. When OFT=1 (output flow through mode), add-with-carry instructions return unspecified results.



ALU add-with-carry operation pipelining: The following sequence of operations will add together the multiple word integers A and B, and leave their sum in C. A[0], B[0] and C[0] contain the least significant words of each integer, A[3], B[3] and C[3] contain the most significant words:

```
C[0] =    IADD   A[0], B[0]
C[1] =    IADDC  A[1], B[1]
C[2] =    IADDC  A[2], B[2]
C[3] =    IADDC  A[3], B[3]
```

RESET OPERATION

Use of the asynchronous hardware reset causes the following events to occur:

- All flags are cleared
- All interrupt enable bits are cleared (prevents interrupt flag from being set)
- The Mode register is set to zero

Therefore, after a reset, the Bx130 will be configured with the following operating modes:

- IEEE mode
- Freeze on interrupt disabled
- Parity checking is disabled
- Even parity is generated
- Round to nearest
- Borrow mode disabled

The parity flag is not sticky
Instruction set is orthogonal
Flag outputs are not multiplexed
Flags are updated only when an appropriate instruction is executed

UNSUPPORTED OPERATIONS

The following operations from the B2110A/2120A, the B3110A/B3120A and the B5110/B5120 are not supported by the Bx130:

NORMX	normalize
PASSn	pass opcode
SCREGR, SCREGW	SC register read/write
ROTX, LROTX	rotate
ROTC, LROTC	concatenated rotate
BITR	bit reverse and rotate
ADDSC	add to SC register
NEGSC	negate SC register

Fixed Point Formats (Integer)

The Bx130 assumes that 32-bit, or smaller, integers will be in the MSW of the X or Y port. If this is not the case, use signals (pins) CLTHX or CLTHY, as appropriate, to copy the LSW to the MSW.

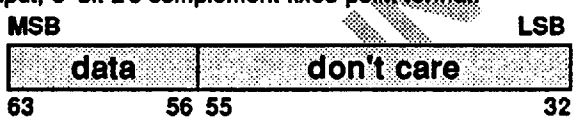
Results for 8-bit, 16-bit, and 32-bit integer operations are written to both the MSW and the LSW of the 64-bit output.

Bytewide (8-bit) and Halfword (16-bit) division, remainder and multiply operations use a different input and output format than 32-bit and 64-bit operations.

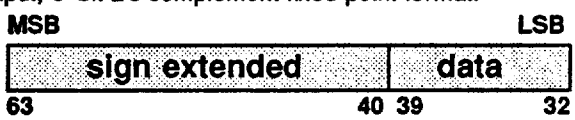
8-bit and 16-bit INPUT FORMATS

For divide/remainder operations:

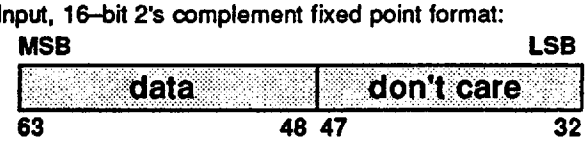
X Input, 8-bit 2's complement fixed point format:



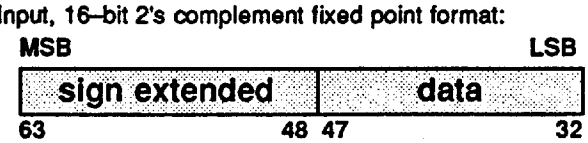
Y Input, 8-bit 2's complement fixed point format:



X Input, 16-bit 2's complement fixed point format:

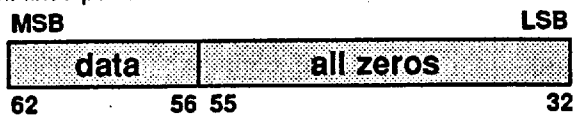


Y Input, 16-bit 2's complement fixed point format:

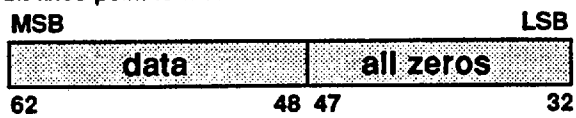


For 8- and 16-bit multiply:

8-bit fixed point format:



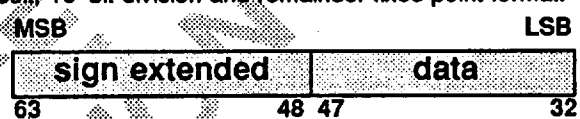
16-bit fixed point format:



8-bit and 16-bit OUTPUT FORMATS

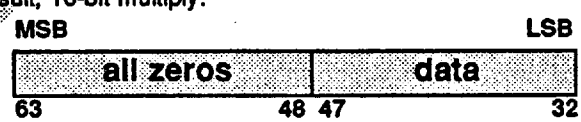
Result, 8-bit division and remainder fixed point format:

Result, 16-bit division and remainder fixed point format:



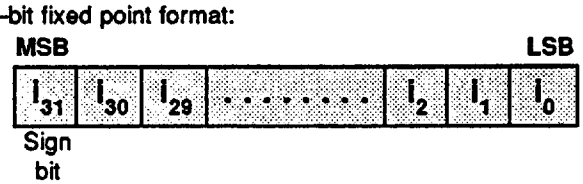
Result, 8-bit multiply:

Result, 16-bit multiply:

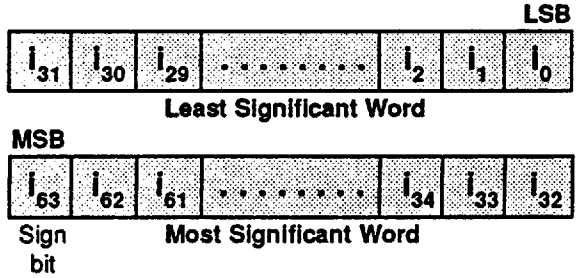


32-bit and 64-bit I/O FORMATS

32-bit fixed point format:



64-bit fixed point format:

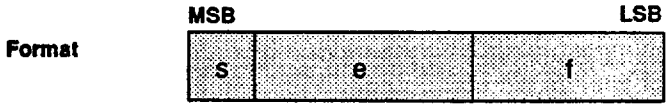


Single Chip Floating Point Processors B2130/B3130/B4130

T-49-12-05

IEEE Floating Point Format

IEEE Standard 754-1985 binary floating point arithmetic single and double precision basic formats are supported by the Bx130. Extended formats are not supported. The floating point data word is made up of three parts: sign bit, biased exponent and fraction. See the IEEE Std. 754 for additional information.



Single	1	8	23
Double	(1)	(11)	(52)

Where s = sign bit
e = biased exponent
f = fraction

PRECISION

The value of the floating point word is determined by the following tables:

Single Precision (NF= 0, mode bit-9)

0/1	255	≠ 0, msb = 1	Signalling NaN
		≠ 0, msb = 0	Quiet NaN
0	255	0	+INF
1	255	0	–INF
0/1	1-254	f	$(-1)^s \cdot 2^{e-127} \cdot 1.f$
0/1	0	≠ 0	DEN
0	0	0	+0
1	0	0	–0

Double Precision (NF= 0, mode bit-9)

Sign(s)	Exponent (e)	Fraction (f)	Interpretation
0/1	2047	≠ 0, msb = 1	Signalling NaN
		≠ 0, msb = 0	Quiet NaN
0	2047	0	+INF
1	2047	0	–INF
0/1	1-2046	f	$(-1)^s \cdot 2^{e-1023} \cdot 1.f$
0/1	0	≠ 0	DEN
0	0	0	+0
1	0	0	–0

ROUNDING

The Bx130 supports all four IEEE-754 rounding modes. The rounding process takes a number and, if necessary, modifies it to fit the destination format. The destination format can be single/double precision floating point or 32-bit/64-bit fixed point.

Round to Nearest
This mode rounds the infinitely precise result to the nearest representable value that fits the destination format. Results that are halfway between two representable values are rounded toward the even result (result with LSB=0 is delivered). This rounding mode is statistically unbiased because over a large quantity of random numbers half will be rounded up and half rounded down.

Round toward Zero
This mode rounds the result to the closest representation whose magnitude is less than or equal to the infinitely precise result. Round to zero truncates all bits that are less significant than the destination fraction's LSB.

Round toward Plus Infinity
This mode rounds the result to the closest representation which is not less than the infinitely precise result. If the pre-rounded result is greater than the maximum representable normalized number, the result is rounded to plus infinity and the overflow flag is set.

Round toward Minus Infinity
This mode rounds the result to the closest representation which is not greater than the infinitely precise result. If the pre-rounded result is less than the minimum representable number, the result is rounded to minus infinity and the overflow flag is set.

OVERFLOWS AND UNDERFLOWS

The result of an operation which overflows or underflows depends on the sign of the result and the rounding mode. The tables below illustrate the possible results.

Overflows

ROUNDING MODE	–OV	+OV
Round to nearest	–INF	+INF
Round to zero	–M	+M
Round to – infinity	–INF	+M
Round to + infinity	–M	+INF

Note: M = Largest magnitude normalized number.

Underflows

ROUNDING MODE	–UF	+UF
Round to nearest	–0	+0
Round to zero	–0	+0
Round to – infinity	–E	+0
Round to + infinity	–0	+E

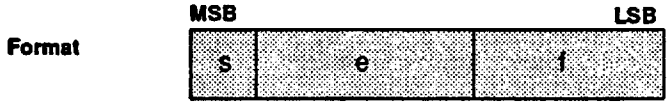
Note: E = Smallest magnitude normalized number.

Bipolar Integrated Technology, Inc.

T-49-12-05

DEC (VAX®) Floating Point Format

The Bx130 also supports the DEC F and G floating point formats. The DEC D and H formats are not supported. DEC floating point arithmetic is very similar to IEEE 754 arithmetic, but does not contain all of the special cases and operands that are defined in the IEEE specification. The F format corresponds to single precision IEEE, while the G format corresponds to double precision. For complete information on DEC format floating point arithmetic, see the VAX Architecture Handbook from Digital Equipment Corporation.



F	1	8	23
G	(1)	(11)	(52)

Where s = sign bit
e = biased exponent
f = fraction

FORMATTING

The value of the floating point word is determined by the following tables.

F Format

Sign(s)	Exponent (e)	Fraction (f)	Interpretation
0	0	0	0
0	0	≠ 0	"dirty" zero
1	0	any	Reserved operand
0/1	1-255	any	$(-1)^s \cdot 0.1f \cdot 2^{e-128}$

G Format

Sign(s)	Exponent (e)	Fraction (f)	Interpretation
0	0	0	0
0	0	≠ 0	"dirty" zero
1	0	any	Reserved operand
0/1	1-2047	any	$(-1)^s \cdot 0.1f \cdot 2^{e-1024}$

ROUNDING

DEC format arithmetic always rounds the infinitely precise result in the same way; there is no choice of rounding modes. The infinitely precise result is rounded to the nearest representable number. If two representable numbers are equally close to the infinitely precise result, then the one with the larger magnitude is chosen. This is slightly different from the IEEE round to nearest mode.

The DEC F and G format sign, exponent, and fraction field widths are the same as their IEEE format counterparts, and both IEEE and DEC formats use a "hidden" bit to increase the resolution of their mantissas. The DEC hidden bit, however, is to the right of the binary point, while the IEEE hidden bit is to the left. Furthermore, the exponent biases of the two standards differ, leading to different representable number ranges.

Normalized Number Range

	Minimum	Maximum
IEEE Single	2^{-126}	$2^{127} \cdot (2 - 2^{-23})$
DEC F	2^{-128}	$2^{126} \cdot (2 - 2^{-23})$
IEEE Double	2^{-1022}	$2^{1023} \cdot (2 - 2^{-52})$
DEC G	2^{-1024}	$2^{1022} \cdot (2 - 2^{-52})$

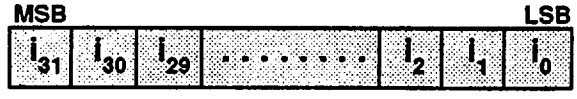
Another difference is that the DEC formats lack denormalized numbers and does not have separate representations for positive and negative zero. A number with a sign of zero, an exponent of zero, and a nonzero mantissa is considered to be a "dirty" zero. On input, dirty zeros are treated exactly the same as the normal zero (except that dirty zeros cause the denormalized input flag to be raised), but they are never returned as a result.

DEC reserved operands are similar to IEEE signaling NaNs. When they are used as an operand, the invalid operation flag is raised and a reserved operand is returned as the result. Reserved operands are also output whenever an invalid operation (such as divide by zero) or overflow occurs. DEC specifies that when a reserved operand is encountered, the destination register should not be changed. It is up to the user to ensure that this happens. Underflows raise the underflow flag and return a result of zero.

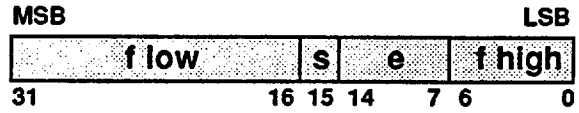
WORD ORDER

The VAX uses different word ordering for integer and floating point numbers. The ordering is not supported by the Bx130. This distinction is important if the same data path is to be used for VAX compatible integer and floating point operations and conversions. The word order of either the floating point or the integer operands will have to be explicitly swapped before the operands are stored in memory.

VAX 32-bit integer:



VAX F floating:

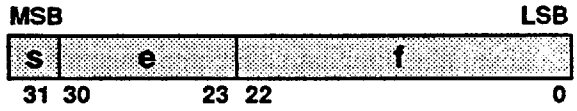


Single Chip Floating Point Processors

B2130/B3130/B4130

T-49-12-05

BIT F floating:

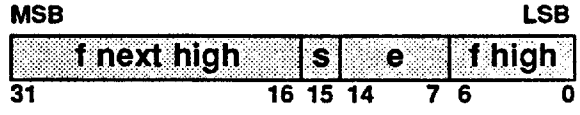


VAX 64–bit integer:

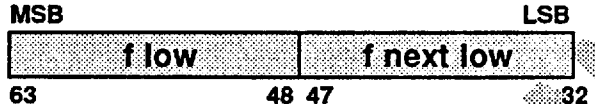
Least Significant Word

Most Significant Word

VAX G floating:

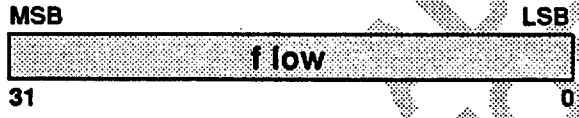


Least Significant Word

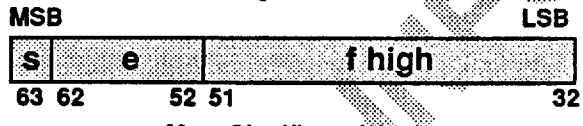


Most Significant Word

BIT G floating:



Least Significant Word



Most Significant Word

MODE REGISTER

To implement DEC format arithmetic the following mode register bits must be set/reset.

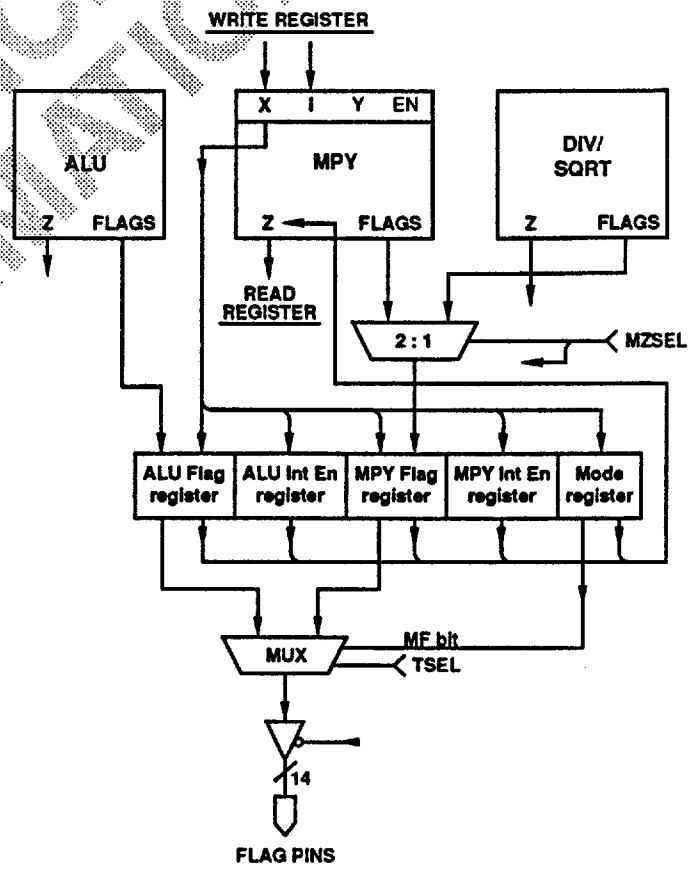
Bit	Value	Name
0	1	IEEE/DEC Format
3	0	IEEE Underflow mode
4	0	IEEE Overflow mode
5	0	IEEE Rounding mode
6	0	IEEE Rounding mode
7	1	Borrow mode (Borrow mode is only required if integer arithmetic is used.)

Control and Status Registers

STATUS REGISTER OPERATIONS AND PIPELINING

All status register read and write-to operations (flag, interrupt enable, or mode) execute through the MPY. (See block diagram, below.) The read or write instruction is input to the MPY I port. Data for status register write operations is taken from the MPY X input register, and read operations return their result on the MPY Z port.

Status register read operations are fully pipelined. The result of a read operation will appear at the Z register with the same pipeline delay as any other operation. Write operations are decoded during the first stage of the pipeline and completed as the write operation is clocked into the second stage of the pipeline. If both a normal operation and a flag register write operation attempt to modify the same flag register at the same time, the flag register write operation will complete while the flags from the normal operation are ignored.



The operation immediately following a write to the mode register operates correctly, using the new modes. Similarly a wrapped to denorm conversion immediately following a write to the ALU flag register operates using the newly written flags. The result returned by a write status register operation is unspecified. ALU operations started at the same time as a mode register write operation return unspecified results.

Note that when in output flow through mode (OFT = 1) the flag registers are transparent; flags flow directly from the functional blocks to the flag pins. In this mode of operation reading and writing the flag registers is not very meaningful, reads return unspecified results while the effect of a write operation is lost as soon as the next opcode is clocked into the multiplier.

Case 1: PLFT = 0, OFT = 0

Pins	mult0	WMR	mult1	mult2	RFR			
Input reg		mult0	WMR	mult1	mult2	RFR		
Pipeline			mult0	WMR	mult1	mult2	RFR	
Z reg				mult0	new	mult1	mult2	m2flags
Result Pins				mult0	new	mult1	mult2	m2flags
Mode reg		old	old	new	new	new	new	new
Clock								

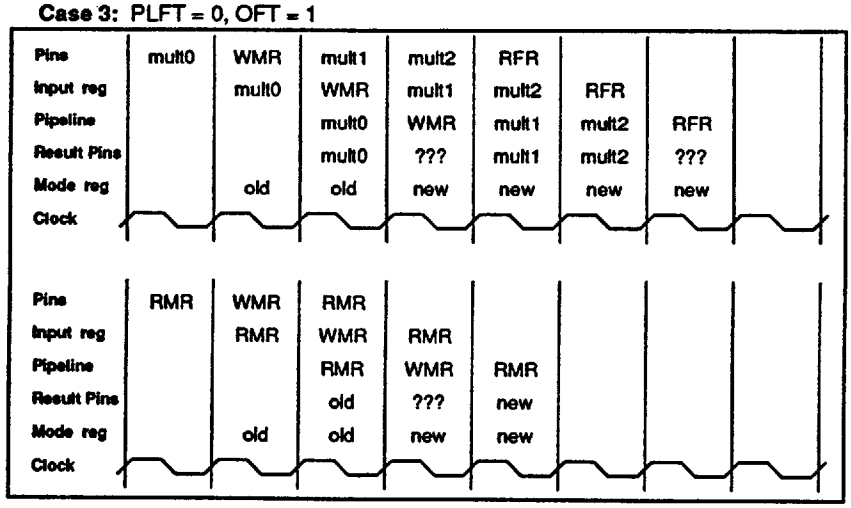
Pins	RMR	WMR	RMR					
Input reg		RMR	WMR	RMR				
Pipeline			RMR	WMR	RMR			
Z reg				old	new	new		
Result Pins				old	new	new		
Mode reg		old	old	new	new	new		
Clock								

Case 2: PLFT = 1, OFT = 0

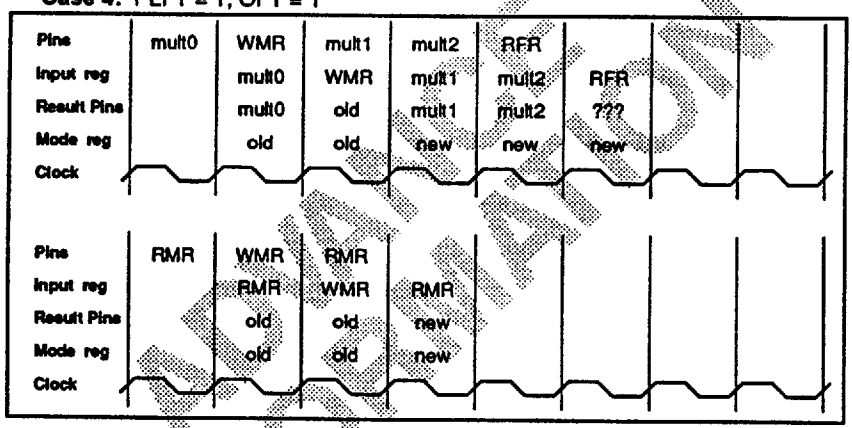
Pins	mult0	WMR	mult1	mult2	RFR			
Input reg		mult0	WMR	mult1	mult2	RFR		
Z reg			mult0	old	mult1	mult2	m2flags	
Result Pins			mult0	old	mult1	mult2	m2flags	
Mode reg		old	old	new	new	new	new	new
Clock								

Pins	RMR	WMR	RMR					
Input reg		RMR	WMR	RMR				
Z reg			old	old	new			
Result Pins			old	old	new			
Mode reg		old	old	new	new	new		
Clock								

Case 3: PLFT = 0, OFT = 1



Case 4: PLFT = 1, OFT = 1



FLAG OUTPUTS

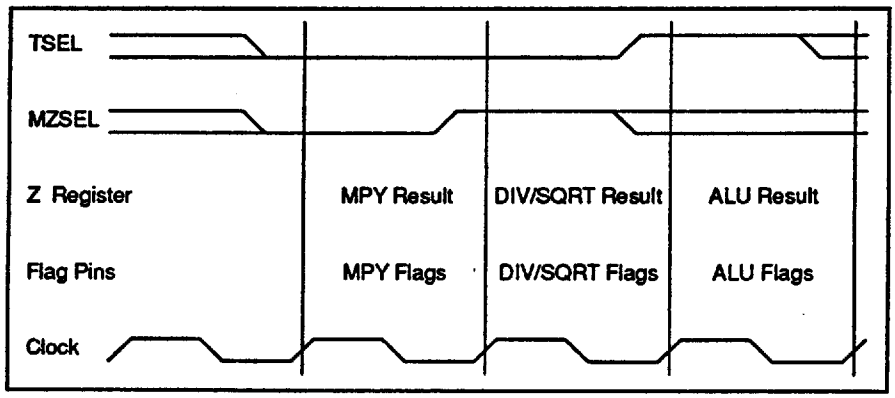
The flag output pins (FLAG[13...0]) from the Bx130 can operate in one of two modes. In the default mode (MF=0, mode register bit-12) a subset of the flags from both the ALU and the MPY are continuously available. If MF=0, FLAG[13...0] are:

FLAG[0]	ALU overflow (AOV)
FLAG[1]	ALU underflow (AUF)
FLAG[2]	ALU invalid operation (AINV)
FLAG[3]	ALU inexact result (AINX)
FLAG[4]	ALU negative (AN)
FLAG[5]	ALU zero (AZR)
FLAG[6]	ALU carry flag (CRY)
FLAG[7]	Division by zero (DIVZ)
FLAG[8]	MPY/DIV/SQRT overflow (MOV)
FLAG[9]	MPY/DIV/SQRT underflow (MUF)
FLAG[10]	MPY/DIV/SQRT invalid operation (MINV)
FLAG[11]	MPY/DIV/SQRT inexact result (MINX)
FLAG[12]	Parity error (PE)
FLAG[13]	Interrupt (INT) caused by ALU, MPY or DIV/SQRT

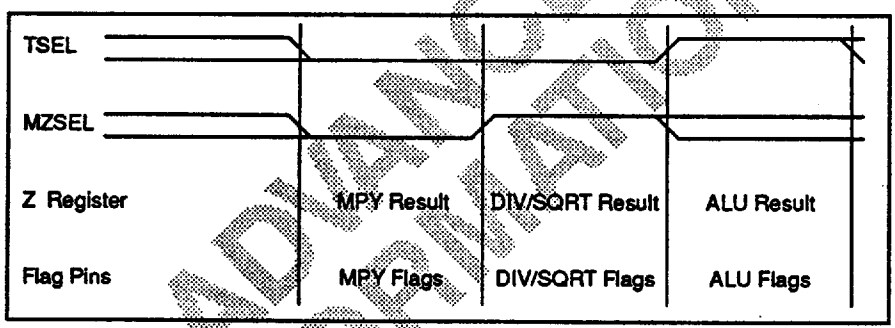
If MF=1, the flags outputs are from the ALU (TSEL=0) or the MPY (TSEL=1). Flags are output on clock if OFT=0, and available immediately if OFT=1. If MF=1, FLAG[13...0] are:

FLAG[0]	OV (Overflow)
FLAG[1]	UF (Underflow)
FLAG[2]	INV (Invalid operation)
FLAG[3]	INX (Inexact result)
FLAG[4]	N (Negative)
FLAG[5]	ZR (Zero)
FLAG[6]	CRY or DIVZ (ALU carry or DIV divide by zero)
FLAG[7]	NaN (Not a number)
FLAG[8]	RND (Rounded up)
FLAG[9]	DEN (Denormalized operand (DX or DY))
FLAG[10]	–not used–
FLAG[11]	–not used–
FLAG[12]	PE (Parity error)
FLAG[13]	INT (Interrupt – caused by selected flag register only)

Note that the PE and INT flags always appear on the same pins regardless of whether the flags are multiplexed or not. Note also that in the multiplexed mode there are two unused flag outputs. In the multiplexed flags mode with OFT=0:



In the multiplexed flags mode with OFT=1, TSEL and MZSEL will select the result and flags will be output without being delayed by the clock:



Control and Status Register Description

ALU and MPY Flag Registers

20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
P2	P1	P0	PY	PX	NY	NX	CRY	DIVZ	DY	DX	NaN	RND	INX	INV	UF	OV	ZR	N	PE	INT
MSB																				LSB

P[2..0], PX, PY — Parity Error Byte Location flags. (ALU Flag register only). When a parity error is detected at port X or Y, bit PX and/or PY is set high to identify the input port(s) containing the error, PX signifies port X and PY signifies port Y. P2 through P0 are encoded high to indicate which byte, within the 64-bit word, contained the error. Code 07H indicates the most significant byte and code 00H the least significant byte. If more than one byte contains a parity error, the highest error-byte will be indicated. If both X and Y ports exhibit errors, the highest-error byte in the X port will be indicated. P[2..0] are valid only if PX or PY is set.

NY, NX — Not a Number input flags. During floating point operations, one or both of these bits will be set if a NaN is received as an operand for either X or Y. NY signals a NaN Y input, and NX flags a NaN X input. If the NaN was received by the ALU, the ALU Flag register will have one of these

bits set; if the MPY or DIV/SQRT received the NaN, the MPY flag register will be updated. Note that if either NX or NY is set, NaN will also be set. In DEC mode, reserved operands received on the inputs will set these flags.

CRY — Carry flag. (ALU flag register only) When the borrow mode bit (BM) in the mode register = 1, the carry flag will be asserted if there is a carry out of the accumulator during integer additions or no carry out during integer subtractions. When BM=0 the carry flag will always indicate a carry out of the ALU during integer operations. During shifts, the carry flag always contains the last bit to leave the ALU. This flag is also used during Min/Max, Compare and Pass operations. See Instruction explanations for details.

DIVZ — Divide by zero flag. (MPY flag register only) This bit is set when a finite non-zero number, or an integer number, is divided by zero.

DY, DX — Denormalized input flags. During floating point operations, one or both of these bits will be set if a denormalized number was received as an operand for either X or Y. DY signals a denormalized Y input, and DX flags a denormalized X input. The bits are set without regard to ID (mode register bit-0). If the denormalized number was received by the ALU, the ALU Flag register will have one of these bits set; if the MPY or DIV/SQRT received the denormalized number, the MPY flag register will be updated.

NaN — Not a number flag. Only valid for floating point operations, this bit indicates that an operand received on one of the two input ports or the result output was Not a Number. In IEEE mode, a signaling NaN input causes both the NaN flag and the invalid operation flag to be set. A quiet NaN causes only the NaN flag to be set. In both cases, the output will be a quiet NaN.

The sign of NaN's follow arithmetic conventions. For example, –NaN multiplied by +NaN = –NaN. During square root operations, the sign of a NaN result is the sign of the X input. The ALU always outputs NaN's with a negative sign.

In DEC mode, reserved operands will set both the NaN and INV flags. A reserved operand is output, and the "value" of the output will be "–0." See DEC format section.

RND — Rounded Up output flag. This bit will be set whenever the normalized output has been rounded away from zero.

INX — Inexact Result flag. This bit will be set whenever the output is not infinitely precise.

INV — Invalid Operation flag. This bit is set when an input operand is invalid for the requested operation. In IEEE mode, the following conditions cause an Invalid Operation flag:

1) A signaling NaN on either input
2) Magnitude subtraction of infinities, i.e. (+INF) + (–INF)
3) Zero multiplied with infinity
4) Zero divided by zero or infinity divided by infinity
5) Square root of a negative number (except –0)
6) Conversion of a floating point number to an integer format when the operand overflows the integer format or is not representable

In DEC mode, INV will be active for the following conditions:
1) Reserved operand on either input
2) Zero/zero
3) Square root of a negative number
4) Case (6) above

UF — Underflow flag. This flag is set if the magnitude of the result of an operation is not zero and is less than the minimum representable normalized number in the chosen format.

OV — Overflow flag. This bit is set if the magnitude of the result of an operation is larger than the maximum representable normalized number in the chosen format.

ZR, N — Zero, Negative flags. ZR indicates a zero result and N indicates a negative result. N is set whenever the most significant bit of the result=1. In addition, these bits indicate the result of a floating point compare operation as follows:

ZR	N	Result
0	0	X > Y
0	1	X < Y
1	0	X = Y
1	1	unordered

PE — Parity Error flag. If a parity error has been detected at either input port, this bit will be set. Odd or even parity, based on the parity mode bit, is checked during every operation except ALU flag register writes. Parity is checked on the bus any time an input enable signal is active. Additionally, both the MSW and LSW are always checked, even during single precision operations. A parity error will occur if the parity bit plus the corresponding byte contain an even number of ones and PM = 1. An input of all zeros (data and parity) is valid if even parity is set, but is a parity error if odd parity is set.

INT — Interrupt flag. This bit is set if all of the following conditions are true: If the IE bit is set in the interrupt enable register, and if a flag register bit, and its corresponding interrupt enable bit are both set. The value of INT (for the register selected by MF) is also the FLAGINT output.

Access to the contents of either flag register is obtained by using the FREGx instructions.

The five parity error status flags are updated only when a parity error occurs. The state of these flags thus represents the byte location of the most recent parity error. These flags can be cleared by the CLRFLAG instruction or by a hardware reset. The parity error (PE) flag is also "sticky" if the SP flag (mode register bit-8) is set.

Every arithmetic operation updates the flag register on the rising edge of CLK, if enabled by the appropriate flag register enable. With the exception of the parity bits, each update will clear the existing bits and set them according to the current operation, unless the flags are set as sticky or frozen (using freeze flags on interrupt mode). Once frozen, the bits will be cleared by the CLRFLAG instruction, by a hardware reset, or by writing zeros to the flag register.

Interrupt Enable Registers

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NY	NX	CRY	DIVZ	dc	DEN	NaN	RND	INX	INV	UF	OV	ZR	N	PE	IE

MSB LSB

NOTE: 1 = enable, 0 = disable

NY, NX — Input NaN interrupt enables. Allows an interrupt if the corresponding flag register NY or NX is set. These bits have no effect on the NaN interrupt enable pin.

CRY — Carry interrupt enable. Used only in the ALU, is don't care in the MPY. Allows an interrupt if the corresponding flag register CRY bit is set.

DIVZ — Divide by zero flag interrupt enable. Used only in the MPY, is don't care in the ALU. Allows an interrupt if the corresponding flag register DIVZ bit is set.

dc — Don't care on write, always read as zero.

DEN — Denormalized number interrupt enable. Allows an interrupt if either of the denormalized number flags (flag register DX and DY bits) are asserted.

NaN — Not a Number interrupt enable. Allows an interrupt if the corresponding flag register NaN bit is set.

RND — Round interrupt enable. Allows an interrupt if the corresponding flag register RND bit is set.

INX — Inexact result interrupt enable. In IEEE mode, allows an interrupt whenever the flag register INX bit is asserted. In DEC mode, this bit should be set to zero.

INV — Invalid operation interrupt enable. Allows an interrupt if the corresponding flag register INV bit is set.

UF — Underflow interrupt enable. Allows an interrupt if the corresponding flag register UF bit is set.

OV — Overflow interrupt enable. Allows an interrupt if the corresponding flag register OV bit is set.

ZR — Zero interrupt enable. Allows an interrupt if the corresponding flag register ZR bit is set.

N — Negative interrupt enable. Allows an interrupt if the corresponding flag register N bit is set.

PE — Parity Error interrupt enable (ALU only). Allows an interrupt if the corresponding flag register PE bit is set.

IE — Master Interrupt enable. Must be set to one for an interrupt to be generated for any of the above conditions.

The ALU and the MPY each have an interrupt enable register which functions in concert with its corresponding flag register. The interrupt enable registers allow the user to determine which condition(s) will activate an interrupt. An interrupt will occur if all of the following conditions are true: If the IE bit is set in the interrupt enable register, and if a flag register bit, and its corresponding interrupt enable bit are both set. The interrupt enable bits will not prevent the flags from being set or reset.

Access to the interrupt enable register is accomplished using the IREGx instructions.

Single Chip Floating Point Processors B2130/B3130/B4130

T-49-12-05

Mode Register

16—13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	MF	PM	EC	NF	SP	BM	R1	R0	IO	IU	EP	FF	ID

MSB LSB

Note: 1 = enable, 0 = disabled

Res These bits are reserved. They must be written as zero if writing to the Mode register. If the Mode register is accessed for reading, these bits will be read as zero. Writing ones to these bits may produce anomalous results.

MF Multiplex flags when one, not multiplexed when zero. See Flag Outputs (page 11) for details.

PM Parity Mode: When high, the input buses are checked for odd parity and the output bus is generated with odd parity. When low, the input buses are checked for even parity and the output bus is generated with even parity. If all of the data bits for a byte are zero, even parity mode will generate a parity bit of zero while odd parity mode will generate a parity bit of one.

EC Enable concurrency. When zero, the flag registers are updated only when the corresponding functional block is performing a specified operation. For example when EC=0 and the ALU receives an undefined or MPY opcode the contents of the ALU flag register will not be modified. When EC=1 the flag registers are updated on every clock cycle. In this case the MPY block will execute a floating point multiplication, with precision given by I [0], in response to all of the ALU opcodes that have I [7]=0 (add, subtract, convert, compare, max, min). In either mode, a flag register may be forced to hold its contents by holding its clock enable pin high. See the instruction description section for more information.

N F NaN Format mode. The definition of a NaN is controlled by this bit and is indicated below:

NF Value	Sign	Exponent	Fraction	NaN Type	Returned NaN
0	0/1	FF (DP = 7FF)	0xxxx...	Quiet	0100...0
	0/1	FF (DP = 7FF)	1xxxx...	Signal	0100...0
1	0/1	FF (DP = 7FF)	1xxxx...	Quiet	1100...0
	0/1	FF (DP = 7FF)	0xxxx...	Signal	1100...0

SP Sticky Parity mode. When SP=1, the parity error flag (PE, flag register bit-1) is sticky, once set it will remain set. When sticky, the parity flag can be reset by asserting the reset pin, executing the CLRFLAG instruction, or writing a zero to the parity error flag bit. This mode bit has no effect on bits [16..20] of the flag register.

BM Borrow mode. Only applies to the ALU. When BM=0, normal carry mode is used, i.e. the CRY flag is set whenever there is a carry out of the ALU. When BM=1, DEC carry mode is used, i.e. CRY flag is set if a carry out of the ALU occurs during addition or if there is no carry (borrow) during subtraction.

R1, R0 Rounding mode. When in DEC mode R1, R0 must be set to 0,0. When in IEEE mode the rounding operation is determined by the following chart.

R1	R 0	ROUNDING MODE
0	0	Round to nearest
0	1	Round to zero
1	0	Round to – infinity
1	1	Round to + infinity

IO IEEE Overflow Mode. When in IEEE mode (ID=0), IO=1 causes overflows to be returned as wrapped numbers. If IO=0, then overflows will be set to either infinity or to the largest finite number, according to section 7.3 of the IEEE standard 754. When ID=1 (DEC mode), this bit must be zero.

IU IEEE Underflow Mode. When in IEEE mode (ID=0), IU=1 causes underflows to be returned as wrapped numbers. If IU=0, then overflows will be set to a properly signed zero and denormalized inputs will be flushed to zero. When ID=1 (DEC mode), this bit must be zero.

EP Enable Parity. When EP=1 parity checking on the input buses will occur and the flags associated will be updated. Parity is always generated on the outputs, regardless of the state of this bit.

FF Freeze Flags on interrupt mode. When FF=1, the bits in the flag registers will freeze (i.e. remain in their current state) once an interrupt has been generated. The flags will remain frozen until a CLRFLAG instruction is executed, the hardware reset pin is asserted, zeros are written to the appropriate flag register or interrupt enable register, or the FF bit is cleared. Status register flags are frozen on the rising edge of CLK if the corresponding flag register enable pin is active. The flag pins reflect the flag registers and will be frozen along with the flag registers.

ID IEEE/DEC mode. When ID=1, the Bx130 operates in DEC (F or G format) mode. When ID=0, IEEE mode is used.

Access to the Mode Register is accomplished through the use of the MREGx instructions.

Bipolar Integrated Technology, Inc.

T-49-12-05

INSTRUCTION SET

The Bx130 instruction set supports a wide range of technical computing applications, from high performance general purpose computing to parallel vector processing with 32- and 64-bit floating point and 32- and 64-bit integer operands.

A single 8-bit instruction stream (I [7..0]) carries opcodes to the internal computing elements of the Bx130. These opcodes are encoded orthogonally so that no code (except as specified) will cause operations on both internal elements. A mode bit is provided which overides this orthogonality.

Three instruction pairs (MADD/DMADD, MSUB/DMSUB and MSUBX/DMSUBX) are encoded so that each opcode is interpreted by the MPY as a multiply and by the ALU as an add or subtract. They are provided to simplify parallel add/subtract and multiply operations.

INSTRUCTION SYMBOLS

SYMBOL	DEFINITION
ALU	Internal Arithmetic Logic Unit
CRY	Carry
DEN	Denormalized number
DIV/SQRT	Internal Divide/Square Root Element
DIVZ	Divide by 0
DP	64-bit floating point number
DX	Denormalized input X
DY	Denormalized input Y
E	Smallest magnitude normalized number
EXP	Exponent
INF	Infinity
INT	Integer
INV	Invalid
INX	Inexact
L	64-bit Long integer
M	Largest magnitude normalized number
MANT	Mantissa
MPY	Internal Multiplier Element
N	Negative
{n}	User determined binary number
N/A	Not applicable
NaN	Not a number
NORM	Normalized number
OV	Overflow
Q	Quiet NaN
R	DEC reserved operand
RND	Rounded up
S	Signaling NaN
sb	Sticky bit
SP	32-bit floating point number
UF	Underflow
WRP	Wrap
ZR	Zero
√	Square root
/	Divide
•	Multiply
*	For Function or Description * indicates compliment (NOT). Within a flag block it indicates the flag is affected
|	Concatenation
( )	Item(s) operated on; ie. SIGN(X) is Sign of X.
| |	Absolute value
[,]	Items within braces are alternative items, one of them must be used

INSTRUCTION SET SUMMARY

MNEMONIC	OPCODE	FUNCTION
FLOATING POINT ARITHMETIC INSTRUCTIONS		
DIV	00H	X/Y
DDIV	01H	DP: X/Y
SQRTX	02H	√X
DSQRTX	03H	DP: √X
MULTWX	04H	WRAPPED X • Y
DMULTWX	05H	DP: WRAPPED X • Y
MULTWY	06H	X • WRAPPED Y
DMULTWY	07H	DP: X • WRAPPED Y
MULT	08H	X • Y
DMULT	09H	DP: X • Y
MULTAY	0AH	X • |Y|
DMULTAY	0BH	DP: X • |Y|
MULTAX	0CH	|X| • Y
DMULTAX	0DH	DP: |X| • Y
MULTA	0EH	|X • Y|
DMULTA	0FH	DP: |X • Y|
ADD	30H	X + Y
DADD	31H	DP: X + Y
SUB	32H	X – Y
DSUB	33H	DP: X – Y
SUBX	34H	Y – X
DSUBX	35H	DP: Y – X
ADDA	38H	|X| + |Y|
DADDA	39H	DP: |X| + |Y|
SUBA	3AH	|X| – |Y|
DSUBA	3BH	DP: |X| – |Y|
SUBXA	3CH	|Y| – |X|
DSUBXA	3DH	DP: |Y| – |X|

Single Chip Floating Point Processors B2130/B3130/B4130

T-49-12-05

INSTRUCTION SET SUMMARY (cont'd)

FLOATING POINT ARITHMETIC INSTRUCTIONS (cont'd)

MNEMONIC	OPCODE	FUNCTION
MIN	24H	MIN[X,Y]
DMIN	25H	DP: MIN[X,Y]
MAX	26H	MAX[X,Y]
DMAX	27H	DP: MAX[X,Y]
ABSX	28H	|X|
DABSX	29H	DP: |X|
NEGX	2AH	–X
DNEGX	2BH	DP: –X
PASSX	2CH	X
DPASSX	2DH	DP: X
CLRS	18H	0 | EXP(X) | MANT(X)
DCLRS	19H	DP: 0 | EXP(X) | MANT(X)
INVS	1AH	SIGN(X)* |EXP(X) | MANT(X)
DINVS	1BH	DP: SIGN(X)* |EXP(X) | MANT(X)
CMPR	36H	COMPARE(X,Y)
DCMPR	37H	DP: COMPARE(X,Y)
CMPRA	3EH	COMPARE(|X|,|Y|)
DCMPRA	3FH	DP: COMPARE(|X|,|Y|)
MADD	12H	X • Y, X + Y
DMADD	13H	DP: X • Y, X + Y
MSUB	14H	X • Y, X – Y
DMSUB	15H	DP: X • Y, X – Y
MSUBX	16H	X • Y, Y – X
DMSUBX	17H	DP: X • Y, Y – X

FLOATING POINT SUPPORT INSTRUCTIONS

MNEMONIC	OPCODE	FUNCTION
PASSXM	10H	MPYZ ← X
DPASSXM	11H	DP: MPYZ ← X
SCALE	20H	SIGN(X) | EXP(X) + Y | MANT(X)
DSCALE	21H	DP: SIGN(X) | EXP(X) + Y | MANT(X)
MERGE	22H	SIGN(X) | EXP(Y) | MANT(X)
DMERGE	23H	DP: SIGN(X) | EXP(Y) | MANT(X)
NOP	58H	NO OPERATION
CLRFLAG	59H	ALU/MPY FLAGS ← 0
PASSXAR	51H	Pass X, set ALU registers
PASSXMR	50H	Pass X, set MPY or DIV/SQRT registers
FREGAR	52H	MPYZ ← AFLAGS
FREGAW	53H	AFLAGS ← X
FREGMR	5AH	MPYZ ← MFLAGS
FREGMW	5BH	MFLAGS ← X
IREGAR	54H	MPYZ ← AINTEN
IREGAW	55H	AINTEN ← X
IREGMR	5CH	MPYZ ← MINTEN
IREGMW	5DH	MINTEN ← X
MREGR	56H	MPYZ ← MODE
MREGW	57H	MODE ← X

CONVERSION INSTRUCTIONS

MNEMONIC	OPCODE	FUNCTION
SDF	76H	SP → DP
DSDF	77H	DP → SP
FFI	7CH	SP → SP format INT
DFFI	7DH	DP → DP format INT
FFIT	7EH	SP → SP format INT (Rnd to 0)
DFFIT	7FH	DP → DP format INT (Rnd to 0)
FCUI	60H	SP → 32-bit unsigned INT
DFCUI	61H	DP → 32-bit unsigned INT
FCSI	62H	SP → 32-bit signed INT
DFCSI	63H	DP → 32-bit signed INT
UICF	64H	SP ← 32-bit unsigned INT
UICDF	65H	DP ← 32-bit unsigned INT
SICF	66H	SP ← 32-bit signed INT
SICDF	67H	DP ← 32-bit signed INT
FCLUI	68H	SP → 64-bit unsigned INT
DFCLUI	69H	DP → 64-bit unsigned INT
FCLSI	6AH	SP → 64-bit signed INT
DFCLSI	6BH	DP → 64-bit signed INT
LUICF	6CH	SP ← 64-bit unsigned INT
LUICDF	6DH	DP ← 64-bit unsigned INT
LSICF	6EH	SP ← 64-bit signed INT
LSICDF	6FH	DP ← 64-bit signed INT
FCUIT	70H	SP → 32-bit unsigned INT (Rnd to 0)
DFCUIT	71H	DP → 32-bit unsigned INT (Rnd to 0)
FCSIT	72H	SP → 32-bit signed INT (Rnd to 0)
DFCSIT	73H	DP → 32-bit signed INT (Rnd to 0)
FCLUIT	78H	SP → 64-bit unsigned INT (Rnd to 0)
DFCLUIT	79H	DP → 64-bit unsigned INT (Rnd to 0)
FCLSIT	7AH	SP → 64-bit signed INT (Rnd to 0)
DFCLSIT	7BH	DP → 64-bit signed INT (Rnd to 0)
WDNM	74H	WRAPPED → DENORM
DWDNM	75H	DP: WRAPPED → DENORM

Bipolar Integrated Technology, Inc.

T-49-12-05

INSTRUCTION SET SUMMARY (cont'd)

INTEGER ARITHMETIC INSTRUCTIONS

MNEMONIC	OPCODE		FUNCTION
BDIV	89H		8-bit signed X / 8-bit signed Y
HDIV	8DH		16-bit signed X / 16-bit signed Y
IDIV	83H		32-bit signed X / 32-bit signed Y
LIDIV	C3H		64-bit signed X / 64-bit signed Y
MIDIV	84H		64-bit signed X / 32-bit signed Y
BREM	8BH		8-bit signed X / 8-bit signed Y
HREM	8FH		16-bit signed X / 16-bit signed Y
IREM	87H		32-bit signed X / 32-bit signed Y
LIREM	C7H		64-bit signed X / 64-bit signed Y
MIREM	88H		64-bit signed X / 32-bit signed Y
IBITR	F5H		Bit reverse X
IMULT	F8H		unsigned X • unsigned Y
IMULTSX	F9H		signed X • unsignedY
IMULTSY	FAH		unsigned X • signed Y
IMULTS	FBH		signed X • signed Y
BMULT	F4H		8-bit signed X • 8-bit signed Y
HMULT	F6H		16-bit signed X • 16-bit signed Y
IMULTH	FCH		unsigned X • unsigned Y → T(32)
IMULTHSX	FDH		signed X • unsigned Y → T(32)
IMULTHSY	FEH		unsigned X • signed Y → T(32)
IMULTHS	FFH		signed X • signed Y → T(32)
LIMULT	F7H		64-bit signed X • 64-bit signed Y
IADD	E0H		X + Y
LIADD	A0H	L:	X + Y
ISUB	E5H		X – Y
LISUB	A5H	L:	X – Y
ISUBX	E6H		Y – X
LISUBX	A6H	L:	Y – X
IADDP	E4H		X + Y + 1
LIADDP	A4H	L:	X + Y + 1
ISUBM	E1H		X – Y – 1
LISUBM	A1H	L:	X – Y – 1
ISUBXM	E2H		Y – X – 1
LISUBXM	A2H	L:	Y – X – 1
IADDC	E8H		X + Y + CRY
LIADDC	A8H	L:	X + Y + CRY
ISUBC	E9H		X – Y – CRY*
LISUBC	A9H	L:	X – Y – CRY*
ISUBXC	EAH		Y – X – CRY*
LISUBXC	AAH	L:	Y – X – CRY*
ISMIN	C6H		signed MIN[X,Y]
LISMIN	86H	L:	signed MIN[X,Y]
ISMAX	C2H		signed MAX[X,Y]
LISMAX	82H	L:	signed MAX[X,Y]
IUMIN	CEH		unsigned MIN[X,Y]
LIUMIN	8EH	L:	unsigned MIN[X,Y]
IUMAX	CAH		unsigned MAX[X,Y]
LIUMAX	8AH	L:	unsigned MAX[X,Y]
ISCMPR	C1H		signed Compare(X,Y)
LISCMPR	81H	L:	signed Compare(X,Y)
IUCMPR	C0H		unsigned Compare(X,Y)
LIUCMPR	80H	L:	unsigned Compare(X,Y)
INEGC	EBH		–X – CRY*
LINEGC	ABH	L:	–X – CRY*
IABSX	EFH		|X|
LIABSX	AFH	L:	|X|
INEGX	E7H		–X
LINEGX	A7H	L:	–X

INTEGER BOOLEAN INSTRUCTIONS

MNEMONIC	OPCODE		FUNCTION
LSS	F0H		Logical shift X w/sb
LLSS	B0H	L:	Logical shift X w/sb
LS	F1H		Logical shift X
LLS	B1H	L:	Logical shift X
AS	F2H		Arithmetic shift X
LAS	B2H	L:	Arithmetic shift X
INAND	D0H		X* or Y*
LINAND	90H	L:	X* or Y*
IORNX	D1H		X* or Y
LIORNX	91H	L:	X* or Y
IORNY	D2H		X or Y*
LIORNY	92H	L:	X or Y*
IOR	D3H		X or Y
LIOR	93H	L:	X or Y
IANDNY	D4H		X and Y*
LIANDNY	94H	L:	X and Y*
IAND	D5H		X and Y
LIAND	95H	L:	X and Y
INOR	D6H		X* and Y*
LINOR	96H	L:	X* and Y*
IANDNX	D7H		X* and Y
LIANDNX	97H	L:	X* and Y
IXNOR	DEH		X xnor Y
LIXNOR	9EH	L:	X xnor Y
IXOR	DFH		X xor Y
LIXOR	9FH	L:	X xor Y
ISET	D8H		All Ones
LISET	98H	L:	All Ones
INOTX	D9H		X*
LINOTX	99H	L:	X*
IPASSY	DAH		Y
LIPASSY	9AH	L:	Y
IPASSX	DBH		X
LIPASSX	9BH	L:	X
ICLR	DCH		0
LICLR	9CH	L:	0
INOTY	DDH		Y*
LINOTY	9DH	L:	Y*

Single Chip Floating Point Processors B2130/B3130/B4130

T-49-12-05

INSTRUCTION SET DESCRIPTION

FLOATING POINT ARITHMETIC INSTRUCTIONS

MNEMONIC	OPCODE	FUNCTION	FLAGS AFFECTED
DIV	00H	X/Y	
DDIV	01H	DP: X/Y	

MPY (DIV/SQRT)

NY	NX	CRY	DIVZ	DY	DX	NaN	RND	INX	INV	UF	OV	ZR	N
*	*	N/A	*	*	*	*	*	*	*	*	*	*	*

Description: Floating Point Division.

Comments: In the tables below, the first entry represents the flag that is set, the second represents the returned result. Once a DIV instruction begins, enabling any input (X, Y or I) will restart the instruction with possibly anomalous results.

IEEE-WRAPPED UNDERFLOW MODE

Y \ X OPERAND	0	DEN	NORM	INF	Q	S
0	INV, NaN/Q	DIVZ/INF	DIVZ/INF	INF	NaN/Q	INV, NaN/Q
DEN	ZR/0	INV, NaN/Q	INX, DIVZ/INF	INF	NaN/Q	INV, NaN/Q
NORM	ZR/0	INX, ZR/0	OV/[WRP, INF, M] NORM UF/WRP	INF	NaN/Q	INV, NaN/Q
INF	ZR/0	ZR/0	ZR/0	INV, NaN/Q	NaN/Q	INV, NaN/Q
Q	NaN/Q	NaN/Q	NaN/Q	NaN/Q	NaN/Q	INV, NaN/Q
S	INV, NaN/Q	INV, NaN/Q	INV, NaN/Q	INV, NaN/Q	INV, NaN/Q	INV, NaN/Q

DEC MODE

Y \ X OPERAND	0	NORM	R
0	INV/R	DIVZ/R	INV/R
NORM	ZR/0	OV/R NORM UF, ZR/0	INV/R
R	INV/R	INV/R	INV/R

IEEE-WRAPPED UNDERFLOW MODE DISABLED

Y \ X OPERAND	0	DEN	NORM	INF	Q	S
0	INV, NaN/Q	INV, NaN/Q	DIVZ/INF	INF	NaN/Q	INV, NaN/Q
DEN	INV, NaN/Q	INV, NaN/Q	DIVZ/INF	INF	NaN/Q	INV, NaN/Q
NORM	ZR/0	ZR/0	OV/[WRP, INF, M] NORM UF [ZR]/[0, E]	INF	NaN/Q	INV, NaN/Q
INF	ZR/0	ZR/0	ZR/0	INV, NaN/Q	NaN/Q	INV, NaN/Q
Q	NaN/Q	NaN/Q	NaN/Q	NaN/Q	NaN/Q	INV, NaN/Q
S	INV, NaN/Q	INV, NaN/Q	INV, NaN/Q	INV, NaN/Q	INV, NaN/Q	INV, NaN/Q

MNEMONIC	OPCODE	FUNCTION
SQRTX	02H	√X
DSQRTX	03H	DP: √X

MPY (DIV/SQRT)

NY	NX	CRY	DIVZ	DY	DX	NaN	RND	INX	INV	UF	OV	ZR	N
0	*	N/A	0	0	*	*	*	*	*	0	0	*	*

Description: Floating Point Square Root. The square root of operand X is returned.

Comments: Once a SQRTX instruction begins, enabling any input (X, Y or I) will restart the instruction with possibly anomalous results. In the tables below, the first entry represents the flag that is set, the second represents the result returned to Z.

IEEE WRAPPED UNDERFLOW MODE

X OPERAND	Z RESULT
S	INV, NaN/Q
Q	NaN/Q
–INF	INV, NaN/Q
–NORM	INV, NaN/Q
–DEN	INV, NaN/Q
–0	ZR/–0
+0	ZR/+0
+DEN	INX, ZR/+0
+NORM	+NORM
+INF	+INF

IEEE WRAPPED UNDERFLOW MODE DISABLED

X OPERAND	Z RESULT
S	INV, NaN/Q
Q	NaN/Q
–INF	INV, NaN/Q
–NORM	INV, NaN/Q
–DEN	ZR/–0
–0	ZR/–0
+0	ZR/+0
+DEN	ZR/+0
+NORM	+NORM
+INF	+INF

DEC MODE

X OPERAND	Z RESULT
R	INV/R
0	0
+NORM	+NORM
–NORM	INV/R

FLOATING POINT ARITHMETIC INSTRUCTIONS (cont'd)

MNEMONIC	OPCODE	FUNCTION
MULTWX	04H	X • Y
DMULTWX	05H	DP: X • Y
MULTWY	06H	X • |Y|
DMULTWY	07H	DP: X • |Y|

FLAGS AFFECTED

MPY

NY	NX	CRY	DIVZ	DY	DX	NaN	RND	INX	INV	UF	OV	ZR	N
*	*	N/A	0	*	*	*	*	*	*	*	0	*	*

Description: Floating Point Multiplication. The wrapped operand is multiplied by [X, Y].

Comments: The chip must have IEEE Underflow mode set (IU=1), or underflows (wrapped outputs) will be set to zero. The wrapped multiply instructions assume that one of their operands is a wrapped underflow. They do not operate on overflows. The result of WRAPPED X • WRAPPED Y is always too small to make a denormalized number, thus the result always underflows and no instruction is provided. The denormalized number flag (DEN) will not be set if the wrapped number looks like a denormalized number. If the result is too small to return as a wrapped number, then the underflow and zero flags are raised and inexact zero is returned. In the tables that follow, the first entry represents the flag that is set, the second represents the returned result.

IEEE-WRAPPED UNDERFLOW

UNWRAPPED OPERAND	Z RESULT
0	ZR/0
DEN	ZR/0
NORM	NORM ZR [, UF]/[0, E] ZR, UF/0[1]
INF	INF
Q	NaN/Q
S	INV, NaN/Q

Note[1]: Double underflow — the result is too small to return a wrapped result.

IEEE-WRAPPED UNDERFLOW DISABLED

UNWRAPPED OPERAND	Z RESULT
0	ZR/0
DEN	ZR/0
NORM	NORM ZR [, UF]/[0, E]
INF	INF
Q	NaN/Q
S	INV, NaN/Q

MNEMONIC	OPCODE	FUNCTION
MULT	08H	X • Y
DMULT	09H	DP: X • Y
MULTAY	0AH	X • |Y|
DMULTAY	0BH	DP: X • |Y|
MULTAX	0CH	|X| • Y
DMULTAX	0DH	DP: |X| • Y
MULTA	0EH	|X • Y|
DMULTA	0FH	DP: |X • Y|

MPY

NY	NX	CRY	DIVZ	DY	DX	NaN	RND	INX	INV	UF	OV	ZR	N
*	*	N/A	0	*	*	*	*	*	*	*	*	*	*

Description: Floating Point Multiplication. [|X|, |Y|] represents the absolute value of [X, Y]; |X• Y| is the absolute value of the result.

Comments: In the tables that follow, the first entry represents the flag that is set, the second represents the returned result.

IEEE-WRAPPED UNDERFLOW MODE

Y \ X OPERAND	0	DEN	NORM	INF	Q	S
0	ZR/0	ZR/0	ZR/0	INV, NaN/Q	NaN/Q	INV, NaN/Q
DEN	ZR/0	INX, ZR/0	INX, ZR/0	INF	NaN/Q	INV, NaN/Q
NORM	ZR/0	INX, ZR/0	OV/[WRP, INF, M] NORM UF/WRP	INF	NaN/Q	INV, NaN/Q
INF	INV, NaN/Q	INF	INF	INF	NaN/Q	INV, NaN/Q
Q	NaN/Q	NaN/Q	NaN/Q	NaN/Q	NaN/Q	INV, NaN/Q
S	INV, NaN/Q	INV, NaN/Q	INV, NaN/Q	INV, NaN/Q	INV, NaN/Q	INV, NaN/Q

DEC MODE

Y \ X OPERAND	0	NORM	R
0	ZR/0	ZR/0	INV/R
NORM	ZR/0	OV/R NORM ZR, UF/0	INV/R
R	INV/R	INV/R	INV/R

IEEE-WRAPPED UNDERFLOW MODE DISABLED

Y \ X OPERAND	0	DEN	NORM	INF	Q	S
0	ZR/0	ZR/0	ZR/0	INV, NaN/Q	NaN/Q	INV, NaN/Q
DEN	ZR/0	ZR/0	INX, ZR/0	INV, NaN/Q	NaN/Q	INV, NaN/Q
NORM	ZR/0	ZR/0	OV/[WRP, INF, M] NORM UF [,ZR]/0, E]	INF	NaN/Q	INV, NaN/Q
INF	INV, NaN/Q	INV, NaN/Q	INF	INF	NaN/Q	INV, NaN/Q
Q	NaN/Q	NaN/Q	NaN/Q	NaN/Q	NaN/Q	INV, NaN/Q
S	INV, NaN/Q	INV, NaN/Q	INV, NaN/Q	INV, NaN/Q	INV, NaN/Q	INV, NaN/Q

Single Chip Floating Point Processors B2130/B3130/B4130

T-49-12-05

FLOATING POINT ARITHMETIC INSTRUCTIONS (cont'd)

MNEMONIC	OPCODE	FUNCTION	FLAGS AFFECTED
ADD	30H	X + Y	
DADD	31H	DP: X + Y	
SUB	32H	X – Y	
DSUB	33H	DP: X – Y	
SUBX	34H	Y – X	
DSUBX	35H	DP: Y – X	
ADDA	38H	|X| + |Y|	
DADDA	39H	DP: |X| + |Y|	
SUBA	3AH	|X| – |Y|	
DSUBA	3BH	DP: |X| – |Y|	
SUBXA	3CH	|Y| – |X|	
DSUBXA	3DH	DP: |Y| – |X|	

ALU

NY	NX	CRY	DIVZ	DY	DX	NaN	RND	INX	INV	UF	OV	ZR	N
*	*	0	N/A	*	*	*	*	*	*	*	*	*	*

Description: Floating point addition and subtraction. [|X|, |Y|] represents the absolute value of [X,Y].

Comments: When the sum of two operands with opposite signs (or the difference of two operands with like signs) is exactly zero, the result is +0 for all rounding modes except round to minus infinity, in which case, the result is –0. Note that (+0) + (+0) = (+0) – (–0) = +0 and (–0) + (–0) = (–0) - (+0) = –0 for all rounding modes. In the tables that follow, the first entry represents the flag that is set, the second represents the result returned to Z.

IEEE-WRAPPED UNDERFLOW MODE

Y OPERAND \ X OPERAND	0	DEN	NORM	INF	Q	S
0	ZR/0	UF/WRP	NORM	INF	NaN/Q	INV, NaN/Q
DEN	UF/WRP	UF/WRP NORM	OV/[WRP, INF, M] NORM UF/WRP	INF	NaN/Q	INV, NaN/Q
NORM	NORM	OV/[WRP, INF,M] NORM UF/WRP	OV/[WRP, INF, M] NORM UF/WRP	INF	NaN/Q	INV, NaN/Q
INF	INF	INF	INF	INF[1] INV, NaN/Q[1]	NaN/Q	INV, NaN/Q
Q	NaN/Q	NaN/Q	NaN/Q	NaN/Q	NaN/Q	INV, NaN/Q
S	INV, NaN/Q	INV, NaN/Q	INV, NaN/Q	INV, NaN/Q	INV, NaN/Q	INV, NaN/Q

DEC MODE

Y OPERAND \ X OPERAND	0	NORM	R
0	ZR/0	NORM	INV/R
NORM	NORM	OV/R NORM ZR, UF/0	INV/R
R	INV/R	INV/R	INV/R

IEEE-WRAPPED UNDERFLOW MODE DISABLED

Y OPERAND \ X OPERAND	0	DEN	NORM	INF	Q	S
0	ZR/0	ZR/0	NORM	INF	NaN/Q	INV, NaN/Q
DEN	ZR/0	ZR/0	NORM	INF	NaN/Q	INV, NaN/Q
NORM	NORM	NORM	OV/[WRP, INF, M] NORM UF/[0, E]	INF	NaN/Q	INV, NaN/Q
INF	INF	INF	INF	INF[1] INV, NaN/Q[1]	NaN/Q	INV, NaN/Q
Q	NaN/Q	NaN/Q	NaN/Q	NaN/Q	NaN/Q	INV, NaN/Q
S	INV, NaN/Q	INV, NaN/Q	INV, NaN/Q	INV, NaN/Q	INV, NaN/Q	INV, NaN/Q

NOTE 1: (+INF) + (–INF) → NaN
(+INF) + (+INF) → +INF
(+INF) – (+INF) → NaN
(–INF) + (–INF) → –INF
(–INF) – (–INF) → NaN
(+INF) – (–INF) → +INF
(–INF) + (+INF) → NaN
(–INF) – (+INF) → –INF

MNEMONIC	OPCODE	FUNCTION
MIN	24H	MIN[X, Y]
DMIN	25H	DP: MIN[X, Y]
MAX	26H	MAX[X, Y]
DMAX	27H	DP: MAX[X, Y]

ALU

NY	NX	CRY	DIVZ	DY	DX	NaN	RND	INX	INV	UF	OV	ZR	N
*	*	*	N/A	*	*	*	0	0	*	*	0	*	*

Description: These floating point instructions return the smaller of the two operands X and Y (MIN/DMIN) or larger (MAX, DMAX).

Comments: The carry flag is reset if X is returned, otherwise it is set. X is returned if X = Y, except that MAX/DMAX (–0, +0) = +0 and MIN/DMIN (+0, –0) = –0. The Invalid Op flag is set if either operand is a signaling NaN. If either operand is not-a-number, then the result is not-a-number. In IEEE Wrapped Underflow mode, a denormalized result is wrapped and the underflow and inexact flags are set. If either operand is a NaN, CRY is unspecified.

Bipolar Integrated Technology, Inc.

T-49-12-05

FLOATING POINT ARITHMETIC INSTRUCTIONS (cont'd)

MNEMONIC	OPCODE	FUNCTION	FLAGS AFFECTED
ABSX	28H	|X|	
DABSX	29H	DP: |X|	
NEGX	2AH	–X	
DNEGX	2BH	DP: –X	
PASSX	2CH	X	
DPASSX	2DH	DP: X	

ALU

NY	NX	CRY	DIVZ	DY	DX	NaN	RND	INX	INV	UF	OV	ZR	N
0	*	*	N/A	0	*	*	0	0	*	*	0	*	*

Description: These single operand floating point instructions use the X operand input. PASSX/DPASSX returns X through the ALU. If X is denormalized and wrapped underflow mode is reset, zero is returned, otherwise, the result is wrapped and the underflow flag is set.

Comments: PASSX and DPASSX with an infinity input sets CRY, otherwise CRY is reset.

CLRS	18H	Single precision Clear sign
DCLRS	19H	DP: Double precision Clear sign
INVS	1AH	Single precision Invert sign
DINVS	1BH	DP: Double precision Invert sign

ALU

NY	NX	CRY	DIVZ	DY	DX	NaN	RND	INX	INV	UF	OV	ZR	N
0	0	0	N/A	0	0	0	0	0	0	0	0	0	*

Description: Exponent and Mantissa of X are passed. Sign(X) is either cleared or inverted as indicated.

Comments: Input operand values are not checked and exceptions are not generated for these instructions.

CMPR	36H	COMPARE(X, Y)
DCMPR	37H	DP: COMPARE(X, Y)
CMPRA	3EH	COMPARE(|X|, |Y|)
DCMPRA	3FH	DP: COMPARE(|X|, |Y|)

ALU

NY	NX	CRY	DIVZ	DY	DX	NaN	RND	INX	INV	UF	OV	ZR	N
*	*	*	N/A	*	*	*	0	0	*	0	0	*	*

Description: Floating Point compare |X|, |Y| represents the absolute value of X, Y.

Comments: Unordered operands (either X or Y is NaN) set the N, ZR, and NaN flags. This allows the two flags N and ZR to completely specify the result of a compare operation. The following values will be returned to the result based on the relative magnitude of operands X and Y.

Input	Output	Flag			
		ZR	N	CRY	NaN
X > Y	1	0	0	1	0
X < Y	–1	0	1	0	0
X = Y	0	1	0	0	0
[X, Y] NaN	NaN	1	1	0	1

Input		Output	Flag			
X	Y		ZR	N	CRY	NaN
±0	±0	0	1	0	0	0
+INF	+INF	0	1	0	0	0
+INF	–INF	1	0	0	1	0
–INF	+INF	–1	0	1	0	0
–INF	–INF	0	1	0	0	0

MADD	12H	X • Y, X + Y
DMADD	13H	DP: X • Y, X + Y
MSUB	14H	X • Y, X – Y
DMSUB	15H	DP: X • Y, X – Y
MSUBX	16H	X • Y, Y – X
DMSUBX	17H	DP: X • Y, Y – X

MPY

NY	NX	CRY	DIVZ	DY	DX	NaN	RND	INX	INV	UF	OV	ZR	N
*	*	N/A	0	*	*	*	*	*	*	*	*	*	*

ALU

NY	NX	CRY	DIVZ	DY	DX	NaN	RND	INX	INV	UF	OV	ZR	N
*	*	0	N/A	*	*	*	*	*	*	*	*	*	*

Description: Floating Point Multiply/Add(or subtract) instruction. Multiplication is performed by the MPY, whereas addition or subtraction is performed by the ALU.

Comments: See the ADD and MULT instructions for details regarding the flags and operation result.

Single Chip Floating Point Processors

B2130/B3130/B4130

T-49-12-05

FLOATING POINT SUPPORT INSTRUCTIONS

MNEMONIC	OPCODE	FUNCTION	FLAGS AFFECTED
PASSXM	10H	MPYZ ← X	MPY
DPASSXM	11H	DP: MPYZ ← X	

NY	NX	CRY	DIVZ	DY	DX	NaN	RND	INX	INV	UF	OV	ZR	N
0	0	N/A	0	0	0	0	0	0	0	0	*	*	

Description: The X input is returned unmodified through the MPY.
Comments: The flags are reset (except for N and ZR).

SCALE	20H	$X \cdot 2^Y$	ALU
DSCALE	21H	DP: $X \cdot 2^Y$	

NY	NX	CRY	DIVZ	DY	DX	NaN	RND	INX	INV	UF	OV	ZR	N
0	*	0	N/A	0	*	*	0	*	*	*	*	*	*

Description: The floating point input X is multiplied by 2 to the power of integer input Y.
Comments: If X is zero, infinity, or NaN then the output will be zero, infinity, or NaN, respectively. Y is always input as a 32-bit integer, however, the least significant 8 bits (single precision), or 11 bits (double precision) are interpreted as a 2's compliment integer. Other bits of Y are ignored.

IEEE-WRAPPED UNDERFLOW MODE

Y \ X OPERAND	0	DEN	NORM	INF	Q	S
0	ZR/0	ZR, UF, INX/0	X	INF	NaN/Q	INV, NaN/Q
OP <>0	ZR/0	ZR, UF, INX/0	OV/[WRP, INF, M] NORM UF/WRP	INF	NaN/Q	INV, NaN/Q

IEEE-WRAPPED UNDERFLOW MODE DISABLED

Y \ X OPERAND	0	DEN	NORM	INF	Q	S
0	ZR/0	ZR/0	X	INF	NaN/Q	INV, NaN/Q
OP <>0	ZR/0	ZR/0	OV/[WRP, INF, M] NORM UF/[0, E]	INF	NaN/Q	INV, NaN/Q

DEC MODE

Y \ X OPERAND	0	NORM	R
0	ZR/0	X	INV/R
OP <>0	ZR/0	OV/R NORM UF, ZR/0	INV, NaN/R

MERGE	22H	SIGN(X) | EXP(Y) | MANT(X)	ALU
DMERGE	23H	DP: SIGN(X) | EXP(Y) | MANT(X)	

NY	NX	CRY	DIVZ	DY	DX	NaN	RND	INX	INV	UF	OV	ZR	N
0	0	0	N/A	0	0	0	0	0	0	*	*	0	*

Description: The exponent field of Y is concatenated with the sign and mantissa field of X.
Comments: If a NaN or INF results, the overflow flag is set. If a denormalized number or zero results, the underflow flag is set.

NOP	58H	No operation

Description: All registers and flags, except PE, remain unchanged if OFT=0 and EC=0. The result is unspecified.
Comments: Parity is checked during NOP's (and all unimplemented instructions) and the PE flag is updated.

CLRFLAG	59H	ALU/MPY FLAGS ← 0	ALU and MPY

NY	NX	CRY	DIVZ	DY	DX	NaN	RND	INX	INV	UF	OV	ZR	N
0	0	0	0	0	0	0	0	0	0	0	0	0	0

Description: The flag registers are cleared, interrupt enable and mode registers are unaffected.
Comments: If an interrupt has frozen the flag register (see freeze on interrupt mode), CLRFLAG will clear and unfreeze the register. While CLRFLAG affects both ALU and MPY flag registers, it executes only in the MPY. CLRFLAG will override clock enables (AFEN* and MFEN*).

Bipolar Integrated Technology, Inc.

T-49-12-05

FLOATING POINT SUPPORT INSTRUCTIONS (cont'd)

MNEMONIC	OPCODE	FUNCTION	FLAGS AFFECTED
PASSXAR	51H	Pass X, set ALU registers	ALU
PASSXMR	50H	Pass X, set MPY or DIV/SQRT registers	MPY (DIV/SQRT)

ALU

NY	NX	CRY	DIVZ	DY	DX	NaN	RND	INX	INV	UF	OV	ZR	N
*	*	*	N/A	*	*	*	*	*	*	*	*	*	*

MPY (DIV/SQRT)

NY	NX	CRY	DIVZ	DY	DX	NaN	RND	INX	INV	UF	OV	ZR	N
*	*	N/A	*	*	*	*	*	*	*	*	*	*	*

Description: This instruction passes the X operand and uses the Y operand to set the ALU, MPY or DIV/SQRT flags (see bit map, below).

Comments: Used to set or restore registers and flags. If the Bx130 is interrupted, data can be read from T[0..63] and the flag pins (FLAG[0..13]) and stored in system memory. By clocking the Bx130 and writing results and flags to memory, a complete "snapshot" of the Bx130 can be taken. Data can then be loaded back into the Bx130 using PASSXAR and PASSXMR as appropriate. If PLFT=0, two PASSXAR instructions are needed to set the ALU input and pipeline registers, and/or three PASSXMR instructions are needed to set the MPY input and pipeline registers and the DIV/SQRT input register. If PLFT=1, one PASSXAR instruction is needed to set the ALU input register, and/or two PASSXMR instructions are needed to set the MPY and DIV/SQRT input registers. Note that only those flags appearing on FLAG[0..13] can be saved using this process (see Flag Outputs, page 11).

Z[63..0] = X[63..0]	DIVZ = Y[61] (M or D)	RND = Y[57]	OV = Y[53]
NY = Y[63]	DY = Y[60]	INX = Y[56]	ZR = Y[52]
NX = Y[62]	DX = Y[59]	INV = Y[55]	N = Y[51]
CRY = Y[61] (ALU)	NaN = Y[58]	UF = Y[54]	

FREGAR	52H	MPYZ ← AFLAGS
FREGAW	53H	AFLAGS ← X
FREGMR	5AH	MPYZ ← MFLAGS
FREGMW	5BH	MFLAGS ← X
IREGAR	54H	MPYZ ← AINTEN
IREGAW	55H	AINTEN ← X
IREGMR	5CH	MPYZ ← MINTEN
IREGMW	5DH	MINTEN ← X
MREGR	56H	MPYZ ← MODE
MREGW	57H	MODE ← X

Description: Register access instructions. FREGx accesses the flag registers, IREGx accesses the interrupt enable registers and MREGx accesses the mode registers.

Comments: The contents of the multiplier X operand register are used to write status registers. The result of status register read operations appear on the multiplier Z port.

CONVERSION INSTRUCTIONS

MNEMONIC	OPCODE	FUNCTION	FLAGS AFFECTED
SDF	76H	SP → DP	ALU

ALU

NY	NX	CRY	DIVZ	DY	DX	NaN	RND	INX	INV	UF	OV	ZR	N
0	*	0	N/A	0	*	*	0	0	*	0	0	*	*

Description: Floating point precision conversion instruction. Conversion is carried out on the X operand.

Single Chip Floating Point Processors

B2130/B3130/B4130

T-49-12-05

CONVERSION INSTRUCTIONS (cont'd)

MNEMONIC	OPCODE	FUNCTION
DSDF	77H	DP → SP

FLAGS AFFECTED

ALU

NY	NX	CRY	DIVZ	DY	DX	NaN	RND	INX	INV	UF	OV	ZR	N
0	*	0	N/A	0	*	*	*	*	*	*	*	*	*

Description: Floating point precision conversion instruction. Conversion is carried out on the X operand.

Comments: CRY is set if a normalized double precision number is output as a single precision infinity. This will occur if overflows are not wrapped, or if the result is too large to be represented by a wrapped overflow. If a result underflows to the extent that it cannot be wrapped, zero is returned and the ZR, UF and INX flags are set. Refer to ANSI/IEEE STD. 754, section 7.3 (overflows) for additional information.

FFI	7CH	SP → SP format integer
DFFI	7DH	DP → DP format integer
FFIT	7EH	SP → SP format integer (Round to 0)
DFFIT	7FH	DP → DP format integer (Round to 0)

ALU

NY	NX	CRY	DIVZ	DY	DX	NaN	RND	INX	INV	UF	OV	ZR	N
0	*	0	N/A	0	*	*	*	*	*	0	0	*	*

Description: Floating point conversion instruction. The floating point number at the X operand input is rounded to an integral valued floating point number in the same format.

Comments: FFIT/DFFIT always rounds toward zero, regardless of the rounding mode. The INX flag will be set if the result is different from the input. The RND flag will be set if the magnitude of the result is greater than the magnitude of the input.

FCUI	60H	SP → 32-bit unsigned integer
DFCUI	61H	DP → 32-bit unsigned integer
FCSI	62H	SP → 32-bit signed integer
DFCSI	63H	DP → 32-bit signed integer
UICF	64H	SP ← 32-bit unsigned integer
UICDF	65H	DP ← 32-bit unsigned integer
SICF	66H	SP ← 32-bit signed integer
SICDF	67H	DP ← 32-bit signed integer
FCLUI	68H	SP → 64-bit unsigned integer
DFCLUI	69H	DP → 64-bit unsigned integer
FCLSI	6AH	SP → 64-bit signed integer
DFCLSI	6BH	DP → 64-bit signed integer
LUICF	6CH	SP ← 64-bit unsigned integer
LUICDF	6DH	DP ← 64-bit unsigned integer
LSICF	6EH	SP ← 64-bit signed integer
LSICDF	6FH	DP ← 64-bit signed integer
FCUIT	70H	SP → 32-bit unsigned integer (Round to 0)
DFCUIT	71H	DP → 32-bit unsigned integer (Round to 0)
FCSIT	72H	SP → 32-bit signed integer (Round to 0)
DFCSIT	73H	DP → 32-bit signed integer (Round to 0)
FCLUIT	78H	SP → 64-bit unsigned integer (Round to 0)
DFCLUIT	79H	DP → 64-bit unsigned integer (Round to 0)
FCLSIT	7AH	SP → 64-bit signed integer (Round to 0)
DFCLSIT	7BH	DP → 64-bit signed integer (Round to 0)

ALU

NY	NX	CRY	DIVZ	DY	DX	NaN	RND	INX	INV	UF	OV	ZR	N
0	*	0	N/A	0	*	*	*	*	*	0	0	*	*

Description: Floating point to integer and integer to floating point conversion instructions. Input operand X is converted to the indicated format. All instructions follow the programmed rounding mode (see mode register), except the xxxT format instructions which always round toward zero.

Comments: When a floating point to integer conversion instruction overflows, the invalid operation flag is set and the result is either the most positive (for positive overflows) or the most negative (for negative overflows) integer. For example:

Input	2^{65}	$-(2^{65})$
32-bit signed result:	7FFFFFFF	80000000
64-bit signed result:	7FFFFFFFFFFFFFFF	8000000000000000
32-bit unsigned result:	FFFFFFFF	00000000
64-bit unsigned result:	FFFFFFFFFFFFFFFF	0000000000000000

If a NaN is converted from floating point to an integer, the result is an overflow with the sign of the NaN.

Integer to floating point conversion instructions can never set NaN flag.

Bipolar Integrated Technology, Inc.

T-49-12-05

CONVERSION INSTRUCTIONS (cont'd)

MNEMONIC	OPCODE	FUNCTION	FLAGS AFFECTED
WDNM	74H	WRAPPED → DENORM	
DWDNM	75H	DP: WRAPPED → DENORM	

ALU

NY	NX	CRY	DIVZ	DY	DX	NaN	RND	INX	INV	UF	OV	ZR	N
0	0	0	N/A	0	0	0	*	*	0	*	0	*	*

Description: This floating point ALU instruction converts the wrapped X input to a denormalized number. Inexact and rounded-up bits are used as additional inputs.

Comments: The UF flag is set if the result is inexact. This corresponds to the IEEE specification, which says that an underflow shall be signaled if a result is denormalized and inexact. The inexact and rounded-up flags are used as inputs to prevent a double-rounding error. These flags must be set equal to the corresponding flags of the operation that produced the wrapped underflow. The rounding mode must also be the same as when the wrapped underflow was produced. The rounded-up flag is unspecified after this operation. In normal operation, the inexact and rounded-up flags are latched externally and written to the ALU just before the WDNM/DWDNM instruction.

INTEGER ARITHMETIC INSTRUCTIONS

MNEMONIC	OPCODE	FUNCTION	FLAGS AFFECTED
BDIV	89H	8-bit signed X / 8-bit signed Y	
HDIV	8DH	16-bit signed X / 16-bit signed Y	
IDIV	83H	32-bit signed X / 32-bit signed Y	
LIDIV	C3H	64-bit signed X / 64-bit signed Y	
MIDIV	84H	64-bit signed X / 32-bit signed Y (32-bit result.)	

MPY (DIV/SQRT)

NY	NX	CRY	DIVZ	DY	DX	NaN	RND	INX	INV	UF	OV	ZR	N
0	0	0	*	0	0	0	0	0	0	0	*	*	*

Description: Integer division, return quotient. Division by Zero sets the DIVZ flag. BDIV and HDIV results are returned sign-extended. Once a DIV instruction begins, enabling any input (X, Y or I) will restart the instruction with possibly anomalous results. Refer to page 6 for BDIV and HDIV word formats.

BREM	8BH	8-bit signed X / 8-bit signed Y
HREM	8FH	16-bit signed X / 16-bit signed Y
IREM	87H	32-bit signed X / 32-bit signed Y
LIREM	C7H	64-bit signed X / 64-bit signed Y
MIREM	88H	64-bit signed X / 32-bit signed Y (32-bit result.)

MPY (DIV/SQRT)

NY	NX	CRY	DIVZ	DY	DX	NaN	RND	INX	INV	UF	OV	ZR	N
0	0	0	*	0	0	0	0	0	0	0	0	*	*

Description: Integer division, return remainder. If Y is zero, the DIVZ flag is set. BREM and HREM results are returned sign-extended. Once a REM instruction begins, enabling any input (X, Y or I) will restart the instruction with possibly anomalous results. Refer to page 6 for BREM and HREM word formats.

IBITR	F5H	Bit-reverse X

MPY

NY	NX	CRY	DIVZ	DY	DX	NaN	RND	INX	INV	UF	OV	ZR	N
0	0	0	N/A	0	0	0	0	0	0	0	0	*	*

Description: Bit-reverse X.

Comments: The 32-bit X operand is bit-reversed, the Y operand is ignored. The result is a 32-bit unsigned integer. IBITR is executed in the MPY unit.

IMULT	F8H	unsigned X • unsigned Y
IMULTSX	F9H	signed X • unsigned Y
IMULTSY	FAH	unsigned X • signed Y
IMULTS	FBH	signed X • signed Y

MPY

NY	NX	CRY	DIVZ	DY	DX	NaN	RND	INX	INV	UF	OV	ZR	N
0	0	0	N/A	0	0	0	0	0	0	0	0	*	*

Description: Integer multiplication instructions returning a 64-bit result.

Single Chip Floating Point Processors

B2130/B3130/B4130

T-49-12-05

INTEGER ARITHMETIC INSTRUCTIONS (cont'd)

MNEMONIC	OPCODE	FUNCTION	FLAGS AFFECTED
LIMULT	F7H	64-bit signed X • 64-bit signed Y	MPY (DIV/SQRT)

0	0	0	N/A	0	0	0	0	0	0	0	*	*	*
NY	NX	CRY	DIVZ	DY	DX	NaN	RND	INX	INV	UF	OV	ZR	N

Description: Long integer multiply, returning a 64-bit result and overflow flag.
Comments: This multiply operation is performed in the DIV/SQRT unit. Once a LIMULT instruction begins, enabling any input (X, Y or I) will restart the instruction with possibly anomalous results.

BMULT	F4H	8-bit signed X • 8-bit signed Y	MPY
HMULT	F6H	16-bit signed X • 16-bit signed Y	
IMULTH	FCH	unsigned X • unsigned Y	
IMULTHSX	FDH	signed X • unsigned Y	
IMULTHSY	FEH	unsigned X • signed Y	
IMULTHS	FFH	signed X • signed Y	

0	0	0	N/A	0	0	0	0	0	0	0	*	*	*
NY	NX	CRY	DIVZ	DY	DX	NaN	RND	INX	INV	UF	OV	ZR	N

Description: Integer multiplication instructions returning the least significant 32 bits. BMULT and HMULT results are returned zero-extended. Refer to page 6 for BMULTV and HMULT word formats.

IADD	E0H	X + Y
LIADD	A0H	L: X + Y
ISUB	E5H	X – Y
LISUB	A5H	L: X – Y
ISUBX	E6H	Y – X
LISUBX	A6H	L: Y – X
IADDP	E4H	X + Y + 1
LIADDP	A4H	L: X + Y + 1
ISUBM	E1H	X – Y – 1
LISUBM	A1H	L: X – Y – 1
ISUBXM	E2H	Y – X – 1
LISUBXM	A2H	L: Y – X – 1
IADDC	E8H	X + Y + CRY
LIADDC	A8H	L: X + Y + CRY
ISUBC	E9H	X – Y – CRY*
LISUBC	A9H	L: X – Y – CRY*
ISUBXC	EAH	Y – X – CRY*
LISUBXC	AAH	L: Y – X – CRY*

ALU

0	0	*	N/A	0	0	0	0	0	0	0	*	*	*
NY	NX	CRY	DIVZ	DY	DX	NaN	RND	INX	INV	UF	OV	ZR	N

Description: Integer addition and subtraction instructions.
Comments: The function of the carry input used for ISUBC, LISUBC, ISUBXC, LISUBXC can be changed with mode register BM (Borrow Mode, bit–7). See mode register section for additional information.

ISMIN	C6H	signed MIN[X, Y]
LISMIN	86H	L: signed MIN[X, Y]
ISMAX	C2H	signed MAX[X, Y]
LISMAX	82H	L: signed MAX[X, Y]
IUMIN	CEH	unsigned MIN[X, Y]
LIUMIN	8EH	L: unsigned MIN[X, Y]
IUMAX	CAH	unsigned MAX[X, Y]
LIUMAX	8AH	L: unsigned MAX[X, Y]

ALU

0	0	*	N/A	0	0	0	0	0	0	0	0	*	*
NY	NX	CRY	DIVZ	DY	DX	NaN	RND	INX	INV	UF	OV	ZR	N

Description: The larger of the two operands X and Y is returned (ISMAX/LISMAX, IUMAX/LIUMAX) or the smaller of the two operands is returned (ISMIN/LISMIN, IUMIN/LIUMIN).
Comments: Sign and zero flags are set based on returned result. The carry flag is reset if X is returned, otherwise it is set. X is returned if X = Y.

Bipolar Integrated Technology, Inc.

T-49-12-05

INTEGER ARITHMETIC INSTRUCTIONS (cont'd)

MNEMONIC	OPCODE	FUNCTION	FLAGS AFFECTED
ISCMPR	C1H	signed Compare(X,Y)	
LISCMPR	81H	L: signed Compare(X,Y)	
IUCMPR	C0H	unsigned Compare(X,Y)	
LIUCMPR	80H	L: unsigned Compare(X,Y)	

ALU

0	0	*	N/A	0	0	0	0	0	0	0	0	*	*
NY	NX	CRY	DIVZ	DY	DX	NaN	RND	INX	INV	UF	OV	ZR	N

Description: Signed and unsigned integer instructions compare X and Y, and return the following values based on the relative magnitude of operands X and Y.

Input	Output	Flag ZR	Flag N	Flag CRY
X > Y	1	0	0	1
X < Y	−1	0	1	0
X = Y	0	1	0	0

INEGC	EBH	−X − CRY*	
LINEGC	ABH	L: −X − CRY*	
IABSX	EFH	|X|	
LIABSX	AFH	L: |X|	
INEGX	E7H	−X	
LINEGX	A7H	L: −X	

ALU

0	0	*	N/A	0	0	0	0	0	0	0	*	*	*
NY	NX	CRY	DIVZ	DY	DX	NaN	RND	INX	INV	UF	OV	ZR	N

Description: Single operand integer arithmetic instructions.

Comments: The function of the carry input used for INEGC and LINEGC can be changed with mode register BM (Borrow Mode, bit−7). See ADD with Carry (page 5), and mode register section for additional information.

INTEGER BOOLEAN INSTRUCTIONS

MNEMONIC	OPCODE	FUNCTION	FLAGS AFFECTED
LSS	F0H	Logical shift X w/sb	
LLSS	B0H	L: Logical shift X w/sb	
LS	F1H	Logical shift X	
LLS	B1H	L: Logical shift X	

ALU

0	0	*	N/A	0	0	0	0	0	0	0	0	*	*
NY	NX	CRY	DIVZ	DY	DX	NaN	RND	INX	INV	UF	OV	ZR	N

Description: Integer shift instructions. The low 7 bits (sign extended) of the Y-operand ({n}) determines the shift. If {n} > 0, then shift left by {n}; if {n} < 0, then shift right by −{n}. If {n} = 0, then no shift is performed. For LSS/LLSS right shifts, all bits of the result shifted out are or'd with the least significant bit position of the result (i.e., sticky bit). Zeros are shifted in during right and left shifts.

Comments: Carry bit receives the last bit shifted out of the operand if {n} ≠ 0. Flags are set based on a signed operand. The overflow flag is reset. Carry is reset if {n} = 0. Y is a 32-bit integer.

AS	F2H	Arithmetic shift X	
LAS	B2H	L: Arithmetic shift X	

ALU

0	0	*	N/A	0	0	0	0	0	0	0	*	*	*
NY	NX	CRY	DIVZ	DY	DX	NaN	RND	INX	INV	UF	OV	ZR	N

Description: Arithmetic shift instruction. The low 7 bits (sign extended) of the Y-operand ({n}) determines the shift. If {n} > 0, then shift left by {n}; if {n} < 0, then shift right by −{n}. If {n} = 0, no shift is performed. For left shifts, zeros are shifted in. For right shifts, AS/LAS shifts in copies of the sign bit.

Comments: Carry bit receives the last bit shifted out of the operand if {n} ≠ 0. Carry is reset if {n} = 0. Flags are set based on a signed operand. Left AS/LAS shifts set the OV flag if any bits shifted out differ from the result sign. Y is a 32-bit integer.

Single Chip Floating Point Processors

B2130/B3130/B4130

T-49-12-05

INTEGER BOOLEAN INSTRUCTIONS (cont'd)

MNEMONIC	OPCODE	FUNCTION
INAND	D0H	X* or Y*
LINAND	90H	L: X* or Y*
IORNX	D1H	X* or Y
LIORNX	91H	L: X* or Y
IORNY	D2H	X or Y*
LIORNY	92H	L: X or Y*
IOR	D3H	X or Y
LIOR	93H	L: X or Y
IANDNY	D4H	X and Y*
LIANDNY	94H	L: X and Y*
IAND	D5H	X and Y
LIAND	95H	L: X and Y
INOR	D6H	X* and Y*
LINOR	96H	L: X* and Y*
IANDNX	D7H	X* and Y
LIANDNX	97H	L: X* and Y
IXNOR	DEH	X xnor Y
LIXNOR	9EH	L: X xnor Y
IXOR	DFH	X xor Y
LIXOR	9FH	L: X xor Y
ISET	D8H	Z = all ones
LISET	98H	L: Z = all ones
INOTX	D9H	Z = X*
LINOTX	99H	L: Z = X*
IPASSY	DAH	Z = Y
LIPASSY	9AH	L: Z = Y
IPASSX	DBH	Z = X
LIPASSX	9BH	L: Z = X
ICLR	DCH	Z = all zeros
LICLR	9CH	L: Z = all zeros
INOTY	DDH	Z = Y*
LINOTY	9DH	L: Z = Y*

FLAGS AFFECTED

ALU

0	0	0	N/A	0	0	0	0	0	0	0	0	*	*
NY	NX	CRY	DIVZ	DY	DX	NaN	RND	INX	INV	UF	OV	ZR	N

Description: Boolean logic instructions.
Comments: Flags are set based on signed operands.

UNUSED OPCODES

Note: The following opcodes should not be used. Flag and data results are undefined. BIT maintains the right to use these opcodes in future products or upgrades.

1CH → 1FH
2EH, 2FH
40H → 4FH
5EH, 5FH
85H, 8CH
A3H, ACH → AEH
B3H → BFH
C4H, C5H, C8H, C9H, CBH → CDH, CFH
E3H, ECH → EEH
F3H

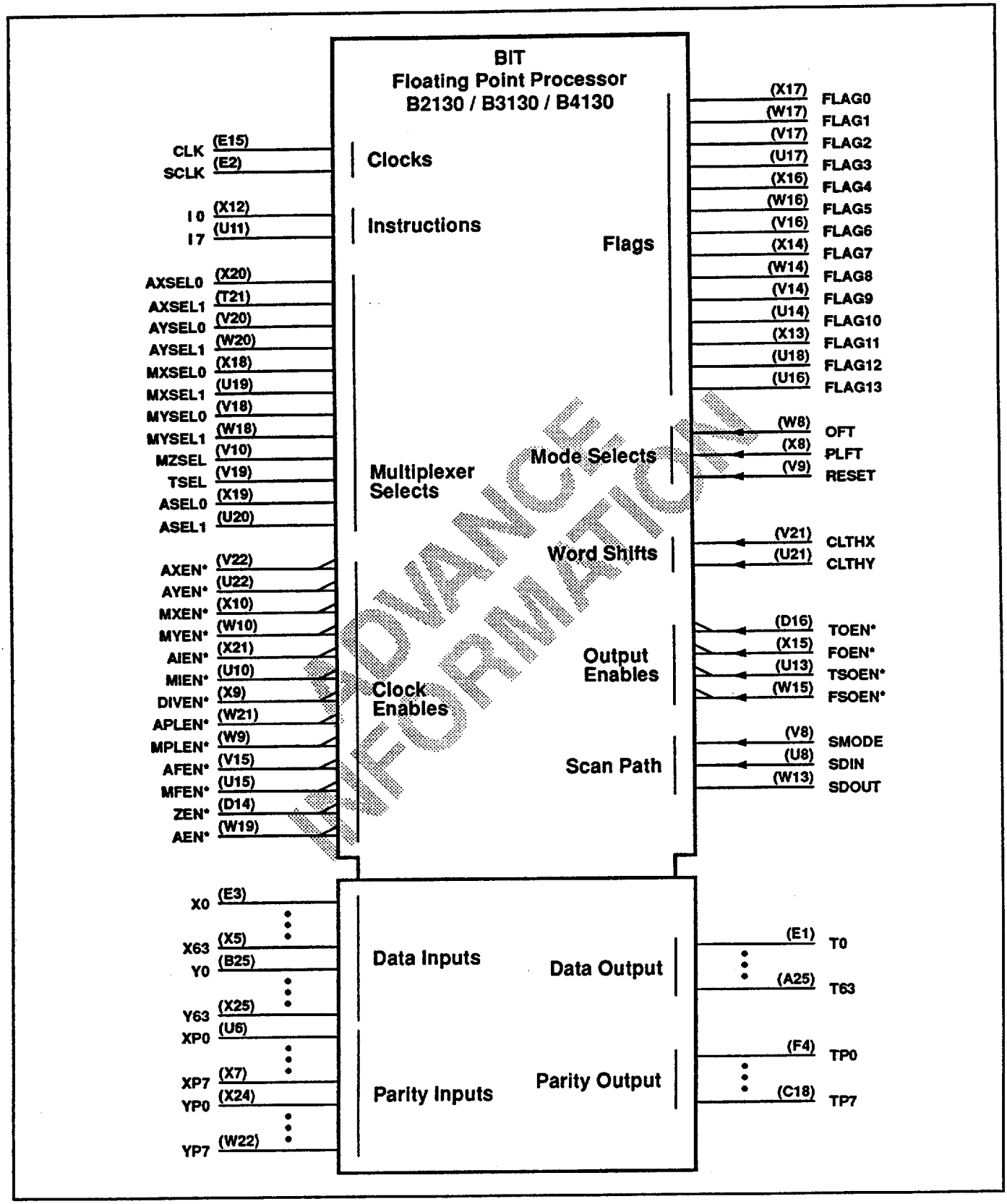


Figure 2 — Pin Assignments

Single Chip
Floating Point Processors

B2130/B3130/B4130

T-49-12-05

Signal Description

DATA

X[63..0] 64-bit X input port.

XP[7..0] Byte parity bits corresponding to the X input port. (XP[0] corresponds to byte-0 of the X input port (X[7..0]), XP[1] to byte-1 (X[15..8]), etc.).

Y[63..0] 64-bit Y input port.

YP[7..0] Byte parity bits corresponding to the Y input port. (YP[0] corresponds to byte-0 of the Y input port (Y[7..0]), YP[1] to byte-1 (Y[15..8]), etc.).

T[63..0] 64-bit T output port. Single precision results are output on both the LSW and the MSW of T.

TP[7..0] Byte parity bits corresponding to the T output port. (TP[0] corresponds to byte-0 of the T output port (T[7..0]), TP[1] to byte-1 (T[15..8]), etc.).

CONTROL

I [7..0] 8-bit instruction bus. Determines the instruction executed by the internal computational elements.

FLAG [13..0] Flag output pins. Flags are output in accordance with FLAG OUTPUTS (see page 11).

OFT Output register mode select. Configures the Z, control and status registers as positive edge triggered registers when low. Bypasses these registers as flowthrough when high. ZEN*, AFEN*, and MFEN* are ignored when OFT is high.

PLFT Pipeline register mode select. Configures the ALU and MPY elements as two stage pipelines when low. Configures these elements as flowthrough when high. APLEN*, and MPLEN* are ignored when PLFT is high.

RESET Reset pin to the device. Resets MODE, STATUS, and INTEN registers to "0" when asserted (high).

AXSEL[1..0] Multiplexer select for ALU input operand X. Selects between one of four sources as follows:
00 - X port
01 - A register
10 - ALU feedback
11 - MPY feedback

AYSEL[1..0] Multiplexer select for ALU input operand Y. Selects between one of four sources as follows:
00 - Y port
01 - A register
10 - ALU feedback
11 - MPY feedback

MXSEL[1..0] Multiplexer select for MPY input operand X. Selects between one of four sources as follows:
00 - X port
01 - A register
10 - ALU feedback
11 - MPY feedback

MYSEL[1..0] Multiplexer select for MPY input operand Y. Selects between one of four sources as follows:
00 - Y port
01 - A register
10 - ALU feedback
11 - MPY feedback

MZSEL Multiplexer select for MPY or DIV/SQRT output operand Z. Selects between the two as follows:
0 - MPY output
1 - DIV/SQRT output

TSEL Multiplexer select for output operand T. Selects between one of two sources as follows:
0 - ALU output Z
1 - MZSEL selected output

ASEL[1..0] Multiplexer select for A register input. Selects between one of four sources as follows:
00 - X port
01 - Y port
10 - ALU feedback
11 - MPY feedback

Word Shift

CLTHX When high, copy the X operand LSW to its MSW. Internally the MSW is used for single precision operations. Must be zero for double precision operations.

CLTHY When high, copy the Y operand LSW to its MSW. Internally the MSW is used for single precision operations. Must be zero for double precision operations.

CLOCK ENABLES

AXEN*, AYEN* Active low enables for ALU operand registers. AXEN* enables the ALU X operand register and AYEN* enables Y.

MXEN*, MYEN* Active low enables for MPY and DIV/SQRT operand registers. MXEN* enables the MPY X operand register and MYEN* enables Y.

AIEN*, MIEN* Active low enables for the instruction registers. AIEN* enables the ALU instruction register, and MIEN* enables the MPY and DIV/SQRT instruction register.

Signal Description (cont'd)

DIVEN* When DIVEN* is low, MXEN*, MYEN* and MIEN* are used to individually enable the DIV/SQRT X and Y operand and instruction registers, while the MPY(X, Y and Z) registers are disabled. When DIVEN* is high, the DIV/SQRT(X,Y and Z) registers are disabled and MXEN*, MYEN* and MIEN* enable the MPY registers.

APLEN*, MPLEN* Active low enables for the ALU and MPY pipeline registers. APLEN* enables the ALU pipe, and MPLEN* enables the MPY.

AFEN*, MFEN* Active low enables for the ALU and MPY flag registers. AFEN* enables the ALU flag register, and MFEN* enables the MPY. Useful for state restoration.

ZEN* Active low enable for the Z output register.

AEN* Active low enable for the A register.

OUTPUT ENABLES

TOEN* Active low output enable for the T output port.

FOEN* Active low output enable for the flag port.

TSOEN* Synchronous active low output enable for the T output port.

FSOEN* Synchronous active low output enable for the flag port.

CLOCKS

CLK Master clock for the device. All internal registers are clocked with this input. Rising edge triggered.

SCLK Scan Clock. Used as the clock input, to shift serial data in the scan path, when SMODE=1. (CLK is used as the clock input when SMODE=0.)

SCAN PATH

SMODE Configures the on-chip registers in the scan path as a long, continous serial shift register when high.

SDIN Input port to the scan path.

SDOUT Output port from scan path.

POWER

VCC Positive supply voltage.

VEE Negative supply voltage.

Single Chip
Floating Point Processors

B2130/B3130/B4130

T-49-12-05

Electrical Specifications

NOTE: While interface signals for the B4130 are compatible with 100K ECL standards, it requires a Vee of –5.0Vdc.

Table 1 — Absolute Maximum Ratings

Note: Permanent damage may occur if any one absolute maximum rating is exceeded. Functional operation is not implied and device reliability may be impaired by exposure to higher than recommended conditions.

Parameter B2130	Symbol	Value		Units
		With Heatsink	Without Heatsink	
Supply Voltage (V_{ee} = 0)	V_{cc}	–0.5 to +7.0	–0.5 to +7.0	V_{dc}
Input Voltage (V_{ee} = 0)	V_{in}	–0.5 to V_{cc} +0.5	–0.5 to V_{cc} +0.5	V_{dc}
Storage Temperature	T_{st}	–55 to +125	–55 to +150	°C
Junction Temperature	T_j	135	225	°C

Parameter B3130 / B4130	Symbol	Value		Units
		With Heatsink	Without Heatsink	
Supply Voltage	V_{ee}	–7.0 to +0.5	–7.0 to +0.5	V_{dc}
Input Voltage	V_{in}	V_{ee} to +0.5	V_{ee} to +0.5	V_{dc}
Output Source Current				
Continuous (50 Ω output)	I_{oc}	30	30	mA_{dc}
Storage Temperature	T_{st}	–55 to +125	–55 to +150	°C
Junction Temperature	T_j	135	225	°C

Table 2 — Recommended Operating and Test Conditions

Note: The following environmental conditions pertain to guaranteed DC and Switching characteristics for chips soldered into circuit boards. These conditions should not be exceeded during normal operation.

Parameter (B2130)	Symbol	Min	Nom	Max	Units
Supply Voltage (V_{ee} = 0)	V_{cc}	4.75	5.0	5.25	V_{dc}
Operating Junction Temperature	T_{jo}*	20		125	°C
Theta, junction to ambient	θ_{ja}**		2.1		°C/Watt
Parameter (B3130)	**Symbol**	**Min**	**Nom**	**Max**	**Units**
Supply Voltage (V_{cc} = 0)	V_{ee}	–5.46	–5.20	–4.75	V_{dc}
Operating Junction Temperature	T_{jo}*	20		125	°C
Output Termination to –2.0 V_{dc}	R_t		50		Ω
Theta, junction to ambient	θ_{ja}**		2.1		°C/Watt
Parameter (B4130)	**Symbol**	**Min**	**Nom**	**Max**	**Units**
Supply Voltage (V_{cc} = 0)	V_{ee}	–5.46	–5.0	–4.75	V_{dc}
Operating Junction Temperature	T_{jo}*	20		125	°C
Output Termination to –2.0 V_{dc}	R_t		50		Ω
Theta, junction to ambient	θ_{ja}**		2.1		°C/Watt

* Worst case junction temperature specified for worst case Supply Current (I_{ee} for ECL and I_{cc} for TTL).

** This assumes an airflow of 500 linear feet per minute across the standard BIT Bx130 PinFin heatsink.

Single Chip Floating Point Processors

B2130/B3130/B4130

T-49-12-05

Table 3 — DC Characteristics

Parameter (B2130)	Symbol	Min		Max	Units
Input Voltage High	V_{ih}	2.0			V_{dc}
Input Voltage Low	V_{il}			0.8	V_{dc}
Output Voltage High	V_{oh}	2.4			V_{dc}
Output Voltage Low	V_{ol}			0.5	V_{dc}

Parameter (B3130)	Symbol	T_{jc} min		T_{jc} nom		T_{jc} max		Units
		Min	Max	Min	Max	Min	Max	
Input Voltage High	V_{ih}	–1.17		–1.13		–1.07		V_{dc}
Input Voltage Low	V_{il}		–1.48		–1.48		–1.45	V_{dc}
Output Voltage High	V_{oh}	–1.02		–0.98		–0.92		V_{dc}
Output Voltage Low	V_{ol}		–1.63		–1.63		–1.60	V_{dc}

Parameter (B4130)	Symbol	Min		Max	Units
Input Voltage High	V_{ih}	–1.165		–0.880	V_{dc}
Input Voltage Low	V_{il}	–1.810		–1.475	V_{dc}
Output Voltage High	V_{oh}	–1.025		–0.880	V_{dc}
Output Voltage Low	V_{ol}	–1.810		–1.620	V_{dc}

Parameter	Symbol	–10	–12	Units
Max Supply Current, B2130	I_{cc}	5.4	4.6	A_{dc}
Max Supply Current, B3130 / B4130	I_{ee}	5.5	4.9	A_{dc}
Max Input Current High, B2130	I_{ih}	0.1	0.1	mA_{dc}
Max Input Current High, B3130 / B4130	I_{ih}	0.3	0.3	mA_{dc}
Max Output Current High, B2130	I_{oh}	–2.0	–2.0	mA_{dc}
Max Output Current Low, B2130	I_{ol}	8.0	8.0	mA_{dc}

Table 4 — Switching Characteristics

Parameter	Symbol	Min				Max		Units
		TTL –10	TTL –12	ECL –10	ECL –12	TTL	ECL	
Setup–hold Time								
Input Register Setup	t_{rs}	3.0	3.0	2.5	2.5			ns
Input Register Hold	t_{rh}	2.5	2.5	1.5	2.0			ns
XSEL/YSEL to CLK Setup	t_{xyss}	3.0	3.0	2.5	2.5			ns
XSEL/YSEL to CLK Hold	t_{xysh}	2.5	2.5	1.5	2.0			ns
TSEL to CLK Setup	t_{tss}	4.5	4.5	4.0	4.0			ns
TSEL to CLK Hold	t_{tsh}	2.5	2.5	1.0	1.0			ns
MZSEL to CLK Setup	t_{mzss}	4.5	4.5	4.0	4.0			ns
MZSEL to CLK Hold	t_{mzsh}	2.5	2.5	1.0	1.0			ns
XEN*/YEN* to CLK Setup	t_{xyes}	3.0	3.0	2.5	2.5			ns
XEN*/YEN* to CLK Hold	t_{xyeh}	2.5	2.5	1.5	2.0			ns
Output Delays								
Register to Register Mode								
Z Register to Data (T)	t_{rod}	2.0	2.0	1.0	1.0	12.0	7.0	ns
Z Register to Parity (TP)	t_{prod}	2.0	2.0	1.0	1.0	12.0	7.0	ns
Flag Register to FLAG[0..12]	t_{frod}	2.0	2.0	1.0	1.0	12.0	7.0	ns
Flag Register to INT (FLAG[13])	t_{frod}	2.0	2.0	1.0	1.0	13.0	8.0	ns

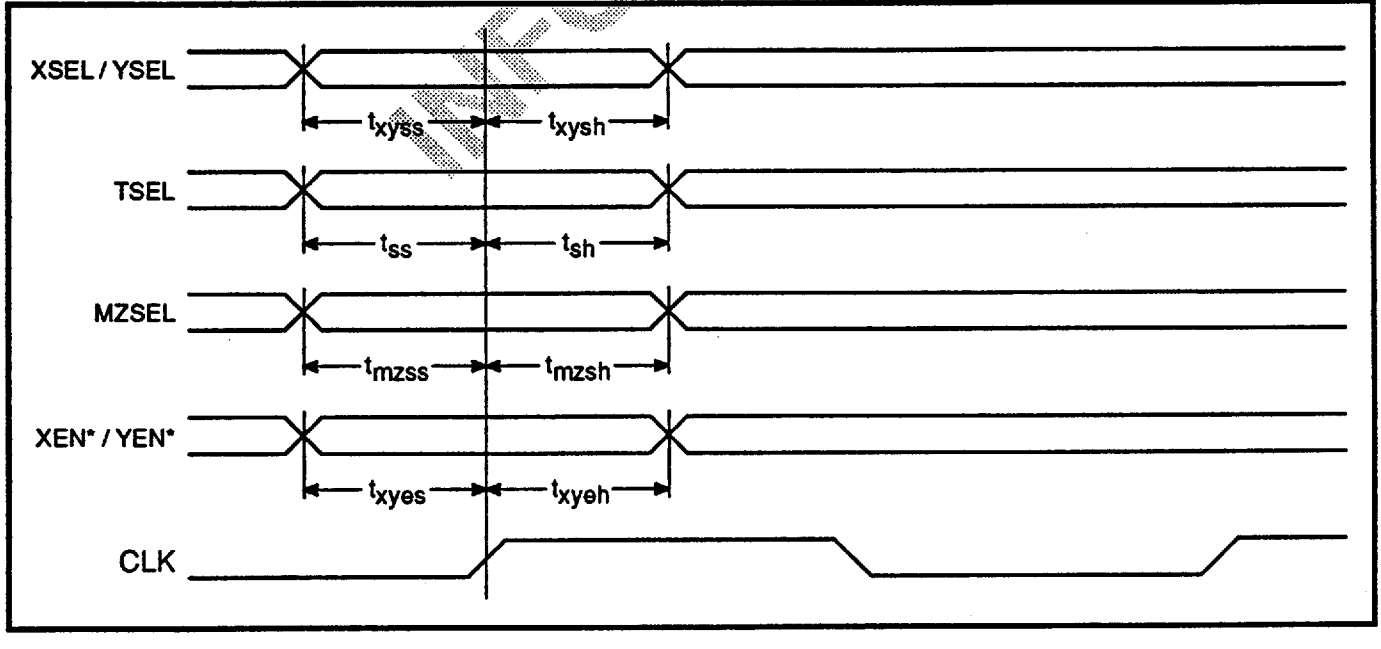


Figure 3 — Timing Diagrams: Setup and Hold Times

Table 5 — Register to Register Mode Operate Times

Parameter	Symbol	Max (–10)	Max (–12)	Units
XREG/YREG to PREG (OPp) Pipeline Mode				
SP/DP FALU		10.0	12.5	ns
IALU		10.0	12.5	ns
SP/DP FMPY		10.0	12.5	ns
IMPY (32)		10.0	12.5	ns
XREG/YREG to ZREG (OP) Non-Pipeline Mode (Latency, or t_{rr})				
DP FALU Operate		20.0	25.0	ns
IALU Operate		20.0	25.0	ns
DP FMPY Operate		20.0	25.0	ns
IMPY (32/32) Operate		20.0	25.0	ns
PASSXMR (in DIV/SQRT Unit)		20.0	25.0	ns
SP FDIV Operate		150.0	150.0	ns
DP FDIV Operate		250.0	250.0	ns
SP SQRT Operate		250.0	250.0	ns
DP SQRT Operate		450.0	450.0	ns
IDIV (32/32) Operate		300.0	300.0	ns
IDIV (64/64) Operate		525.0	525.0	ns
IDIV (64/32) Operate		525.0	525.0	ns
IMPY (64/64) Operate		300.0	300.0	ns

Table 6 — Register to Flowthrough Mode Operate Times

Parameter	Symbol	Max	Units
Pipeline Mode			
PREG Register to Data (T)	t_{rpd}	OPp + t_{rod}	ns
PREG Register to Parity (TP)	t_{prpd}	OPp + t_{prod}	ns
PREG Register to FLAG[0..12]	t_{frpd}	OPp + t_{frod}	ns
PREG Register to INT (FLAG[13])	t_{irpd}	OPp + t_{irod}	ns
Non-Pipeline Mode			
X/Y Register to Data (T)	t_{rd}	OP + t_{rod}	ns
X/Y Register to Parity (TP)	t_{prd}	OP + t_{prod}	ns
X/Y Register to FLAG[0..12]	t_{frd}	OP + t_{frod}	ns
X/Y Register to INT (FLAG[13])	t_{ird}	OP + t_{irod}	ns

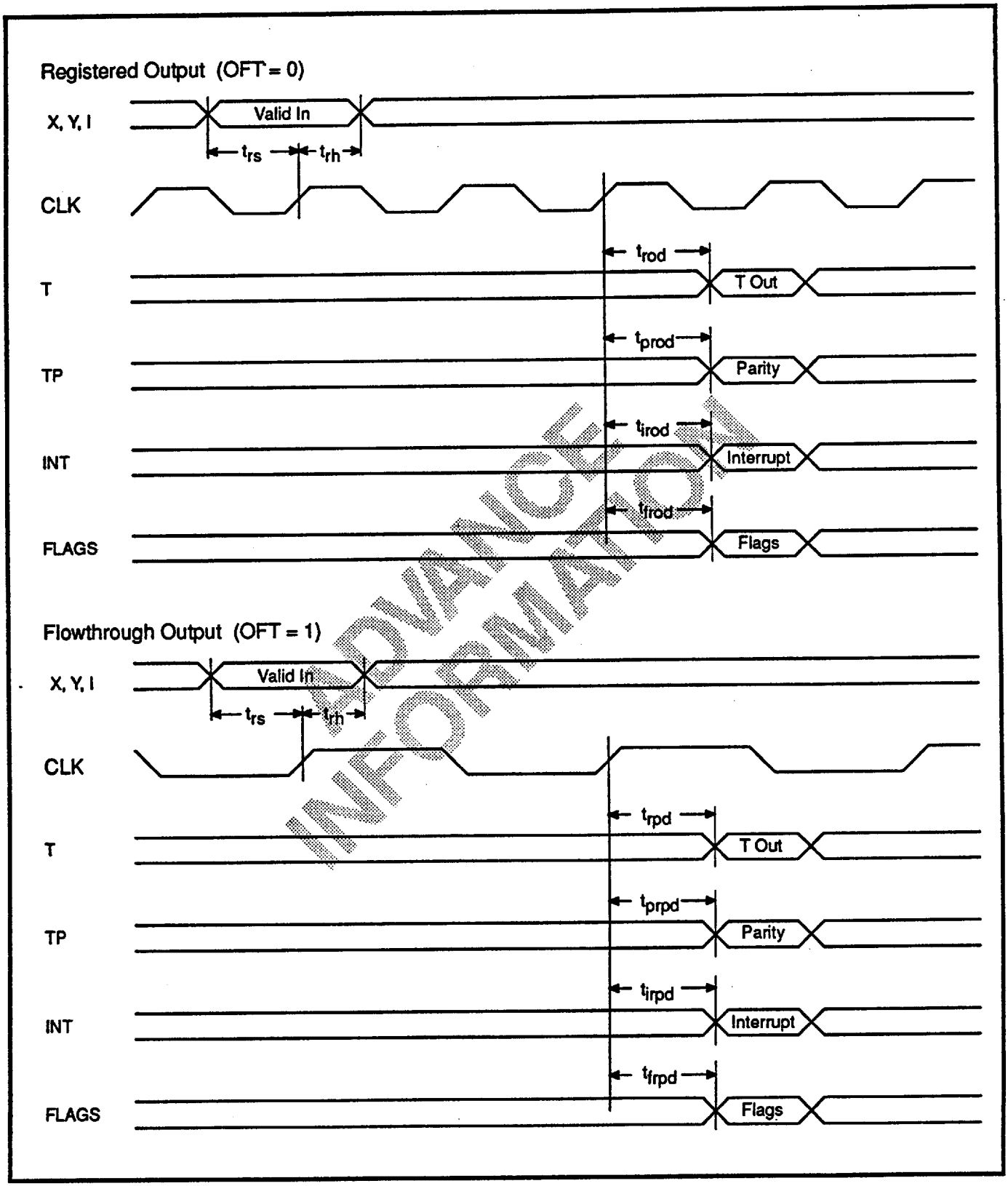


Figure 4 — Timing Diagrams: Pipelined Operation (PLFT = 0)

Single Chip Floating Point Processors

B2130/B3130/B4130

T-49-12-05

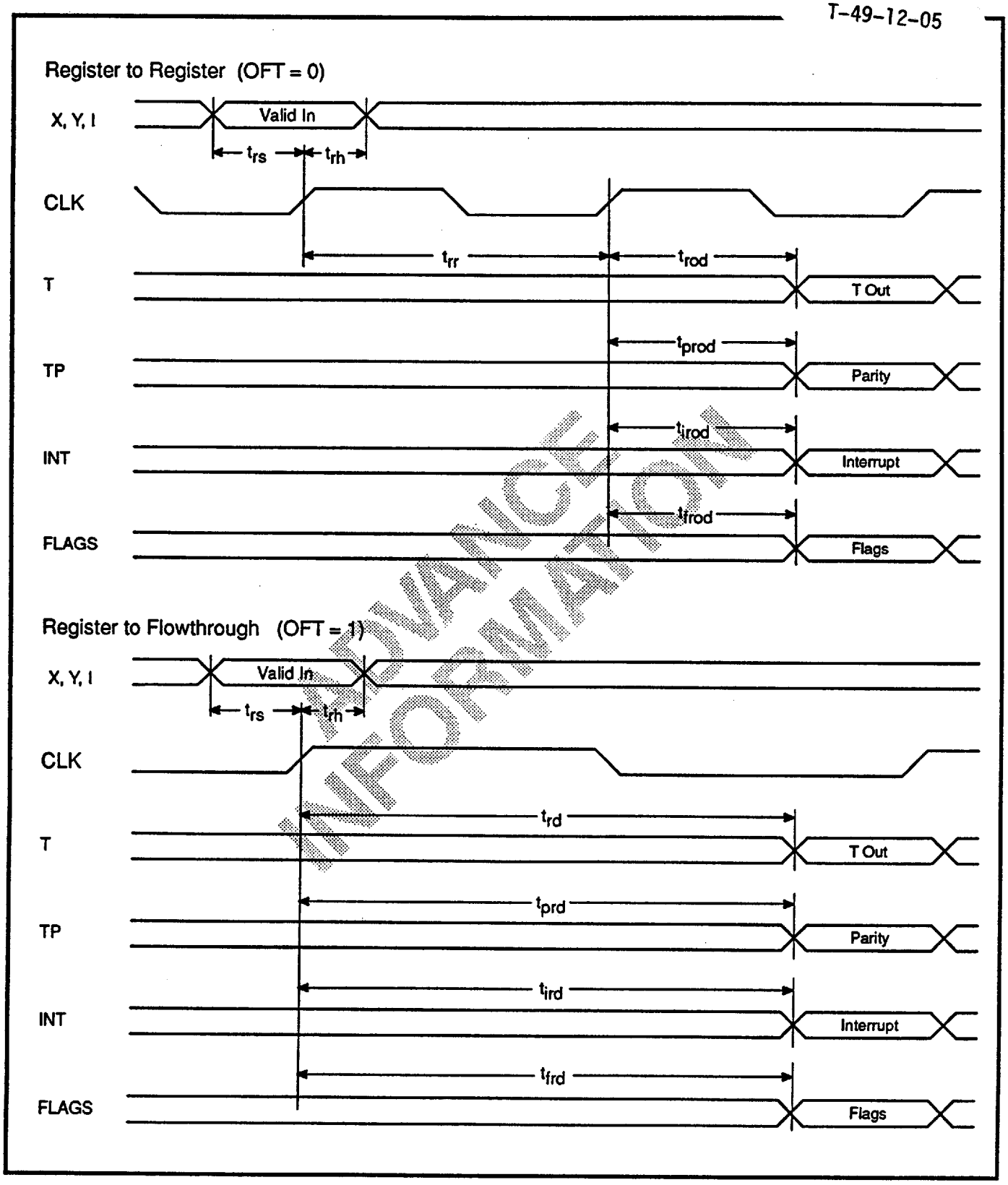
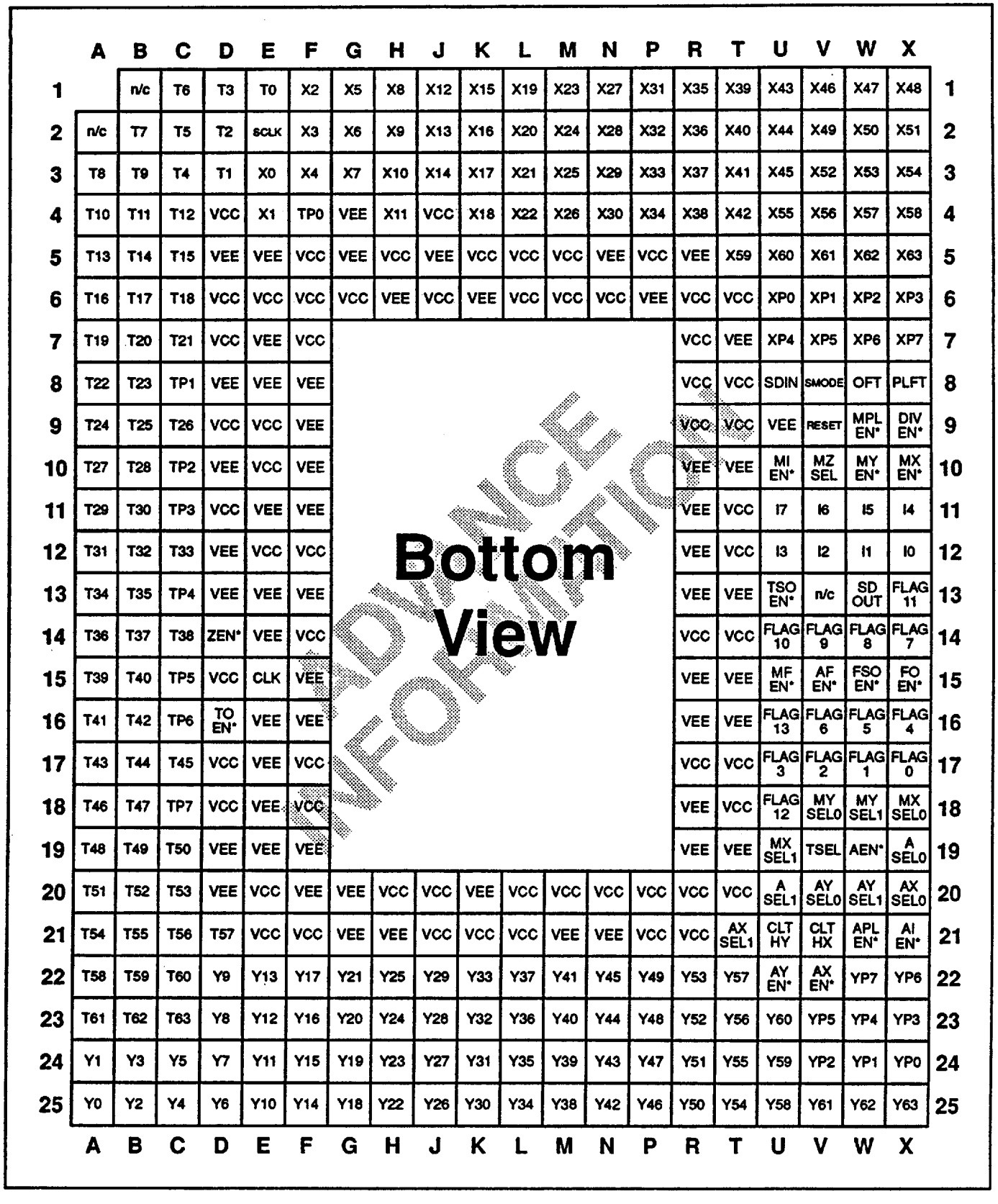


Figure 5 — Timing Diagrams: Non-Pipelined Operation (PLFT = 1)



	A	B	C	D	E	F	G	H	J	K	L	M	N	P	R	T	U	V	W	X	
1		n/c	T6	T3	T0	X2	X5	X8	X12	X15	X19	X23	X27	X31	X35	X39	X43	X46	X47	X48	1
2	n/c	T7	T5	T2	SCLK	X3	X6	X9	X13	X16	X20	X24	X28	X32	X36	X40	X44	X49	X50	X51	2
3	T8	T9	T4	T1	X0	X4	X7	X10	X14	X17	X21	X25	X29	X33	X37	X41	X45	X52	X53	X54	3
4	T10	T11	T12	VCC	X1	TP0	VEE	X11	VCC	X18	X22	X26	X30	X34	X38	X42	X55	X56	X57	X58	4
5	T13	T14	T15	VEE	VEE	VCC	VEE	VCC	VEE	VCC	VCC	VCC	VEE	VCC	VEE	X59	X60	X61	X62	X63	5
6	T16	T17	T18	VCC	VCC	VCC	VCC	VEE	VCC	VEE	VCC	VCC	VCC	VEE	VCC	VCC	XP0	XP1	XP2	XP3	6
7	T19	T20	T21	VCC	VEE	VCC									VCC	VEE	XP4	XP5	XP6	XP7	7
8	T22	T23	TP1	VEE	VEE	VEE									VCC	VCC	SDIN	SMODE	OFT	PLFT	8
9	T24	T25	T26	VCC	VCC	VEE									VCC	VCC	VEE	RESET	MPL EN*	DIV EN*	9
10	T27	T28	TP2	VEE	VCC	VEE									VEE	VEE	MI EN*	MZ SEL	MY EN*	MX EN*	10
11	T29	T30	TP3	VCC	VEE	VEE									VEE	VCC	I7	I6	I5	I4	11
12	T31	T32	T33	VEE	VCC	VCC									VEE	VCC	I3	I2	I1	I0	12
13	T34	T35	TP4	VEE	VEE	VEE									VEE	VEE	TSO EN*	n/c	SD OUT	FLAG 11	13
14	T36	T37	T38	ZEN*	VEE	VCC									VCC	VCC	FLAG 10	FLAG 9	FLAG 8	FLAG 7	14
15	T39	T40	TP5	VCC	CLK	VEE									VEE	VEE	MF EN*	AF EN*	FSO EN*	FO EN*	15
16	T41	T42	TP6	TO EN*	VEE	VEE									VEE	VEE	FLAG 13	FLAG 6	FLAG 5	FLAG 4	16
17	T43	T44	T45	VCC	VEE	VCC									VCC	VCC	FLAG 3	FLAG 2	FLAG 1	FLAG 0	17
18	T46	T47	TP7	VCC	VEE	VCC									VEE	VCC	FLAG 12	MY SEL0	MY SEL1	MX SEL0	18
19	T48	T49	T50	VEE	VEE	VEE									VEE	VEE	MX SEL1	TSEL	AEN*	A SEL0	19
20	T51	T52	T53	VEE	VCC	VEE	VEE	VCC	VCC	VEE	VCC	VCC	VCC	VCC	VCC	VCC	A SEL1	AY SEL0	AY SEL1	AX SEL0	20
21	T54	T55	T56	T57	VCC	VCC	VEE	VEE	VCC	VCC	VCC	VEE	VEE	VCC	VCC	AX SEL1	CLT HY	CLT HX	APL EN*	AI EN*	21
22	T58	T59	T60	Y9	Y13	Y17	Y21	Y25	Y29	Y33	Y37	Y41	Y45	Y49	Y53	Y57	AY EN*	AX EN*	YP7	YP6	22
23	T61	T62	T63	Y8	Y12	Y16	Y20	Y24	Y28	Y32	Y36	Y40	Y44	Y48	Y52	Y56	Y60	YP5	YP4	YP3	23
24	Y1	Y3	Y5	Y7	Y11	Y15	Y19	Y23	Y27	Y31	Y35	Y39	Y43	Y47	Y51	Y55	Y59	YP2	YP1	YP0	24
25	Y0	Y2	Y4	Y6	Y10	Y14	Y18	Y22	Y26	Y30	Y34	Y38	Y42	Y46	Y50	Y54	Y58	Y61	Y62	Y63	25
	A	B	C	D	E	F	G	H	J	K	L	M	N	P	R	T	U	V	W	X	

Figure 6 — B2130 Pin Out

Single Chip Floating Point Processors

B2130/B3130/B4130

T-49-12-05

	A	B	C	D	E	F	G	H	J	K	L	M	N	P	R	T	U	V	W	X	
1		n/c	T6	T3	T0	X2	X5	X8	X12	X15	X19	X23	X27	X31	X35	X39	X43	X46	X47	X48	1
2	n/c	T7	T5	T2	SCLK	X3	X6	X9	X13	X16	X20	X24	X28	X32	X36	X40	X44	X49	X50	X51	2
3	T8	T9	T4	T1	X0	X4	X7	X10	X14	X17	X21	X25	X29	X33	X37	X41	X45	X52	X53	X54	3
4	T10	T11	T12	VCC	X1	TP0	VCC	X11	VCC	X18	X22	X26	X30	X34	X38	X42	X55	X56	X57	X58	4
5	T13	T14	T15	VCC	VCC	VCC	VCC	VCC	VCC	VCC	VCC	VCC	VCC	VCC	VCC	X59	X60	X61	X62	X63	5
6	T16	T17	T18	VCC	VCC	VCC	VCC	VEE	VCC	VCC	VCC	VCC	VCC	VEE	VCC	VCC	XP0	XP1	XP2	XP3	6
7	T19	T20	T21	VCC	VEE	VCC									VCC	VCC	XP4	XP5	XP6	XP7	7
8	T22	T23	TP1	VCC	VEE	VCC									VCC	VCC	SDIN	SMODE	OFT	PLFT	8
9	T24	T25	T26	VCC	VCC	VCC									VCC	VCC	VEE	RESET	MPL EN*	DIV EN*	9
10	T27	T28	TP2	VEE	VCC	VEE									VEE	VEE	MI EN*	MZ SEL	MY EN*	MX EN*	10
11	T29	T30	TP3	VCC	VCC	VEE									VEE	VCC	I7	I6	I5	I4	11
12	T31	T32	T33	VCC	VCC	VCC									VCC	VCC	I3	I2	I1	I0	12
13	T34	T35	TP4	VCC	VEE	VCC									VCC	VEE	TSO EN*	n/c	SD OUT	FLAG 11	13
14	T36	T37	T38	ZEN*	VEE	VCC									VCC	VCC	FLAG 10	FLAG 9	FLAG 8	FLAG 7	14
15	T39	T40	TP5	VCC	CLK	VEE									VEE	VCC	MF EN*	AF EN*	FSO EN*	FO EN*	15
16	T41	T42	TP6	TO EN*	VCC	VEE									VEE	VEE	FLAG 13	FLAG 6	FLAG 5	FLAG 4	16
17	T43	T44	T45	VCC	VCC	VCC									VCC	VCC	FLAG 3	FLAG 2	FLAG 1	FLAG 0	17
18	T46	T47	TP7	VCC	VEE	VCC									VCC	VCC	FLAG 12	MY SEL0	MY SEL1	MX SEL0	18
19	T48	T49	T50	VEE	VEE	VCC									VCC	VEE	MX SEL1	TSEL	AEN*	A SEL0	19
20	T51	T52	T53	VCC	VCC	VCC	VEE	VCC	VCC	VCC	VCC	VCC	VCC	VCC	VCC	VCC	A SEL1	AY SEL0	AY SEL1	AX SEL0	20
21	T54	T55	T56	T57	VCC	VCC	VCC	VCC	VCC	VCC	VCC	VEE	VCC	VCC	VCC	AX SEL1	CLT HY	CLT HX	APL EN*	AI EN*	21
22	T58	T59	T60	Y9	Y13	Y17	Y21	Y25	Y29	Y33	Y37	Y41	Y45	Y49	Y53	Y57	AY EN*	AX EN*	YP7	YP6	22
23	T61	T62	T63	Y8	Y12	Y16	Y20	Y24	Y28	Y32	Y36	Y40	Y44	Y48	Y52	Y56	Y60	YP5	YP4	YP3	23
24	Y1	Y3	Y5	Y7	Y11	Y15	Y19	Y23	Y27	Y31	Y35	Y39	Y43	Y47	Y51	Y55	Y59	YP2	YP1	YP0	24
25	Y0	Y2	Y4	Y6	Y10	Y14	Y18	Y22	Y26	Y30	Y34	Y38	Y42	Y46	Y50	Y54	Y58	Y61	Y62	Y63	25
	A	B	C	D	E	F	G	H	J	K	L	M	N	P	R	T	U	V	W	X	

Bottom View

Figure 7 — B3130 / B4130 Pin Out

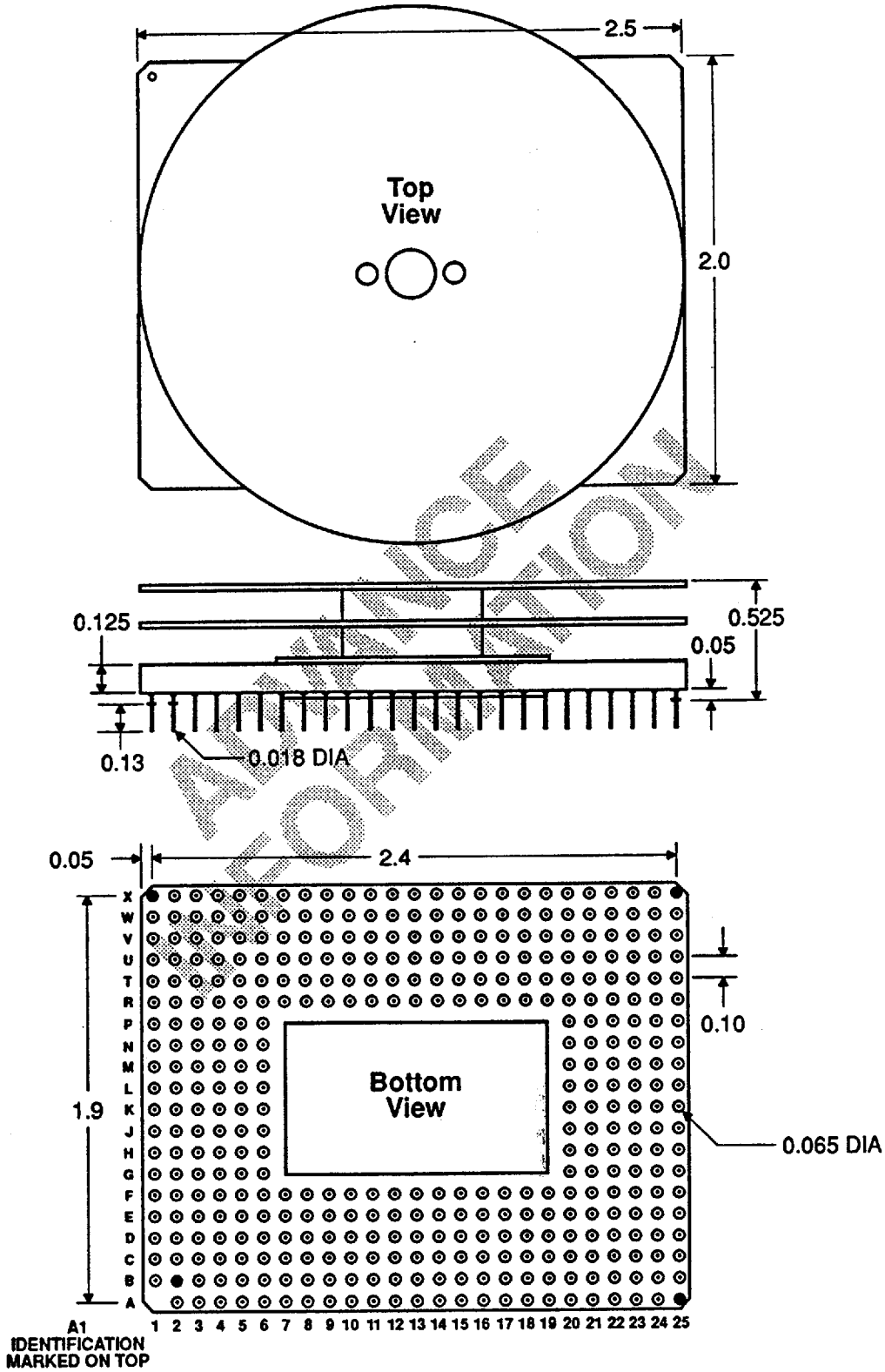


Figure 8 — Package Dimensions with Flat Round Heatsink

Single Chip Floating Point Processors B2130/B3130/B4130

T-49-12-05

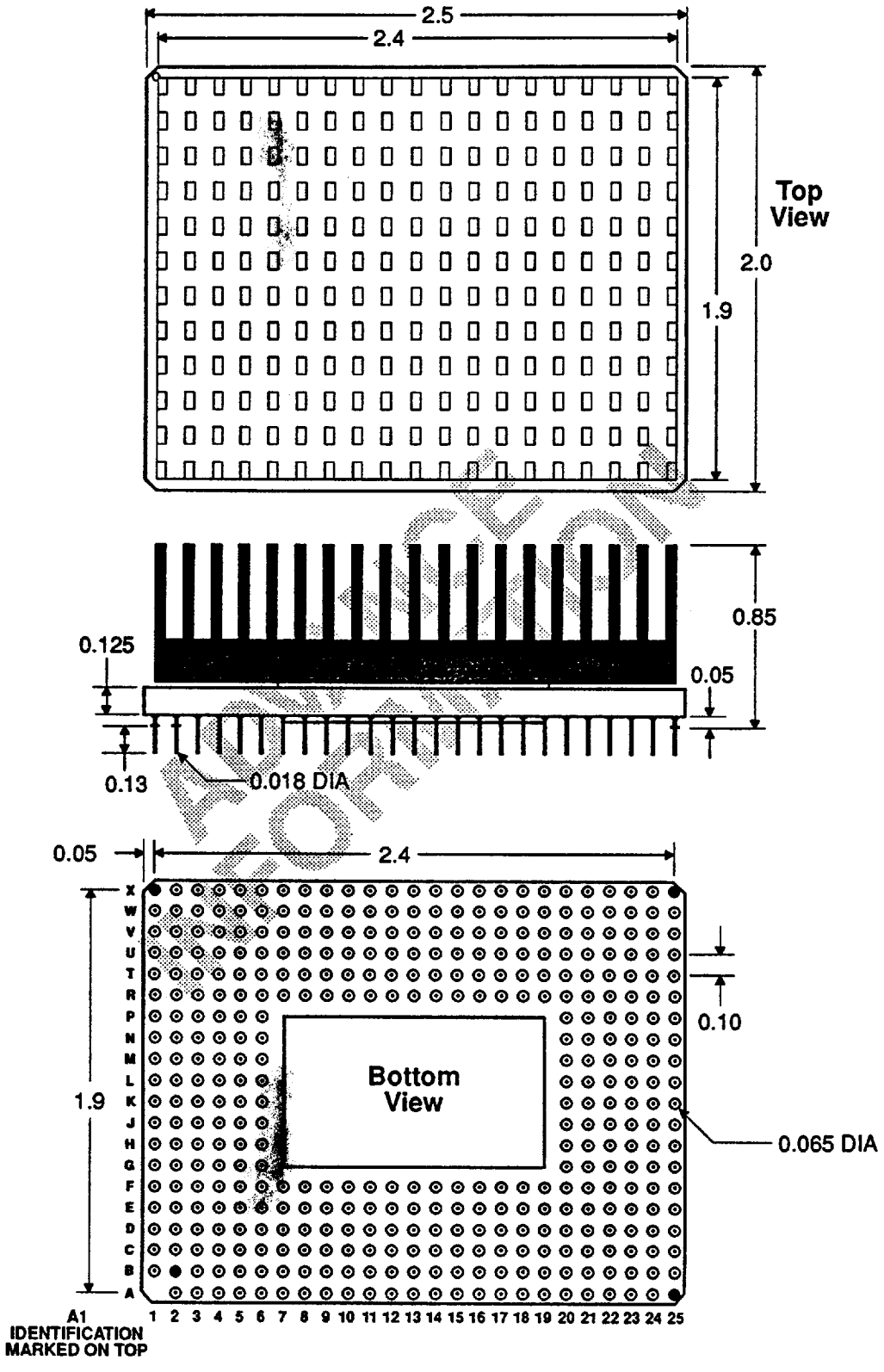


Figure 9 — Package Dimensions with PinFin Heatsink