WTL 1164/WTL 1165 64-BIT IEEE FLOATING POINT MULTIPLIER/ DIVIDER AND ALU

PRELIMINARY DATA

Features

A COMPLETE FLOATING POINT ARITHMETIC SOLUTION FOR HIGH-SPEED PROCESSORS AND COPROCESSORS

FULL 32-BIT AND 64-BIT FLOATING POINT FORMAT AND OPERATIONS, CONFORMING TO THE IEEE STANDARD FOR FLOATING POINT ARITHMETIC

INTERFACE CHIPS TO POPULAR 32-BIT MICROPROCESSORS AVAILABLE

FULL FUNCTION

Addition
Subtraction
Multiplication
Division
Conversion to and from 32-bit two's complement integers
Absolute value
Compare

INTERFACES DIRECTLY TO 32-BIT BUSES

SINGLE PHASE, EDGE-TRIGGERED CLOCKED INTERFACE WITH FULLY REGISTERED INPUTS AND OUTPUTS

HIGH SPEED

2.38 MFlops (420 ns) 32-bit add/subtract/convert and compare
1.85 MFlops (540 ns) 64-bit add/subtract/convert and compare
2.38 MFlops (420 ns) 32-bit multiply
1.67 MFlops (600 ns) 64-bit multiply
0.52 MFlops (1.92 μs) 32-bit divide
0.26 MFlops (3.78 μs) 64-bit divide
Up to 3.33 MFlops (300 ns) for pipelined operations
Up to 3.33 MFlops (300 ns) for chained operations
32-bit data input or 32-bit data output operation every 60 ns

STANDARD 64-PIN DIP AND 68-PIN PGA PACKAGES

Description

The WTL 1164 floating point multiplier and the WTL 1165 floating point arithmetic logic (ALU) unit are designed to provide high-speed 32-bit and 64-bit floating point processing capability. By virtue of their high performance, simple interface and ease of control, the WTL 1164/1165 are ideal choices for high performance floating point coprocessors. In fact, the bus interface and control of the WTL 1164/1165 are so simple that all of the logic required to implement a memory-mapped coprocessor to industry standard microprocessors can be put on a single chip. A short application note explaining how a single chip, the WTL 1163, can interface the WTL 1164/1165 to Intel's 80386 is attached.

This two-chip set can handle both single (32-bit) and double (64-bit) precision IEEE formats and operations, as well as full 32-bit two's complement integers. The NMOS VLSI design allows all of the functional elements to be combined on two chips while very flexible input/output path and control signals permit the WTL 1164/1165 to work in a broad range of bus systems and a wide variety of applications.

The data format and floating point operations conform to the requirements of the IEEE Standard for Binary Floating Point, including all rounding modes, infinity, NaN, denormalized and zero operand representations, and the treatment of exceptions, such as overflow, underflow, divide-by-zero, invalid and inexact operations. This assures complete software portability between systems designed using these parts and other general purpose computer systems which may be used to prototype algorithms and applications software. A "FAST" mode of operation, which removes the time penalty of underflow exception handling by substitution of zero for denormalized numbers while retaining all the other features, is included.

The WTL 1164/1165 use dedicated circuit arrays to perform the required functions, providing significantly faster processing than designs which must rely solely on sequential, clocked logic. The array flowthrough time for the WTL 1164 is under 240 ns for a single precision (32-bit) multiply and under 360 ns for a double precision (64-bit) multiply. For the WTL 1165, the array flowthrough time is under 240 ns for both single and double precision functions. These figures include the time for performing the arithmetic function as well as denormalization, renormalization and exponent adjustments.

Description, continued

Although division operations do not occur as frequently as addition, subtraction or multiplication operations, systems performing floating point arithmetic in close accordance with the IEEE standard require a floating point division operation which meets the requirements for accuracy specified by the standard. The division operation provided on the WTL 1165 floating point ALU fully conforms with the IEEE standard.

Data input and output transfers may occur at a rate of one transfer per clock cycle, permitting the devices to be used in variety of bus configurations without degrading performance. Two double precision inputs can be loaded through the 32-bit input data port, X, in four clock cycles, and the result unloaded in two clock cycles. Two single precision inputs can be loaded in two clock cycles and the result unloaded in a single clock cycle. One or both operands can be stored internally, allowing multiple functions or repeated operations with a constant to be performed on two operands.

All inputs and outputs are fully registered. All are loaded on each positive-going transition of the clock. Transfers from input registers to the ALU or multiplier array are accomplished with load controls. Unload controls select transfers from the array to the output registers and control tri-stating of the output registers.

A mode register selects optional characteristics that are not often changed, such as the selection of IEEE format, operation timing and rounding modes. The 16-bit mode register is loaded in 4-bit segments from the function input registers.

Arithmetic functions are selected by a 6-bit function control, which is transferred at the same time as the final operand. A 4-bit status output flags arithmetic exceptions and conditions.

Pin Definitions

X_{31-0}

32-bit Input/Output Port

S_{3-0}

4-bit status port which indicates execeptions or conditions resulting from floating point operations.

F_{4-0}

5-bit function control port

L_{3-0}

The 4-bit load control is used to specify the site and destination of the data on the X Port and start the instruction.

U

Unload control is used to specify the most or least significant half of the result latch.

CSL–

Load enable; load control loaded only if CSL– is a logic "0"

CSUX–

This signal controls the direction of the internal X1 Bus and serves as the synchronous output enable for the X Port. CSUX– must *not* be asserted when loading data from the X Port.

CSUS–

Synchronous status output enable

OE–

Asynchronous status and output enable; to activate/deactivate outputs both CSUX– and OE– must be a logic "0"/logic "1" with the proper timing. Status outputs are similarly controlled by OE– and CSUS–.

CLK

System clock; all registers are synchronized to the positive-going edge of the clock.

V_{CC}

5V power supply for input/output port; all V_{CC} pins must be connected.

V_{DD}

5V power supply for internal circuits; all V_{DD} pins must be connected.

GND

Ground; all GND pins must be connected.

Block Diagram

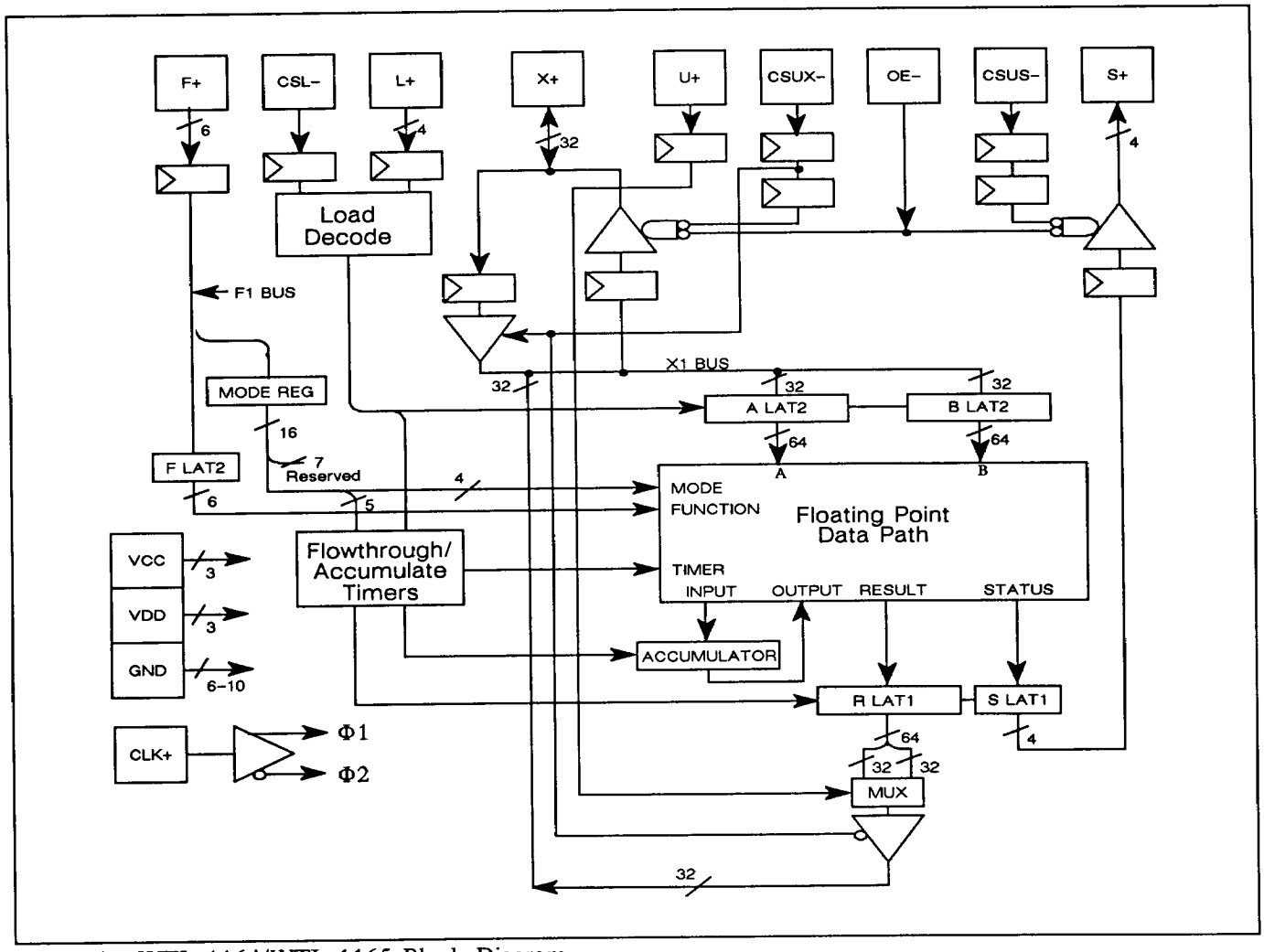


Figure 1. WTL 1164/WTL 1165 Block Diagram

NOTATION

The signal names used are described by the data path diagram (see Figure 1). The notation "–" following a signal (i.e., OE–) indicates a negative true signal. The notation "SIGNAL1" indicates a signal that is produced by a single full register from another signal, named "SIGNAL", and "SIGNAL2" indicates a signal produced by a second level of register. A register with no clock signal indicated is clocked unconditionally by the CLK signal, while others may have gated or enabled clocks.

In some cases, half-registers (latches) are used instead of registers. These are marked "LAT1" if they are transparent when the CLK signal is high and latching when CLK is low, or "LAT2" if they are transparent when CLK is low and latching when CLK is high.

Individual bits in a multi-bit signal are numbered from zero, starting with the least significant bit (little-endian).

Block Diagram, continued

All input signals, with the sole exception of OE–, are synchronized on the rising edge of CLK which, in the tables below, is denoted edge #0. Actions which are taken on the rising edge following the associated input signal are listed under edge #1, and actions taken after two rising edges are listed under edge #2. Actions which are taken on the falling edge following the associated input signal are listed under edge #0.5, and actions taken after two falling edges are listed under edge #1.5.

Specifications

ABSOLUTE MAXIMUM RATINGS (Above Which The Useful Life May Be Impaired)

Supply voltage ..... –0.5 to 7.0 V
Input voltage ..... –0.5 to 5.5 V
Output voltage ..... –0.5 to 5.5 V
Operating temperature range (T_{CASE}) –55°C to 125°C
Storage temperature range ..... –65°C to 150°C
Lead temperature (10 seconds) ..... 300°C
Junction temperature ..... 175°C

RECOMMENDED OPERATING CONDITIONS

PARAMETER	1164/1165-60 COMMERCIAL			1164/1165-80 COMMERCIAL			UNIT
	MIN	NOM	MAX	MIN	NOM	MAX	
Supply voltage, V_{CC}	4.75	5.0	5.25	4.75	5.0	5.25	V
Supply voltage, V_{DD}	4.75	5.0	5.25	4.75	5.0	5.25	V
Operating temperature, T_{CASE}	0		80	0		80	°C

DC ELECTRICAL CHARACTERISTICS, 1

PARAMETER		TEST CONDITIONS	MIN	MAX	UNIT	NOTES
V_{IH}	High-level input voltage	V_{DD}/V_{CC} = MIN	2.0		V	
V_{IL}	Low-level input voltage	V_{DD}/V_{CC} = MIN		0.8	V	
V_{OH}	High-level output voltage	V_{DD}/V_{CC} = MIN, I_{OH} = –1.0 mA	2.4		V	
V_{OL}	Low-level output voltage	V_{DD}/V_{CC} = MIN, I_{OL} = 4.0 mA		0.4	V	
I_{LI}	Input leakage current	V_{DD}/V_{CC} = MAX, V_{IN} = 0 to V_{CC}		10	µA	
I_{LO}	Output leakage current (output disabled)	V_{DD}/V_{CC} = MAX, V_{IN} = 0 or V_{CC}		10	µA	
C_{IN}	Input capacitance	V_{CC} = MAX, V_{IN} = 0 to Vcc		5	pF	Note 2
C_{OUT}	Output capacitance	V_{CC} = MAX, V_{OUT} = 0 to Vcc		8	pF	Note 2
C_{CLK}	Clock Input capacitance	Vcc = MAX, V_{IN} = 0 to Vcc		15	pF	Note 2
I_{CC}+ I_{DD}	Supply current, WTL1164	V_{DD}/V_{CC}= MAX; T_{CY} = MIN; TTL inputs		600	mA	Note 3
I_{CC}+ I_{DD}	Supply current, WTL1165	V_{DD}/V_{CC}= MAX; T_{CY} = MIN; TTL inputs		700	mA	Note 3

Note 1: Tested from T_{CASE} = 0° C min to 80° C max; V_{DD} and Vcc from 4.75 to 5.25V.
Note 2: Not tested.
Note 3: Worst case power is at 0° C. See Figure 4 for typical power vs. temperature curve.

Specifications, continued

AC ELECTRICAL CHARACTERISTICS, 1, 4

PARAMETER	TEST CONDITIONS	1164/1165-60		1164/1165-80		UNIT
		MIN	MAX	MIN	MAX	
T_{CY} Clock cycle time	Figure 3	60	DC	80	DC	ns
T_{CH} Clock high time	Figure 3	27	DC	35	DC	ns
T_{CL} Clock low time	Figure 3	27	1000	35	1000	ns
T_S Input setup time	Figure 2	15		15		ns
T_H Input hold time	Figure 2	3		2		ns
T_{DO} Output delay time	Figure 2		35		35	ns
T_{VO} Output valid time	Figure 2; Note 6	5		3		ns
T_{OZ} Output disable time	Figure 2; Notes 5 and 6	3	35	3	35	ns
T_{ZO} Output enable time	Figure 2	3	35	3	35	ns
T_{FL} Flowthrough time						
WTL1164 Floating point multiplier			240		240	ns
WTL1165 Floating point ALU			240		240	ns
T_{AC} Accumulation time						
WTL1164 Floating point multiplier			120		160	ns
WTL1165 Floating point ALU			60		80	ns
T_{NC} Operand encoding time WTL 1164			60		80	ns
T_{OP} Operation time (using T_{CY} min)						
WTL1164 Floating point multiplier						
32-bit multiply			240		240	ns
64-bit multiply			360		400	ns
WTL1165 Floating point ALU						
32-bit divide			1740		2240	ns
64-bit divide			3480		4560	ns
all other 32-bit functions			240		240	ns
all other 64-bit functions			240		240	ns
T_{LA} Total latency (using T_{CY} min)						
WTL1164 Floating point multiplier						
32-bit multiply			420		480	ns
64-bit minimum latency multiply			600		700	ns
WTL1165 Floating point ALU						
32-bit divide			1920		2480	ns
64-bit divide			3780		4900	ns
32-bit single operand instructions			360		400	ns
64-bit single operand instructions			420		480	ns
32-bit double operand instructions			420		480	ns
64-bit double operand instructions			540		640	ns

Note 4: All transitions except Toz are measured at 1.5V level, as shown in Figure 2, with TTL inputs of 0.4V and 3.5V. The test load in Figure 5 is used.
Note 5: The test load in Figure 6 is used.
Note 6: Guaranteed but not tested.
Specifications subject to change without notice.

I/O Chracteristics

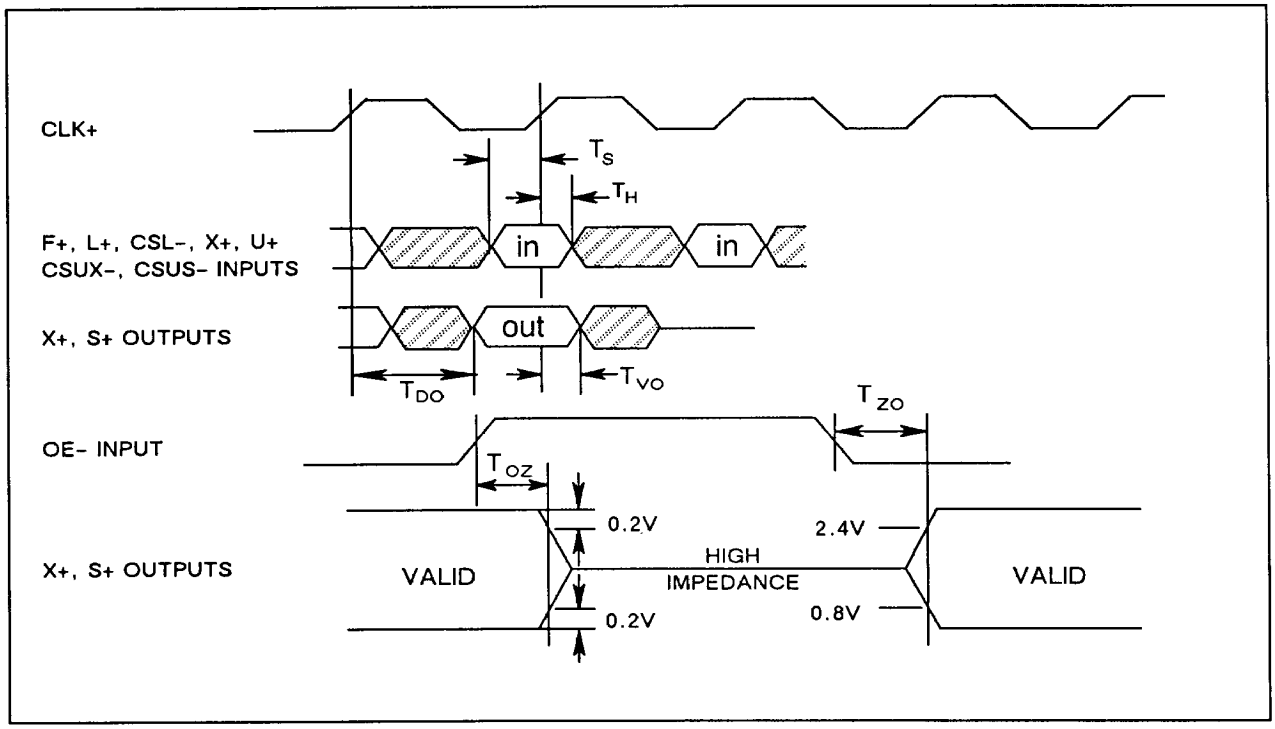


Figure 2. WTL 1164/WTL 1165 Input/Output Timing

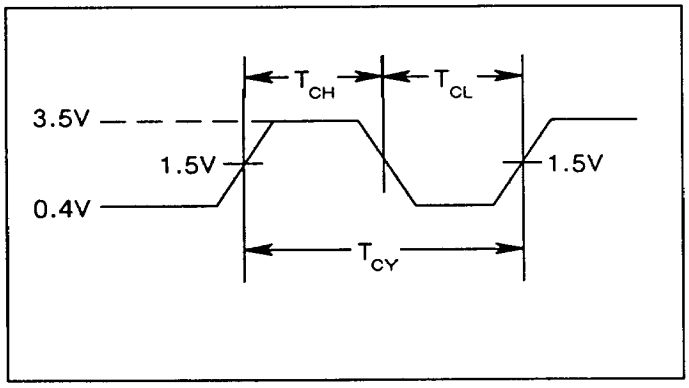


Figure 3. WTL 1164/WTL1165 Clock Timing

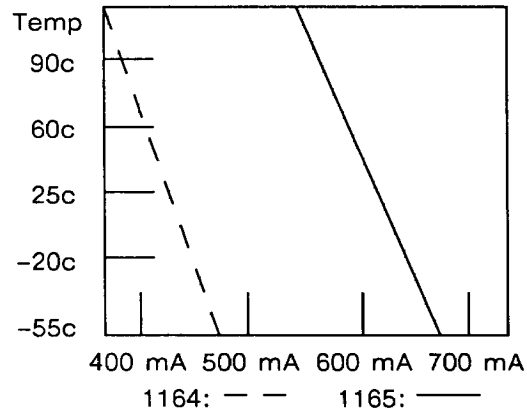


Figure 4. Typical $I_{CC} + I_{DD}$ vs. Temperature

I/O Chracteristics, continued

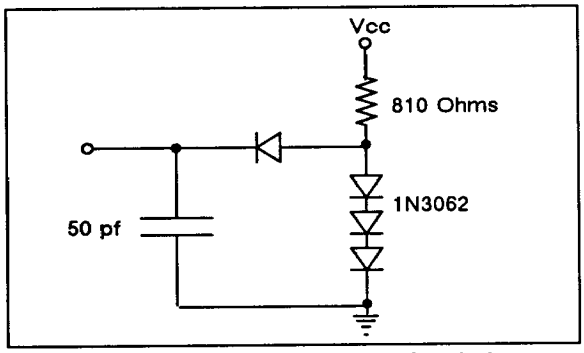


Figure 5. Normal Load Circuit for Delay Measurements

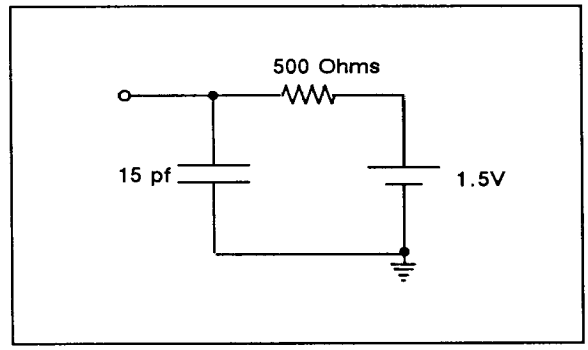


Figure 6. Toz Measurement Circuit

Method of Operation

LOAD CONTROLS

The CSL– and L input signals control the loading of operands into the on-chip registers and into the arithmetic path. The signals also control the loading of operating modes into on-chip mode registers from the 6-bit function code bus.

Using CSL– and L , data presented on the external bus, X , may be written into the on-chip registers, A and B, where they pass directly into the arithmetic unit.

The CSL– signal, if de-asserted (1), causes no load operations to be performed, regardless of the state of the L signal. Load operations are performed only if the CSL– signal is asserted.

LOAD NOTES

CSL– is often used as a chip select to allow the sharing of the L and F fields. An inverter can be used to generate CSL– for the other device if the L Field specifies a NOP when no loading is desired.

If Lo is set, indicating a function is to be started, the F Register (FREG) is loaded and an operation is initiated. This also resets the pipeline advance timer and the accumulator advance timer.

If Lo and CSL– are both asserted while an operation is in progress, it will cause that operation to abort but will not start the new operation. The result of an aborted operation is undefined.

For the conversion of integer to single or double precision functions, the 32-bit integer operand must be loaded into the A Operand Register with the LAL (#12) or LALF (#13) load commands.

Divide uses the A Operand Register for intermediate values. The A Register must be reloaded prior to starating another instruction following a divide operation.

Single precision loads modify both halves of the A or B register.

Method of Operation, continued

$L_3 L_2 L_1 L_0$	Mnemonic	EDGE #0.5	
0 0 0 0	LNOP		NOP
0 0 0 1	LF		F1 → FREG
0 0 1 0	LBS	X1 → B [single]	
0 0 1 1	LBSF	X1 → B [single]	F1 → FREG
0 1 0 0	LBL	X1 → B [LSW double]	
0 1 0 1	LBLF	X1 → B [LSW double]	F1 → FREG
0 1 1 0	LBM	X1 → B [MSW double]	
0 1 1 1	LBMF	X1 → B [MSW double]	F1 → FREG
1 0 0 0	LMODE		LOAD MODE
1 0 0 1	– reserved		
1 0 1 0	LAS	X1 → A [single]	
1 0 1 1	LASF	X1 → A [single]	F1 → FREG
1 1 0 0	LAL	X1 → A [LSW double]	
1 1 0 1	LALF	X1 → A [LSW double]	F1 → FREG
1 1 1 0	LAM	X1 → A [MSW double]	
1 1 1 1	LAMF	X1 → A [MSW double]	F1 → FREG

Table 1. L Input

UNLOAD CONTROLS

The unload controls, U, CSUX– and CSUS–, select the data to be presented on the external bus, X, and the status bus, S. The U signal controls whether the most significant word or the least significant word is sent to the X Bus. Single precision floating point results are unloaded by selecting the most significant word, while results in integer format are unloaded by selecting the least significant word.

The CSUS– and CSUX– signals control tri-stating of the S and X buses, respectively, in a synchronous fashion. The OE– signal controls tri-stating of both the S and X Buses and is a fully asynchronous signal. In the chart below, X1 refers to the X1 Bus shown in Figure 1.

CSUS –	CSUX –	U	Mnemonic	EDGE #0.5	EDGE #1
0	0	0	ULST	LSW result → X1	STREG1 → S, X1 → X
0	0	1	UMST	MSW result → X1	STREG1 → S, X1 → X
0	1	X	UST	X → X1	STREG1 → S, X tri-stated
1	0	0	UL	LSW result → X1	S tri-stated, X1 → X
1	0	1	UM	MSW result → X1	S tri-stated, X1 → X
1	1	X	UNOP	X → X1	S tri-stated, X tri-stated

Table 2. U Input

FUNCTION CONTROLS

The function controls select the operation to be performed by the floating point units. Each chip has a separate set of function code definitions.

The F signal is received on each clock edge and buffered as the F1 Bus. An operation is started when ordered by the Lo signal. Starting an operation loads F1 → F LAT2, and resets the flowthrough and accumulate timers.

In the tables below, F32 denotes a single precision floating point operand or result; F64 denotes a double precision value. F32/64 represents either a single or double precision floating point operand. W32 and W64 denote single and double precision wrapped, normalized operands and results. U32 and U64 denote single and double precision unrounded, wrapped, normalized operands and results. I32 represents a 32-bit integer value.

Method of Operation, continued

When two operands are indicated, the A operand is given on the left and the B operand on the right. When only one operand is indicated, it must be loaded into the A Operand Register.

When a double precision result is indicated, operands may be either single precision or double precision or mixed single and double precision. The load controls indicate the precision of the operands as they are loaded. Mixed precision operations must be loaded in the specified loading sequence (figures 7, 8 and 14).

If an instruction is started when another instruction is in progress, in either the multiplier or ALU, both instructions are corrupted. While divide operations are in progress a new instruction may be started by first starting some arbitrary instruction (killing both instructions), waiting three cycles, then beginning the third instruction.

Function Controls For Floating Point Multiplier (WTL 1164)

For the multiplier, the timing of the function is determined by the precision of the B operand. When the B operand is single precision, the multiplication is performed in the flowthrough operation time, T_{FL}; when the B operand is double precision, the time is increased by the accumulate time, T_{AC}.

The floating point multiplier uses functions provided on the floating point ALU for the handling of denormalized operands and results, significantly increasing multiplier performance and reducing system cost. A mode bit is provided to select whether full handling of denormalized operands is required.

In IEEE mode, denormalized operands are detected in the multiplier and exceptions are generated when they are found, indicating which operands are denormalized. The original operands may then be sent to the floating point ALU, where the wrap function normalizes the operands and extends the range of the exponent. The modified operands may then be multiplied, using function codes F_{2-0} = #2 through #7 in the floating point multiplier. In "FAST" mode, denormalized operands are treated as zero.

Whether or not the operands are denormalized, the multiplier may also produce a number whose exponent is too small to be represented as a normalized floating point value. In IEEE mode, one of the exceptions, Underflow or Underflow & Inexact, is generated, and the value returned is a normalized number with extended range and without rounding. This value may be converted to a denormalized number or zero value by using the Unwrap Exact or Unwrap Inexact functions of the floating point ALU. In "FAST" mode, these results are given zero values and the Underflow or Underflow & Inexact exception is generated.

Certain restrictions are imposed on the timing with which operands are loaded into the WTL 1164 floating point multiplier. The A operand is internally encoded into a more usable form immediately after the operand is loaded. The encoded A operand is loaded into an internal register when a multiplication function is indicated by the L_O signal. The time required for the encoding is T_{NC}, which is normally no greater than the time required to load the B operand. Thus, if the A operand is loaded first, followed by the B operand, the multiplication function can begin immediately upon loading the B operand. If the B operand must be loaded first, an additional cycle is required before starting the operation.

In some applications, a series of multiplications may be performed with one of the two operands held constant. Ideally, in order to maximize performance, this constant operand is the A operand, so each multiplication can begin immediately upon loading the B operand.

Method of Operation, continued

$F_2F_1F_0$	Operation			Description
0 0 0	F32	* F32	→ F32 / U32	single multiply
0 0 1	F32 / F64	* F32 / F64	→ F64 / U64	double multiply
0 1 0	W32	* F32	→ F32 / U32	single multiply, A wrapped
0 1 1	W32 / W64	* F32 / F64	→ F64 / U64	double multiply, A wrapped
1 0 0	F32	* W32	→ F32 / U32	single multiply, B wrapped
1 0 1	F32 / F64	* W32 / W64	→ F64 / U64	double multiply, B wrapped
1 1 0	W32	* W32	→ F32 / U32	single multiply, A & B wrapped
1 1 1	W32 / W64	* W32 / W64	→ F64 / U64	double multiply, A & B wrapped

$F_5F_4F_3$	Operation	Description
0 0 0	A * B	multiply
0 0 1	|A| * B	B times magnitude of A
0 1 0	A * |B|	A times magnitude of B
0 1 1	|A| * |B|	magnitude of A times B
1 0 0	− A * B	multiply and negate
1 0 1	− |A| * B	B times negative value of A
1 1 0	− A * |B|	A times negative value of B
1 1 1	− |A| * |B|	negative value of A times B

Table 3. Function Controls for Floating Point Multiplier

Function Controls For Floating Point ALU (WTL 1165)

The function controls for the WTL 1165 are shown in Table 4. Mixed precision operations and divide operations may generate one of the denormalized input exceptions; all other operations can handle denormalized operands directly without the exception.

When a DNRM exception is encountered in a mixed-mode operation, the DNRM input must first be converted to double precision before the instruction can be re-started.

Like the WTL 1164, the WTL 1165 may produce a result whose exponent is too small to be represented as a normalized floating point value. When this result appears in IEEE mode, one of the exceptions Underflow or Underflow & Inexact is generated and the value returned is a normalized number with extended range and without rounding. This value may be converted to a denormalized number or zero value by using the Unwrap Exact or Unwrap Inexact functions of the floating point ALU. In "FAST" mode, these results are given zero values and the Underflow or Underflow & Inexact exception is generated.

When performing the divide function in IEEE mode, denormalized operands are detected and exceptions are generated when they are found, indicating which operands are denormalized. The operands may then be loaded into the A operand of the floating point ALU, where the wrap function normalizes the operand and extends the range of the exponent. The modified operands may then be divided using function codes F_{5-0} = #14–15, #22–23, #30–31 in the floating point ALU. In "FAST" mode, denormalized operands are treated as zero, yielding a zero or infinite result and no denormalized operand exceptions. A divide-by-zero exception will occur, however, if a denormalized operand is used as a divisor in "FAST" mode.

Method of Operation, continued

F_5 F_4 F_3 F_2 F_1 F_0	Operation	Description
0 0 0 0 0 0 (0)	F32 − F32 → F32/U32	single subtract
0 0 0 0 0 1 (1)	F32/F64 − F32/F64 → F64/U64	double subtract
0 0 0 0 1 0 (2)	|F32 − F32| → F32/U32	single magnitude of difference
0 0 0 0 1 1 (3)	|F32/F64 − F32/F64| → F64/U64	double magnitude of difference
0 0 0 1 0 0 (4)	|F32| − |F32| → F32/U32	single subtract magnitudes
0 0 0 1 0 1 (5)	|F32/F64| − |F32/F64| → F64/U64	double subtract magnitudes
0 0 0 1 1 0 (6)	F32 ÷ F32 → F32/U32	single divide
0 0 0 1 1 1 (7)	F32/F64 ÷ F32/F64 → F64/U64	double divide
0 0 1 0 0 0 (8)	−F32 + 0 → F32/U32	single negate plus zero
0 0 1 0 0 1 (9)	−F32/F64 + 0 → F64/U64	double negate plus zero
0 0 1 0 1 0 (10)		
0 0 1 0 1 1 (11)		
0 0 1 1 0 0 (12)		
0 0 1 1 0 1 (13)		
0 0 1 1 1 0 (14)	W32 ÷ F32 → F32/U32	single divide, A wrapped
0 0 1 1 1 1 (15)	W32/W64 ÷ F32/F64 → F64/U64	double divide, A wrapped
0 1 0 0 0 0 (16)	F32 + F32 → F32/U32	single add
0 1 0 0 0 1 (17)	F32/F64 + F32/F64 → F64/U64	double add
0 1 0 0 1 0 (18)	|F32 + F32| → F32/U32	single magnitude of sum
0 1 0 0 1 1 (19)	|F32/F64 + F32/F64| → F64/U64	double magnitude of sum
0 1 0 1 0 0 (20)	|F32| + |F32| → F32/U32	single add magnitude
0 1 0 1 0 1 (21)	|F32/F64| + |F32/F64| → F64/U64	double add magnitude
0 1 0 1 1 0 (22)	F32 ÷ W32 → F32/U32	single divide, B wrapped
0 1 0 1 1 1 (23)	F32/F64 ÷ W32/W64 → F64/U64	double divide, B wrapped
0 1 1 0 0 0 (24)	F32 + 0 → F32/U32	single plus zero
0 1 1 0 0 1 (25)	F32/F64 + 0 → F64/U64	double plus zero
0 1 1 0 1 0 (26)		
0 1 1 0 1 1 (27)		
0 1 1 1 0 0 (28)	|F32| + 0 → F32/U32	single absolute value
0 1 1 1 0 1 (29)	|F32/F64| + 0 → F64/U64	double absolute value
0 1 1 1 1 0 (30)	W32 ÷ W32 → F32/U32	single divide, A & B wrapped
0 1 1 1 1 1 (31)	W32/W64 ÷ W32/W64 → F64/U64	double divide, A & B wrapped
1 0 0 0 0 0 (32)	Compare F32 − F32	single compare
1 0 0 0 0 1 (33)	Compare F32 / F64 − F32 / F64	double compare
1 0 0 0 1 0 (34)		
1 0 0 0 1 1 (35)		
1 0 0 1 0 0 (36)	Compare |F32| − |F32|	single compare magnitude
1 0 0 1 0 1 (37)	Compare |F32 / F64| − |F32 / F64|	double compare magnitude
1 0 0 1 1 0 (38)		
1 0 0 1 1 1 (39)		
1 0 1 0 0 0 (40)	Compare F32 − 0	single compare with zero
1 0 1 0 0 1 (41)	Compare F32 / F64 − 0	double compare with zero
1 0 1 0 1 0 (42)		
1 0 1 0 1 1 (43)		
1 0 1 1 0 0 (44)		
1 0 1 1 0 1 (45)		
1 0 1 1 1 0 (46)		
1 0 1 1 1 1 (47)		
1 1 0 0 0 0 (48)	U32 → F32 (Exact)	single unwrap exact value
1 1 0 0 0 1 (49)	U64 → F64 (Exact)	double unwrap exact value
1 1 0 0 1 0 (50)	F32 → W32	single wrap denormalized value
1 1 0 0 1 1 (51)	F64 → W64	double wrap denormalized value
1 1 0 1 0 0 (52)	U32 → F32 (Inexact)	single unwrap inexact value
1 1 0 1 0 1 (53)	U64 → F64 (Inexact)	double unwrap inexact vlaue
1 1 0 1 1 0 (54)		
1 1 0 1 1 1 (55)		

Table 4. Function Controls for Floating Point ALU

Method of Operation, continued

F_5 F_4 F_3 F_2 F_1 F_0	Operation	Description
1 1 1 0 0 0 (56)	I32 → F32	single convert to integer
1 1 1 0 0 1 (57)	I32 → F64	double convert to integer
1 1 1 0 1 0 (58)	I32 → F32	single float
1 1 1 0 1 1 (59)	I32 → F64	double float
1 1 1 1 0 0 (60)	F64 → F32 / U32	convert double to single
1 1 1 1 0 1 (61)	F32 → F64	convert single to double
1 1 1 1 1 0 (62)		
1 1 1 1 1 1 (63)		

Table 4. Function Controls for Floating Point ALU, continued

Mode Controls

The mode controls are not directly loaded from external pins. Instead, when the load mode command is presented on the L signal, F_{5-4} selects one of four mode control segments to be loaded with the signal F_{3-0}, according to the following table.

F_5 F_4	EDGE #0	EDGE #0.5
0 0 (0)	F → F1	$F1_{3-0}$ → MODE $_{3-0}$
0 1 (1)	F → F1	$F1_{3-0}$ → MODE $_{7-4}$
1 0 (2)	F → F1	$F1_{3-0}$ → MODE $_{11-8}$
1 1 (3)	F → F1	$F1_{3-0}$ → MODE $_{15-12}$

Mode controls must not be altered while a floating point function is in progress.

"FAST"/IEEE FORMAT MODE CONTROL

$MODE_0$ controls the floating point format used for all operations and the manner in which the denormalized operands and results are handled, according to the following table.

$MODE_0$	Function	
0	Multiplier and ALU generate denormalized operand exceptions and produce UNRM values on underflow exceptions.	(IEEE mode)
1	Multiplier and ALU flush denormalized operands to zero and round underflow results to zero.	(FAST mode)

ROUNDING MODE CONTROL FOR ALU FIX FUNCTION

$MODE_1$ controls the IEEE rounding mode for the ALU functions Convert Single to Integer and Convert Double to Integer, according to the following table.

$MODE_1$	Function
0	Round according to default rounding mode (MODE $_{3-2}$)
1	Round toward zero, regardless of the default rounding mode

Mode Controls, continued

ROUNDING MODE CONTROL

$MODE_{3-2}$ control the IEEE rounding mode for all operations, except for the ALU functions Convert Single to Integer and Convert Double to Integer, according to the following table.

$MODE_{3-2}$	Function
00	Round toward nearest value or even significand if a tie
01	Round toward zero
10	Round toward positive infinity
11	Round toward negative infinity

FLOWTHROUGH TIMER CONTROL

$MODE_{7-5}$ allow the number of clock cycles required to meet the flowthrough time T_{FL} to be selected using clock periods other than 60 ns. The value for T_{FL} which should be used is specified on page 5 under "AC ELECTRICAL CHARACTERISTICS". Using 240 ns for T_{FL} and the recommended clock speed of 60 ns, the value, k, which should be loaded into $MODE_{7-5}$ is three. To determine k for other clock periods or flowthrough times, the following formula may be used where $\lceil$ means round up to the next integer greater than or equal to the result:

$$MODE_{7-5} = \lceil (T_{FL} \div \text{Clock Period}) - 1$$

Clock Frequency (MHz)	Clock Period (ns)	$\lceil T_{FL} \div$ Clock Period	$MODE_{7-5}$
20.0	50	5	100 (4)
16.7	60	4	011 (3)
14.3	70	4	011 (3)
12.5	80	3	010 (2)
8.33	120	2	001 (1)

The flowthrough timer should never be set to 0.

ACCUMULATE TIMER CONTROL

If the B operand on the WTL 1164 is single precision, the multiplication is performed as a flowthrough operation, but if the B operand is double precision two passes must be made through the multiplier array. After the first pass the partial results of the multiplication are held in an accumulate register. These partial results are added to the results of the second pass to form the unrounded product. The minimum time required for the first pass through the array is called the accumulation time. For a given clock frequency the time taken for this accumulation operation is controlled by $MODE_{11-10}$. $MODE_{11-10}$ control when the Internal Accumulator Register in the multiplier is clocked. Loading a value, k, into $MODE_{11-10}$ causes the accumulator to be clocked k + 1 cycles after an operation is started.

Most functions of the WTL 1165 are performed as a flowthrough operation, except for divide operations. Divide operations on the WTL 1165 are performed in a bit-sequential fashion and require 25 iterations for a single precision divide and 54 iterations for a double precision divide. The time taken for each iteration is controlled by $MODE_{11-10}$. $MODE_{11-10}$ control the rate at which the internal accumulator register in the divide circuitry is clocked. Loading a value, k, into $MODE_{11-10}$ causes the accumulator to be clocked every k + 1 cycles after the start of a divide operation.

Mode Controls, continued

In general, the value used for MODE11-10 will not be the same for both chips. The value for TAC which should be used is specified on page 5 under "AC ELECTRICAL CHARACTERISTICS". The formula for calculating the value to be loaded into MODE11-10 is given below:

$$MODE11\text{-}10 = \lceil (T_{AC} \div \text{Clock Period}) - 1$$

Where $\lceil$ means round up to the next integer greater than or equal to the result.

The value which should be loaded into MODE11-10 for the WTL 1164 using 120 ns for TAC and a 60 ns clock is one. Values for other clock frequencies are shown in the table below:

Clock Frequency (MHz)	Clock Period (ns)	$\lceil T_{AC} \div$ Clock Period	MODE 11-10
20.0	50	3	10 (2)
16.7	60	2	01 (1)
14.3	70	2	01 (1)
11.1	90	2	01 (1)
8.33	120	1	00 (0)

The value which should be loaded into MODE11-10 for the WTL 1165 using 60 ns for TAC and a 60 ns clock is zero. Values for other clock frequencies are shown in the table below:

Clock Frequency (MHz)	Clock Period (ns)	$\lceil T_{AC} \div$ Clock Period	MODE 11-10
20.0	50	2	01 (1)
16.7	60	1	00 (0)
14.3	70	1	00 (0)
11.1	90	1	00 (0)
8.33	120	1	00 (0)

After the final firing of the accumulate timer on either the WTL 1164 or WTL 1165, the flowthrough timer is started, terminating the operation after the flowthrough timer expires. Operation timing can thus be determined from the following formulae.

Operation	Time required in cycles
Multiply, B operand is single precision	T_{FL}
Multiply, B operand is double precision	$T_{AC} + T_{FL}$
Divide, result is single precision	$25 * T_{AC} + T_{FL}$
Divide, result is double precision	$54 * T_{AC} + T_{FL}$
Other WTL1165 operations	T_{FL}

RESERVED MODE CONTROL BITS

MODE15-12, MODE9-8 and MODE4 on the floating point ALU chip and the floating point multiply chip are currently reserved for future definition. These mode control bits should be set to zero to assure compatibility with future derivatives of the WTL 1164/1165.

Result Status

The S Bus indicates any exceptions or conditions that result from operations performed by the floating point units. The status bits associated with an operation have the same timing as the results. The S Bus is controlled by the CSUS– control, which causes the S Bus to be driven two clock cycles after CSUS is asserted. For comparison operations, the S Bus indicates the condition resulting from the comparison; for all other operations, the S Bus indicates exception status arising from the associated operation. For all operations, the S Bus has the following meanings.

$S_3 S_2 S_1 S_0$	Comparison Condition	Exception Status
0 0 0 0 (0)	Equal	Result = +0 or –0, exact
0 0 0 1 (1)	Less than	Result = +infinity or –infinity, exact
0 0 1 0 (2)	Greater than	Result finite and $\neq$ 0, exact
0 0 1 1 (3)		Result finite and $\neq$ 0, inexact
0 1 0 0 (4)		– not used
0 1 0 1 (5)		Overflow & inexact
0 1 1 0 (6)		Underflow
0 1 1 1 (7)		Underflow & inexact
1 0 0 0 (8)		Operand A is Denormalized
1 0 0 1 (9)		Operand B is Denormalized
1 0 1 0 (10)		Operands A & B are Denormalized
1 0 1 1 (11)		Divide by Zero
1 1 0 0 (12)		Operand A is NaN
1 1 0 1 (13)		Operand B is NaN
1 1 1 0 (14)		Operands A & B are NaN
1 1 1 1 (15)	Unordered	Invalid Operation

Exception status signals #0–#3 indicate normal completion of the operation. The specific value of the status signal indicates the type of result produced, either zero, non-zero or infinity as well as exact or inexact. Signals #0–#2 indicate an exact result, as defined by the IEEE standard. Signals #3, #5, #6 and #7 indicate an inexact result. While #6 does not imply inexact directly, the result after unwrapping may be exact or inexact and the status bits correctly indicate exactness at the conclusion of the unwrap operation.

Signals #6 and #7 have special meaning in IEEE mode. These signals indicate the result is smaller than the minimum magnitude normalized number representable in the specified format and a UNRM value is presented as a result. This value should be converted to a legal denormalized number by the Unwrap Exact (for signal #6) or Unwrap Inexact (for signal #7) functions of the floating point ALU. The exception status indicated by the ALU at the conclusion of the unwrap operation, either #6 or #7, properly indicates the exactness of the result.

Status numbers eight through 11 signify denormalized input values (DIN) in IEEE mode. Status numbers 12 through 15 signify invalid operations (INV) and source NaNs.

Under certain conditions, multiple exceptions can occur. These exceptions may be resolved with the following priority table, where higher-priority exceptions will mask lower-priority exceptions.

Result Status, continued

Priority	Exception
Highest	Operands A & B are NaN
	Operand A is NaN
	Operand B is NaN
	Invalid Operation
	Divide by Zero
	Operands A & B are denormalized
	Operand A is denormalized
	Operand B is denormalized
	Underflow & inexact
	Underflow
	Overflow & Inexact
Lowest	Result is finite and <> 0, inexact

Timing

The WTL 1164 and WTL 1165 have very flexible loading controls to optimize loading order for total system performance. Figures 3 through 11 show loading sequences which minimize flowthrough times and are the preferred loading sequences. Figures 15 and 16 show loading sequences which are also valid but incur somewhat longer flowthrough times. While all of the loading sequences shown are guaranteed, restricting the loading order to the preferred loading sequences will minimize timing changes when using higher clock rates on future WTL 1164/1165 compatable devices.

PRECISION PREFERRED LOADING SEQUENCES

To conform to the preferred loading sequence, the loading order must be followed except where indicated. Other sequences may still be acceptable (see alternate loading sequences) but minor timing changes may be required for use with future WTL 1164/1165 family devices at higher clock speeds.

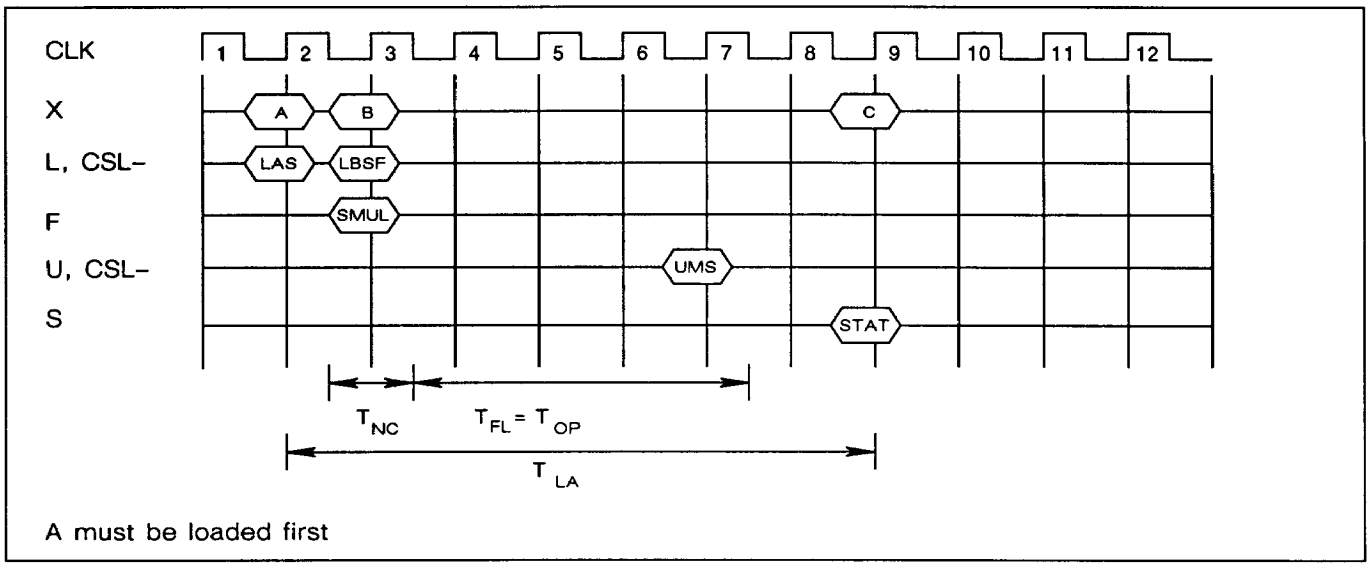


Figure 7. Single Precision Multiply

Timing, continued

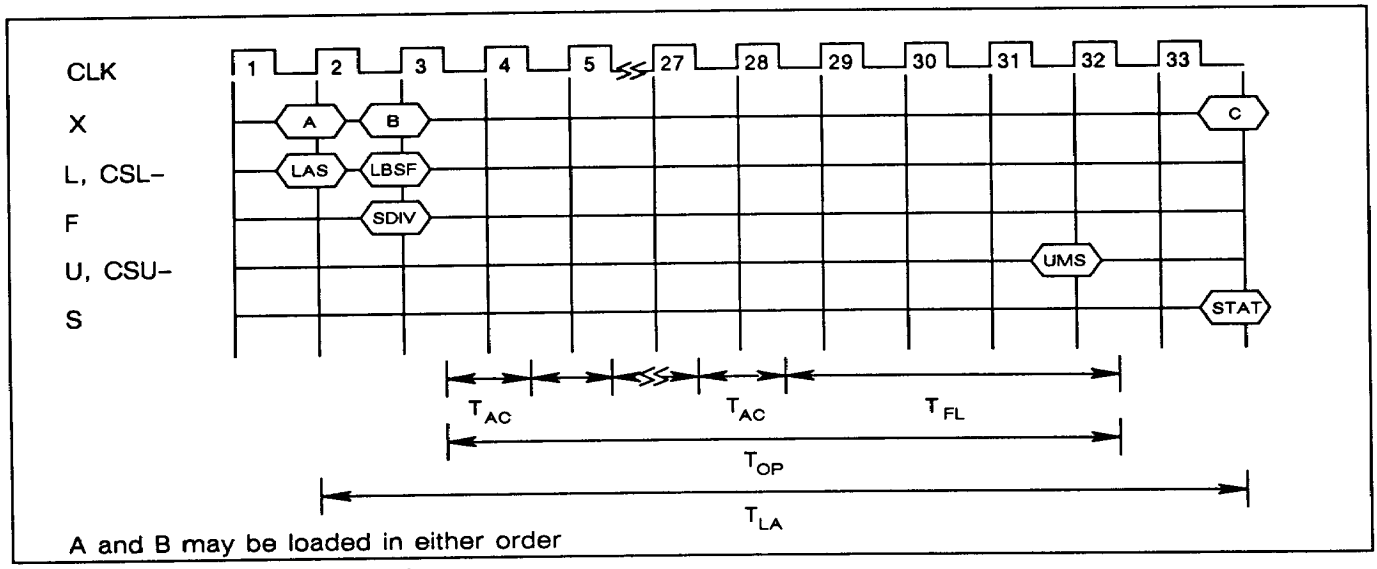


Figure 8. Single Precision Divide

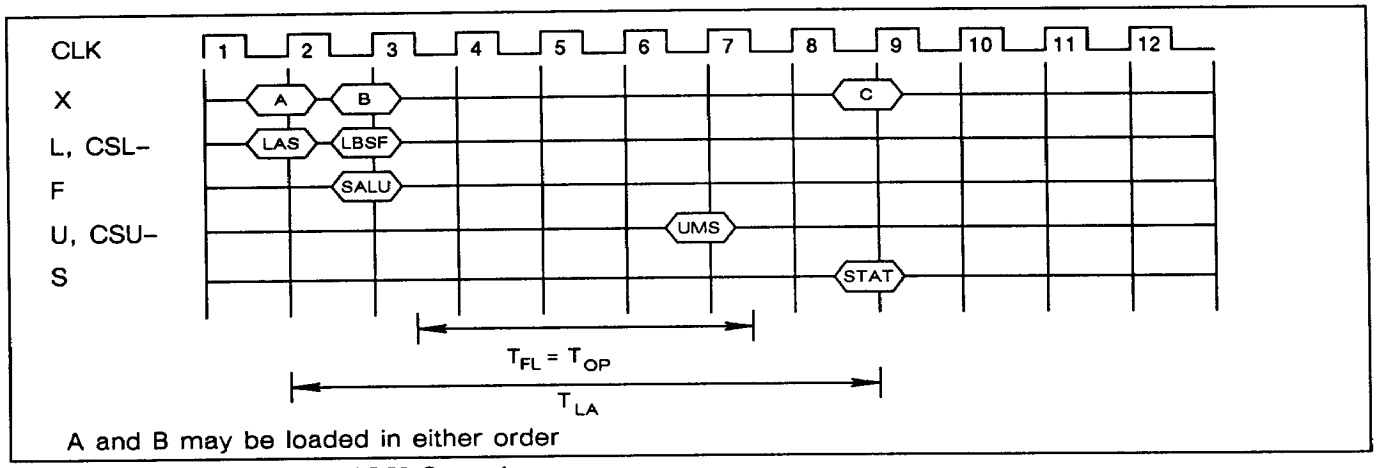


Figure 9. Single Precision ALU Operation

Timing, continued

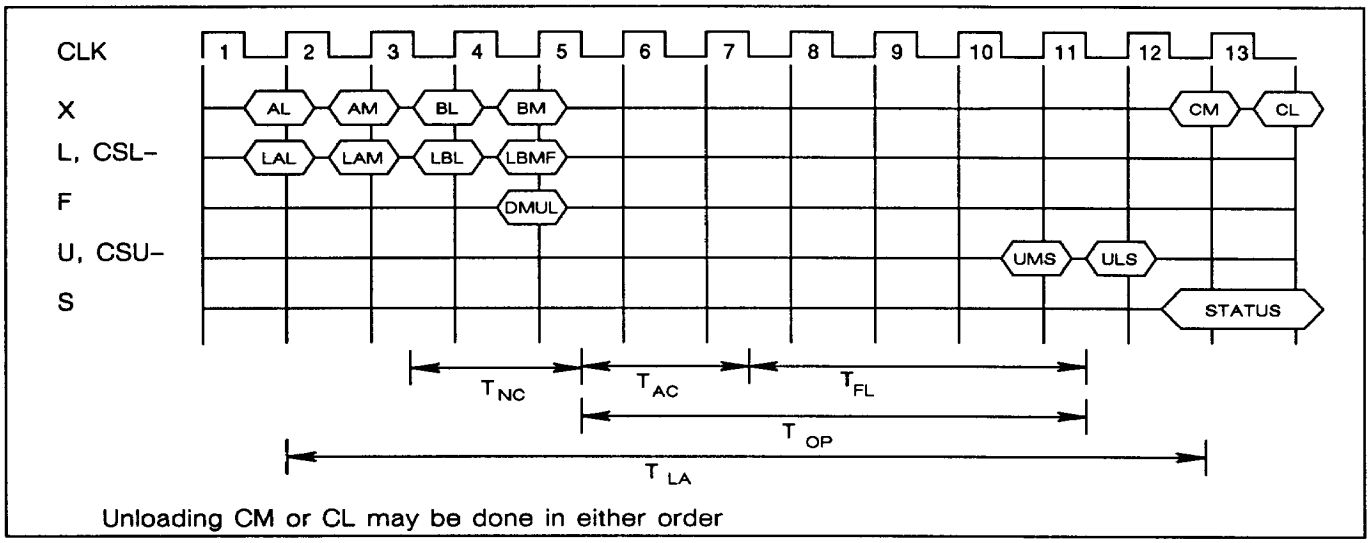


Figure 10. Double Precision Multiply

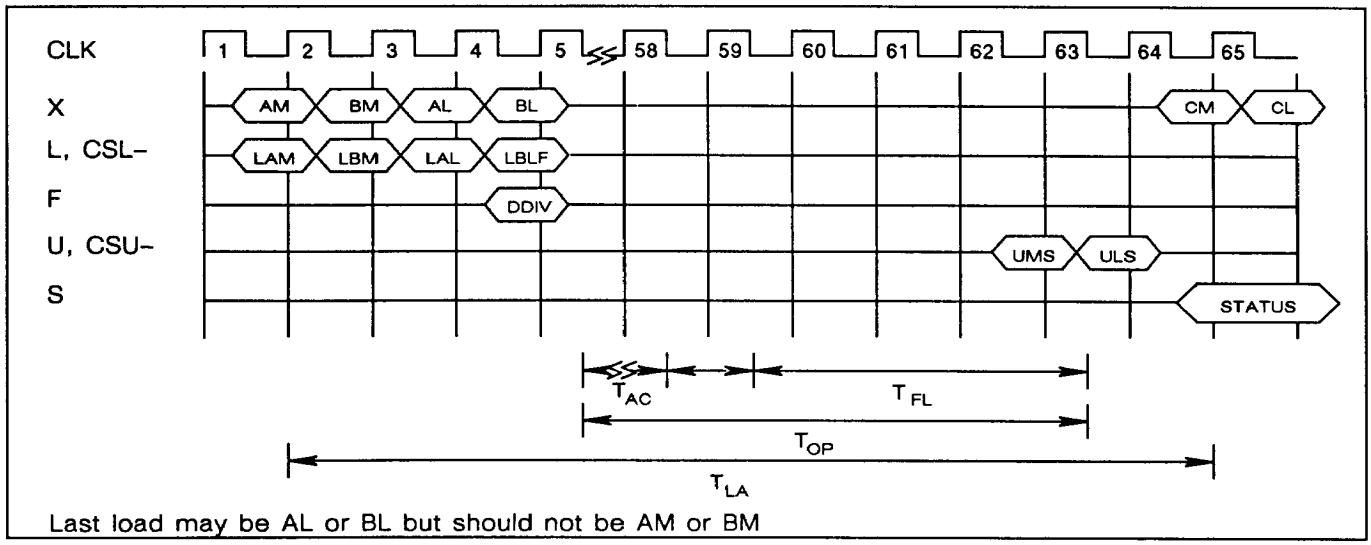


Figure 11. Double Precision Divide

Timing, continued

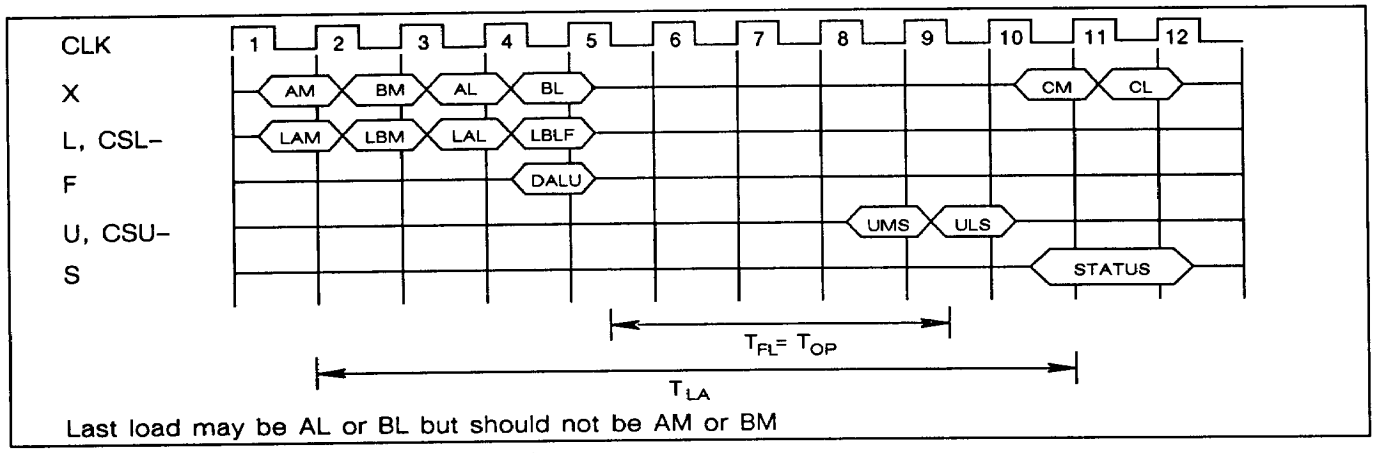


Figure 12. Double Precision ALU Operation

MIXED-MODE LOADING SEQUENCES

When operating with mixed single and double precision operands, the loading order is especially important. Special values such as zero, infinity and denormalized numbers require conversion from single to double precision values before the operation can be completed. For that reason, mixed-mode operations must be loaded as specified below.

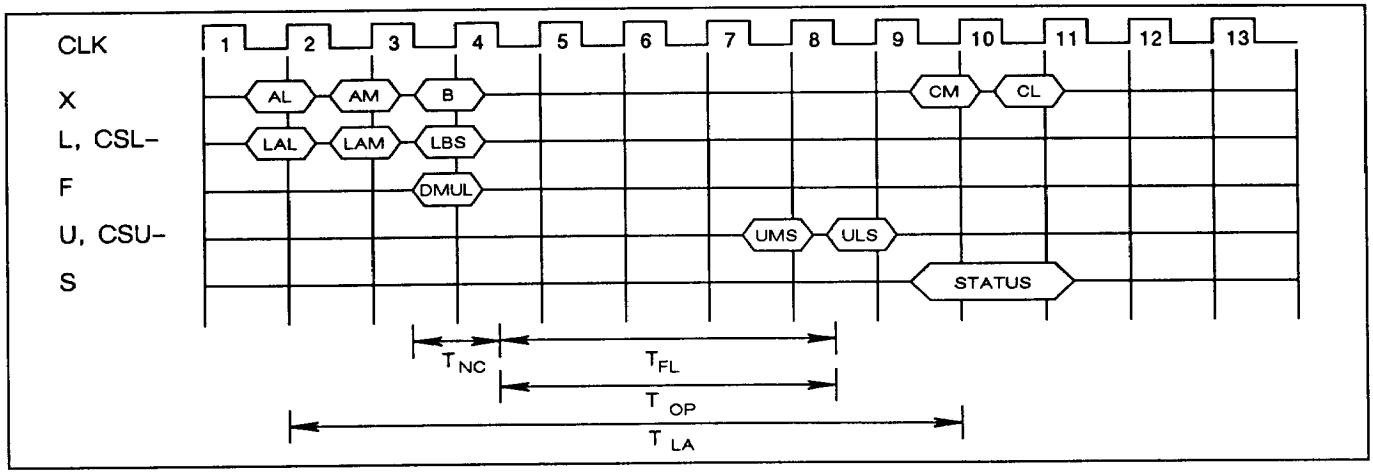


Figure 13. Mixed-mode Minimum Latency Multiply

Timing, continued

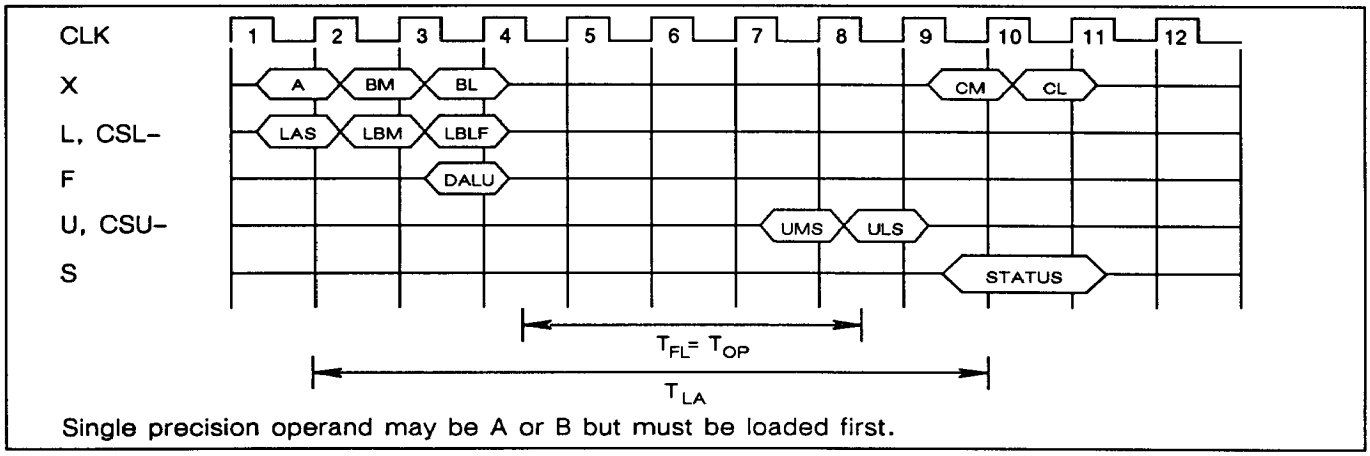


Figure 14. Mixed-mode ALU Operation

ALTERNATE LOADING SEQUENCES

Alternate loading sequences are those which cause flowthrough or encode times to be greater than the preferred loading sequences. Except for mixed-mode, operands may be loaded in any order. Alternate loading sequences are guaranteed by the T_{FL} and T_{NC} specifications but may not be optimum. mixed-mode operands must be loaded in the order specified by figures 13 and 14.

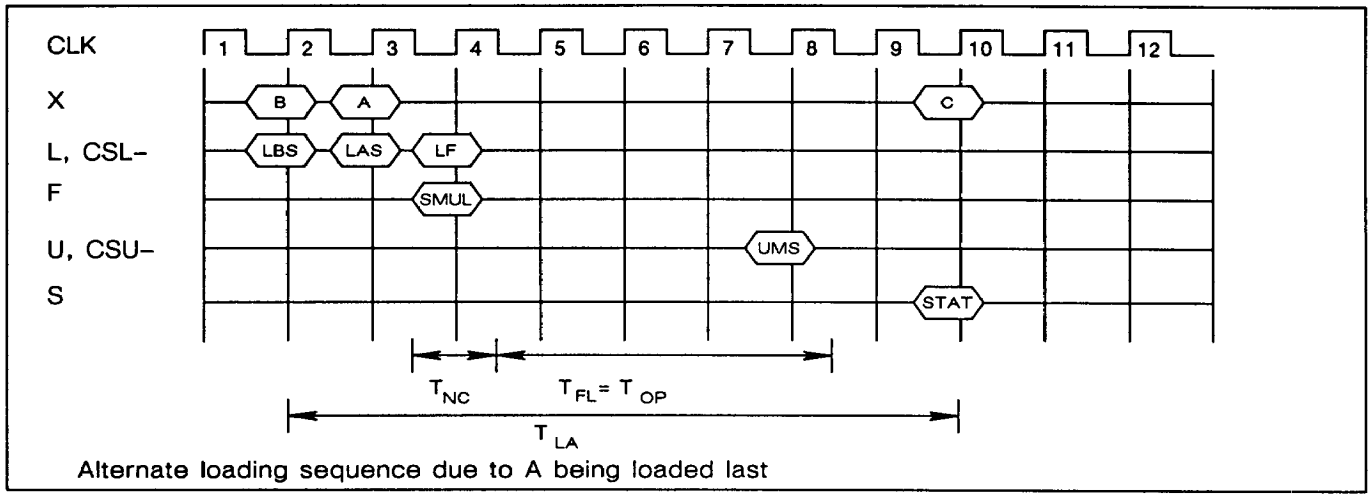


Figure 15. Single Precision Multiply, B Operand First

Timing, continued

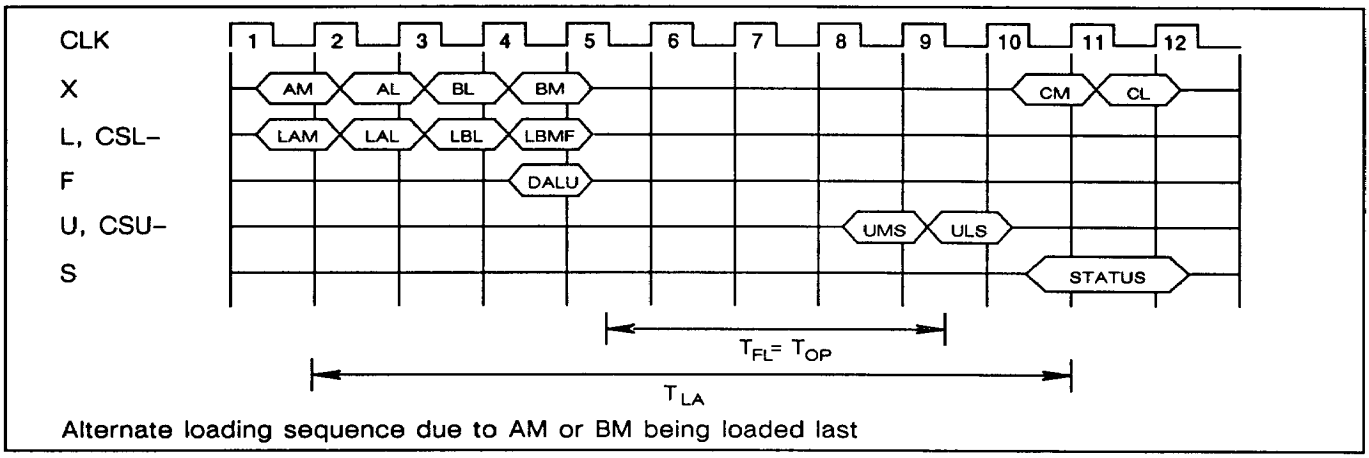


Figure 16. Double Precision ALU Operation, MSW Loaded Last

OPTIMIZING FLOATING POINT PERFORMANCE

On the WTL 1164, if the least significant half of the B operand is loaded first, the floating point multiplication can be started before loading the most significant half of the B operand. (See figures 17, 18 and 19.) The most significant half must be loaded on the following cycle and the cycle time must be less than the accumulate time, T_{AC}. This is possible because the initial accumulation operation requires only the least significant word of the B operand. The preferred loading order for the A operand is LSW followed by MSW.

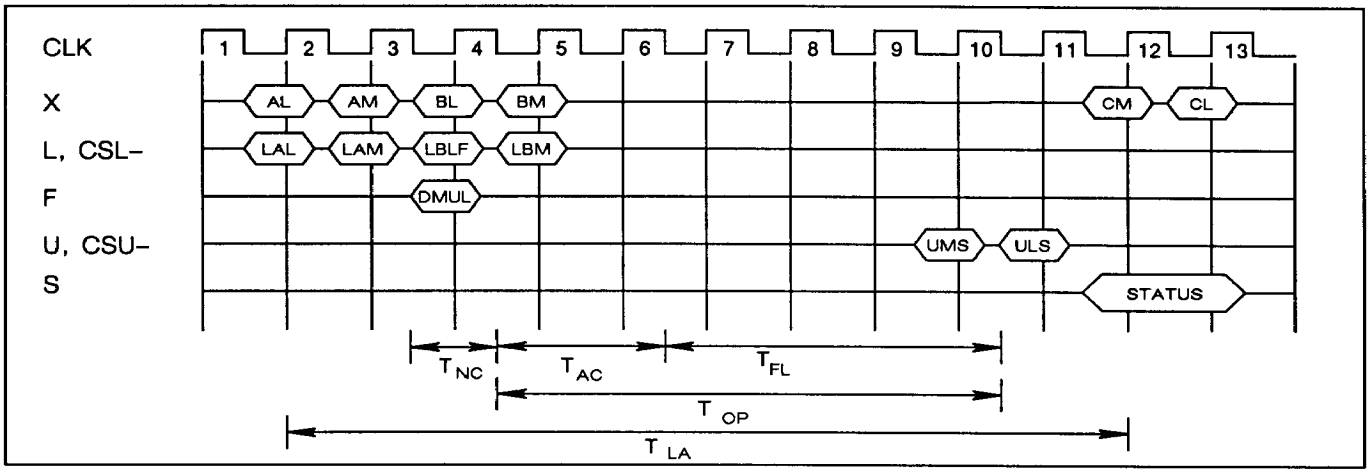


Figure 17. Double Precision Minimum Latency Multiply, Preferred Loading

Timing, continued

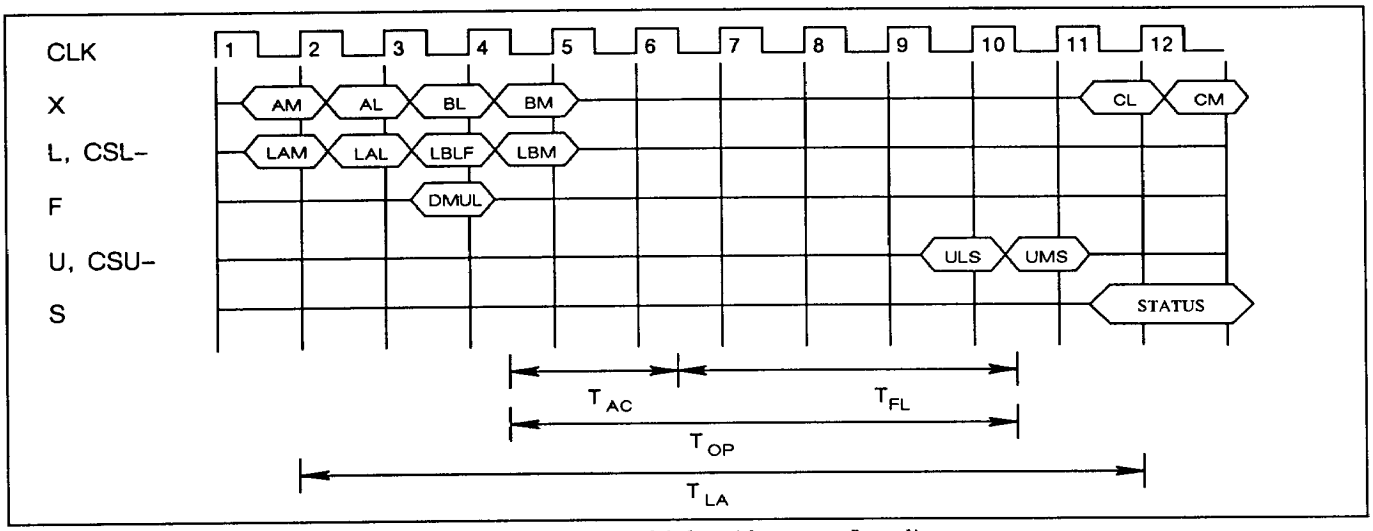


Figure 18. Double Precision Minimum Latency Multiply, Alternate Loading

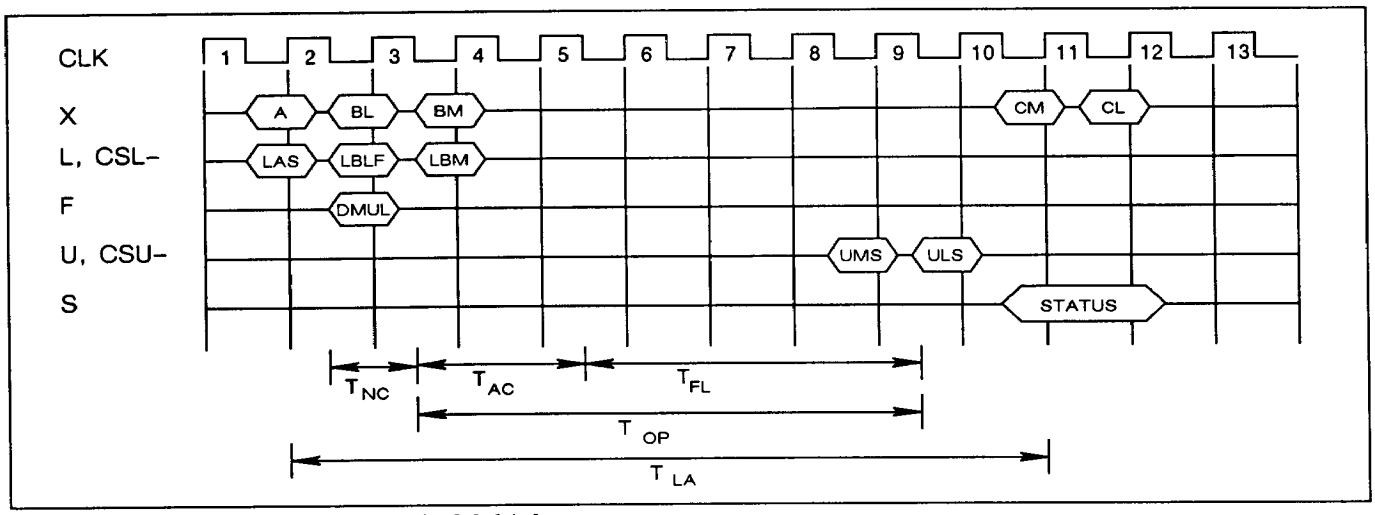


Figure 19. Alternate Mixed Mode Multiply

The WTL 1164/1165 contain a result latch at the end of the floating point hardware, allowing results to be loaded directly into one or both of the operand registers (a chained operation) or to start another operation before unloading the result (a pipelined operation).

Results may be chained from a result register back to an operand register by selecting both a load and unload operation on the same clock cycle. The result is passed onto the internal bus and then clocked into the indicated operand register. Normally, the result is also presented on the X Bus on the next clock cycle, which can be useful for ensuring that sufficient internal state is retained externally for aborting and later resuming an operation of this kind. If avoiding driving the X Bus is desirable, it can be tri-stated by bringing the OE– pin to an inactive (high) level. The time for a series of operations where the result is chained back to one of the operand registers, and another operand is loaded from the X Bus may be calculated as:

$$T_{RC} = T_{CY} + T_{OP}$$

For single precision operations (see Figure 20), and for double precision operations (see Figure 21), the time for each chained operation is:

$$T_{RC} = 3 * T_{CY} + T_{OP}$$

Timing, continued

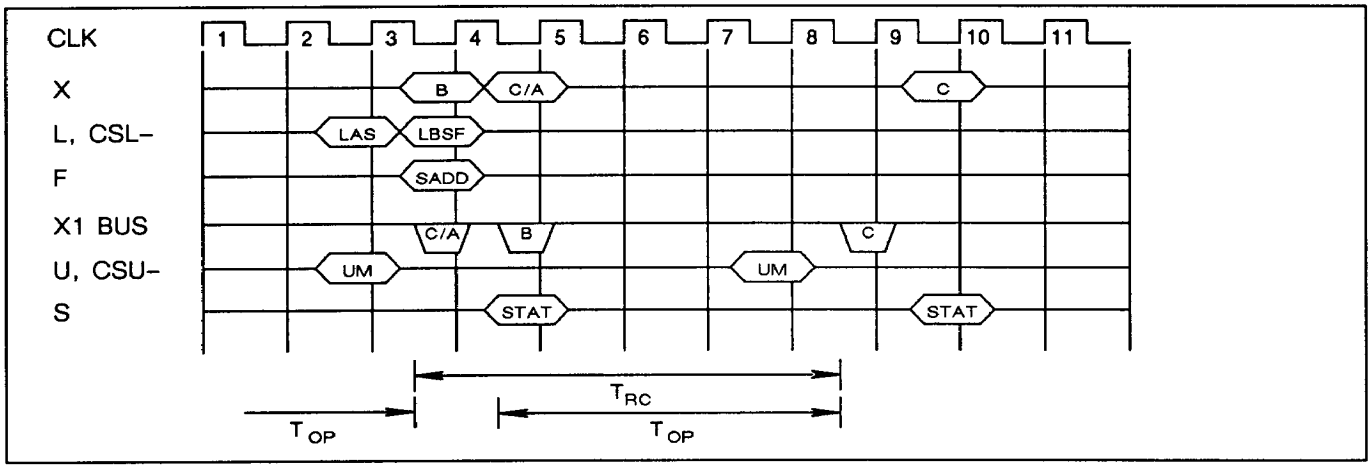


Figure 20. Single Precision Chained Add Operation

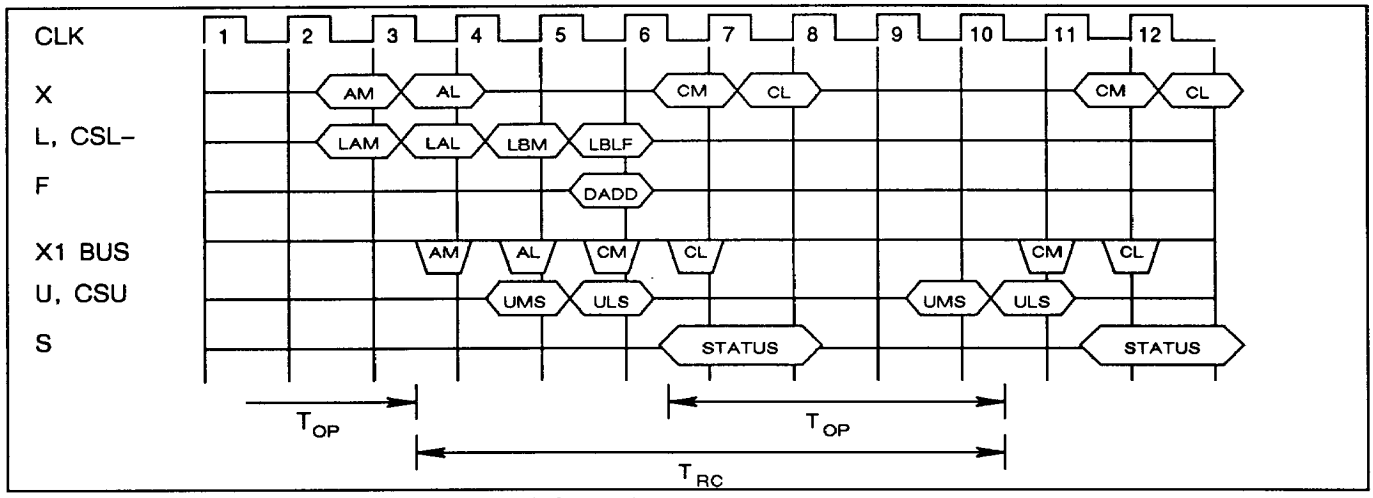


Figure 21. Double Precision Chained Add Operation

Timing, continued

The WTL 1164/1165 may be used in a pipelined fashion by loading new operands and starting a second operation before unloading the result of the previous operation. This previous result must be unloaded by selecting unload operations immediately after loading the final operand for the second operation. Due to the one cycle delay between the external X Bus and the internal X1 Bus in each direction, two cycles will elapse before the result appears on the bus. In order to assure that sufficient time to unload the result before the completion of the next operation reloads the result register, T_{OP} must be at least $3*T_{CY}$ for single precision results and at least $4*T_{CY}$ for double precision results. The effective pipeline rate can be calculated from the formula:

$$T_{RC} = \max (T_{CY} + T_{OP} , 5 * T_{CY})$$

For single precision operations (see Figure 22), and for double precision operations (see Figure 23), the effective pipeline rate is:

$$T_{RC} = \max (3 * T_{CY} + T_{OP} , 8 * T_{CY})$$

If a double precision multiply is performed when loading the least-significant B operand first, the effective pipeline rate is:

$$T_{RC} = \max (2 * T_{CY} + T_{OP} , 8 * T_{CY})$$

Note that T_{OP} must be at least $4*T_{CY}$ for this operation.

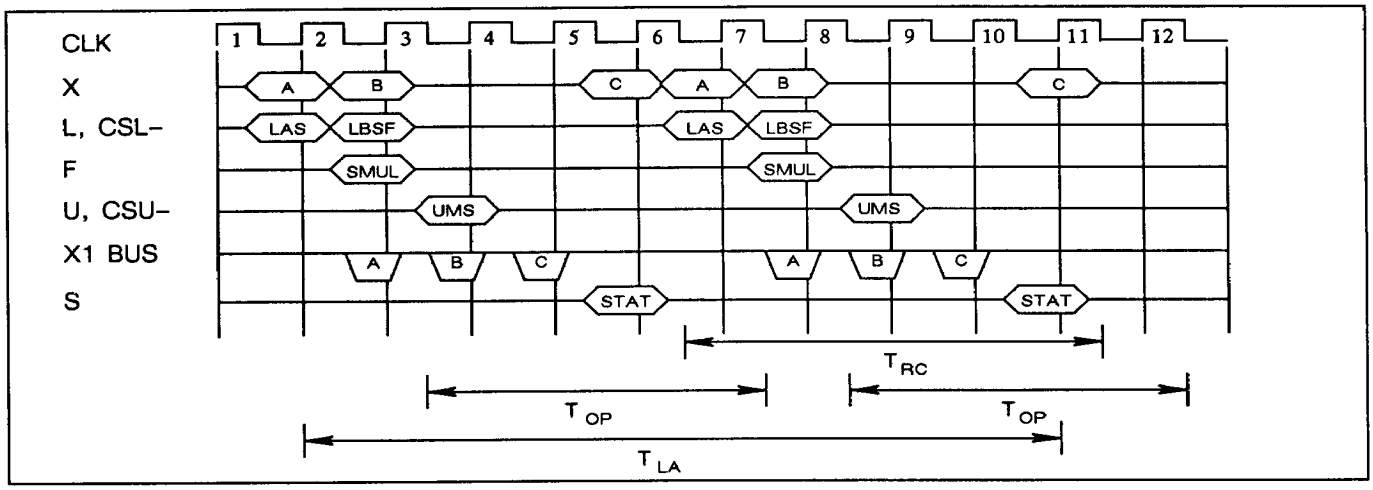


Figure 22. Single Precision Pipelined Multiply

Timing, continued

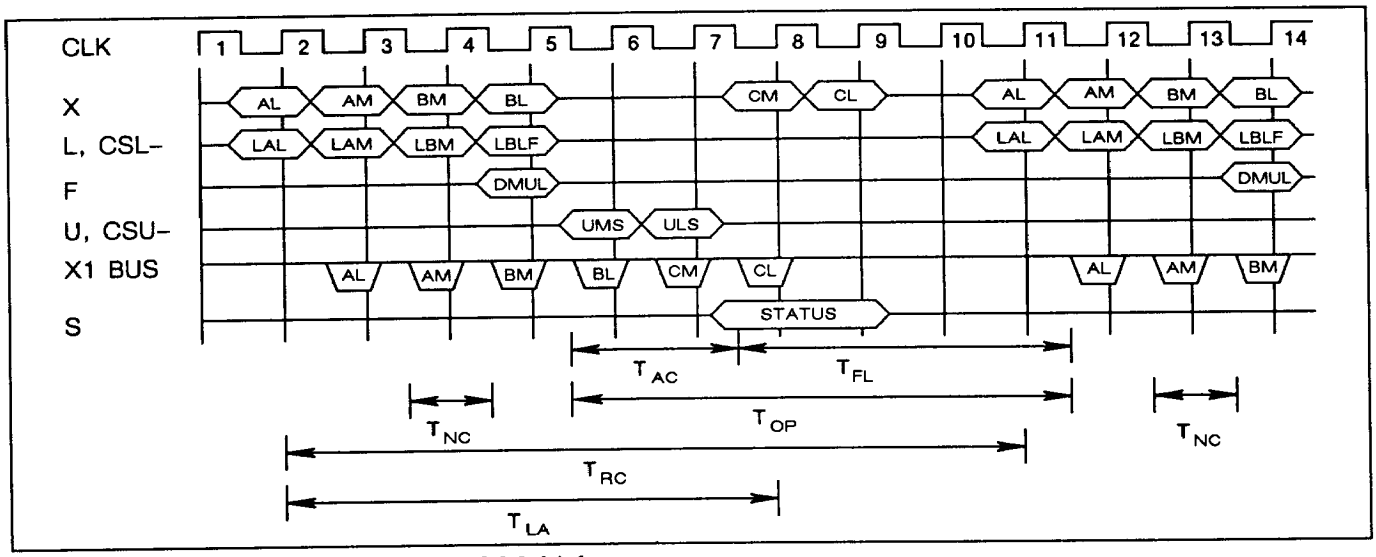


Figure 23. Double Precision Pipelined Multiply

USING FASTER AND SLOWER CLOCK RATES

Because the WTL 1164 and WTL 1165 use both clocked and unclocked logic in the arithmetic data path units, certain optimizations can be made when the parts are operated with clock rates that differ from the standard maximum clock rate. This permits greater performance in applications where the standard clock rate cannot be easily maintained.

Application Note

WEITEK FLOATING POINT COPROCESSOR INTERFACE

The simple interface and high speed of the WTL 1164/1165 make it an ideal chip set for accelerating the floating point performance of popular microprocessors. All of the control required to implement the interface between the microprocessor and the WTL 1164/1165 can be put onto one chip. The WTL 1163 furnishes the interface between the WTL 1164/1165 and Intel's 80386. The three chip set can then provide 3.5 megaWhetstone performance. The chip set is fully supported with Fortran, Pascal and C compilers. A block diagram of the three chip set showing the system interface is shown below.

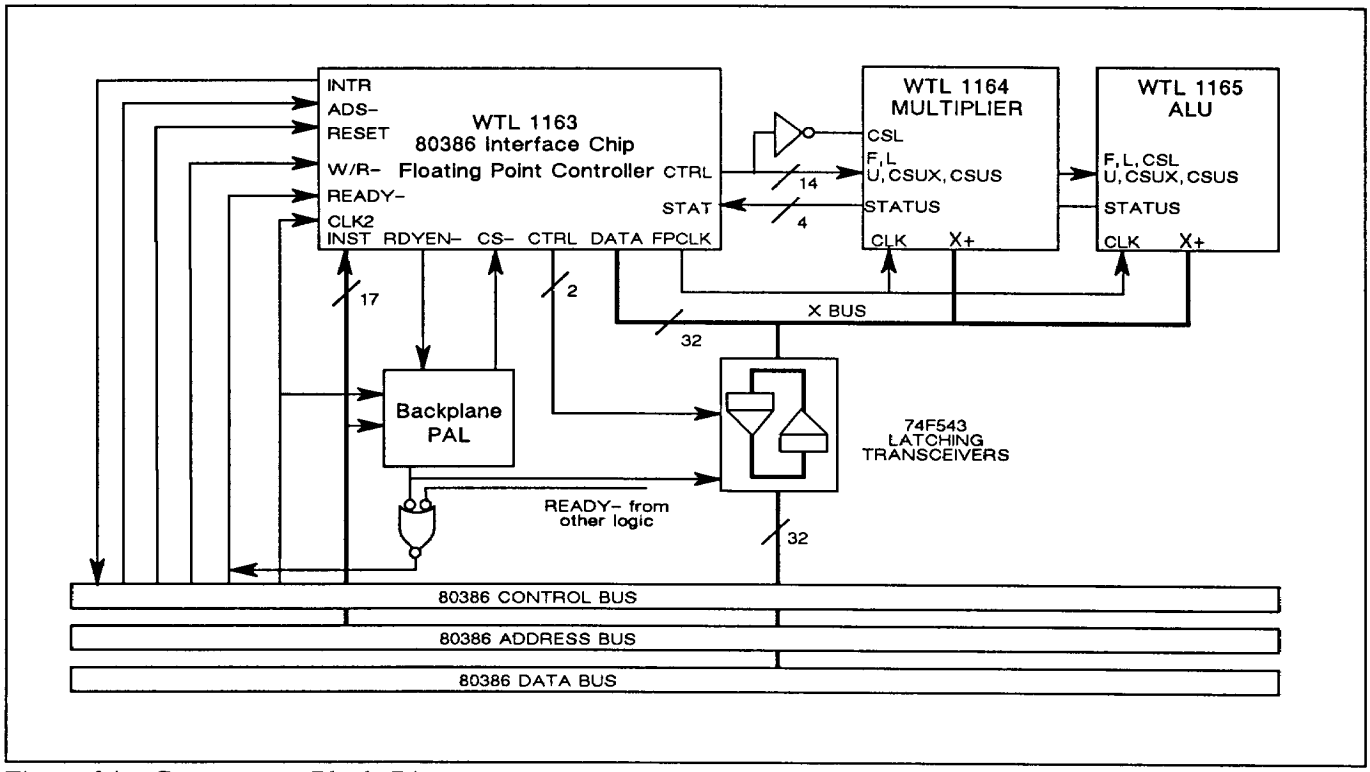


Figure 24. Coprocessor Block Diagram

For more information on coprocessor interface chips, call WEITEK.

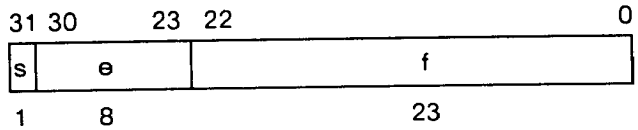
IEEE Compatibility

The IEEE Standard for Binary Floating Point specifies floating point processor data formats, rounding modes and exception handling. The WTL 1164 and WTL 1165 conform to the specification. The discussion below reviews IEEE implementation on the WTL 1164/1165. A separate note describes how denormalized numbers (DNRMs) are handled.

DATA FORMATS

The WTL 1164/1165 perform both 32-bit and 64-bit IEEE standard floating point operations. The 32-bit format has a 24-bit sign-magnitude fraction field and an 8-bit exponent, in the following format:

IEEE STANDARD SINGLE PRECISION



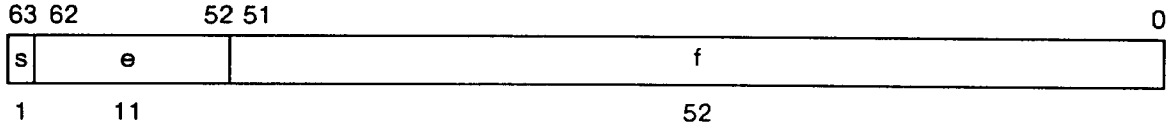
e	f	value	name
255	not 0	none	NaN (Not A Number)
255	0	$(-1)^s$ * infinity	Infinity
1 .. 254	any	$(-1)^s * 2^{e-127} * (1.f)$	Normalized number
0	not 0	$(-1)^s * 2^{-126} * (0.f)$	Denormalized number
0	0	$(-1)^s * 0$	Zero

Exponent values for normalized single precision numbers range from one to 254, with exponents of zero and 255 being reserved for special operands. To calculate the value of a number in this format, the exponent is decremented by 127 (the "exponent bias" is +127), and the fraction has a one inserted before the binary point. (This is called the hidden bit.) The value of the number is then $(-1)^s \times 2^{e-127} \times (1.f)$.

The 64-bit format has a 53-bit signed-magnitude fraction field and an 11-bit exponent, in the following format:

IEEE Compatibility, continued

IEEE STANDARD DOUBLE PRECISION



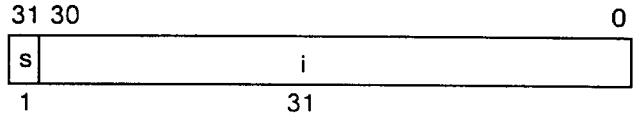
e	f	value	name
2047	not 0	none	NaN (Not A Number)
2047	0	$(-1)^s$ * infinity	Infinity
1 .. 2046	any	$(-1)^s * 2^{e-1023} * (1.f)$	Normalized number
0	not 0	$(-1)^s * 2^{-1022} * (0.f)$	Denormalized number
0	0	$(-1)^s * 0$	Zero

Exponent values for normalized double precision numbers range from one to 2,046 with exponents of zero and 2,047 being reserved for special operands. To calculate the value of a number in this format, the exponent is decremented by 1,023 (the exponent bias is +1,023), and the fraction has a one inserted before the binary point. Thus the value of a double precision number is $(-1)^s \times 2^{e-1023} \times (1.f)$.

INTEGER FORMAT

The WTL 1164/1165 family supports conversion operations to and from 32-bit two's complement integers. The format is given below.

INTEGER FORMAT



s	value	name
0	i	positive integer
1	$i - 2^{32}$	negative integer

IEEE Compatibility, continued

Several number types are required to implement the standard. These are normalized numbers, denormalized numbers, wrapped numbers, infinity and zero.

Normalized Numbers (NRM)

Most caculations are performed on normalized numbers. For single precision, normalized numbers have an exponent that ranges from 00000001 to 11111110 (one to 254) and a normalized fraction field (the leftmost or hidden bit is a one). In decimal notation, this allows one to represent a range of both positive and negative numbers from roughly 10^{38} to 10^{-38} with accuracy to seven decimal places. Double precision numbers have an exponent ranging from one to 2,046 and a normalized fraction field.

Infinity (INF)

Infinity has an exponent of all ones and a fraction field equal to zero. Both positive and negative infinity are allowed.

ZERO

ZERO has an exponent of zero, a hidden bit equal to zero and a value of zero in the fraction field. Both +0 and −0 are supported.

Wrapped Numbers (WNRM)

A wrapped number is created by normalizing a DNRM's fraction field and subtracting from the exponent the number of shift positions required. (Normalizing is accomplished by left shifting until the hidden bit contains a one.) The value of the exponent is equal to (1 − (the number of shifts)) and is represented in two's complement.

Unrounded Normalized Number (UNRM)

A UNRM is the result of an operation that has a magnitude less than the minimum representable normalized number. A UNRM has a normalized fraction field, a wrapped exponent and a hidden bit equal to one. The minimum UNRM is attained by multiplying two DNRM.MINs. UNRMs are turned into DNRM values using the ALU's unwrap function.

Denormalized Numbers (DNRM)

Denormalized numbers have a zero exponent and a denormalized (hidden bit equal to zero) non-zero fraction field.

Not A Number (NaN)

NaN, or not a number, is a special data format usually used as a flag for data flow control, for uninitialized variables or to signify an invalid operations such as 0 x ∞. The format for a NaN is an exponent of all ones and a non-zero fraction.

IEEE SINGLE PRECISION FORMATS SUPPORTED BY THE WTL 1164/1165

OPERAND	EXPONENT	FRACTION	HIDDEN BIT	VALUE
NaN	255	ANY	N/A	NONE
INFINITY	255	ALL 0's	1	$(-1)^{s} \infty$
NORM.MAX	254	ALL 1's	1	$(-1)^{s} \times 2^{127} \times (2)$
NORM	1 to 254	ANY	1	$(-1)^{s} \times 2^{e-127} \times (1.f)$
NORM.MIN	1	ALL 0's	1	$(-1)^{s} \times 2^{-126} \times (1)$
DNRM.MAX	0	ALL 1's	0	$(-1)^{s} \times (2^{-126} - 2^{-149})$
DNRM	0	ANY	0	$(-1)^{s} \times 2^{-126} \times (0.f)$
DNRM.MIN	0	000...01	0	$(-1)^{s} \times 2^{-126} \times 2^{-23}$
WNRM.MAX	0	ALL 1's	1	$(-1)^{s} \times 2^{-126}$
WNRM	0 to (−22)	ANY	1	$(-1)^{s} \times 2^{e-127} \times (1.f)$
WNRM.MIN	−22	ALL 0's	1	$(-1)^{s} \times 2^{-149}$
UNRM.MAX	0	ALL 1's	1	$(-1)^{s} \times 2^{-126}$
UNRM.MIN	−171	ALL 0's	1	$(-1)^{s} \times 2^{-298}$
ZERO	0	ALL 0's	0	$(-1)^{s} 0$

IEEE Compatibility, continued

The same formats are supported in double precision. The range of double precision numbers and their values are obtained by substituting the double precision mantissa and exponent in the pattern shown above. A partial table of values is given below.

E	F	VALUE	NAME	MNEMONIC
2047	Not all zeros	None	Not a number	NaN
2047	All zeros	$(-1)^S *$ Infinity	Infinity	INF
1-2046	Any	$(-1)^S * 2^{E-1023} * (1.F)$	Normalized number	NOR
0	Not all zeros	$(-1)^S * 2^{-1022} * (0.F)$	Denormalized number	DNRM
0	Zero	$(-1)^S * 0$	Zero	ZERO

ROUNDING OPTIONS

The WTL 1164/1165 support all four rounding modes of the IEEE standard — round to nearest, round toward zero, round toward plus infinity and round toward minus infinity. Rounding may be biased or unbiased. Biased rounding introduces a small offset in the direction of the bias. Positive bias, negative bias or a bias toward zero are specified in the IEEE format. Unbiased rounding rounds the result to the nearest representable number. In the case of a number exactly halfway between two representable numbers, the number is rounded toward the closest even number, resulting in half the numbers rounding up and half rounding down, on average.

Round To Nearest

Rounds the result to the nearest representable value. If two numbers are equally near the result, the even number is chosen.

Round Toward Zero

Rounds the result to the value closest to but not greater than the magnitude of the result.

Round Toward Plus Infinity

Rounds the result to the value closest to but not less than the result.

Round Toward Minus Infinity

Rounds the result to the value closest to but not greater than the result.

EXCEPTION HANDLING

The WTL 1164/1165 generate the exceptions specified in the IEEE standard for floating point operations. The status word corresponding to an operation is propagated through the array and pipeline registers in synchrony with the operands and partial results. The status outputs are registered when the output data is clocked into the output register and is valid until the next rising edge of the clock. The status outputs also indicate the result of a Compare operation. Thus Compare precludes the indication of other exceptions.

Inexact (NXT)

NXT is generated on the WTL 1164/1165 whenever there is a loss of accuracy. The chips compute results to higher precision than the number of mantissa bits that appear in the result. If any of the fraction bits less than the LSB was equal to one prior to rounding, then the inexact bit will be high. NXT will be signaled in the WTL 1165 if there is a partial or complete loss of significance in a float-to-fixed operation.

IEEE Compatibility, continued

Overflow (OVF)

OVF is generated when the result of a floating point operation overflows the largest representable normalized number. The result produced at the output is either infinity or the largest representable positive or negative number, depending upon the rounding mode as follows:

+MAX.NRM	if ((RM or RZ) and the result is positive)
-MAX.NRM	if ((RP or RZ) and the result is negative)
$+\infty$	if ((RN or RP) and the result is positive)
$-\infty$	if ((RN or RM) and the result is negative)

Overflow is also generated when converting floating-point-to-fixed-point and the result overflows the 32-bit format.

Underflow (UNF)

When the result of an operation after rounding is less than the minimum normalized number, UNF is asserted. A result of exactly zero does not underflow.

Divide By Zero (DVZ)

The WTL 1165 will assert a DVZ exception when performing division on a normalized dividend and a zero divisor. The result is a properly signed infinity.

Invalid Operation (INV)

INV (status codes 12 through 15) will be asserted if a NaN is an input or if an invalid operation occurs. The invalid WTL 1164 operation is $\infty \cdot 0$. Invalid WTL 1165 operations include $\infty \div \infty$, $0 \div 0$, subtraction of like infinities ($\infty - \infty$) and addition of opposite infinities ($+\infty - \infty$). When either input is NaN the status code indicates the source of the NaN operand(s). The result of any invalid operation (12 through 15) is NaN with the fraction and exponent all ones. The sign bit is zero.

Denormalized Input (DIN)

DIN is asserted whenever an operand is denormalized and the chip is in IEEE mode.

GRADUAL UNDERFLOW

The minimum normalized number has an exponent of one and a fraction field of zero. Zero has an exponent of zero and a fraction field of all zeros. This gives users the ability to deal with numbers between NORM.MIN and ZERO. These numbers are known as denormals. Their format is given in the number format section. The IEEE standard has specified gradual underflow as the way to handle denormals. Many of the features of the WTL 1164/1165 are included to deal with denormals in a manner consistent with IEEE Standard 754, Version 10.0. Since denormals are very close to zero, many applications can substitute zero for a denormal without a signifcant loss of accuracy. For these applications, a "FAST" mode is included which substitutes zero for all denormalized inputs to the WTL 1164 and 1165. ZERO is also inserted for all UNRM outputs in "FAST" mode.

For all arithmetic operations other than divide, the WTL 1165 handles denormalized inputs directly as it would handle any other number.

Unfortunately, a floating point multiplier must either operate exclusively on normalized numbers or suffer large cost and performance penalties in dealing directly with denormals. A normalized format that yields an equivalent to a given denormalized number is the wrapped format. The number format table shows the equivalence of wrapped and denormalized numbers. To translate a denormalized number to a wrapped number, the fraction is normalized (shifted up so that a one is in the hidden bit) and one is subtracted from the exponent for every position shifted. The WTL 1164 can correctly multiply either two wrapped numbers or a wrapped and a normalized number. To understand the full procedure, consider the following case.

Assume one of the two input operands to the WTL 1164 is a denormalized number. Four cycles after the input, the denorm exception is flagged. The denormalized operand must then be sent to the WTL 1165 to be wrapped. Once wrapped, the operand can be sent back to the WTL 1164 for multiplication. The result of the multiplication will either be a normalized number or a UNRM.

If the result is a UNRM, status bit So indicates either UNF (if all the truncated bits are equal to zero) or UNF-NXT (if any of the truncated bits is equal to one).

No rounding will occur regardless of the rounding mode specified.

IEEE Compatibility, continued

The underflowed number may then be sent to the WTL 1165 for "unwrapping". To unwrap a number, the fraction field is shifted right and the exponent incremented by one for each shift position. Status bit So must be used to conditionally execute the Unwrap Inexact or Unwrap Exact instruction. The rounding must be performed in the ALU. The unwrapping may have three possible results:

RESULT	EXCEPTION	COMMENT
DNRM	UNF	When the denormalized result is exact. Note that his result is possible only if the UNRAP EXACT instruction is is possible (i. e., both the input and the result must be exact.)
DNRM	UNF-NXT	If the UNWRAP INEXACT instruction is executed *or* if the result of the UNWRAP EXACT instruction is inexact.
ZERO	UNF-NXT	The result is zero, but the unwrapping has resulted in the loss of precision.

Operations

The following tables delineate the results that are obtained for all combinations of input data formats and rounding options, for both the WTL 1164 and the WTL 1165 in IEEE as well as "FAST" mode. The format used in the tables is STATUS:(status code)-Result.

TABLE 5: FLOATING POINT ADD/SUBTRACT ("FAST" MODE)

A/B	ZERO	DNRM	NRM	INF	NaN
NaN	INV:12-Nan	INV:12-NaN	INV:12-NaN	INV:12-NaN	INV:14-NaN
INF	OK*:1-INF	OK*:1-INF	OK*:1-INF	INV:15-NaN (2) OK*:1-INF (1)	INV:13-NaN
NRM	OK:2-NRM	OK:2,3-NRM	OVF:5-(4) OK:2,3-NRM UNF:6,7-ZERO OK:0-ZERO	OK*:1-INF	INV:13-NaN
DNRM	OK:0-ZERO (3)	OK:0-ZERO	OK:2,3-NRM	OK*:1-INF	INV:13-NaN
ZERO	OK:0-ZERO (3)	OK:0-ZERO (3)	OK:2-NRM	OK*:1-INF	INV:13-NaN

*If an operand is INF, OK will be signaled rather than OVF (see Note 1)

Operations, continued

Notes:

1. +INF+INF → +INF
   −INF−INF → −INF

2. +INF−INF → NaN (invalid operation)
   −INF+INF → NaN (invalid operation)

3. +ZERO+ZERO → +ZERO (RN, RZ, RP, RM)
   −ZERO−ZERO → −ZERO (RN, RZ, RP, RM)
   +ZERO−ZERO → +ZERO (RN, RZ, RP)
   +ZERO−ZERO → −ZERO (RM)
   −ZERO+ZERO → +ZERO (RN, RZ, RP)
   −ZERO+ZERO → −ZERO (RM)

4. OVF will produce INF or MAX.NRM, depending upon the rounding mode:
   +MAX.NRM IF [(RM, RZ) AND (RESULT IS +)]
   −MAX.NRM IF [(RP, RZ) AND (RESULT IS −)]
   +INF IF [(RN, RP) AND (RESULT IS +)]
   −INF IF [(RN, RM) AND (RESULT IS −)]

TABLE 6: FLOATING POINT MULTIPLICATION ("FAST" MODE)

A/B	ZERO	DNRM	NRM	INF	NaN
NaN	INF:12−NaN	INV:12−NaN	INV:12−NaN	INV:12−NaN	INV:14−NaN
INF	INV:15−NaN	INV:15−NaN	OK:1−INF	OK:1−INF	INV:13−NaN
NRM	OK:0−ZERO	OK:0−ZERO	OVF:5−(1) OK:2,3−NRM UNF:6,7−ZERO	OK:1−INF	INV:13−NaN
DNRM	OK:0−ZERO	OK:−ZERO	OK:0−ZERO	INV:15−NaN	INV:13−NaN
ZERO	OK:0−ZERO	OK:0−ZERO	OK:0−ZERO	INV:15−NaN	INV:13−NaN

Notes:

1. OVF will produce INF or MAX.NRM, depending upon the rounding mode:
   +MAX.NRM IF [(RM, RZ) AND (RESULT IS +)]
   −MAX.NRM IF [(RP, RZ) AND (RESULT IS −)]
   +INF IF [(RN, RP) AND (RESULT IS +)]
   −INF IF [(RN, RM) AND (RESULT IS −)]

Operations, continued

TABLE 7: FLOATING POINT ADD/SUBTRACT (IEEE MODE)					
A/B	ZERO	DNRM	NRM	INF	NaN
NaN	INF:12–NaN	INV:12–NaN	INV:12–NaN	INV:12–NaN	INV:14–NaN
INF	OK:1–INF	OK:1–INF	OK:1–INF	INV:15–NaN (2) OK:1–INF (1)	INV:13–NaN
NRM	OK:2–NRM	OVF:5–(4) OK:2,3–NRM UNF:6,7–UNRM	OVF:5–(4) OK:2,3–NRM UNF:6,7–UNRM OK:0–ZERO	OK:1–INF	INV:13–NaN
DNRM	UNF:6–UNRM	OK:0–ZERO (3) UNF:6–UNRM OK:2–NRM	OK:2,3–NRM UNF:6,7–UNRM OVF:5–(4)	OK:1–INF	INV:13–NaN
ZERO	OK:0–ZERO (3)	UNF:6–UNRM	OK:2–NRM	OK:1–INF	INV:13–NaN

Notes:

1. +INF+INF → +INF
   −INF−INF → −INF

2. +INF−INF → NaN
   −INF+INF → NaN

3. +ZERO+ZERO → +ZERO (RN, RZ, RP, RM)
   −ZERO−ZERO → −ZERO (RN, RZ, RP, RM)
   +ZERO−ZERO → +ZERO (RN, RZ, RP)
   +ZERO−ZERO → −ZERO (RM)
   −ZERO+ZERO → +ZERO (RN, RZ, RP)
   −ZERO+ZERO → −ZERO (RM)

4. OVF will produce INF or MAX.NRM, depending upon the rounding mode:
   +MAX.NRM IF [(RM, RZ) AND (RESULT IS +)]
   −MAX.NRM IF [(RP, RZ) AND (RESULT IS −)]
   +INF IF [(RN, RP) AND (RESULT IS +)]
   −INF IF [(RN, RM) AND (RESULT IS −)]

Operations, continued

TABLE 8: FLOATING POINT MULTIPLICATION (IEEE MODE)					
A/B	ZERO	DNRM	NRM	INF	NaN
NaN	INF:12-NaN	INV:12-NaN	INV:12-NaN	INV:12-NaN	INV:14-NaN
INF	INV:15-NaN	OK:1-INF	OK:1-INF	OK:1-INF	INV:13-NaN
NRM	OK:0-ZERO	DIN:9-(U)*	OVF:5-(2) OK:2,3-NRM UNF:6,7-UNRM	OK:1-INF	INV:13-NaN
DNRM	OK:0-ZERO	DIN:10-(U)*	DIN:8-(U)*	OK:1-INF	INV:13-NaN
ZERO	OK:0-ZERO	OK:1-ZERO	OK:1-ZERO	INV:15-NaN	INV:13-NaN

Note:

(1) (U) result is undefined.

(2) OVF will produce INF or MAX.NRM, depending upon the rounding mode:

+MAX.NRM	IF [(RM, RZ) AND (RESULT IS +)]
+MAX.NRM	IF [(RP, RZ) AND (RESULT IS -)]
+INF	IF [(RN, RP) AND (RESULT IS +)]
-INF	IF [(RN, RM) AND (RESULT IS -)]

TABLE 9: FLOATING POINT DIVIDE ("FAST" MODE)					
A/B	ZERO	DNRM	NRM	INF	NaN
NaN	INV:12-NaN	INV:12-NaN	INV:12-NaN	INV:12-NaN	INV:14-NaN
INF	OK:1-INF	OK:1-INF	OK:1-INF	INV:15-NaN	INV:13-NaN
NRM	DVZ:11-INF	DVZ:11-INF	OK:2,3-NRM OVF:5-(1) UNF:6,7-UNRM	OK:0-ZERO	INV:13-NaN
DNRM	INV:15-NaN	INV:15-NaN	OK:0-ZERO	OK:0-ZERO	INV:13-NaN
ZERO	INV:15-NaN	INV:15-NaN	OK:0-ZERO	OK:0-ZERO	INV:13-NaN

Operations, continued

Note:

(1) OVF will produce INF or MAX.NRM, depending upon the rounding mode:

+MAX.NRM	IF	[(RM, RZ) AND (RESULT IS +)]
+MAX.NRM	IF	[(RP, RZ) AND (RESULT IS -)]
+INF	IF	[(RN, RP) AND (RESULT IS +)]
-INF	IF	[(RN, RM) AND (RESULT IS -)]

TABLE 10: FLOATING POINT DIVIDE (IEEE MODE)

A/B	ZERO	DNRM	NRM	INF	NaN
NaN	INV:12-NaN	INV:12-NaN	INV:12-NaN	INV:12-NaN	INV:14-NaN
INF	OK:1-INF	BDIN:9-(2)	INF:1-INF	INV:15-NaN	INV:13-NaN
NRM	DVZ:11-INF	BDIN:9-(2)	OK:2,3-NRM OVF:5-(1) UNF:6,7-UNRM	OK:0-ZERO	INV:13-NaN
DNRM	DVZ:11-INF	ABDIN:10-(2)	ADIN:8-(2)	ADIN:8-(2)	INV:13-NaN
ZERO	INV:15-NaN	BDIN:9-(2)	OK:0-ZERO	OK:0-ZERO	INV:13-NaN

Note:

(1) OVF will produce INF or MAX.NRM, depending upon the rounding mode:

+MAX.NRM	IF	[(RM, RZ) AND (RESULT IS +)]
+MAX.NRM	IF	[(RP, RZ) AND (RESULT IS -)]
+INF	IF	[(RN, RP) AND (RESULT IS +)]
-INF	IF	[(RN, RM) AND (RESULT IS -)]

(2) Result is undefined.

Operations, continued

TABLE 11: FLOATING POINT COMPARE STATUS								
A/B	NaN	-INF	-NRM	-DNRM	ZERO	+DNRM	+NRM	+INF
NaN	U:15	U:15	U:15	U:15	U:15	U:15	U:15	U:15
+INF	U:15	G:2	G:2	G:2	G:2	G:2	G:2	E:0
+NRM	U:15	G:2	G:2	G:2	G:2	G:2	:0, 1, 2	L:1
+DNRM	U:15	G:2	G:2	G:2	G:2	:0, 1, 2	L:1	L:1
ZERO	U:15	G:2	G:2	G:2	E:0	L:1	L:1	L:1
-DNRM	U:15	G:2	G:2	:0, 1, 2	L:1	L:1	L:1	L:1
-NRM	U:15	G:2	:0, 1, 2	L:1	L:1	L:1	L:1	L:1
-INF	U:15	E:0	L:1	L:1	L:1	L:1	L:1	L:1

FORMAT: Condition: Status Code(s)

U:15 := unordered (status = 15)
E:0 := equal (status = 0)
L:1 := A < B (status = 1)
G:2 := A > B (status = 2)
:0, 1, 2 := may be A = B, A < B, or A > B, depending upon data values

TABLE 12: CONVERT SINGLE TO DOUBLE			
F32 → F64			
F32 OPERAND	F64 RESULT	STATUS	COMMENTS
7FFFFFFF or FFFFFFFF	7FFFFFFF FFFFFFFF	12	A operand is NaN
7F800000	7FF00000 00000000	1	+INF
7F7FFFFF	47EFFFFF E0000000	2	Input operand is +MAX.NRM
3F800000	3FF00000 00000000	2	+1
00800000	38100000 00000000	2	Input operand is +MIN.NRM
007FFFFF	380FFFFF C0000000	2	Input operand is +MAX.DNRM Result = 0 in "FAST" mode
00000001	36A00000 00000000	2	Input operand is +MIN.DNRM Result = 0 in "FAST" mode
00000000	00000000 00000000	0	+ZERO

Note: Sign bit is orthogonal; it is directly copied from the input operand to the output result (except for NaN which is clamped to zero).

Operations, continued

TABLE 13: CONVERT DOUBLE TO SINGLE			
F64 → F32 (Round to Nearest)			
F64 OPERAND	F32 RESULT	STATUS	COMMENTS
7FFFFFFF FFFFFFFF	7FFFFFFF	12	A operand is NaN
7FF00000 00000000	7F800000	1	+INF
7FEFFFFF FFFFFFFF	7F800000	5	+MAX.NRM OVERFLOWS
47EFFFFF F0000000	7F800000	5	+OVF RESULT
47EFFFFF E0000000	7F7FFFFF	2	+MAX.NRM RESULT
3FF00000 00000000	3F800000	2	+1
38100000 00000000	00800000	2	+MIN.NRM RESULT
380FFFFF FFFFFFFF	00800000	3	Result after rounding is +MIN.NRM
38000000 00000000	00000000 (UNRM) 1165	6	Result underflows
36FFFFFF FFFFFFFF	77FFFFFF (UNRM) 1165	7	Produces UNRM + NXT
36A00000 00000000	75000000 (UNRM) 1165	6	+MIN.DNRM RESULT
00000000 00000001	40000000 (UNRM) 1165	7	Input is DNRM
00000000 00000000	00000000	0	ZERO

Note: Sign bit is orthogonal; it is directly copied from the input operand to the output result (except for NaN which is clamped to zero).

Operations, continued

TABLE 14: DOUBLE FLOAT			
I32 → F64			
I32 OPERAND	F64 RESULT	STATUS	COMMENTS
7FFFFFFF	41DFFFFF FFC00000	2	Largest Positive Integer
00000001	3FF00000 00000000	2	+1
00000000	00000000 00000000	0	ZERO
FFFFFFFF	BFF00000 00000000	2	-1
80000000	C1E00000 00000000	2	Largest Negative Integer

TABLE 15: SINGLE FLOAT			
I32 → F32			
I32 OPERAND	F32 RESULT	STATUS	COMMENTS
7FFFFFFF	4F000000	3	Largest Positive Integer
7FFFFFC0	4F000000	3	INEXACT
7FFFFF80	4EFFFFFF	2	EXACT
00000001	3F800000	2	+1
00000000	00000000	0	ZERO
FFFFFFFF	BF800000	2	-1
80000080	CEFFFFFF	2	EXACT
80000040	CF000000	3	INEXACT
80000000	CF000000	2	Largest Negative Integer

Operations, continued

TABLE 16: DOUBLE FIX			
F64 → I32 (Round to Nearest)			
F64 OPERAND	I32 RESULT	STATUS	COMMENTS
7FFFFFFF FFFFFFFF	7FFFFFFF	12	Input is NaN
7FF00000 00000000	7FFFFFFF	5	+INF
7FEFFFFF FFFFFFFF	7FFFFFFF	5	Input is +MAX.NRM
41DFFFFF FFC00000	7FFFFFFF	2	Largest Positive Integer Result
3FF00000 00000000	00000001	2	+1
3FE80000 00000000	00000001	3	INEXACT
00100000 00000000	00000000	3	Input is MIN.NRM
00000000 00000001	00000000	3	Input is MIN.DNRM
00000000 00000000	00000000	0	+ZERO
80000000 00000000	00000000	0	-ZERO
8FF00000 00000000	00000000	3	Small Negative Number
BFF00000 00000000	FFFFFFFF	2	-1
C1E00000 00000000	80000000	5	Largest Negative Integer Result
FFF00000 00000000	80000000	5	-INF
FFFFFFFF FFFFFFFF	7FFFFFFF	12	-NaN

Operations, continued

TABLE 17: SINGLE FIX			
F32 → I32			
F32 OPERAND	I32 RESULT	STATUS	COMMENTS
7FFFFFFF	7FFFFFFF	12	Input is NaN
7F800000	7FFFFFFF	5	+INF
7F7FFFFF	7FFFFFFF	5	Input is +MAX.NRM
4F000000	7FFFFFFF	5	+OVF
4EFFFFFF	7FFFFF80	2	EXACT
3F800000	00000001	2	+1
3F400000	00000001	3	INEXACT
00800000	00000000	3	Input is +MIN.NRM
00000001	00000000	3	Input is +MIN.DNRM
00000000	00000000	0	+ZERO
80000000	00000000	0	-ZERO
8F800000	00000000	3	Small Negative Number
BF800000	FFFFFFFF	2	-1
CEFFFFFF	80000080	2	Large Negative Number
CF000001	80000000	5	-OVF
FF800000	80000000	5	-INF

TABLE 18: DOUBLE WRAP DENORMALIZED VALUE			
F64 → W64			
F64 OPERAND	W64 RESULT	STATUS	COMMENTS
00000000 00000001	7CD00000 00000000	6	Input is +MIN.DNRM
00080000 00000000	00000000 00000000	6	Always Exact
000FFFFF FFFFFFFF	000FFFFF FFFFFFFE	6	Input is +MAX.DNRM

TABLE 19: SINGLE WRAP DENORMALIZED VALUE			
F32 → W32			
F32 OPERAND	W32 RESULT	STATUS	COMMENTS
00000001	75000000	6	Input is -MIN.DNRM
00400000	00000000	6	Always Exact
007FFFFF	007FFFFE	6	Input is -MAX.DNRM

Operations, continued

TABLE 20: DOUBLE UNWRAP EXACT VALUE			
U64 → D64			
U64 OPERAND	D64 RESULT	STATUS	COMMENTS
000FFFFF FFFFFFFF	00100000 00000000	3	Result is NRM + NXT
00000000 00000000	00080000 00000000	6	UNF + EXACT
7FFFFFFF FFFFFFFF	00080000 00000000	7	UNF + NXT
40000000 00000000	00000000 00000000	7	UNF + NXT

TABLE 21: SINGLE UNWRAP EXACT VALUE			
U32 → D32			
U32 OPERAND	D32 RESULT	STATUS	COMMENTS
007FFFFF	00800000	3	Result is NRM
007FFFFE	007FFFFF	6	Result is DNRM
00000000	00400000	6	UNF + EXACT
7FFFFFFF	00400000	7	UNF + NXT
40000000	00000000	7	UNF + NXT

Note: For single and double unwrap functions, the sign bit of the output result is directly copied from the sign bit of the input operand.

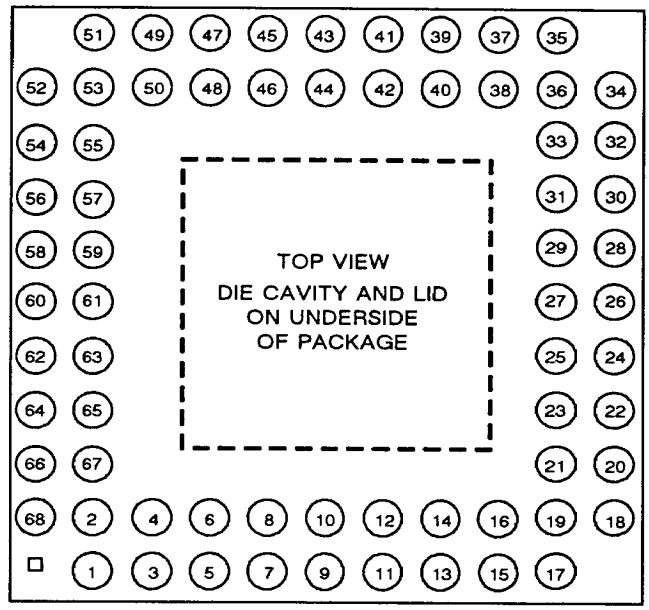
Pin Configuration

The WTL 1164/WTL 1165 will be supplied in 64-pin DIP and 68-pin pin grid array packages. Pins are allocated as shown. The four additional pins on the 68-pin packages are not connected.

- Θ JC = 6° C/Watt
- Θ JA = 13° C/Watt – 200 CFM, with heat sink
- Θ JA = 20° C/Watt – 200 CFM, no heat sink

* New PGA footprint; use for systems designed after July 11, 1986.

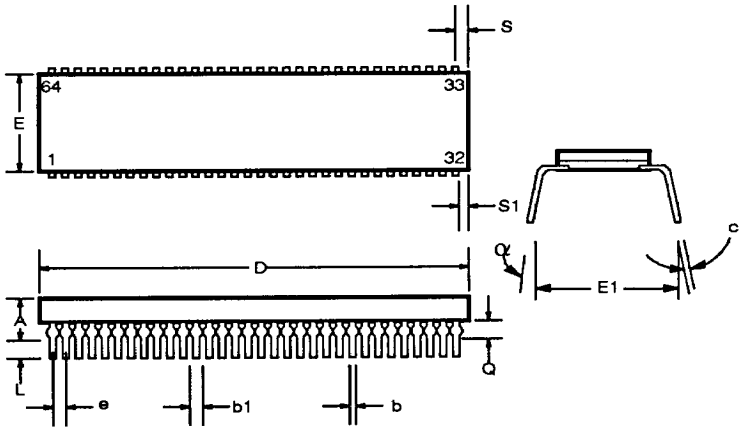
This part is desginated with the GCB suffix in the ordering information.



PIN #	PGA	DIP	PIN #	PGA	DIP	PIN #	PGA	DIP	PIN#	PGA	DIP
1	OE	X17	18	X25	X2	35	VCC	L1	52	GND	X28
2	CSUS-	X16	19	X24	X1	36	GND	L2	53	CLK	X27
3	VDD	X15	20	X23	X0	37	X9	L3	54	F0	X26
4	GND	X14	21	X22	GND	38	X8	VDD	55	F1	GND
5	S3	X13	22	X21	VDD	39	X7	GND	56	F2	VDD
6	S2	X12	23	X20	CSUX-	40	X6	VCC	57	F3	X25
7	S1	X11	24	X19	U	41	X5	OE-	58	F4	X24
8	S0	X10	25	X18	GND	42	X4	CSUS-	59	F5	X23
9	X31	VCC	26	X17	CLK	43	X3	VDD	60	CSL-	X22
10	X30	GND	27	X16	F0	44	X2	GND	61	L0	X21
11	X29	X9	28	X15	F1	45	X1	S3	62	L1	X20
12	X28	X8	29	X14	F2	46	X0	S2	63	L2	X19
13	X27	X7	30	X13	F3	47	GND	S1	64	L3	X18
14	X26	X6	31	X12	F4	48	VDD	S0	65	VDD	
15	GND	X5	32	X11	F5	49	CSUX	X31	66	GND	
16	VCC	X4	33	X10	CSL-	50	NC	X30	67	VCC	
17	N/C	X3	34	NC	L0	51	U	X29	68	NC	

Physical Dimensions

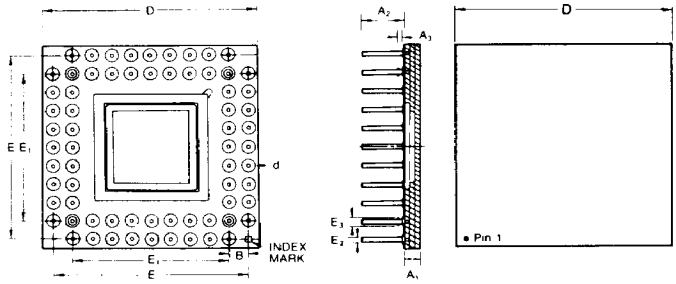
64-PIN DUAL IN-LINE PACKAGE



Symbol	LIMITS INCHES		MM	
	MIN	MAX	MIN	MAX
A***	.135	.250	3.43	11.43
b	.015	.022	0.38	0.56
b1	.030	.060	0.76	1.52
c	.008	.013	0.20	0.33
D	3.140	3.260	79.76	82.80
E	.775	.825	19.69	20.96
E1	.880	.920	22.35	23.37
e	.090	.110	2.29	2.79
L	.120	.160	3.05	4.06
Q	.040	.100	1.02	2.54
α	0°	15°	0°	15°
S*		.098		2.49
S1**	.005		1.27	

* From centerline of end lead
** From edge of end lead
*** Includes heat sink

68-PIN PIN GRID ARRAY PACKAGE



BOTTOM VIEW SIDE VIEW TOP VIEW

Symbol	Measurements	
	Inches	Millimeters
A_1	.157 ± 0.017	4.04 ± .43
A_2	0.160 ± 0.008	4.06 ± .20
A_3	0.050 ± 0.006	1.27 ± .150
D	1.100 ± 0.012 sq.	27.94 ± .30
E	1.000 ± 0.012	25.4 ± .30
E_1	0.800 ± 0.012	20.32 ± .30
E_2	0.018 dia. typ.	.46 dia. typ.
E_3	0.050 dia. typ	1.27 dia. typ

Ordering Information

PACKAGE TYPE	TEMPERATURE RANGE (T_{CASE})	ORDER NUMBER
Hermetic DIP	T = 0 to +80°C	WTL 1164-080-JC/WTL1165-080-JC
Pin Grid Array	T = 0 to +80°C	WTL 1164-080-GC/WTL1165-080-GC
Hermetic DIP	T = 0 to +80°C	WTL 1164-060-JC/WTL1165-060-JC
Pin Grid Array	T = 0 to +80°C	WTL 1164-060-GC/WTL1165-060-GC

Revision Summary

1.	Features	Revised	page 1
2.	Description	Revised	pages 1–2
3.	Pin Definitions	Revised	page 2
4.	Recommended Operating Conditions	Revised	page 4
5.	DC Electrical Characteristics	Revised	page 4
6.	AC Switching Characteristics	Revised	page 5
7.	I/O Characteristics	Revised	page 6
8.	Load Controls	Revised	page 7
9.	Load Notes	New	page 7
10.	Function Controls for Floating Point ALU	Revised	page 10
11.	Table 4. Function Controls for Floating Point ALU	Revised	page 13
12.	Result Status	Revised	page 15
13.	Figure 7. Single Precision Multiply	Revised	page 16
14.	IEEE Compatability	Revised	pages 27–28
15.	Gradual Underflow	Revised	pages 31–32
16.	Operations (Tables 5–21)	Revised	pages 32–41
17.	Pin Configuration	Revised	page 42