FEBRUARY 1986

- STL-AS Technology
- Parallel 8-Bit ALU with Expansion Inputs and Outputs
- 13 Arithmetic and Logic Functions
- 8 Conditional Shifts (Single and Double Length)
- 4 Instructions that Manipulate Bits
- Add and Subtract Immediate Instructions
- Absolute Value Instruction
- Signed Magnitude to/from Two's Complement Conversion
- Single- and Double-Length Normalize
- Select Functions
- Signed and Unsigned Divides with Overflow Detection; Input does not Need to be Prescaled
- Signed, Mixed, and Unsigned Multiplies
- Three-Operand, 16-Word Register File
- Full Carry Look Ahead Support
- Sign, Carry Out, Overflow, and Zero-Detect Status Capabilities
- Excess-3 BCD Arithmetic
- Internal Shift Multiplexers that Eliminate the Need for External Shift Control Parts
- ALU Bypass Path to Increase Speeds of Multiply, Divide, and Normalize Instructions and to Provide New Instructions such as Bit Set, Bit Reset, and Bit Test
- 3-Operand Register Files to Allow an Operation and a Move Instruction to be Combined
- Bit Masks that are Shared with Register Address Fields to Minimize Control Store Word Width
- 3 Data Input/Output Paths to Maximize Data Throughput

description

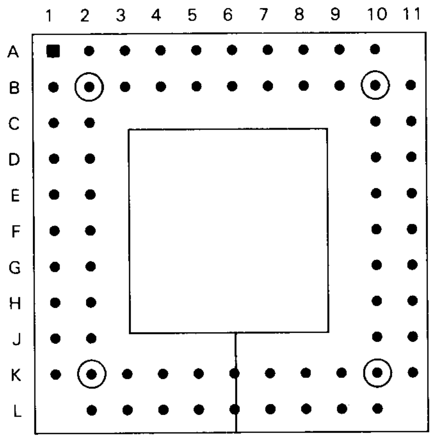
These 8-bit Advanced Schottky TTL integrated circuits are designed to implement high performance digital computers or controllers. An architecture and instruction set has been chosen that supports a fast system clock, a narrow micro-code word width, and a high system throughput. The powerful instruction set allows high-speed system architecture to be implemented and also allows an existing system's performance to be upgraded while protecting software investments. These processors are non-cascadable versions of the 'AS888. They are designed for 8-bit applications only.

The SN54AS887 is characterized for operation over the full military temperature range of −55°C to 125°C. The SN74AS887 and SN74AS887-1 are characterized for operation from 0°C to 70°C.

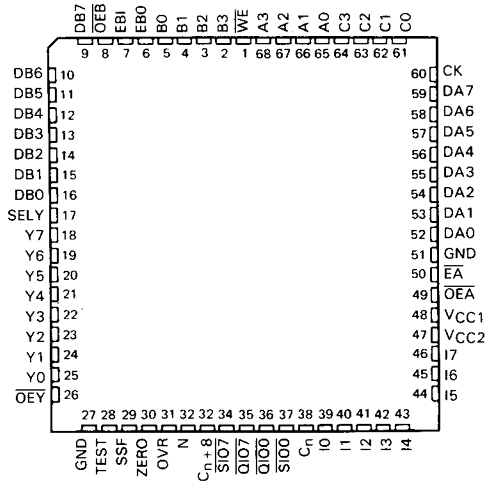
Package options include both plastic and ceramic chip carriers in addition to a 68-pin grid array ceramic package.

TEXAS INSTRUMENTS
POST OFFICE BOX 225012 • DALLAS, TEXAS 75265

SN54AS887, SN74AS887 . . . GB PACKAGE
(TOP VIEW)



SN54AS887 . . . FK PACKAGE
SN74AS887 . . . FN PACKAGE
(TOP VIEW)



GB PACKAGE PIN ASSIGNMENTS

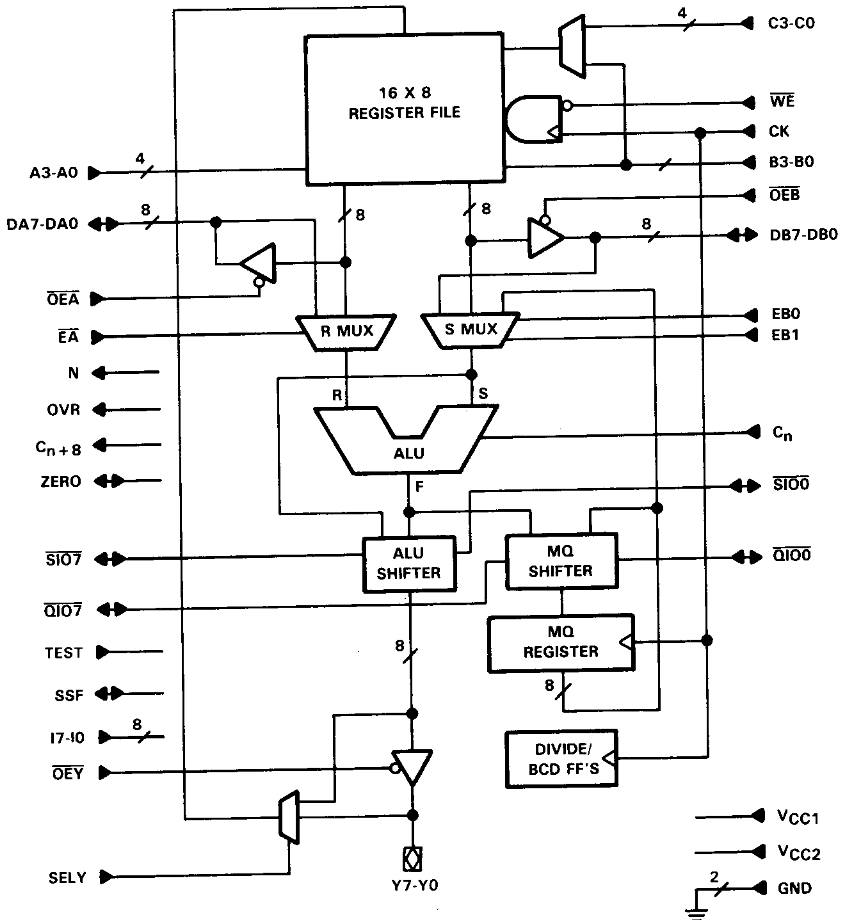
PIN	NAME	PIN	NAME	PIN	NAME	PIN	NAME
A-2	C_n	B-9	$\overline{OEY}$	F-10	Y3	K-4	C2
A-3	$\overline{SIO0}$	B-10	Y0	F-11	DB2	K-5	A0
A-4	$\overline{QIO0}$	B-11	Y1	G-1	DA2	K-6	A3
A-5	$\overline{QIO7}$	C-1	I5	G-2	DA0	K-7	$\overline{WE}$
A-6	C_{n+8}	C-2	V_{CC2}	G-10	DB0	K-8	DB7
A-7	N	C-10	Y4	G-11	DB3	K-9	$\overline{OEB}$
A-8	OVR	C-11	Y6	H-1	DA3	K-10	EB0
A-9	ZERO	D-1	I6	H-2	DA1	K-11	EB1
A-10	TEST	D-2	V_{CC1}	H-10	DB6	L-2	CK
B-1	I2	D-10	Y5	H-11	DB4	L-3	C1
B-2	I3	D-11	Y7	J-1	DA4	L-4	C3
B-3	I1	E-1	I7	J-2	DA5	L-5	A1
B-4	I0	E-2	$\overline{OEA}$	J-10	SELY	L-6	A2
B-5	I4	E-10	Y2	J-11	DB5	L-7	B3
B-6	$\overline{SIO7}$	E-11	DB1	K-1	DA6	L-8	B2
B-7	SSF	F-1	$\overline{EA}$	K-2	DA7	L-9	B1
B-8		F-2	GND	K-3	C0	L-10	B0

PIN GRID ARRAY	CHIP CARRIER	NAME	I/O	DESCRIPTION
A-10	28	TEST	I	Test input pin. Connected to ground for normal operation.
B-7	29	SSF	I/O	Special shift function. Used to transfer required information between packages during special instruction execution.
A-9	30	ZERO	I/O	Device zero detection, open collector. Input during certain special instructions.
A-8	31	OVR	O	ALU overflow, low active.
A-7	32	N	O	ALU negative, low active.
A-6	33	C_{n+8}	O	ALU ripple carry output.
B-6	34	$\overline{SIO7}$	I/O	Bidirectional shift pin, low active.
A-5	35	$\overline{QIO7}$	I/O	
A-4	36	$\overline{QIO0}$	I/O	
A-3	37	$\overline{SIO0}$	I/O	
A-2	38	C_n	I	ALU carry input.
B-4	39	I0	I	Instruction input.
B-3	40	I1	I	
B-1	41	I2	I	
B-2	42	I3	I	
B-5	43	I4	I	
C-1	44	I5	I	
D-1	45	I6	I	
E-1	46	I7	I	
C-2	47	V_{CC2}		Low voltage power supply (2 V).
D-2	48	V_{CC1}		I/O interface supply voltage (5 V).
E-2	49	$\overline{OEA}$	I	DA bus enable, low active.
F-1	50	$\overline{EA}$	I	ALU input operand select. High state selects external DA bus and low state selects register file.
F-2	51	GND		Ground pin.
G-2	52	DA0	I/O	A port data bus. Outputs register file data ($\overline{EA}$ = 0) or inputs external data ($\overline{EA}$ = 1).
H-2	53	DA1	I/O	
G-1	54	DA2	I/O	
H-1	55	DA3	I/O	
J-1	56	DA4	I/O	
J-2	57	DA5	I/O	
K-1	58	DA6	I/O	
K-2	59	DA7	I/O	
L-2	60	CK	I	Clocks all synchronous registers on positive edge.
K-3	61	C0	I	Register file write address select.
L-3	62	C1	I	
K-4	63	C2	I	
L-4	64	C3	I	
K-5	65	A0	I	Register file A port read address select.
L-5	66	A1	I	
L-6	67	A2	I	
K-6	68	A3	I	

PIN GRID ARRAY	CHIP CARRIER	NAME	I/O	DESCRIPTION
K-7	1	$\overline{WE}$	I	Register file (RF) write enable. Data is written into RF when $\overline{WE}$ is low and a low-to-high clock transition occurs. RF write is inhibited when $\overline{WE}$ is high.
L-7	2	B3	I	Register file B port read address select. (0 = LSB).
L-8	3	B2	I	
L-9	4	B1	I	
L-10	5	B0	I	
K-10	6	EB0	I	ALU input operand select. EB0 and EB1 selects the source of data that the S multiplexer provides for the S bus. Independent control of the DB bus and data path selection allow the user to isolate the DB bus while the ALU continues to process data.
K-11	7	EB1	I	
K-9	8	$\overline{OEB}$	I	DB bus enable, low active.
K-8	9	DB7	I/O	B port data bus. Outputs register data ($\overline{OEB}$ = 0) or used to input external data ($\overline{OEB}$ = 1), (0 = LSB).
H-10	10	DB6	I/O	
J-11	11	DB5	I/O	
H-11	12	DB4	I/O	
G-11	13	DB3	I/O	
F-11	14	DB2	I/O	
E-11	15	DB1	I/O	
G-10	16	DB0	I/O	
J-10	17	SELY	I	Y bus select, high active.
D-11	18	Y7	I/O	Y port data bus. Outputs instruction results ($\overline{OEY}$ = 0) or used to put external data into register file ($\overline{OEY}$ = 1).
C-11	19	Y6	I/O	
D-10	20	Y5	I/O	
C-10	21	Y4	I/O	
F-10	22	Y3	I/O	
E-10	23	Y2	I/O	
B-11	24	Y1	I/O	
B-10	25	Y0	I/O	
B-9	26	$\overline{OEY}$	I	Y bus output enable, low active.
F-2	27	GND		Ground pin

TEXAS INSTRUMENTS
POST OFFICE BOX 225012 • DALLAS, TEXAS 75265

functional block diagram



architectural elements

3-port register file

Working registers consist of 128 storage elements organized into sixteen 8-bit words. These storage elements appear to the user as 16 positive edge-triggered registers. The three port addresses, one write (C) and two reads (A and B), are completely independent of each other to implement a 3-operand register file. Data is written into the register file when $\overline{WE}$ is low and a low-to-high clock transition occurs. The ADD and SUBTRACT immediate instructions require only one source operand. The B address is used as the source address, and the bits of the A address are used to provide a constant field. The SET, RESET, and TEST BIT instructions use the B addressed register as both the source and destination register while the A and C addresses are used as masks. These instructions are explained in more detail in the instruction section.

S multiplexer

The S multiplexer selects the ALU operand, as follows:

EB1	EB0	S bus
Low	Low	RF data
Low	High	MQ data
High	Low	DB data
High	High	MQ data

DB port

The 8-bit bidirectional DB port inputs external data to the ALU or outputs the register file. If $\overline{OEB}$ is low, the DB bus is active; if $\overline{OEB}$ is high, the DB bus is in the high impedance state. Notice that the DB port may be isolated at the same time that register file data is passed to the ALU.

R multiplexer

The R multiplexer selects the other operand of the ALU. Except for those instructions that require constants or masks, the R bus will contain DA if $\overline{EA}$ is high or the RF data pointed to by A if $\overline{EA}$ is low.

DA port

The 8-bit bidirectional DA port inputs external data to the ALU or outputs the register file. If $\overline{OEA}$ is low, the DA bus is active; if $\overline{OEA}$ is high, the DA bus is in the high-impedance state.

Notice that the DA bus may be isolated while register file data is passed to the ALU.

ALU

The shift instructions are summarized in Table 4 and illustrated in Figure 2. The ALU can perform seven arithmetic and six logical instructions on two 8-bit operands. It also supports multiplication, division, normalization, bit set, reset, test, byte operations, and excess-3 BCD arithmetic. These source operands are the outputs of the S and R multiplexers.

ALU and MQ shifters

ALU and MQ shifters perform all of the shift, multiply, divide, and normalize functions. Table 4 shows the value of the $\overline{SIO7}$ and $\overline{QIO7}$ pins of the most significant package. The standard shifts may be made into conditional shifts and the serial data may be input or output with the aid of two three-state gates. These capabilities are discussed further in the arithmetic and logic section.

TEXAS INSTRUMENTS
POST OFFICE BOX 225012 • DALLAS, TEXAS 75265

MQ register

The multiplier-quotient (MQ) register has specific functions in multiplication, division, and normalization. This register may also be used as a temporary storage register. The MQ register may be loaded if the instruction code on pins I7-I0 is E1-E7 or E9-EE (See Table 1).

Y bus

The Y bus contains the output of the ALU shifter if $\overline{OEY}$ is low and is a high impedance input if $\overline{OEY}$ is high. SELY must be low to pass the internal ALU shift bus and must be high to pass the external Y bus to the register file.

status

Four status pins are available on the most significant package, overflow (OVR), sign (N), carry out (C_{n+8}), and zero (ZERO). The C_{n+8} line signifies the ALU result while OVR, ZERO, and N refer the status after the ALU shift has occurred. Notice that the ZERO pin cannot be used to detect whether an input placed on a high impedance Y bus is zero.

divide BCD flip-flops

The multiply-divide flip-flops contain the status of the previous multiply or divide instruction. They are affected by the following instructions:

DIVIDE REMAINDER FIX
SIGNED DIVIDE QUOTIENT FIX
SIGNED MULTIPLY
SIGNED MULTIPLY TERMINATE
SIGNED DIVIDE INITIALIZE
SIGNED DIVIDE START
SIGNED DIVIDE ITERATE
UNSIGNED DIVIDE START
UNSIGNED DIVIDE ITERATE
UNSIGNED MULTIPLY
SIGNED DIVIDE TERMINATE
UNSIGNED DIVIDE TERMINATE

The excess-3 BCD flip-flops are affected by all instructions except NOP. The clear function clears these flip-flops. They preserve the carry from each nibble (4-bits) in excess-3/BCD operations.

test pin (test)

This pin should be connected to ground.

special shift function (SSF) pin

Conditional shifting algorithms may be implemented via control of the SSF pin. The applied voltage to this pin may be set as a function of a potential overflow condition (the two most significant bits are not equal) or any other condition (see Group 1 instructions).

instruction set

The 'AS887 bit-slice processor uses bits I7-I0 as instruction inputs. A combination of bits I3-I0 (Group 1 instructions) and bits I7-I4 (Group 2-5 instructions) are used to develop the 8-bit op code for a specific instruction. Group 1 and Group 2 instructions can be combined to perform arithmetic or logical functions plus a shift function in one instruction cycle. A summary of the instruction set is given in Table 1.

TABLE 1. INSTRUCTION SET

GROUP 1 INSTRUCTIONS

INSTRUCTION BITS (I3-I0) HEX CODE	MNEMONIC	FUNCTION
0		Accesses Group 4 instructions
1	ADD	$R+S+C_n$
2	SUBR	$\overline{R}+S+C_n$
3	SUBS	$R+\overline{S}+C_n$
4	INCS	$S+C_n$
5	INCNS	$\overline{S}+C_n$
6	INCR	$R+C_n$
7	INCNR	$\overline{R}+C_n$
8		Accesses Group 3 instructions
9	XOR	R XOR S
A	AND	R AND S
B	OR	R OR S
C	NAND	R NAND S
D	NOR	R NOR S
E	ANDNR	$\overline{R}$ AND S
F		Accesses Group 5 instructions

GROUP 2 INSTRUCTIONS

INSTRUCTION BITS (I7-I4) HEX CODE	MNEMONIC	FUNCTION
0	SRA	Arithmetic Right Single
1	SRAD	Arithmetic Right Double
2	SRL	Logical Right Single
3	SRLD	Logical Right Double
4	SLA	Arithmetic Left Single
5	SLAD	Arithmetic Left Double
6	SLC	Circular Left Single
7	SLCD	Circular Left Double
8	SRC	Circular Right Single
9	SRCD	Circular Right Double
A	MQSRA	Pass (F→Y) and Arithmetic Right MQ
B	MQSRL	Pass (F→Y) and Logical Right MQ
C	MQSLL	Pass (F→Y) and Logical Left MQ
D	MQSLC	Pass (F→Y) and Circular Left MQ
E	LOADMQ	Pass (F→Y) and Load MQ (F=MQ)
F	PASS	Pass (F→Y)

TEXAS INSTRUMENTS
POST OFFICE BOX 225012 • DALLAS, TEXAS 75265

TABLE 1. INSTRUCTION SET (Continued)

GROUP 3 INSTRUCTIONS

INSTRUCTION BITS (I7-I0) HEX CODE	MNEMONIC	FUNCTION
08	SET1	Set Bit
18	SET0	Reset Bit
28	TB1	Test Bit (One)
38	TB0	Test Bit (Zero)
48	ABS	Absolute Value
58	SMTC	Sign Magnitude/Two's Complement
68	ADDI	Add Immediate
78	SUBI	Subtract Immediate
88		Reserved
98		Reserved
A8		Reserved
B8		Reserved
C8		Reserved
D8		Reserved
E8		Reserved
F8		Reserved

GROUP 4 INSTRUCTIONS

INSTRUCTION BITS (I7-I0) HEX CODE	MNEMONIC	FUNCTION
00		Reserved
10	SEL	Select S/R
20	SNORM	Single Length Normalize
30	DNORM	Double Length Normalize
40	DIVRF	Divide Remainder Fix
50	SDIVQF	Signed Divide Quotient Fix
60	SMULI	Signed Multiply Iterate
70	SMULT	Signed Multiply Terminate
80	SDIVIN	Signed Divide Initialize
90	SDIVIS	Signed Divide Start
A0	SDIVI	Signed Divide Iterate
B0	UDIVIS	Unsigned Divide Start
C0	UDIVI	Unsigned Divide Iterate
D0	UMULI	Unsigned Multiply Iterate
E0	SDIVIT	Signed Divide Terminate
F0	UDIVIT	Unsigned Divide Terminate

TABLE 1. INSTRUCTION SET (Concluded)

GROUP 5 INSTRUCTIONS

INSTRUCTION BITS (I7-I0) HEX CODE	MNEMONIC	FUNCTION
0F	CLR	Clear
1F	CLR	Clear
2F	CLR	Clear
3F	CLR	Clear
4F	CLR	Clear
5F	CLR	Clear
6F	CLR	Clear
7F	BCDBIN	BCD to Binary
8F		Reserved
9F	EX3C	Excess-3 Word Correction
AF	SDIVO	Signed Divide Overflow Check
BF	CLR	Clear
CF	CLR	Clear
DF	BINEX3	Binary to Excess-3
EF	CLR	Clear
FF	NOP	No Operation

group 1 instructions

TABLE 2. GROUP 1 INSTRUCTIONS

INSTRUCTION BITS (I3-I0) HEX CODE	MNEMONIC	FUNCTION
0		Accesses Group 4 instructions
1	ADD	$R + S + C_n$
2	SUBR	$\overline{R} + S + C_n$
3	SUBS	$R + \overline{S} + C_n$
4	INCS	$S + C_n$
5	INCNS	$\overline{S} + C_n$
6	INCR	$R + C_n$
7	INCNR	$\overline{R} + C_n$
8		Accesses Group 3 instructions
9	XOR	R XOR S
A	AND	R AND S
B	OR	R OR S
C	NAND	R NAND S
D	NOR	R NOR S
E	ANDNR	$\overline{R}$ AND S
F		Accesses Group 5 instructions

Group 1 instructions (excluding hex codes 0, 8, and F), shown in Table 2, may be used in conjunction with Group 2 shift instructions to perform arithmetic or logical functions plus a shift function† in one instruction cycle (hex codes 0, 8, and F are used to access Group 4, 3, and 5 instructions, respectively). Each shift may be made into a conditional shift by forcing the special shift function (SSF) pin into the proper state. If the SSF pin is high or floating, the shifted ALU output will be sent to the output buffers. If the SSF pin is pulled low externally, the ALU result will be passed directly to the output buffers. Conditional shifting is useful for scaling inputs in data arrays or in signal processing algorithms.

These instructions set the BCD flip-flop for the excess-3 correct instruction. The status is set with the following results (C_{n+8} is ALU carry out and is independent of shift operation; others are evaluated after shift operation).

†Double-precision shifts involve both the ALU and MQ register.

Status is set with the following results:

Arithmetic

N → MSB of result
OVR → Signed arithmetic overflow
C_{n+8} → Carry out equal one
Z → Result equal zero

Logic

N → MSB of result
OVR → None (force to zero)
C_{n+8} → None (force to zero)
Z → Result equal zero

group 2 instructions

TABLE 3. GROUP 2 INSTRUCTIONS

INSTRUCTION BITS (I7-I4) HEX CODE	MNEMONIC	FUNCTION
0	SRA	Arithmetic Right Single
1	SRAD	Arithmetic Right Double
2	SRL	Logical Right Single
3	SRLD	Logical Right Double
4	SLA	Arithmetic Left Single
5	SLAD	Arithmetic Left Double
6	SLC	Circular Left Single
7	SLCD	Circular Left Double
8	SRC	Circular Right Single
9	SRCD	Circular Right Double
A	MQSRA	Pass (F→Y) and Arithmetic Right MQ
B	MQSRL	Pass (F→Y) and Logical Right MQ
C	MQSLL	Pass (F→Y) and Logical Left MQ
D	MQSLC	Pass (F→Y) and Circular Left MQ
E	LOADMQ	Pass (F→Y) and Load MQ (F = MQ)
F	PASS	Pass (F→Y)

The processor's shift instructions are implemented using Group 2 instructions (Table 3). The connections are the same on all instructions including multiply, divide, and normalization functions.

The following external connections are required:

$\overline{SIO7}$ to $\overline{SIO0}$
$\overline{QIO7}$ to $\overline{QIO0}$

Single- and double-precision shifts are supported. Double-precision shifts assume the most significant half has come through the ALU and will be placed (if $\overline{WE}$ is low) into the register file on the rising edge of the clock and the least significant half lies in the MQ register. All Group 2 shifts may be made conditional. (see previous page)

The following definitions apply to Group 2 shift instructions:

Arithmetic right shifts copy the sign of the number if no overflow occurs from the ALU calculation; if overflow occurs, the sign bit is inverted.
Arithmetic left shifts do not retain the sign of the number if an overflow occurs. A zero is filled into the LSB if not forced externally.
Logical right shifts fill a zero in the MSB position if not forced externally.
Circular right shifts fill the LSB in the MSB position.
Circular left shifts fill the MSB in the LSB position.
Shifting left is defined as moving a bit position towards the MSB (doubling).
Shifting right is defined as moving a bit towards the LSB (halving).

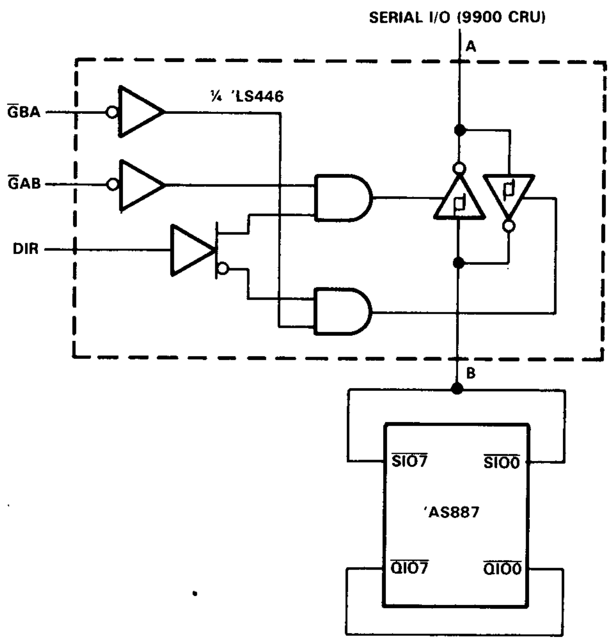


FIGURE 1. SERIAL I/O

TEXAS INSTRUMENTS
POST OFFICE BOX 225012 • DALLAS, TEXAS 75265

Serial input may be performed using the circuitry shown in Figure 1. A single-/or double-precision arithmetic left or logical right shift fills the complement of the data on $\overline{SIO0}$ and $\overline{SIO7}$ into the LSB or MSB of the data word(s). Note that if $\overline{SIO0}$ and $\overline{SIO7}$ are floating (HI-Z), a zero will be filled as an end condition.

Serial output may be performed with circular instructions.

The shift instructions are summarized in Table 4 and illustrated in Figure 2. In Figure 2 and all succeeding figures that illustrate instruction execution, the following definitions apply:

- QBT — End fill for signed divide.
- MQF — End fill for unsigned divide.
- SRF — End fill for signed multiply and the arithmetic right shifts.

TABLE 4. SHIFT INSTRUCTIONS

OP CODE†	SHIFT FUNCTION‡	$\overline{SIO7}$ • $\overline{SIO0}$ WIRED VALUE	$\overline{QIO7}$ • $\overline{QIO0}$ WIRED VALUE
0N	Arithmetic Right Single	ALU-LSB Output	–
1N	Arithmetic Right Double	MQ-LSB Output	ALU-LSB Output
2N	Logical Right Single	Input to ALU-MSB	ALU-LSB Output
3N	Logical Right Double	Input to ALU-MSB	ALU-LSB Output
4N	Arithmetic Left Single	Input to ALU-LSB	ALU-MSB Output
5N	Arithmetic Left Double	Input to MQ-LSB	MQ-MSB Output
6N	Circular Left Single	ALU-MSB Output	–
7N	Circular Left Double	ALU-MSB Output	MQ-MSB Output
8N	Circular Right Single	ALU-LSB Output	–
9N	Circular Right Double	MQ-LSB Output	ALU-LSB Output
AN	Arithmetic Right (MQ only)	MQ-LSB Output	MQ-LSB Output
BN	Logical Right (MQ only)	MQ-LSB Output	Input to MQ-MSB
CN	Logical Left (MQ only)	Input to MQ-LSB	MQ-MSB Output
DN	Circular Left (MQ only)	MQ-MSB Output	MQ-MSB Output

†Op Code N ≠ 0, 8, or F; these select special instruction Groups 4, 3, and 5 respectively.
‡Shift I/O pins are active low. Therefore, inputs and outputs must be inverted if true logical values are required.

Status is set with the following results:

Arithmetic

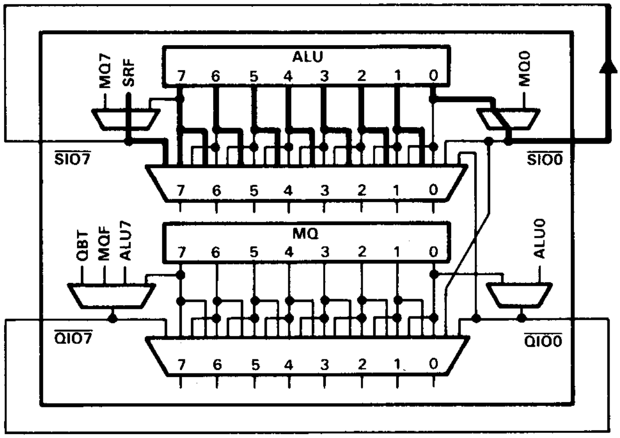
- N → Result MSB equal one
- OVR → Signed arithmetic overflow†
- C_{n+8} → Carry out equal one
- Z → Result equal zero

Logic

- N → Result MSB equal one
- OVR → Zero
- C_{n+8} → Zero
- Z → Result equal zero

† For the SLA and SLAD instructions, OVR is set if signed arithmetic overflow or if the ALU result MSB XOR MSB-1 equals one.

ARITHMETIC RIGHT SINGLE



ARITHMETIC RIGHT DOUBLE

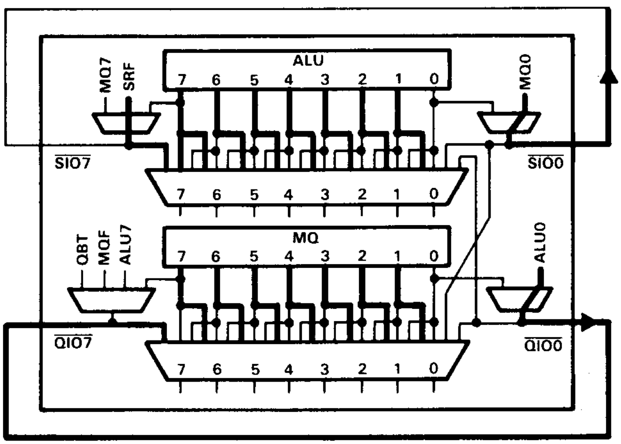
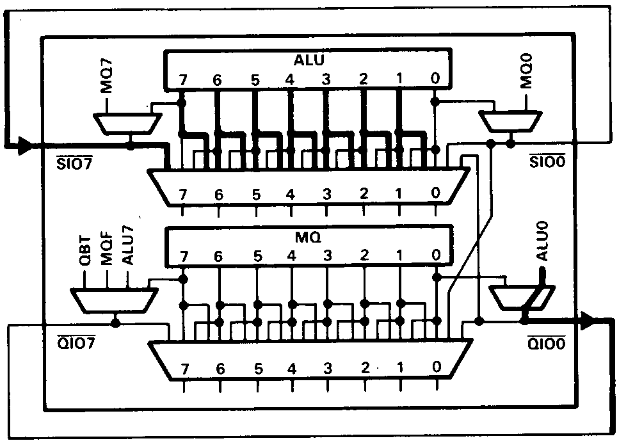


FIGURE 2. SHIFT INSTRUCTIONS

LOGICAL RIGHT SINGLE

FILLS ZERO IF NOT FORCED



LOGICAL RIGHT DOUBLE

FILLS ZERO IF NOT FORCED

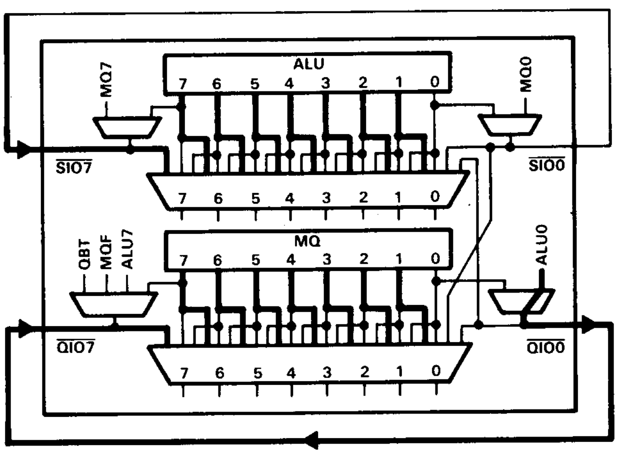
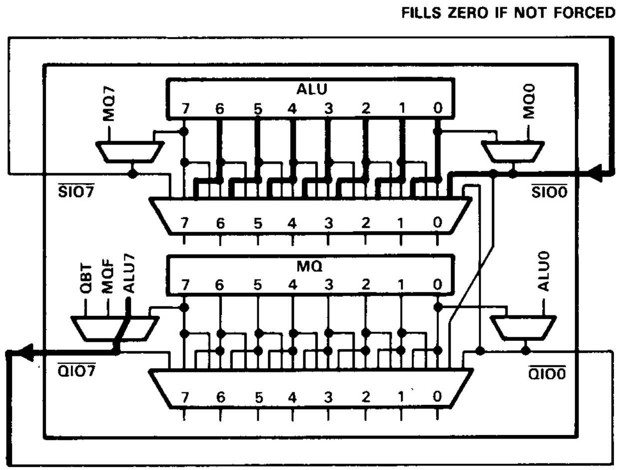


FIGURE 2. SHIFT INSTRUCTIONS (Continued)

2

LSI Devices

ARITHMETIC LEFT SINGLE



ARITHMETIC LEFT DOUBLE

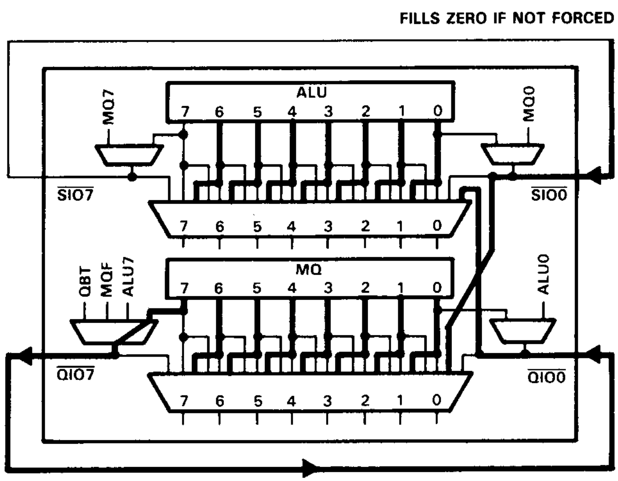
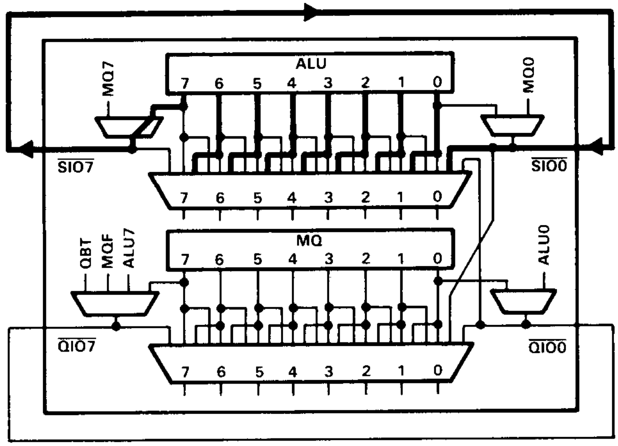


FIGURE 2. SHIFT INSTRUCTIONS (Continued)

TEXAS INSTRUMENTS
POST OFFICE BOX 225012 • DALLAS, TEXAS 75265

CIRCULAR LEFT SINGLE



CIRCULAR LEFT DOUBLE

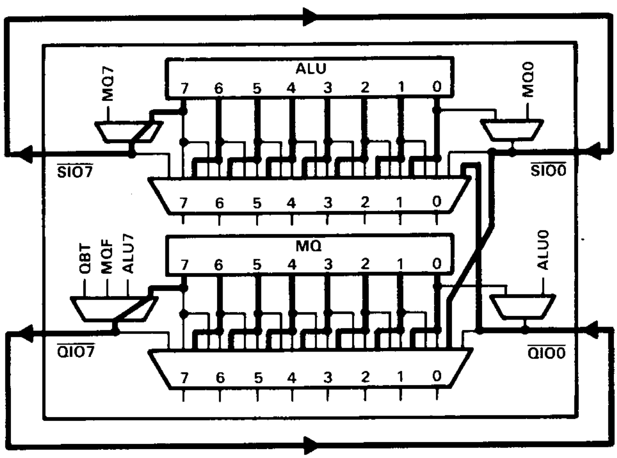
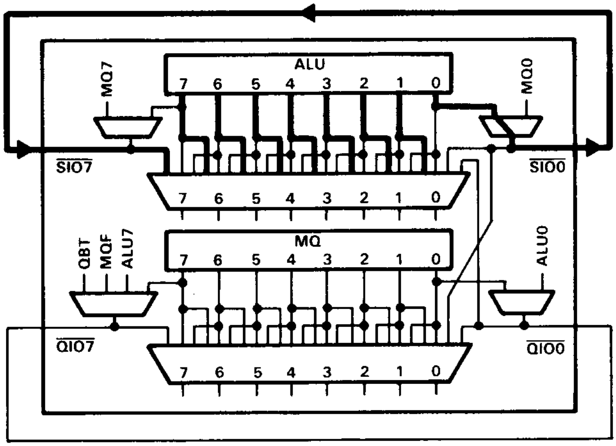


FIGURE 2. SHIFT INSTRUCTIONS (Continued)

CIRCULAR RIGHT SINGLE



CIRCULAR RIGHT DOUBLE

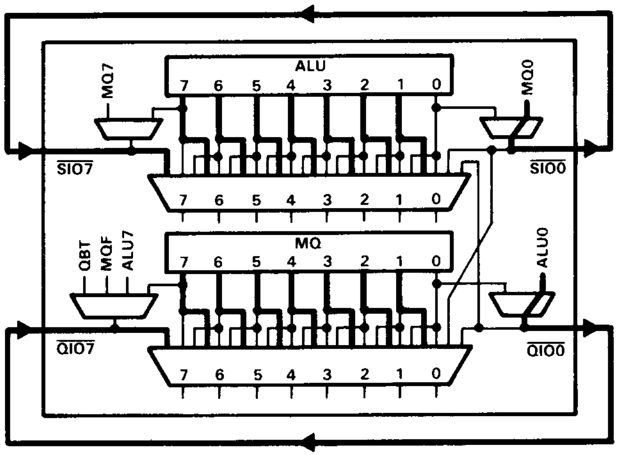
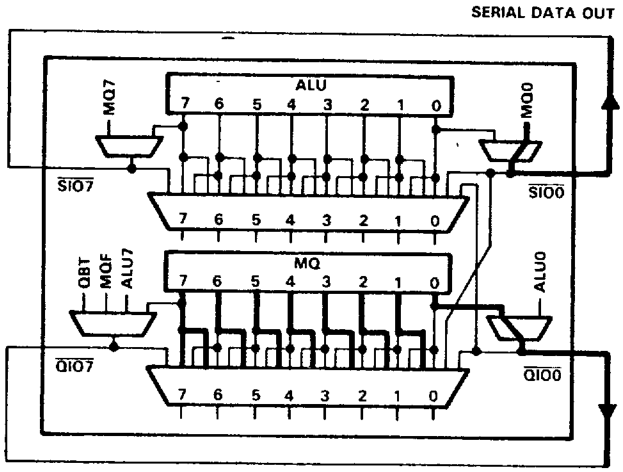


FIGURE 2. SHIFT INSTRUCTIONS (Continued)

TEXAS INSTRUMENTS
POST OFFICE BOX 225012 • DALLAS, TEXAS 75265

ARITHMETIC RIGHT (MQ ONLY)



LOGICAL RIGHT (MQ ONLY)

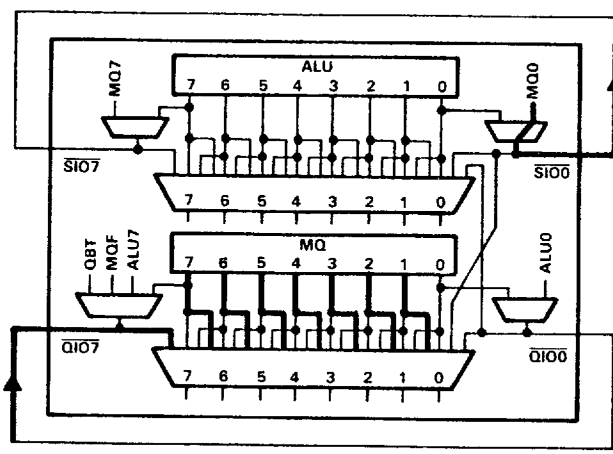
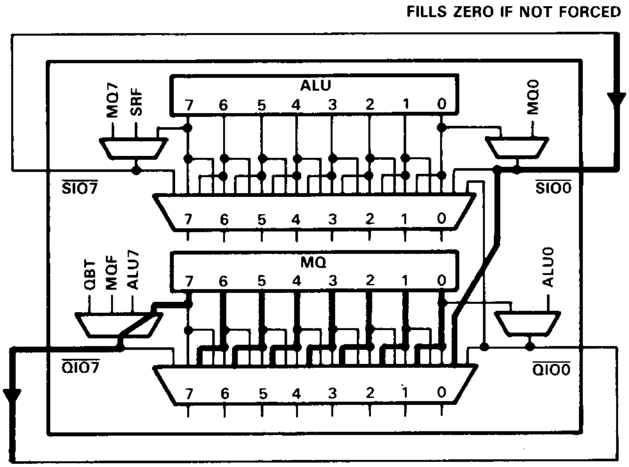


FIGURE 2. SHIFT INSTRUCTIONS (Continued)

LOGICAL LEFT (MQ ONLY)



CIRCULAR LEFT (MQ ONLY)

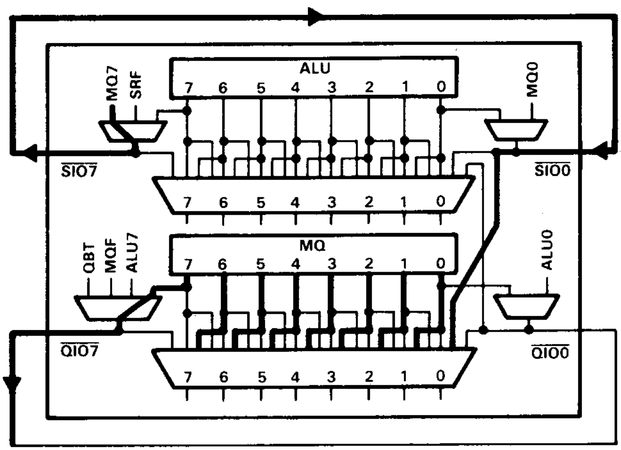


FIGURE 2. SHIFT INSTRUCTIONS (Concluded)

TEXAS INSTRUMENTS
POST OFFICE BOX 225012 • DALLAS, TEXAS 75265

group 3 instructions

Hex code 8 of Group 1 instructions is used to access Group 3 instructions. Group 3 instructions are summarized in Table 5.

TABLE 5. GROUP 3 INSTRUCTIONS

INSTRUCTION BITS (I7-I0) OP CODE (HEX)	MNEMONIC	FUNCTION
08	SET1	Set Bit
18	SET0	Reset Bit
28	TB1	Test Bit (One)
38	TB0	Test Bit (Zero)
48	ABS	Absolute Value
58	SMTC	Sign Magnitude/Two's Complement
68	ADDI	Add Immediate
78	SUBI	Subtract Immediate
88		Reserved
98		Reserved
A8		Reserved
B8		Reserved
C8		Reserved
D8		Reserved
E8		Reserved
F8		Reserved

set bit instruction (set1): I7-I0 = 08_{16}

This instruction (Figure 3) is used to force selected bits to one (any combination of zero to eight bits). The desired bits are specified by an 8-bit mask (C3-C0)::(A3-A0)† consisting of register file address ports that are not required to support this instruction. All bits that are in the same bit positions as ones in the mask are forced to a logical one. The B3-B0 address field is used for both source and destination of this instruction. The S bus is the source word for this instruction. $\overline{\text{SIO0}}$ must be forced low for proper operation. If $\overline{\text{SIO0}}$ is high, data on the S bus is passed unaltered. The status set by the set bit instruction is as follows:

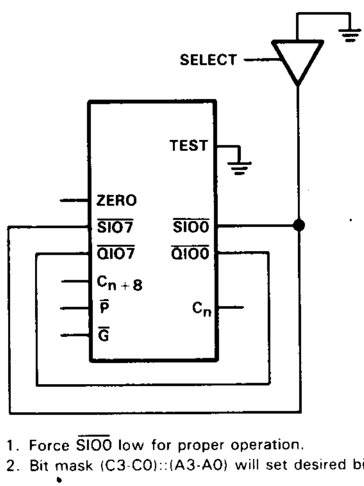
N	→	None (force to zero)
OVR	→	None (force to zero)
C_{n+8}	→	None (force to zero)
Z	→	Result equal zero

† The symbol '::' is concatenation operator

reset bit instruction (set0): I7-I0 = 18_{16}

This instruction (Figure 3) is used to force selected bits to zero. The desired bits are specified by an 8-bit mask (C3-C0)::(A3-A0) consisting of register file address ports that are not required to support this instruction. All bits in the selected byte(s) that are in the same bit positions as ones in the mask are reset. The B3-B0 address field is used for both source and destination of this instruction. The S bus is the source word for this instruction. $\overline{\text{SIO0}}$ must be forced low for proper operation. If $\overline{\text{SIO0}}$ is high, data on the S bus is passed unaltered. The status set by the reset bit instruction is as follows:

N	→	None (force to zero)
OVR	→	None (force to zero)
C_{n+8}	→	None (force to zero)
Z	→	Result equal zero



NOTES: 1. Force $\overline{\text{SIO0}}$ low for proper operation.
2. Bit mask (C3-C0)::(A3-A0) will set desired bits to one.

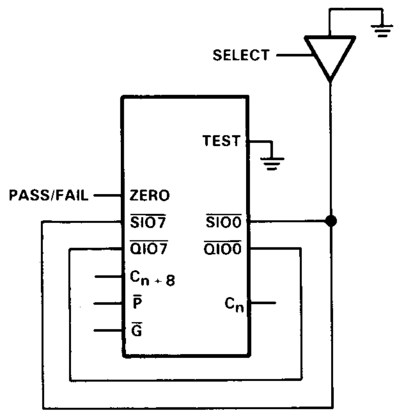
FIGURE 3. SET BIT (OR RESET BIT)

TEXAS INSTRUMENTS
POST OFFICE BOX 225012 • DALLAS, TEXAS 75265

test bit (one) instruction (TB1): I7-I0 = 28_{16}

This instruction (Figure 4) is used to test selected bits for ones. Bits to be tested are specified by an 8-bit mask (C3-C0)::(A3-A0) consisting of register file address ports that are not required to support this instruction. Write Enable ($\overline{WE}$) is internally disabled during this instruction. The test will pass if the selected byte has ones at all bit locations specified by the ones of the mask (Figure 5). The S bus is the source word for this instruction. $\overline{SIO0}$ must be forced low for proper operation. The status set by the test bit (one) instruction is as follows:

N → None (force to zero)
OVR → None (force to zero)
C_{n+8} → None (force to zero)
Z → Pass



NOTES: 1. Force $\overline{SIO0}$ low for proper operation.
2. Bit mask (C3-C0)::(A3-A0) will define bits for testing.
3. Pass/fail is indicated on Z output.

FIGURE 4. TEST BIT

test bit (zero) instruction (TB0): I7-I0 = 38_{16}

This instruction (Figure 4) is used to test selected bits for ones. Bits to be tested are specified by an 8-bit mask (C3-C0)::(A3-A0) consisting of register file address ports that are not required to support this instruction. Write Enable ($\overline{WE}$) is internally disabled during this instruction. The test will pass if the selected byte has zeros at all bit locations specified by the ones of the mask (Figure 6). The S bus is the source word for this instruction. $\overline{SIO0}$ must be forced low for proper operation. The status set by the test bit (zero) instruction is as follows:

N → None (force to zero)
OVR → None (force to zero)
C_{n+8} → None (force to zero)
Z → Pass

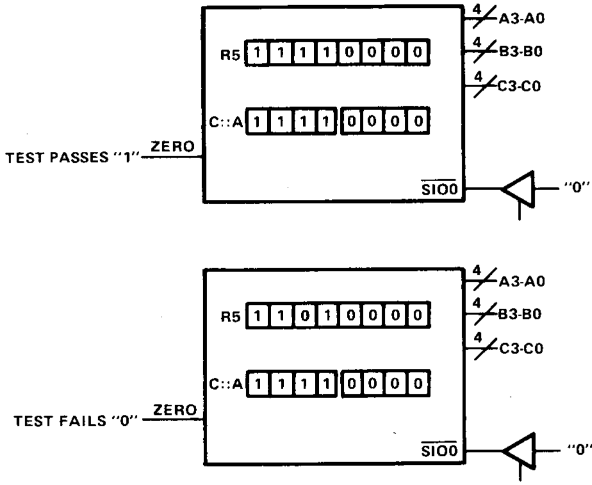


FIGURE 5. TEST BIT ONE EXAMPLES

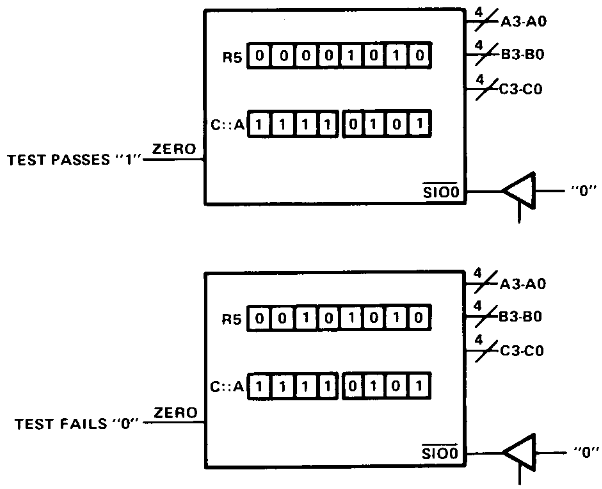


FIGURE 6. TEST BIT ZERO EXAMPLES

TEXAS INSTRUMENTS
POST OFFICE BOX 225012 • DALLAS, TEXAS 75265

absolute value instruction (ABS): I7-I0 = 48_{16}

This instruction is used to convert two's complement numbers to their positive value. The operand placed on the S bus is the source for this instruction. The 'AS887 will test the sign of the S bus and force the SSF pin to the proper value. The status set by the absolute value instruction is as follows:

N	→	Input MSB equal one
OVR	→	Input equal 80 (hex)
C_{n+8}	→	S = 0
Z	→	Result equal zero

sign magnitude/two's complement instruction (SMTC): I7-I0 = 58_{16}

This instruction allows conversion from two's complement representation to sign magnitude representation, or vice-versa, in one clock cycle. The operand placed on the S bus is the source for this instruction.

When a negative zero (80 hex) is converted, the result is 00 with an overflow. If the input is in two's complement notation, the overflow indicates an illegal conversion. The status set by the sign magnitude/two's complement instruction is as follows:

N	→	Result MSB equal one
OVR	→	Input equal 80 (hex)
C_{n+8}	→	Input equal 00 (hex)
Z	→	Result equal zero

add immediate instruction (ADDI): I7-I0 = 68_{16}

This instruction is used to add a specified constant value to the operand placed on the S bus. The constant will be between the values of 0 and 15. The constant value is specified by the unused register file address (A port) not required to support this instruction. Forcing the carry input will add an additional one to the result. The status set by the add immediate instruction is as follows:

N	→	Result MSB equal one
OVR	→	Arithmetic signed overflow
C_{n+8}	→	Carry out equal one
Z	→	Result equal zero

subtract immediate instruction (SUBI): I7-I0 = 78_{16}

This instruction is used to subtract a specified constant value from the operand placed on the S bus. The constant value is specified by the unused register file address (A port) that is not required to support this instruction. The constant applied is the least significant four bits of a two's complement number. The device sign extends the constant over the entire word length. The status set by the subtract immediate instruction is as follows:

N	→	Result MSB equal one
OVR	→	Arithmetic signed overflow
C_{n+8}	→	Carry out equal one
Z	→	Result equal zero

group 4 instructions

Hex code 0 of Group 1 instructions is used to access Group 4 instructions. Group 4 instructions are summarized in Table 6.

TABLE 6. GROUP 4 INSTRUCTIONS

INSTRUCTION BITS (I7-I0) OP CODE (HEX)	MNEMONIC	FUNCTION
00		Reserved
10	SEL	Select S/R
20	SNORM	Single Length Normalize
30	DNORM	Double Length Normalize
40	DIVRF	Divide Remainder Fix
50	SDIVQF	Signed Divide Quotient Fix
60	SMULI	Signed Multiply Iterate
70	SMULT	Signed Multiply Terminate
80	SDIVIN	Signed Divide Initialize
90	SDIVIS	Signed Divide Start
A0	SDIVI	Signed Divide Iterate
B0	UDIVIS	Unsigned Divide Start
C0	UDIVI	Unsigned Divide Iterate
D0	UMULI	Unsigned Multiply Iterate
E0	SDIVIT	Signed Divide Terminate
F0	UDIVIT	Unsigned Divide Terminate

select S/R instruction (SEL): I7-I0 = 10_{16}

This instruction is used to pass either the S bus or the R bus to the output depending on the state of the SSF input pin. Normally, the preceding instruction would test the two operands and the resulting status information would be used to force the SSF input pin. SSF = 0 will output the R bus and SSF = 1 will output the S bus. The status set by the select S/R instruction is as follows:

N → Result MSB equal one
OVR → None (force to zero)
C_{n+8} → None (force to zero)
Z → Result equal zero

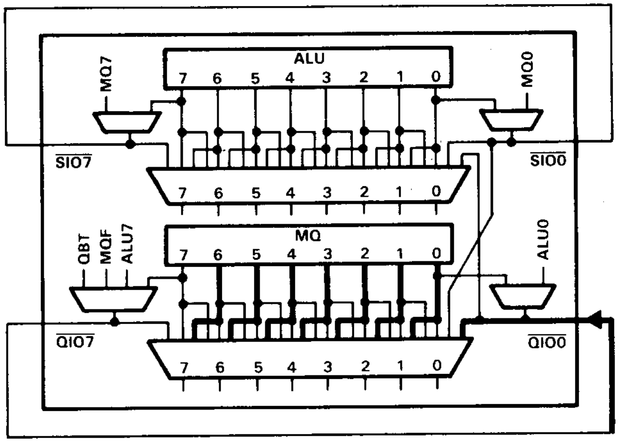
single-length normalize instruction (SNORM): I7-I0 = 20_{16}

This instruction will cause the contents of the MQ register to shift toward the most significant bit. Zeros are shifted in via the $\overline{QIO0}$ input. The number of shifts performed can be counted and stored in one of the register files by forcing a high at the C_n input. When the two most significant bits are of opposite value, normalization is complete. This condition is indicated on the microcycle that completes the normalization at the OVR output.

The chip contains conditional logic that inhibits the shift function (and also inhibits the register file increment) if the number within the MQ register is already normalized at the beginning of the instruction (Figure 7). The status set by the single-length normalize instruction is as follows:

N → MSB of result
OVR → MSB XOR 2nd MSB
C_{n+8} → Carry out equal one
Z → Result equal zero

SINGLE-LENGTH NORMALIZE



DOUBLE-LENGTH NORMALIZE

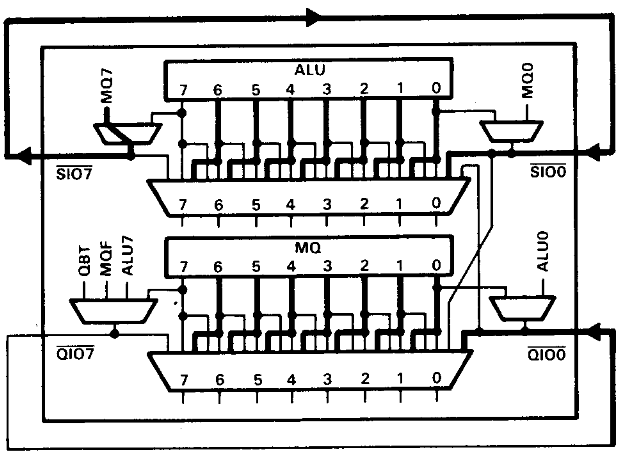


FIGURE 7. SINGLE- AND DOUBLE-LENGTH NORMALIZE

double-length normalize instruction (DNORM): I7-I0 = 30_{16}

This instruction will cause the contents of a double-length word (register file contains the most significant half and the MQ register contains the least significant half) to shift toward the most significant bit. Zeros are shifted in via the $\overline{QIO0}$ input. When the two most significant bits are of opposite value, normalization is complete. This condition is indicated on the microcycle that completes the normalization at the OVR output.

The chip contains conditional logic which inhibits the shift function if the number is already normalized at the beginning of the instruction (Figure 7). The most significant half of the operand must be placed on the S bus. The status set by the double-length normalize instruction is as follows:

N → MSB of result
OVR → MSB XOR 2nd MSB
C_{n+8} → None (force to zero)
Z → Result equal zero

multiply operations

The ALU performs three unique types of 8 by 8 multiplies each of which produces a 16-bit result (Figure 8). All three types of multiplication proceed via the following recursion:

$P(J+1) = 2[P(J) + \text{Multiplicand} \times M\ (8\text{-}J)]$

where

P(J) = partial product at iteration number J

P(J + 1) = partial product at iteration number J + 1

J varies from 0 to 8

M (8-J) = mode bit (unique to multiply type)

2 denotes some type of shift (unique to multiply)

Notice that by proper choice of mode terms and shifting operations, signed, unsigned, and mixed multiplies (signed times unsigned) may be performed.

All multiplies assume that the multiplier is stored in MQ before the operation begins (in the case of mixed multiply, the unsigned number must be the multiplier).

The processor has the following multiply instructions:

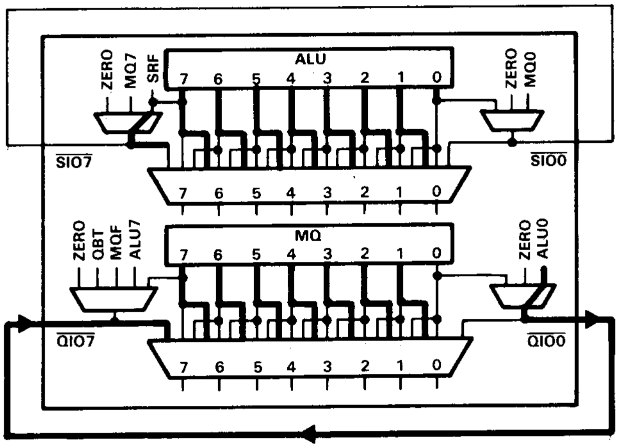
1. SIGNED MULTIPLY ITERATE (SMULI): I7-I0 = 60_{16}
2. SIGNED MULTIPLY TERMINATE (SMULT): I7-I0 = 70_{16}
3. UNSIGNED MULTIPLY ITERATE (UMULI): I7-I0 = $D0_{16}$

The signed multiply iterate (SMULI) instruction performs a signed times signed iteration. This instruction interprets M(8-J) as the 8-J bit of the multiplier. The shift is a double-precision right shift one bit. This instruction is repeated 7 times for a 8 × 8 signed multiply. This instruction will be used 7 consecutive times for a mixed multiplication.

The signed multiply terminate (SMULT) instruction provides correct (negative) weighting of the sign bit of a negative multiplier in signed multiplication. The instruction is identical to signed multiply iterate (SMULI) except that M(8-J) is interpreted as −1 if the sign bit of the multiplier is 1, and 0 if the sign bit of the multiplier is 0.

TEXAS INSTRUMENTS
POST OFFICE BOX 225012 • DALLAS, TEXAS 75265

SMULI, SMULT



UMULI

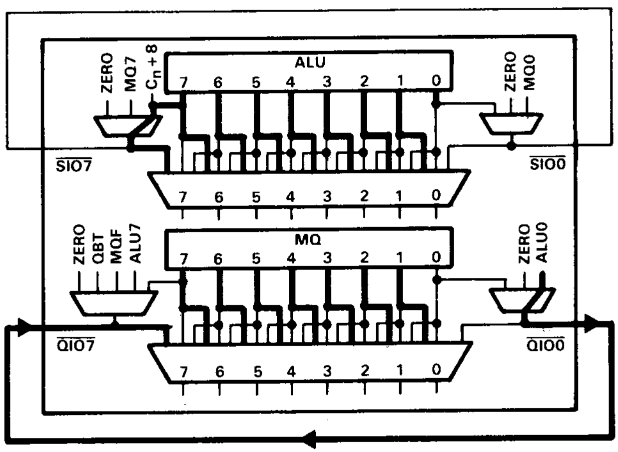


FIGURE 8. MULTIPLICATION OPERATIONS

The unsigned multiply iterate (UMULI) performs an unsigned multiplication iteration. This instruction interprets M(8-J) as the 8-J bit of the multiplier. The shift is a double-precision right shift with the carry out from the P(J) + Multiplicand × M(8-J) operation forced into bit 8 of P(J + 1). This instruction is used in unsigned and mixed multiplication.

signed multiplication

Signed multiplication performs a ten clock cycle, two's complement multiply. The instructions necessary to produce an algebraically correct result proceed in the following manner:

Zero register to be used for accumulator

Load MQ with multiplier

SMULI (repeat 7 times)	S port	=	Accumulator
	R port	=	Multiplicand
	F port	=	Iteration Result
SMULT	S port	=	Accumulator
	R port	=	Multiplicand
	F port	=	Product (MSH)

At completion, the accumulator will contain the 8 most significant bits and MQ will contain the 8 least significant bits of the product.

The status for the signed multiply iterate should not be used for any testing (overflow is not set by SMULI). The following status is set for the signed multiply terminate instruction:

N	→	Result MSB equal one
OVR	→	Forced to zero
C_{n+8}	→	Carry out equal to one
Z	→	Double precision result is zero

unsigned multiplication

Unsigned multiplication produces an unsigned times unsigned product in ten clocks. The instructions necessary to produce an algebraically correct result proceed in the following manner:

Zero register to be used for accumulator

Load MQ with multiplier

UMULI (8 times)	S port	=	Accumulator
	R port	=	Multiplicand
	F port	=	Iteration result (product MSH on final result)

Upon completion, the accumulator will contain the 8 most significant bits and MQ will contain the 8 least significant bits of the product.

The status set by the unsigned multiply iteration is meaningless except on the final execution of the instruction. The status set by the unsigned multiply iteration instruction is as follows:

N → Result MSB equal one
OVR → Forced to zero
C_{n+8} → Carry out equal to one
Z → Double-precision result is zero

mixed multiplication

Mixed multiplication multiplies a signed multiplicand times an unsigned multiplier to produce a signed result in ten clocks. The steps are as follows:

Zero register used for accumulator

Load MQ with unsigned multipler

SMULI (8 times) S port = Accumulator
R port = Multiplicand
F port = Iteration result

Upon completion, the accumulator will contain the 8 most significant bits and MQ will contain the 8 least significant bits of the product.

The following status is set by the last SMULI instruction:

N → Result MSB equal one
OVR → Forced to zero
C_{n+8} → Carry out equal to one
Z → Double-precision result is zero

divide operations

The divide uses a nonrestoring technique to perform both signed and unsigned division of a 16 bit integer dividend and an 8 bit integer divisor (Figure 9). It produces an 8 integer quotient and remainder.

The remainder and quotient will be such that the following equation is satisfied:

$$(\text{Quotient}) \times (\text{Divisor}) + \text{Remainder} = \text{Dividend}$$

The processor has the following divide instructions:

1. UNSIGNED DIVIDE START (UDIVIS): I7-I0 = $B0_{16}$
2. UNSIGNED DIVIDE ITERATE (UDIVI): I7-I0 = $C0_{16}$
3. UNSIGNED DIVIDE TERMINATE (UDIVIT): I7-I0 = $F0_{16}$
4. SIGNED DIVIDE INITIALIZE (SDIVIN): I7-I0 = 80_{16}
5. SIGNED DIVIDE OVERFLOW TEST (SDIVO): I7-I0 = AF_{16}
6. SIGNED DIVIDE START (SDIVIS): I7-I0 = 90_{16}
7. SIGNED DIVIDE ITERATE (SDIVI): I7-I0 = $A0_{16}$
8. SIGNED DIVIDE TERMINATE (SDIVIT): I7-I0 = $E0_{16}$
9. DIVIDE REMAINDER FIX (DIVRF): I7-I0 = 40_{16}
10. SIGNED DIVIDE QUOTIENT FIX (SDIVQF): I7-I0 = 50_{16}

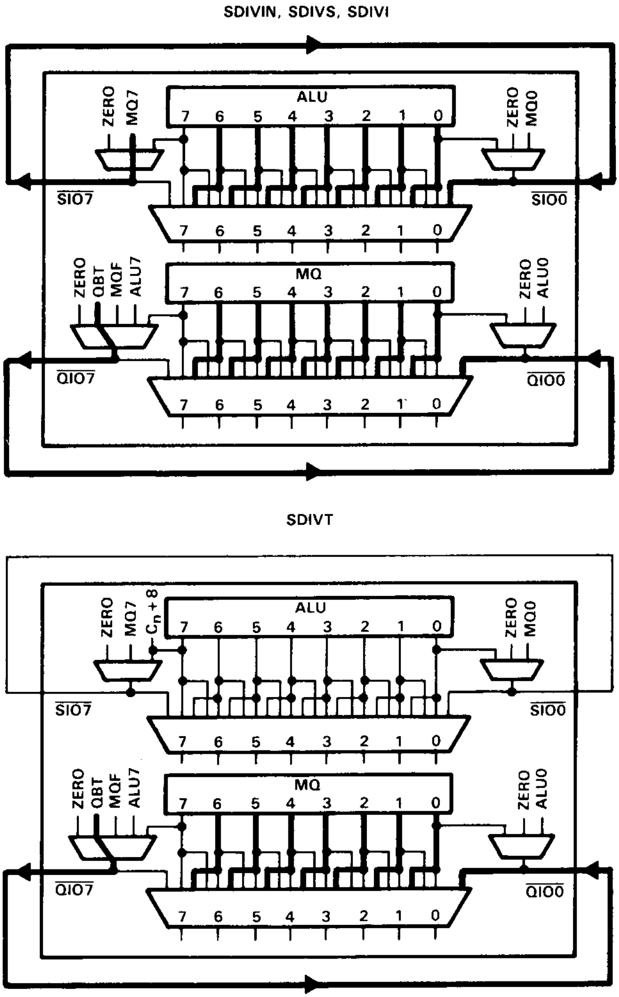
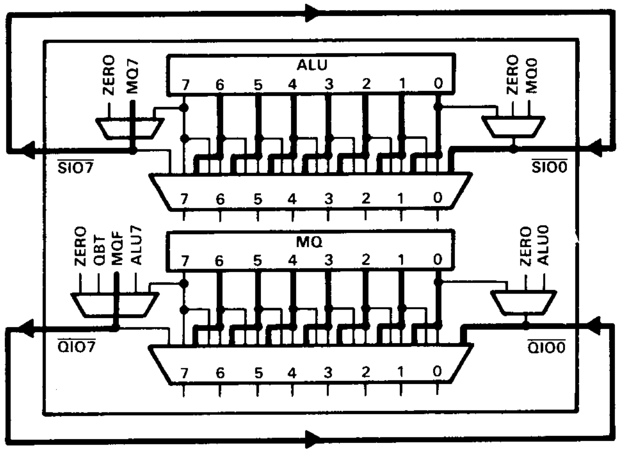


FIGURE 9. DIVIDE OPERATIONS

TEXAS INSTRUMENTS
POST OFFICE BOX 225012 • DALLAS, TEXAS 75265

UDIVS, UDIVI



UDIVT

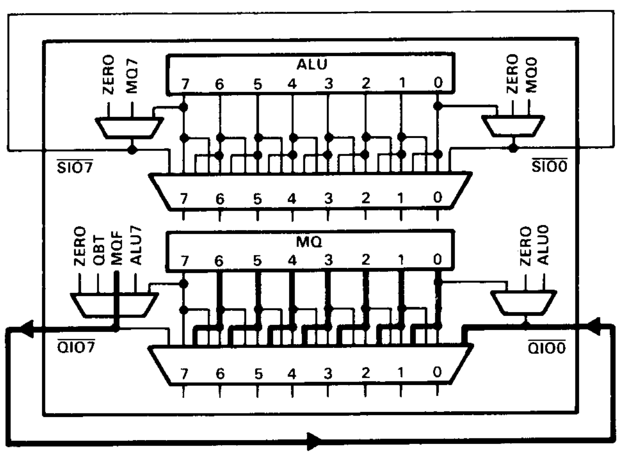


FIGURE 9. DIVIDE OPERATIONS (Continued)

The unsigned divide iterate start (UDIVIS) instruction begins the iterate procedure while testing for overflow. Overflow is reported when the first subtraction of the divisor from the MSH of the dividend produces carry out. The test detects quotient overflow and divide by zero.

The unsigned divide iterate terminate (UDIVIT) instruction completes the iterate procedure generating the last quotient bit.

The signed divide initialize (SDIVIN) instruction prepares for iteration by shifting the dividend and storing the sign of the dividend for use in the following instructions and overflow tests.

The signed divide overflow test (SDIVO) checks for overflow possibilities. This instruction may be deleted from the divide operation if the OVR pin is ignored. If it is removed some overflow conditions will go undetected. $\overline{\text{WE}}$ must be high (writing inhibited) when this instruction is used.

The signed divide iterate start (SDIVIS) instruction calculates the difference between the divisor and MSH of the dividend. Partial detection of overflow is also done during this instruction. Operations with like signs (positive quotient) and division by zero will overflow during this instruction (including zero divisor). Operations with unlike signs are tested for overflow during the signed divide quotient fix instruction (SDIVQF). Partial overflow results are saved and will be used during SDIVQF when overflow is reported.

The signed divide iterate (SDIVI) instruction forms the quotient and remainder through iterative subtract/add-shift operations of the divisor and dividend. One quotient bit is generated on each clock.

The signed divide iterate terminate (SDIVIT) instruction completes the iterate procedure, generating the last quotient bit. It also tests for a remainder equal to zero, which determines the action to be taken in the following correction (fix) instructions.

The divide remainder fix (DIVRF) instruction corrects the remainder. If a zero remainder was detected by the previous instructions, the remainder is forced to zero. For nonzero remainder cases where the remainder and dividend have the same sign, the remainder is correct. When the remainder and dividend have unlike signs, a correction add/subtract of the divisor to the remainder is performed.

The signed divide quotient fix (SDIVQF) instruction corrects the quotient if necessary. This correction requires adding one to the incorrect quotient. An incorrect quotient results if the signs of the divisor and dividend differ and the remainder is nonzero. An incorrect quotient also results if the sign of the divisor is negative and the remainder is zero.

Overflow detection is completed during this instruction. Overflow may be generated for differing signs of the dividend and divisor. The partial overflow test result performed during SDIVIS is ORed with this test result to produce a true overflow indication.

signed divide usage

The instructions necessary to perform an algebraically correct division of signed numbers are as follows:

Load MQ with the least significant half of the dividend

SDIVIN	S port = MSH of dividend R port = Divisor F port = Intermediate result
SDIVO	S port = Result of SDIVIN R port = Divisor F port = Test result ($\overline{WE}$ must be high)
SDIVIS	S port = Result of SDIVIN R port = Divisor F port = Intermediate result
SDIVI (8N-2 times)	S port = Result of SDIVIS (or SDIVI) R port = Divisor F port = Intermediate result
SDIVIT	S port = Result of last SDIVI R port = Divisor F port = Intermediate result
DIVRF	S port = Result of SDIVIT R port = Divisor F port = Remainder
SDIVQF	S port = MQ register R port = Divisor F port = Quotient

The status of all signed divide instructions except SDIVIN, DIVRF, and SDIVQF is as follows:

N → Forced to zero
OVR → Forced to zero
C_{n+8} → Carry out equal to one
Z → Intermediate result is zero

The status of the SDIVIN instruction is as follows:

N → Forced to zero
OVR → Forced to zero
C_{n+8} → Forced to zero
Z → Divisor is zero

The status of the DIVRF instruction is as follows:

N → Forced to zero
OVR → Forced to zero
C_{n+8} → Carry out equal to one
Z → Remainder is zero

2

LSI Devices

The status of the SDIVQF instruction is as follows:

N → Sign of quotient
OVR → Divide overflow
C_{n+8} → Carry out equal to one
Z → Quotient is zero

The quotient is stored in the MQ register and the remainder is stored in the register file location that originally held the most significant word of the dividend. If fractions are divided, the quotient must be shifted right one bit and the remainder right three bits to obtain the correct fractional representations.

The signed division algorithm is summarized in Table 7.

TABLE 7. SIGNED DIVISION ALGORITHM

OP CODE	MNEMONIC	CLOCK CYCLES	INPUT S PORT	INPUT R PORT	OUTPUT F PORT
E4	LOADMQ	1	Dividend (LSH)	–	Dividend (LSH)
80	SDIVIN	1	Dividend (MSH)	Divisor	Remainder
AF	SDIVO	1	Remainder	Divisor	Test Result
90	SDIVIS	1	Remainder	Divisor	Remainder
A0	SDIVI	7	Remainder	Divisor	Remainder
E0	SDIVIT	1	Remainder	Divisor	Remainder (Unfixed)
40	DIVRF	1	Remainder (Unfixed)	Divisor	Remainder
50	SDIVQF	1	MQ Register	Divisor	Quotient

unsigned divide usage

The instructions necessary to perform an algebraically correct division of unsigned numbers are as follows:

Load MQ with the least significant half of the dividend

UDIVIS
S port = MSH of dividend
R port = Divisor
F port = Intermediate result

UDIVI (8-1 times)
S port = Result of UDIVIS (OR UDIVI)
R port = Divisor
F port = Intermediate result

UDIVIT
S port = Result of last UDIVI
R port = Divisor
F port = Remainder (unfixed)

DIVRF
S port = Result of UDIVIT
R port = Divisor
F port = Remainder

The status of all unsigned divide instructions except UDIVIS is as follows:

N → Forced to zero
OVR → Forced to zero
C_{n+8} → Carry out equal to one
Z → Intermediate result is zero

The status of the UDIVIS instruction is as follows:

N → Forced to zero
OVR → Divide overflow
C_{n+8} → Carry out equal to one
Z → Intermediate result is zero

If fractions are divided, the remainder must be shifted right two bits to obtain the correct fractional representation. The quotient is correct as is. The quotient is stored in the MQ register at the completion of the divide.

The unsigned division algorithm is summarized in Table 8.

TABLE 8. UNSIGNED DIVISION ALGORITHM

OP CODE	MNEMONIC	CLOCK CYCLES	INPUT S PORT	INPUT R PORT	OUTPUT F PORT
E4	LOADMQ	1	Dividend (LSH)	—	Dividend (LSH)
B0	UDIVIS	1	Dividend (MSH)	Divisor	Remainder
C0	UDIVI	7	Remainder	Divisor	Remainder
F0	UDIVIT	1	Remainder	Divisor	Remainder (Unfixed)
40	DIVRF	1	Remainder (Unfixed)	Divisor	Remainder

group 5 instructions

Hex code F of Group 1 instructions is used to access Group 5 instructions. Group 5 instructions are summarized in Table 9.

TABLE 9. GROUP 5 INSTRUCTIONS

INSTRUCTION BITS (I7-I0) OP CODE (HEX)	MNEMONIC	FUNCTION
0F	CLR	Clear
1F	CLR	Clear
2F	CLR	Clear
3F	CLR	Clear
4F	CLR	Clear
5F	CLR	Clear
6F	CLR	Clear
7F	BCDBIN	BCD to Binary
8F		Reserved
9F	EX3C	Excess-3 Word Correction
AF	SDIVO	Signed Divide Overflow Check
BF	CLR	Clear
CF	CLR	Clear
DF	BINEX3	Binary to Excess-3
EF	CLR	Clear
FF	NOP	No Operation

clear instructions (CLR)

There are 11 clear instructions listed in Table 9. The instructions force the ALU output to be zero and the BCD flip-flops to be cleared. The status set by the clear instruction is as follows:

N → None (force to zero)
OVR → None (force to zero)
C_{n+8} → None (force to zero)
Z → Active (one)

no operation instruction (NOP): I7-I0 = FF_{16}

This instruction is identical to the clear instructions except that the BCD flip-flops retain their old value.

excess-3 correction instructions (EX3C): $9F_{16}$

This instruction corrects excess-3 additions (subtractions). For correct excess-3 arithmetic, this instruction must follow each add/subtract. The operand must be on the S port.

NOTE: The previous arithmetic overflow should be ignored.

The status set by the EX3C instruction is as follows:

N → MSB of result
OVR → Signed overflow
C_{n+8} → Carry out equal one
Z → None (force to one)

radix conversions

Conversions between decimal and binary number representations are performed with the aid of two special instructions: BINEX3 and BCDBIN.

BCD to binary instructions (BCDBIN): I7-I0 = $7F_{16}$

This instruction (Figure 10) allows the user to convert a 2-digit BCD number to an 8-bit binary number in 12 clocks. This function sums the R bus, the S bus, and the C_n bit, performs an arithmetic left shift on the ALU result, and simultaneously circular shifts the MQ left. The status set by the BCD to binary instruction is as follows:

N → MSB of result
OVR → Signed arithmetic overflow†
C_{n+8} → Carry out equal one
Z → Result equal zero

† Overflow may be the result of an ALU operation or the arithmetic left shift operation.

TEXAS INSTRUMENTS
POST OFFICE BOX 225012 • DALLAS, TEXAS 75265

The following code illustrates the BCD to binary conversion technique.

Let ACC be an accumulator register
Let NUM be the register which contains the BCD number
Let MSK be a mask register

```
LOADMQ NUM                      ; LOAD MQ WITH BCD NUMBER
SUB ACC, ACC, SLCMQ             ; CLEAR ACC AND ALIGN MQ
SUB, MSK, MSK, SLCMQ            ; CLEAR MSK AND ALIGN MQ
SLCMQ                           ; ALIGN
SLCMQ                           ; ALIGN
ADDI ACC, MSK, 15(10)           ; MSK = 15(10)
AND MQ, MSK, R1, SLCMQ          ; EXTRACT ONE DIGIT
                                ; ALIGN MQ
ADD, ACC, R1, R1, SLCMQ         ; ACC + DIGIT
                                ; IS STORED IN R1
                                ; ALIGN MQ
BCDBIN, R1, R1, ACC             ; 4 x (ACC + DIGIT)
                                ; IS STORED IN ACC
                                ; ALIGN MQ
BCDBIN, ACC, R1, ACC            ; 10 x (ACC + DIGIT)
                                ; IS STORED IN ACC
                                ; ALIGN MQ
AND MQ, MSK, R1                 ; FETCH LAST DIGIT
ACC+R1 → ACC                    ; ADD IN LAST DIGIT
```

The previous code generates a binary number by executing the standard conversion formula for a 2-digit BCD number.

$$AB = A \times 10 + B$$

Notice that the conversion begins with the most significant BCD digit and that the addition is performed in radix 2.

binary to excess-3 instructions (BINEX3): I7-I0 = DF_{16}

This instruction (Figure 11) allows the user to convert an 8-bit binary number to 2-digit excess-3 number representation in 19 clocks. The data on the R and S ports are summed with the MSB of the MQ register. The MQ register is simultaneously shifted left circularly. The status set by the binary to excess-3 instruction is as follows:

N	→	MSB of result
OVR	→	Signed arithmetic overflow
C_{n+8}	→	Carry out equal one
Z	→	Result equal zero

BCDBIN

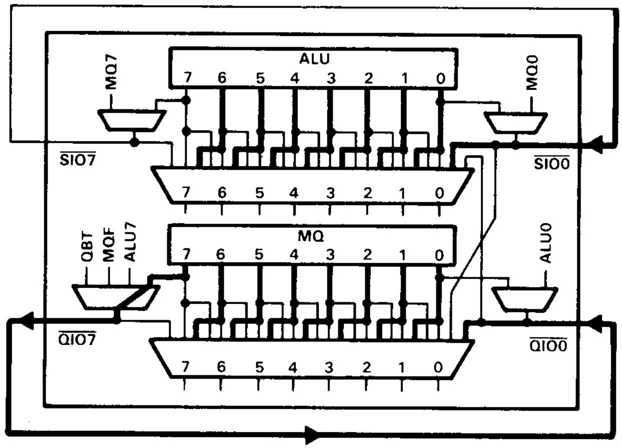


FIGURE 10. BCD TO BINARY

BINEX3

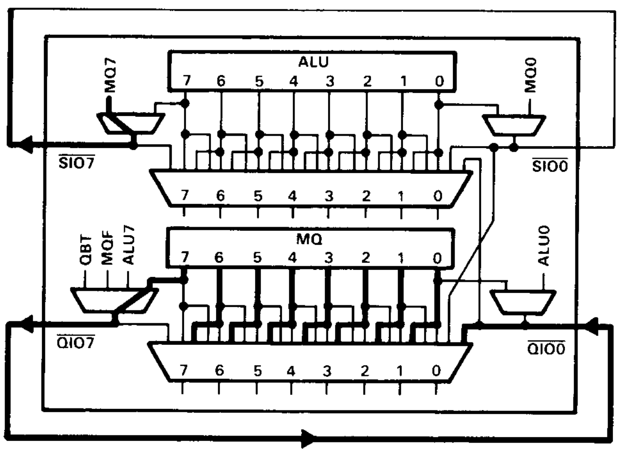


FIGURE 11. BINARY TO EXCESS-3

The following illustrates the binary to excess-3 conversion technique.

Let NUM be a register containing an unsigned binary number
Let ACC be an accumulator

```
M1:        LOADMQ NUM              ; LOAD MQ WITH BINARY
                                   ; NUMBER
M2:        CLEAR ACC               ; CLEAR ACC
M3:        SET1 ACC H/33/          ; ACC → HEX/3333 . . .
L1:        BINEX3 ACC, ACC, ACC    ; DOUBLE ACC AND ADD IN
                                   ; MSB OF MQ
                                   ; ALIGN MQ
L2:        EX3C ACC, ACC           ; EXCESS 3 CORRECT
                                   ; REPEAT L1 AND L2
                                   ; 7 TIMES
```

The previous code generates an excess-3 number by executing the standard conversion formula for a binary number.

$$a_n 2^n + a_{n-1} 2^{n-1} + a_{n-2} 2^{n-2} + \ldots a_0 2^0 = [(2a_n + a_{n-1})2 + a_{n-2}]2 + \ldots a_0$$

Notice that the conversion begins with the most significant binary bit and that the addition is performed in radix-10 (excess-3).

decimal arithmetic

Decimal numbers are represented in excess-3 code. Excess-3 code numbers may be generated by adding three to each digit of a Binary Coded Decimal (BCD) number. The hardware necessary to implement excess-3 arithmetic is only slightly different from binary arithmetic. Carries from one digit to another during addition in BCD occur when the sum of the two digits plus the carry-in is greater than or equal to ten. If both numbers are excess-3, the sum will be excess-6, which will produce the proper carries. Therefore, every addition or subtraction operation may use the binary adder. To convert the result from excess-6 to excess-3, one must consider two cases resulting from a BCD digit add: (1) where a carry-out is produced, and (2) where a carry-out is not produced. If a carry-out is not produced, three must be subtracted from the resulting digit. If a carry is produced, the digit is correct as a BCD number. For example, if BCD 5 is added to BCD 6, the excess-3 result would be 8 + 9 = 1 (with a carry). A carry rolls the number through the illegal BCD representations into a correct BCD representation. Binary 3 must be added to digit positions that produce a carry-out to correct the result to an excess-3 representation. Every addition and subtraction instruction stores the carry generated from each 4-bit digit location for use by the excess-3 correction function. The correction instruction must be executed in the clock cycle immediately after the addition or subtraction operation.

absolute maximum rating over operating free-air temperature range (unless otherwise noted)

Supply voltage, V_{CC1} . . . . 7 V
Supply voltage, V_{CC2} . . . . 3 V
Input voltage . . . . 7 V
High-level voltage applied to 3-state outputs . . . . 5.5 V
Operating case temperature range: SN54AS887 . . . . −55°C to 125°C
Operating free-air temperature range: SN74AS887, SN74AS887-1 . . . . 0°C to 70°C
Storage temperature range . . . . −65°C to 150°C

recommended operating conditions

			SN54AS887			SN74AS887 SN74AS887-1			UNIT
			MIN	NOM	MAX	MIN	NOM	MAX	
V_{CC1}	I/O supply voltage		4.5	5	5.5	4.5	5	5.5	V
V_{CC2}	STL internal logic supply voltage		1.9	2	2.1	1.9	2	2.1	V
V_{IH}	High-level input voltage		2			2			V
V_{IL}	Low-level input voltage				0.8			0.8	V
I_{OH}	High-level output current				−1			−2.6	mA
I_{OL}	Low-level output current	All output except N and ZERO			8			8	mA
		N			16			16	
		ZERO			48			48	
T_C	Operating case temperature		−55		125				°C
T_A	Operating free-air temperature					0		70	

electrical characteristics over recommended operating free-air temperature range (unless otherwise noted)

PARAMETER		TEST CONDITIONS	SN54AS887			SN74AS887 SN74AS887-1			UNIT
			MIN	TYP†	MAX	MIN	TYP†	MAX	
V_{IK}		V_{CC1} = 4.5 V, I_I = −18 mA			−1.2			−1.2	V
V_{OH}	All outputs except ZERO	V_{CC1} = 4.5 V to 5.5 V, I_{OH} = −0.4 mA	V_{CC} − 2			V_{CC} − 2			V
		V_{CC1} = 4.5 V, I_{OH} = −1 mA	2.4						
		V_{CC1} = 4.5 V, I_{OH} = −2.6 mA				2.4			
I_{OH}	ZERO	V_{CC1} = 4.5 V, V_{OH} = 5.5 V			0.1			0.1	mA
V_{OL}	All outputs except N and ZERO	V_{CC1} = 4.5 V, I_{OL} = 8 mA			0.5			0.5	V
	N	V_{CC1} = 4.5 V, I_{OL} = 16 mA			0.5			0.5	
	ZERO	V_{CC1} = 4.5 V, I_{OL} = 48 mA			0.5			0.5	
I_I	I/O	V_{CC1} = 5.5 V, V_I = 5.5 V			0.1			0.1	mA
	All others	V_{CC1} = 5.5 V, V_I = 7 V			0.1			0.1	
I_{IH}‡		V_{CC1} = 5.5 V, V_I = 2.7 V			20			20	μA
I_{IL}‡		V_{CC1} = 5.5 V, V_I = 0.5 V			−0.4			−0.4	mA
I_O§		V_{CC1} = 5.5 V, V_O = 2.25 V	−30		−112	−30		−112	mA
I_{CC1}		V_{CC1} = 5.5 V			150			130	mA
I_{CC2}		V_{CC2} = 2.1 V			410			390	mA

†All typical values are at V_{CC} = 5 V, T_A = 25°C.
‡For I/O ports, the parameters I_{IH} and I_{IL} include the off-state current.
§The output conditions have been chosen to produce a current that closely approximates one-half the true short-circuit current, I_{OS}.

TEXAS INSTRUMENTS
POST OFFICE BOX 225012 • DALLAS, TEXAS 75265

SN54AS887 maximum switching characteristics, V_{CC} = 4.5 V to 5.5 V, T_C = −55 °C to 125 °C (see Note 1)

PARAMETER	FROM (INPUT)	TO (OUTPUT)										UNIT
		Y	C_{n+8}	$\overline{G}$, $\overline{P}$	Z†	N	OVR	DA	DB	$\overline{QIO}$	$\overline{SIO}$	
t_{pd}	A3-A0 B3-B0	62	42	48	69	62	60	18	18	65	66	ns
	DA7-DA0, DB7-DB0	47	28	28	58	50	42	–	–	50	50	
	C_n	25	14	–	32	24	18	–	–	32	32	
	$\overline{EA}$	54	32	35	62	52	52	–	–	58	58	
	$\overline{EB}$	54	32	35	62	52	52	–	–	58	58	
	I7-I0	58	32	32	62	52	41	–	–	58	58	
	$\overline{OEB}$	–	–	–	–	–	–	–	14	–	–	
	$\overline{OEY}$	14	–	–	–	–	–	–	–	–	–	
	$\overline{QIO}$ (n) Shift	15	–	–	24	–	–	–	–	–	–	
	$\overline{SIO}$ (n) Shift	15	–	–	24	22	–	–	–	–	–	
	CK	68	60	56	62	50	68	38	38	70	70	
	$\overline{OEA}$	–	–	–	–	–	–	14	–	–	–	

† Load resistor R1 = 100 Ω.
NOTE 1: Load circuit and voltage waveforms are shown in Section 1.

SN74AS887 maximum switching characteristics, V_{CC} = 4.5 V to 5.5 V, T_A = 0 °C to 70 °C (see Note 1)

PARAMETER	FROM (INPUT)	TO (OUTPUT)										UNIT
		Y	C_{n+8}	$\overline{G}$, $\overline{P}$	Z†	N	OVR	DA	DB	$\overline{QIO}$	$\overline{SIO}$	
t_{pd}	A3-A0 B3-B0	54	36	42	60	52	50	18	18	58	58	ns
	DA7-DA0, DB7-DB0	44	26	26	52	46	38	–	–	44	44	
	C_n	25	8	–	32	24	18	–	–	31	31	
	$\overline{EA}$	49	29	29	58	49	47	–	–	54	54	
	$\overline{EB}$	49	29	29	58	49	47	–	–	54	54	
	I7-I0	55	30	30	60	49	39	–	–	54	54	
	$\overline{OEB}$	–	–	–	–	–	–	–	12	–	–	
	$\overline{OEY}$	12	–	–	–	–	–	–	–	–	–	
	$\overline{QIO}$ (n) Shift	15	–	–	24	–	–	–	–	–	–	
	$\overline{SIO}$ (n) Shift	15	–	–	24	19	–	–	–	–	–	
	CK	58	55	52	61	52	62	35	35	60	60	
	$\overline{OEA}$	–	–	–	–	–	–	12	–	–	–	

† Load resistor R1 = 100 Ω.
NOTE 1: Load circuit and voltage waveforms are shown in Section 1.

SN74AS887-1 maximum switching characteristics, V_{CC} = 4.5 V to 5.5 V, T_A = 0°C to 70°C (see Note 1)

PARAMETER	FROM (INPUT)	TO (OUTPUT) Y	C_{n+8}	$\overline{G}$, $\overline{P}$	Z†	N	OVR	DA	DB	$\overline{QIO}$	$\overline{SIO}$	UNIT
t_{pd}	A3-A0 B3-B0	44	30	36	50	44	44	17	17	48	48	ns
	DA7-DA0, DB7-DB0	36	24	24	46	41	32	–	–	40	40	
	C_n	22	8	–	27	21	16	–	–	25	25	
	$\overline{EA}$	40	25	25	49	41	41	–	–	44	44	
	$\overline{EB}$	40	25	25	49	41	41	–	–	44	44	
	I7-I0	46	27	27	50	42	35	–	–	45	45	
	$\overline{OEB}$	–	–	–	–	–	–	–	12	–	–	
	$\overline{OEY}$	12	–	–	–	–	–	–	–	–	–	
	$\overline{QIO}$ (n) Shift	14	–	–	20	–	–	–	–	–	–	
	$\overline{SIO}$ (n) Shift	14	–	–	20	18	–	–	–	–	–	
	CK	50	46	46	50	50	50	30	30	50	50	
	$\overline{OEA}$	–	–	–	–	–	–	12	–	–	–	

† Load resistor R1 = 100 Ω.

NOTE 1: Load circuit and voltage waveforms are shown in Section 1.

register file write setup and hold times

PARAMETER		SN54AS887 MIN	SN54AS887 MAX	SN74AS887 MIN	SN74AS887 MAX	SN74AS887-1 MIN	SN74AS887-1 MAX	UNIT
t_{su}	C3-C0	8		7		6		ns
	DB§	14		12		11		
	I7-I4	16		14		13		
	I3-I0	24		22		21		
	$\overline{OEY}$	4		3		3		
	Y7-Y0	2		2		2		
	$\overline{WE}$	8		6		6		
	$\overline{QIO}$(n), $\overline{SIO}$(n)	6		5		5		
	SELY	8		6		6		
t_h	C3-C0	0		0		0		ns
	DB§	0		0		0		
	I7-I4	0		0		0		
	I3-I0	0		0		0		
	$\overline{OEY}$	6		5		5		
	Y7-Y0	10		10		10		
	$\overline{WE}$	3		2		2		
	$\overline{QIO}$(n), $\overline{SIO}$(n)	0		0		0		
	SELY	8		6		6		

§ DB (during select instruction) through Y port.

TEXAS INSTRUMENTS
POST OFFICE BOX 225012 • DALLAS, TEXAS 75265

special instruction switching characteristics

The SSF pin is used internally during certain instructions. The following tables list the instructions which force the SSF pin during their execution. The propagation delay from various inputs is also shown. The parameter which limits normal system performance is indicated by a dagger.

SN54AS887 SSF PIN DELAYS AND SETUP TIMES

MNEMONIC	HEX CODE	INPUT → SSF (ns)				SSF SETUP TIME (ns)
		C_n	$I_{(n)}$	CK	$B_{(n)}$	
SNORM	20	–	29†	46	–	20
DNORM	30	–	29	55	40†	20
DIVRF	40	–	29†	46	–	20
SDIVQF	50	–	26†	–	–	18
SMULI	60	–	26†	43	–	0
SDIVIN	80	–	48	64	44†	0
SDIVIS	90	26†	51	64	55	0
SDIVI	A0	26†	51	64	55	0
UDIVIS	B0	18†	45	64	46	0
UDIVI	C0	18†	50	54	40	0
UMULI	D0	–	25†	48	–	0
SDIVIT	E0	26†	50	56	54	0
ABX	48	–	34	62	39†	20
SMTC	58	–	29	58	39†	20
BINEX3	DF	–	29†	58	–	18
LOADMQ (Arith)		23†	34	62	40	0
LOADMQ (Log)		–	33	62	40†	0

† This parameter limits normal system performance.

2

LSI Devices

SN74AS887 SSF PIN DELAYS AND SETUP TIMES

MNEMONIC	HEX CODE	INPUT → SSF (ns)				SSF SETUP TIME (ns)
		C_n	$I_{(n)}$	CK	$B_{(n)}$	
SNORM	20	–	26†	40	–	17
DNORM	30	–	26	52	37†	17
DIVRF	40	–	26†	40	–	17
SDIVQF	50	–	25†	–	–	17
SMULI	60	–	25†	40	–	0
SDIVIN	80	–	38	60	40†	0
SDIVIS	90	24†	48	60	52	0
SDIVI	A0	24†	48	60	52	0
UDIVIS	B0	17†	43	60	45	0
UDIVI	C0	17†	44	52	37	0
UMULI	D0	–	26†	40	–	0
SDIVIT	E0	25†	46	52	49	0
ABX	48	–	32	60	38	17
SMTC	58	–	26	52	38†	17
BINEX3	DF	–	26†	40	–	17
LOADMQ (Arith)		22†	32	50	38	0
LOADMQ (Log)		–	32	50	38†	0

† This parameter limits normal system performance.

TEXAS INSTRUMENTS
POST OFFICE BOX 225012 • DALLAS, TEXAS 75265

SN74AS887-1 SSF PIN DELAYS AND SETUP TIMES

MNEMONIC	HEX CODE	INPUT → SSF (ns)				SSF SETUP TIME (ns)
		C_n	$I_{(n)}$	CK	$B_{(n)}$	
SNORM	20	–	23†	28	–	14
DNORM	30	–	23	40	34†	14
DIVRF	40	–	23†	27	–	14
SDIVQF	50	–	23†	–	–	14
SMULI	60	–	22†	27	–	0
SDIVIN	80	–	35	46	35†	0
SDIVIS	90	22†	42	48	42	0
SDIVI	A0	22†	42	46	42	0
UDIVIS	B0	16†	42	46	38	0
UDIVI	C0	16†	36	46	34	0
UMULI	D0	–	22†	27	–	0
SDIVIT	E0	21†	40	44	42	0
ABX	48	–	28	46	30†	14
SMTC	58	–	24	44	30†	14
BINEX3	DF	–	23†	27	–	14
LOADMQ (Arith)		19†	28	40	30	0
LOADMQ (Log)		–	28	35	30†	0

† This parameter limits normal system performance.

2

LSI Devices