pdp11

KE11-E and KE11-F instruction set options user's manual

digital

EK-KE11E-OP-001

KE11-E and KE11-F instruction set options user's manual

digital equipment corporation • maynard, massachusetts

1st Edition, September 1976

Printed in U.S.A.

This document was set on DIGITAL's DECset-8000 computerized typesetting system.

CONTENTS

ILLUSTRATIONS

TABLES

INTRODUCTION

This manual describes the KE11-E Extended Instruction Set (EIS) and KE11-F Floating Instruction Set (FIS) Options to the KD11-A Programmed Data Processor for the PDP-11/40 System. These two options are described in one manual because of their interdependency, in that KE11-F cannot be installed without the KE11-E being first installed. The purpose of this manual is to:

1. Provide an overall understanding of the functions of these options in a PDP-11/40 System.

2. Explain how the KE11-E and KE11-F can be used in software operating systems.

In this manual each chapter is split in two with the first half of the chapter presenting information concerning the KE11-E Option and the second half being devoted to comparable information for the KE11-F Option. This organization is intended to facilitate greater ease in use by those customers who utilize only the EIS hardware.

Chapter 1 provides an introduction to the options and lists brief specifications. Chapter 2 contains installation information. Chapter 3 contains programming information, listing instructions and illustrating their formats.

Detailed descriptions of processor, console, Unibus, and memory logic that interface with these options are provided in the following related documents:

PDP-11/40 System Maintenance Manual	DEC-11-H40SA-A-D
KD11-A Central Processor Unit Maintenance Manual	EK-KD11A-MM-001

CHAPTER 1
GENERAL DESCRIPTION

This chapter contains a general description of both the KE11-E and KE11-F Options. Mechanical descriptions are given together with engineering specifications for each option. The chapter is divided in half with the EIS information presented first, followed by comparable information for the FIS hardware.

1.1 KE11-E EXTENDED INSTRUCTION SET

The KE11-E Extended Instruction Set is a hardware option to the basic PDP-11/40 Computer System. It is supplied as a pluggable option to the KD11-A Central Processor.

1.1.1 Purpose

The KE11-E Option expands the instruction set of the KD11-A Central Processor to provide extended manipulation of fixed-point numbers. When installed, it adds the capability of Arithmetic Shift, Arithmetic Shift Combined, Multiply, and Divide. With these additional instructions, the system can multiply and divide signed 16-bit numbers, and can shift signed 16-bit or 32-bit numbers. Condition codes are set in the processor on the result of each instruction.

1.1.2 Configuration

The KE11-E Option consists of one module. The single-hex × 8-1/2 in. M7238 module plugs directly into slot 2 (A–F) of the processor system unit. This is a dedicated prewired slot such that no other modules need be moved to accommodate its installation. When installed, the module functions as an extension of the basic KD11-A data paths, branch control, and control ROM. Basic timing of the processor is not degraded by use of this module, nor is the NPR latency affected when its instructions are being executed. Interrupts are serviced at the end of each instruction in the standard manner.

1.1.3 Specifications

Specifications for the KE11-E Option are given in Table 1-1.

Table 1-1
KE11-E (EIS) Specifications

Instructions	Arithmetic Shift (ASH) Arithmetic Shift Combined (ASHC) Multiply (MUL) Divide (DIV)
Operations	Multiplication and division of signed 16-bit numbers Arithmetic shifting of signed 16-bit or 32-bit numbers

Table 1-1 (Cont)
KE11-E (EIS) Specifications

Addressable Registers	None in option. Operands fetched from core or processor general registers.
Timing	Time = SRC Time + EF Time

SRC Mode	SRC Time
0	0.28 μs
1	0.78 μs
2	0.98 μs
3	1.74 μs
4	0.98 μs
5	1.74 μs
6	1.74 μs
7	2.64 μs

Instr	EF Time	Notes
MUL	8.88 μs	
DIV	11.30 μs	
ASH (right)	2.58 μs	+0.30 μs/shift
ASH (left)	2.78 μs	+0.30 μs/shift
ASHC (no shift)	2.78 μs	
ASHC (shift)	3.26 μs	+0.30 μs/shift

Size	Single Hex module (M7238)
Power Required	+5V, 2.3A

1.2 KE11-F FLOATING INSTRUCTION SET

The KE11-F Floating Instruction Set is a hardware option to the basic PDP-11/40 Computer System. It is supplied as a pluggable option to the KD11-A Central Processor and requires that the KE11-E described above be installed as a prerequisite.

1.2.1 Purpose

The KE11-F Floating Instruction Set (FIS) enables direct operations on single-precision 32-bit words in floating-point arithmetic. Since the KE11-E is a prerequisite to the KE11-F, extended manipulation of fixed-point numbers is available as well. The KE11-F Option further extends the PDP-11/40 instruction set to include Floating Add, Floating Subtract, Floating Multiply, and Floating Divide. As with the KE11-E, condition codes in the Processor Status Register are set on the result of each instruction. The prime advantage of this option is increased speed without the necessity of writing complex floating-point software routines.

1.2.2 Configuration

The KE11-F Option consists of one single-quad X 8-1/2 in. M7239 module with the KE11-E Option described above being a prerequisite. This FIS module plugs directly into slot 1 (A–D) also a dedicated prewired slot in the basic KD11-A. No degradation of processor timing or NPR latency is effected by the use of this option. Floating instructions are aborted if a BR request is issued before the instruction is within approximately 8 μs of completion, at which time the Program Counter (PC) is adjusted to point to the aborted floating instruction so that the instruction will be restarted upon return from the interrupt.

1.2.3 Specifications

Specifications for the KE11-F Option are given in Table 1-2.

Table 1-2
KE11-F (FIS) Specifications

Prerequisite	KE11-E Extended Instruction Set Option
Instructions	Floating-point Addition (FADD) Floating-point Subtraction (FSUB) Floating-point Multiply (FMUL) Floating-point Divide (FDIV)
Operations	Single-precision floating-point addition, subtraction, multiplication, and division of 24-bit numbers
Addressable Registers	None in option. Operands fetched from core.
Size	Single-quad module (M7239)
Power Required	+5V, 1.1A (typical)
Timing	Time = Basic Time + Binary Point Alignment Time + Normalization Time

Instr	Basic Time* μs	Binary Point Alignment Time Per Shift μs	Normalization Time Per Shift μs
FADD	18.78	0.30	0.34
FSUB	19.08	0.30	0.34
FMUL	29.00	---	0.34
FDIV	46.27	---	0.34

*Basic instruction times for FADD and FSUB assume exponents are equal or differ by one.

CHAPTER 2
INSTALLATION

2.1 KE11-E PROCEDURE

When the KE11-E is included as part of the initial PDP-11/40 System, the M7238 module is installed prior to shipment. If it is being added to an existing system, proceed as follows:

a. Insert the M7238 module in 2(A–F).

b. Remove the jumper (W1) on processor module M7233 (IR DECODE) at location 5(A–F).

c. Install the three "over the back" cables from J1, J2, and J3 of the M7238 module to J1, J2, and J3 respectively of the M7232 (U Word) module at location 3(A–D).

2.2 KE11-F PROCEDURE

When the KE11-F is to be added to a system, the KE11-E must also be added. Proceed as follows:

a. Perform steps a. through c. above.

b. Insert the M7239 module in 1(A–D).

c. On the M7238 module, remove the following jumpers:

    1. W1 from C02F2 to ground.

    2. W2 from A02B1 to ground.

    3. W3 from D02L1 to ground.

NOTE

If these jumpers are not removed, the KE11-E Option will still execute EIS instructions but will not execute FIS instructions.

When the above steps are performed, the KE11-E and KE11-F Options are ready to be checked out using the diagnostic programs supplied with the options.

CHAPTER 3
PROGRAMMING

This chapter is devoted to general programming information for the KE11-E and KE11-F Options. It provides general descriptions of their operation, the formats and instructions for each. In addition, programming examples are supplied for each option. This chapter is intended merely as an introduction to the programming of this hardware. For more detailed information refer to the pertinent software documentation generated for these options. As with Chapter 1, information has been separated for each option.

3.1 KE11-E EXTENDED INSTRUCTION SET

There are no addressable registers in the KE11-E Option. EIS operands are fetched from either core memory or from the general processor registers. The result of each operation is stored in the general registers.

3.1.1 Operation

When the Arithmetic Shift (ASH) instruction is used, the contents of the selected register is shifted right or left the number of places specified by a count. This shift count is a 6-bit, 2's complement number which is the least significant 6 bits of the source operand. If the count is positive, the number is shifted left; if it is negative, the number is shifted right. This allows for shifts from 31 positions left to 32 positions right (+31 to −32) although a shift of greater than 16 places loses all significance. A count of 0 causes no change in the number.

When the Arithmetic Shift Combined (ASHC) instruction is used, the contents of the register (R) and the register ORed with 1 (RV1) are treated as a single 32-bit word. Register RV1 represents bits (15:00), register R represents bits (31:16). This 32-bit word is shifted right or left the number of places specified by a count. This shift count is the same as that described for the ASH instruction and permits shifts from +31 to −32. If the selected register (R) is an odd number, then R and RV1 are the same. In this case, the right shift becomes a rotate and the 16-bit word is rotated right the number of bits specified by the count for up to 16 shifts.

When the MULtiply (MUL) instruction is used, the contents of the Destination Register and the source are multiplied as 2's complement integers. The result is stored in the Destination Register R and the register ORed with 1 (RV1). If the register is odd, only the low-order product is stored. This instruction multiplies full 16-bit numbers.

When the DIVide (DIV) instruction is used, a 32-bit dividend in R and RV1 is divided by a 16-bit divisor to provide a 16-bit quotient and a 16-bit remainder. The sign of the remainder is always the same as the sign of the dividend unless the remainder is 0. Overflow is indicated if more than 16 bits are required to express the quotient. In this case, the instruction is aborted. If the content of the Source Register is 0, indicating divide by 0, an overflow is indicated.

3.1.2 Formats

The number formats for the KE11-E Option are shown in Figure 3-1. A single word is 16-bits long and a double word is 32-bits long. In the single word, bit 15 is the sign of the number; and in the double word, the sign bit is bit 15 of the high number part. The S bit is 0 for positive quantities and is 1 for negative quantities.

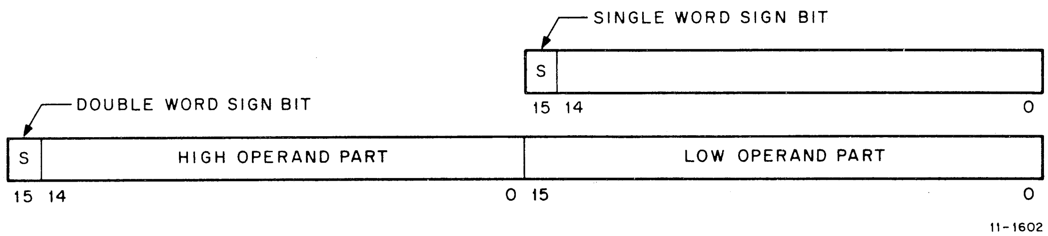


Figure 3-1 EIS Number Formats

3.1.3 Instructions

The EIS instruction format is shown in Figure 3-2. It is a double operand instruction in which bits (15:09) comprise the Op code, bits (08:06) designate the Destination Register field (RRR), bits (05:03) indicate the Source Address Mode (SSS), and bits (02:00) specify the Source Address Register (SSS). The octal coding is in the form 07XRSS. There are four EIS instructions, as follows:

MUL 070RSS

MULtiply

Operation: R, RV1 ← R X(SRC)

Condition Codes:

N: set if product is < 0; cleared otherwise.
Z: set if product is = 0; cleared otherwise.
V: cleared
C: set if the result is less than -2^{15} or is greater than or equal to $2^{15}-1$; cleared otherwise.

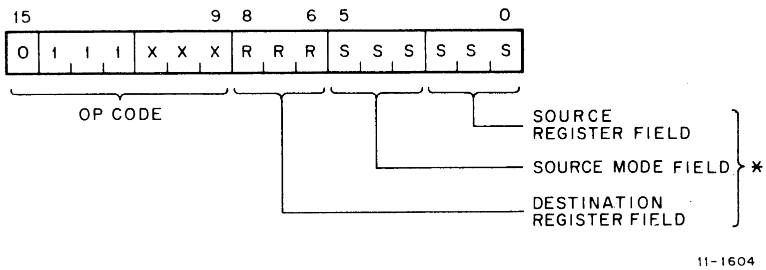
Description: The contents of the Destination Register R and source taken as 2's complement integers are multiplied and stored in the Destination Register R and the succeeding register RV1 (if R is even). If R is odd, only the low-order product is stored. Assembler syntax is:

MUL S, R. (Note that the actual destination is R, RV1 which reduces to just R when R is odd.)

Example: 16-bit product (R is odd)

```
000241        ,   CLC              ;Clear carry condition code
012701,400    ,   MOV #400, R1
070127, 10    ,   MUL #10, R1
1034xx        ,   BCS ERROR        ;Carry will be set if
                                   ;product is less than
                                   ;-2^15 or greater than or
                                   ;equal to 2^15
                                   ;no significance lost
```

Before	**After**
(R1)=000400	(R1)=004000



***Note that for the EIS instructions the Source Register is considered the Destination since the answer is stored in that register. The Destination Mode and Register Field are considered to be the source. This is not consistent with other PDP-11 family instruction formats but is used throughout the discussions of the EIS instructions in this manual.**

Figure 3-2 EIS Instruction Format

DIV 071RSS

DIVide

Operation: R ← R, RV1 ÷ (SRC) RV1 ← Remainder

Condition Codes:

N: set if quotient < 0; cleared otherwise.
Z: set if quotient = 0; cleared otherwise.
V: set if source = 0 or if the absolute value of the register is larger than the absolute value of the source. (In this case, the instruction is aborted because the quotient would exceed 16 bits.)
C: set if divide by 0 attempted; cleared otherwise.

Description: The 32-bit 2's complement integer in R and RV1 is divided by the source operand (SSS). The quotient is placed in R; the remainder is placed in RV1 with the same sign of the dividend. R must be even.

Example:

```
005000          ,    CLR R0
012701,20001    ,    MOV #20001,R1
071027,2        ,    DIV #2, R0
```

Before	**After**	
(R0)=000000	(R0)=010000	Quotient
(R1)=020001	(R1)=000001	Remainder

ASH 072RSS

Arithmetic SHift

Operation: R ← R shifted arithmetically NN places to right or left, where NN = low-order 6 bits of source.

Condition Codes:

N: set if result < 0; cleared otherwise.
Z: set if result = 0; cleared otherwise.
V: set if sign of register changed during left shift; cleared otherwise.
C: loaded from last bit shifted out of register.

Description: The contents of the register are shifted right or left the number of times specified by the shift count. The shift count is taken as the low-order 6 bits of the source operand (SSS). This number ranges from −32 to +31. Negative is a right shift and positive is a left shift (Figure 3-3).

Example: ASH R0, R3

Before	**After**
(R3)=000003	(R3)=000003
(R0)=001234	(R0)=012340

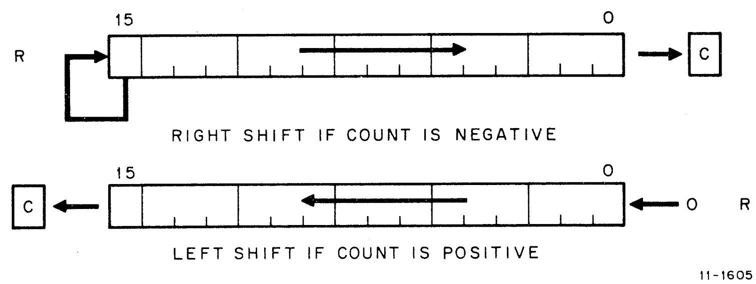


Figure 3-3 ASH Operation

ASHC 073RSS

Arithmetic SHift Combined

Operation: R, RV1 ← R, RV1. The double word is shifted NN places to the right or left, where NN = low-order six bits of source.

Condition Codes:

N: set if result < 0; cleared otherwise.
Z: set if result = 0; cleared otherwise.
V: set if sign bit changes during the left shift; cleared otherwise.
C: loaded with the last bit shifted out of the register.

Description: The contents of the register and the register ORed with 1 are treated as one 32-bit word. RV1 (bits 15:00) and R (bits 31:16) are shifted right or left the number of times specified by the shift count. The shift count is taken as the low-order 6 bits of the source operand. This number ranges from −32 to +31. Negative is a right shift and positive is a left shift (Figure 3-4). When the register chosen is an odd number, the register and the register ORed with 1 are the same. In this case, the right shift becomes a rotate. The 16-bit word is rotated right the number of bits specified by the shift count for up to 16 shifts.

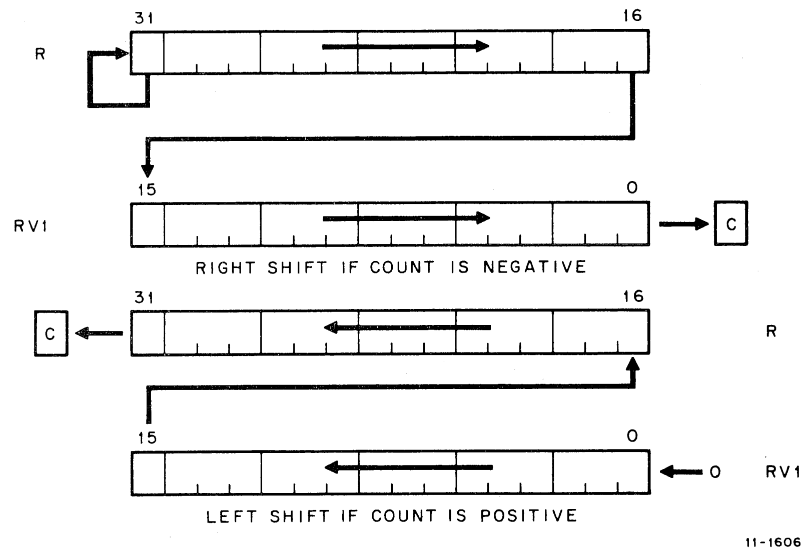


Figure 3-4 ASHC Operation

3.2 KE11-F FLOATING INSTRUCTION SET

There are no addressable registers in the KE11-F Option. FIS operands are fetched from core memory and the result of each operation is stored in core memory. Operands are ordered on the stack in Polish Notation (Paragraph 4.2), thereby reducing the number of operations necessary to achieve a result.

3.2.1 Operation

For Floating ADD, the A argument from the stack is added to the B argument from the stack with the result stored in the A argument position on the stack.

For Floating SUBtract, the B argument from the stack is subtracted from the A argument on the stack with the result stored in the A argument position on the stack.

The Floating MULtiply instruction multiplies the A argument on the stack by the B argument on the stack and stores the result in the A argument position on the stack.

The Floating DIVide instruction divides the A argument on the stack by the B argument on the stack and stores the result in the A argument position on the stack.

3.2.2 Formats

The number format for the KE11-F Option is shown in Figure 3-5. The KE11-F word is 32 bits long with bit 15 of the high argument designating the sign of the fraction. Note that the 8-bit exponent separates the fraction from its associated sign. In floating point, representation of binary numbers is in three parts: a sign bit, an exponent, and a mantissa. The mantissa is a fraction expressed in sign and magnitude format with the binary point positioned to the left of the most significant bit of the mantissa. The mantissa is assumed to be normalized. The MSB of the mantissa is not stored in core because it is redundant. Leading 0s are removed by shifting the mantissa left; however, each left shift of the mantissa must be followed by a decrement of the exponent value to maintain the true value of the number. The exponent value represents the power of 2 by which the mantissa is multiplied to obtain the value to be used.

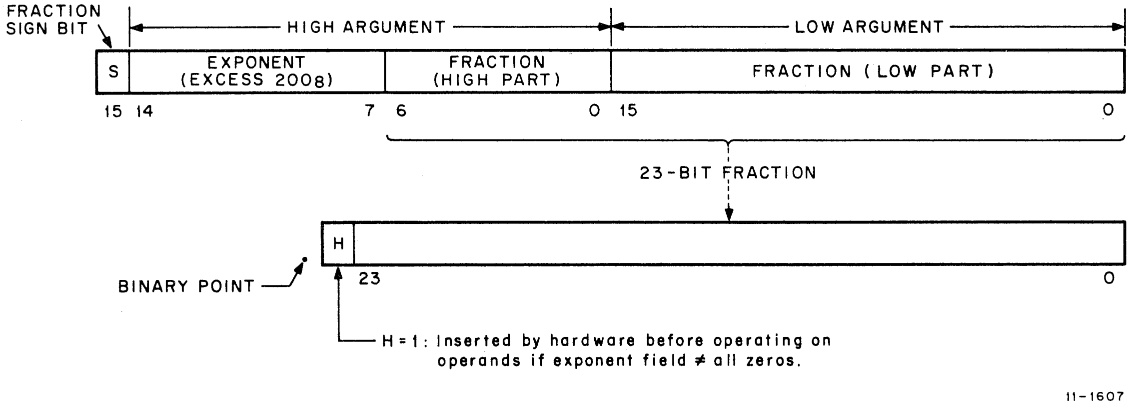


Figure 3-5 FIS Number Format

The KE11-F Option stores the exponent in excess 200_8 (128_{10}) notation. As a result, exponent values from −128 to +127 are represented by the binary equivalent of 0 to 255 (octal 0–377). Mantissas are represented in sign magnitude form.

The binary radix point is to the left. The results of the floating-point operations are always rounded away from 0, increasing the absolute value of the number.

If the exponent is equal to 0, the number is assumed to be 0 regardless of the sign bit or fraction value. The hardware generates a clean 0 (32-bit word of all 0s) in this case.

3.2.3 Instructions

The FIS instruction format is shown in Figure 3-6. It is a double operand instruction in which the low three bits (R,R,R) specify a register that is utilized as a stack pointer for the floating-point operands. The register may be any one of the eight general registers, but some caution must be used if using the PC (R7). It is unlikely that the PC would be desirable as a pointer.

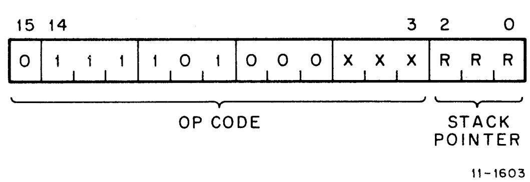


Figure 3-6 FIS Instruction Format

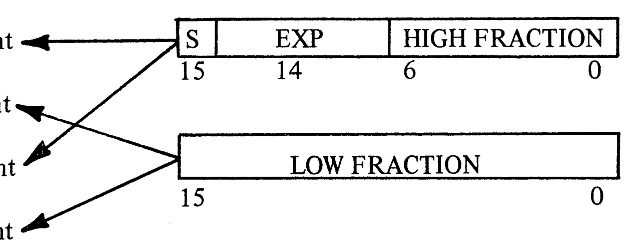
The operands are located on the stack as follows:

(R) = High B Argument

(R)+2 = Low B Argument

(R)+4 = High A Argument

(R)+6 = Low A Argument



The floating-point answers are stored as follows:

(R)+4 = High Answer
(R)+6 = Low Answer

The floating-point stack pointer is repositioned to point to (R)+4 (High Answer).

The floating-point octal coding is in the form 0750XR. There are four FIS instructions, as follows:

FADD 07500R

Floating-ADD

Operation:	[(R) +4 □ (R) +6] ← [(R) +4 □ (R) +6] + [(R) □ (R) +2], if result $\geqslant 2^{-128}$; else [(R) +4 □ (R) +6] ← 0
Condition Codes: **(See Note Below)**	N: set if result < 0; cleared otherwise. Z: set if result = 0; cleared otherwise. V: cleared C: cleared
Description:	Adds the B argument to the A argument and stores the result in the A argument position on the stack. A ← A+B

FSUB 07501R

Floating-SUBtract

Operation:	[(R) +4 □ (R) +6] ← [(R) +4 □ (R) +6] - [(R) □ (R) +2], if result $\geqslant 2^{-128}$; else [(R) +4 □ (R) +6] ←0
Condition Codes: **(See Note Below)**	N: set if result < 0; cleared otherwise. Z: set if result = 0; cleared otherwise. V: cleared C: cleared
Description:	Subtracts the B argument from the A argument and stores the result in the A argument position on the stack. A ← A−B

FMUL 07502R

Floating-MULtiply

Operation:	[(R) +4 □ (R) +6] ← [(R) +4 □ (R) +6] * [(R) □ (R) +2], if result $\geqslant 2^{-128}$; else [(R) +4, (R) +6]
Condition Codes: **(See Note Below)**	N: set if result < 0; cleared otherwise. Z: set if result = 0; cleared otherwise. V: cleared C: cleared

Description: Multiplies the B argument by the A argument and stores the result in the A argument position on the stack. A ← A*B. If the result is $< 2^{-128}$, then underflow occurs and the destination address will contain the A argument.

FDIV 07503R

Floating-DIVide

Operation: [(R) +4 □ (R) +6] ← [(R) +4 □ (R) +6] / [(R) □ (R) +2], if result $\geqslant 2^{-128}$;
else [(R) +4 □ (R) +6]

Condition Codes:
(See Note Below)

N: set if result <0; cleared otherwise.
Z: set if result = 0; cleared otherwise.
V: cleared
C: cleared

Description: Divides the A argument by the B argument and stores the result in the A argument position on the stack. If the B argument (divisor) is equal to 0, the stack is left untouched. A ← A/B. If the result is $< 2^{-128}$, then the destination address will contain the A argument.

NOTE

If a trap occurs as a function of a floating instruction, the condition codes are reinterpreted as follows:

N: set if underflow, cleared if overflow.
Z: cleared
V: set if underflow, overflow, divide by 0 (error conditions).
C: set if divide by 0, otherwise cleared.

Traps occur through the vector 244. (R) is reset to point to high B argument on the stack. The arguments are left untouched.

3.2.4 Programming Example

A sample floating-point program is given below.

```

        000000'         .CSECT
                        .TITLE FISEXM
                ;
                ;       COPYRIGHT 1972 BY DIGITAL EQUIPMENT CORPORATION,
                ;               MAYNARD, MASSACHUSETTS,
                ;
                ;
                ;    EXAMPLE OF PDP-11/40 FLOATING INSTRUCTION SET USAGE (FIS)
                ;
                ;    COMPUTE LARGER ROOT OF QUADRATIC EQUATION:
                ;
                ;       A*X*X + B*X + C = 0
                ;
                ;    ALGORITHM IS:
                ;
                ;       ROOT1 = (-B + SQRT(B*B - 4*A*C))/(2*A)
                ;
```

```
                    ;       INITIAL VALUES OF A, B, AND C ARE
                    ;           PLACED IN MEMORY LOCATIONS A, B, AND C.
                    ;           RESULT IS COMPUTED AND STORED AT ROOT1.
                    ;
                    ;           NORMAL TERMINATION IS A HALT AT LOCATION DONE.
                    ;           IF DISCRIMINANT IS NEGATIVE THEN HALT AT LOCATION
                    ;           IMAG. HALT AT AZERO IF A = 0.
                    ;
                    ;       NORMAL REGISTER DECLARATIONS:
             000000 R0      =%0
             000001 R1      =%1
             000002 R2      =%2
             000003 R3      =%3
             000004 R4      =%4
             000005 R5      =%5
             000006 SP      =%6
             000007 PC      =%7
                    ;
                    ;       PROGRAM STARTS HERE
                    ;
 00000 012706 START:  MOV     #STACK,SP       ;INITIALIZE PROCESSOR STACK
       000442'
 00004 016746         MOV     B+2,-(SP)       ;B TO STACK
       000204
 00010 016746         MOV     B,-(SP)
       000176
 00014 016746         MOV     B+2,-(SP)       ;AGAIN
       000174
 00020 016746         MOV     B,-(SP)
       000166
 00024 075026         FMUL    SP              ;FORM B*B
 00026 005046         CLR     -(SP)           ;4.0 TO STACK
 00030 012746         MOV     #^F4.0,-(SP)
       040600
 00034 016746         MOV     A+2,-(SP)       ;A TO STACK
       000150
 00040 016746         MOV     A,-(SP)
       000142
 00044 001457         BEQ     AZERO           ;HALT IF A = 0.
 00046 016746         MOV     C+2,-(SP)       ;C TO STACK
       000146
 00052 016746         MOV     C,-(SP)
       000140
 00056 075026         FMUL    SP              ;FORM A*C
 00060 075026         FMUL    SP              ;FORM 4.*A*C
 00062 075016         FSUB    SP              ;FORM B*B-4.*A*C (DISCRIMINANT)
 00064 100446         BMI     IMAG            ;BRANCH IF NEGATIVE
 00066 012667         MOV     (SP)+,TEMP1     ;STORE DISCRIMINANT
       000130
 00072 012667         MOV     (SP)+,TEMP1+2
       000126
 00076 004567         JSR     R5,SQRT         ;CALL FORTRAN SQUARE ROOT ROUTINE
       000000G
 00102 000401         BR      .+4
 00104 000222         .WORD   TEMP1
 00106 010067         MOV     R0,TEMP2        ;STORE RESULT
       000114
 00112 010167         MOV     R1,TEMP2+2
       000112
                    ;COMPUTE ROOT1
 00116 016746         MOV     B+2,-(SP)       ;B TO STACK
       000072
 00122 016746         MOV     B,-(SP)
       000064
 00126 062716         ADD     #100000,@SP     ;NEGATE B ON STACK
       100000
```

```
00132 016746          MOV     TEMP2+2,-(SP)  ;SQUARE ROOT TO STACK
      000072
00136 016746          MOV     TEMP2,-(SP)
      000064
00142 075006          FADD    SP             ;FORM -B+SQRT
00144 016746          MOV     CONST+2,-(SP)  ;2.0 TO STACK
      000064
00150 016746          MOV     CONST,-(SP)
      000056
00154 016746          MOV     A+2,-(SP)      ;A TO STACK
      000030
00160 016746          MOV     A,-(SP)
      000022
00164 075026          FMUL    SP             ;FORM 2.*A
00166 075036          FDIV    SP             ;FORM (-B+SQRT)/(2.*A)
00170 012667          MOV     (SP)+,ROOT1    ;SAVE RESULT
      000042
00174 012667          MOV     (SP)+,ROOT1+2
      000040
00200 000000 DONE:    HALT
00202 000000 IMAG:    HALT
00204 000000 AZERO:   HALT
             ;
             ;
00206        A:       .BLKW   2
00212        B:       .BLKW   2
00216        C:       .BLKW   2
00222        TEMP1:   .BLKW   2
00226        TEMP2:   .BLKW   2
00232 040400 CONST:   .FLT2   2.0
00234 000000
00236        ROOT1:   .BLKW   2
                      .GLOBL  SQRT           ;EXTERNAL SUBROUTINE
                      .BLKW   100            ;ROOM FOR STACK
00442        STACK:   .BLKW   1              ;START OF STACK IS TOP OF AREA
      000001'         .END
```

APPENDIX A
GLOSSARY OF TERMS

Table A-1 contains a collection of some of the terms used in this manual that may need defining. It does not include all terms, only those that it is thought might be confusing. Listing is in alphabetical order.

Table A-1
Glossary of Terms

Term	Definition
ADD	Add (instruction)
ADR	Address
ALU	Arithmetic Logic Unit
ALUM	Arithmetic Logic Unit Mode
ARGA	Argument A (f/f)
ASH	Arithmetic shift (instruction)
ASHC	Arithmetic shift combined (instruction)
BBSY	Bus busy
BRQ	Bus request
BUS	Unibus
BUS U	Bus microprogram
BUSY	Busy
BUT	Branch microprogram test
CIN	Carry-in (ALU)
CLK	Clock
CLKB	Clock B Register
CLKBA	Clock BA Register
CLKD	Clock D Register
CLKOFF	Clock off
CLR	Clear C,V,N,Z (instruction)
CON	Constant
COUT MUX	Carry-out multiplexer (ALU)
C1 BUS	C1 of Unibus
DAD	Discrete alteration of data
DEST	Destination
DIV	Divide (instruction)
DMUX	Data multiplexer
EINSTR	Extended Instruction
EIS	Extended arithmetic instruction set

Table A-1 (Cont)
Glossary of Terms

Term	Definition
EPS	Extended Processor Status
EUB	Extended microprogram bus
EUPP	Extended microprogram pointer
EXP	Exponent
f	Function of
FADD	Floating add (instruction)
FC1BUS	Floating C1 Bus
FDIV	Floating divide (instruction)
FETCH	Fetch (Processor State)
FINSTR	Floating Instruction
FIS	Floating instruction set
FMUL	Floating multiply (instruction)
FSUB	Floating subtract (instruction)
FUB	Floating microprogram bus
IR	Instruction register
ISP	Instruction set processor
JAMUPP	Jam microprogram pointer
MUL	Multiply (instruction)
MUX	Multiplexer
NO-OP	No operation
OVFL	Overflow
PC	Program Counter
PS	Processor Status Register
R(x)	Scratch Pad Register
RSVD INSTR	Reserved instruction
SALU	Select arithmetic logic unit
SALUM	Select arithmetic logic unit mode
SBC	Select B constant
SERVICE	Service
SET COND CODES	Set condition codes
SF	Source field
SFV1	Source field ORed with 1
SRC	Source (processor major state)
STPM	Special Trap Pointer Marker
TRAP	User call
U	Microprogram
UBF	Microprogram branch field
UNFL	Underflow
UPP	Microprogram pointer
U WORD	Microprogram word
VECT	Vector
XOR	Exclusive OR (V)
ZB	"Z" bit previous state (flip-flop)

KE11-E and KE11-F INSTRUCTION SET OPTIONS MANUAL
EK-KE11E-TM-002

Reader's Comments

Your comments and suggestions will help us in our continuous effort to improve the quality and usefulness of our publications.

What is your general reaction to this manual? In your judgment is it complete, accurate, well organized, well written, etc.? Is it easy to use? ______

What features are most useful? ______

What faults do you find with the manual? ______

Does this manual satisfy the need you think it was intended to satisfy? ______

Does it satisfy *your* needs? ______ Why? ______

Would you please indicate any factual errors you have found. ______

Please describe your position. ______

Name ______ Organization ______

Street ______ Department ______

City ______ State ______ Zip or Country ______

CUT OUT ON DOTTED LINE

Fold Here

Do Not Tear - Fold Here and Staple

FIRST CLASS
PERMIT NO. 33
MAYNARD, MASS.

BUSINESS REPLY MAIL
NO POSTAGE STAMP NECESSARY IF MAILED IN THE UNITED STATES

Postage will be paid by:

Digital Equipment Corporation
Technical Documentation Department
146 Main Street
Maynard, Massachusetts 01754

digital equipment corporation

Printed in U.S.A.