SCIENCE RESEARCH COUNCIL

RUTHERFORD LABORATORY
ATLAS COMPUTING DIVISION

FR80 TECHNICAL PAPER 27

III-15 DIVS Hardware Error

issued by
D C Sutcliffe
& R W Witty

9 December 1977

DISTRIBUTION:

F R A Hopgood
L O Ford
A H Francis
D C Sutcliffe
D Green (III)

R W Witty
D A Duce
R E Thomas
R Brandwood
R Haddow (III)

1. INTRODUCTION

There appears to be a design fault in the DIVS instruction hardware of the FR80's III-15 computer. DIVS is the signed division of a 36 bit number by an 18 bit number. DIVS on the III-15 does not always produce the same results as DIVS on ACD's PDP-15 computer in the case of divide overflow.

2. DIVS (PDP-15) v DIVS (III-15)

Figure 2 is the PDP-15 Reference Manual description of DIVS. The dividend is 34 bits and 2 sign bits. This is understandable if one thinks of it as the result of multiplying two signed 17 bit numbers together. The underlined sentence defines the condition for divide overflow. This occurs when the quotient is too big to be represented as a signed 17-bit number. The underlined section of the PDP-15 manual describes the overflow condition two ways. "Magnitude of the quotient exceeds 17 bit plus sign capacity of the MQ" is correct but it is not the same thing as saying "the divisor is not greater than the AC portion of the dividend".

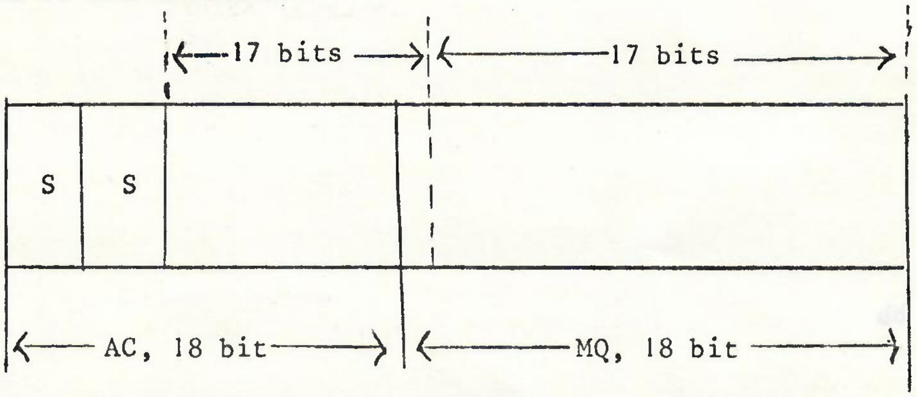


Figure 1 Layout of Dividend

This is because the AC portion of the dividend only includes the top 16 bits of the most significant half of the dividend; the 17th bit is lurking in the top bit of the MQ register, see Figure 1. The PDP-15 hardware correctly uses the bottom 16 bits of the AC and the top bit of the MQ when testing against the divisor for divide overflow. The error is just in the manual (see Figure 4).

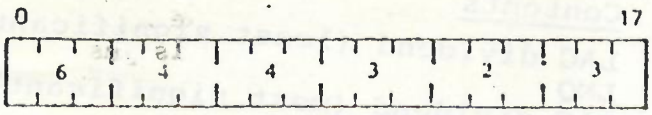
Figure 3 is the III-15 reference manual's description of DIVS. Note that only the erroneous description of the test for divide overflow has been cribbed from the PDP-15 manual. Figure 4 is a III-15 test program to see if the error was in the III-15 manual or in the hardware. Figure 4 shows the error to be in both as the III-15 is only testing the contents of the AC for divide overflow. The centre test run in Figure 4 should have generated divide overflow (LINK=77) but did not on the III-15. Divide overflow was generated correctly by the PDP-15 (last line of the test program reslts).

3. CONCLUSION

There is a serious design fault in the III-15 DIVS instruction. DIVS does not always generate divide overflow when the quotient arithmetically exceeds the single length result capacity of a sign plus 17 bit number, ie it gives wrong answers sometimes!

bb

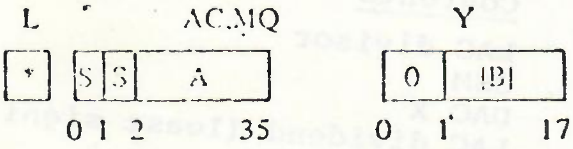
DIVS Signed Division



Execute Time: 7.65 μs

Divide the contents of the AC and MQ (a 36-bit signed dividend with the sign in AC_0 and bits 0 and 01, and the remaining 34 bits devoted to magnitude) by the contents of memory location Y (the divisor). The resulting quotient appears in the MQ with the algebraically determined sign in MQ bit 0 and the magnitude (1's complement) in MQ bits 1 through 17. The remainder is in the AC with AC bit 0 containing the magnitude (1's complement). The address of Y is taken to be sequential to the address of the DIVS instruction word. The contents of Y are taken to be the absolute value of the divisor: the contents of the Link are taken to be the orignal sign of the divisor (DIVS assumes previous execution of an EAE GSM instruction). Prior to this DIVS instruction, the dividend must be entered in the AC and MQ (LAC of least significant half, LMQ, and LAC of most significant half). The MQ portion of a negative dividend is 1's complemented prior to the division. If the divisor is not greater than the AC portion of the dividend, divide overflow occurs (magnitude of the quotient exceeds the 17 bit plus sign capacity of the MQ), and the Link is set to one to signal the overflow condition: data in the AC and the MQ are of no value. A valid division halts when the step counter, initialized to the 2's complement of 23_8 (19_{10} steps), counts up to 0 (the six low-order bits of the DIVS instruction word specify the step count). The content of the Link is cleared to 0. The contents of Y are unchanged. The program resumes at the next instruction (memory location Y + 1).

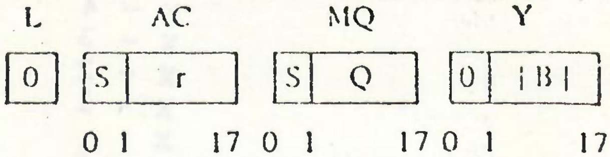
Pre-execution



*original sign of B

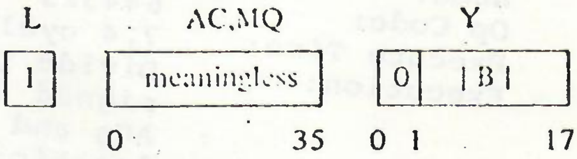
Post-execution

(no overflow)



(S=Sign A) (S=Sign A ⊻ L)

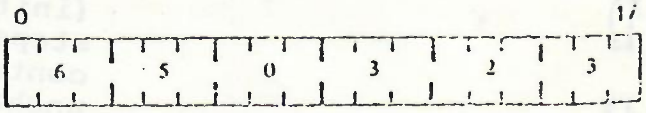
(overflow)



Instruction Sequence:

Memory Location	Contents
Y - 7	LAC Divisor
Y - 6	GSM
Y - 5	DAC Divisor in Y
Y - 4	LAC Dividend (least significant half)
Y - 3	LMQ
Y - 2	LAC Dividend (most significant half)
Y - 1	DIVS
Y	Divisor (absolute value)
Y + 1	Next Instruction

FRDIV Fraction Divide Unsigned



Execute Time 7.65 μs

Divide the contents of the AC and the MQ (A contains an 18-bit fractional dividend. MQ = zeroed at setup) by the contents of mem location Y (the divisor). The binary point assumed at the left of AC (bit 0). The

Figure 2. PDP-15

Instruction Sequence:

Location	Contents
X - 4	LAC dividend (least significant)
X - 3	LMQ
X - 2	LAC dividend (most significant)
X - 1	DIV
X	Divisor
X + 1	Next instruction

Mnemonic: DIVS

Name: Divide, Signed

Op Code: 644323

Execute Time: 7.4 cycles

Execution: Divide the contents of the AC and MQ (a signed 36-bit dividend with the sign in AC_0 and AC_1) by the contents of memory location X (the absolute divisor). The quotient appears in the MQ with the algebraic sign in MQ_0. The remainder is in the AC, with AC_0 containing the sign of the dividend and AC_{1-17} containing the magnitude in 1's complement notation. The address of X (the absolute divisor) must be sequential to the DIVS instruction address.

Prior to the execution of the DIVS instruction, the absolute value of the divisor must be stored in X and the link must contain the sign of the divisor. This is accomplished by loading the AC with the signed divisor, executing a GSM instruction, and then storing the AC in X. The dividend must then be entered into the AC and MQ. If the divisor is not greater than the most significant half of the dividend (the AC), divide overflow occurs; the link is set to 1, and the divide operation terminates with the resulting data, contained in the AC and MQ, being of no value.

A valid division terminates when the SC (initialized to the 2's complement of 23_8) steps to zero. The link is cleared. The contents of X (the absolute divisor) remain unchanged. The program resumes execution at address X + 1.

Instruction Sequence:

Location	Contents
X - 7	LAC divisor
X - 6	GSM
X - 5	DAC X
X - 4	LAC dividend (least significant)
X - 3	LMQ
X - 2	LAC dividend (most significant)
X - 1	DIVS
X	Absolute divisor
X + 1	Next instruction

Figure 3. III - 15

Figure F.

```
/TEST
BEGIN,
        LAC (000001)
        GSM
        DAC X
        LAC (400000)
        LMQ
        LAC (000001)
        DIVS
X,      0
        DAC Y
        LACQ
        DAC Z
        SNL
        DZM LINK
        JMP I (037777)
Y,      0
Z,      0
LINK,   77
START BEGIN
```

III-15 and PDP-15 source prog. ⟷

```
/TEST
BEGIN,
        LAC (000001)
        GSM
        DAC X
        LAC (400000)
        LMQ
        LAC (000000)
        DIVS
X,      0
        DAC Y
        LACQ
        DAC Z
        SNL
        DZM LINK
        JMP I (037777)
Y,      0
Z,      0
LINK,   77
START BEGIN
```

```
/TEST
BEGIN,
        LAC (000001)
        GSM
        DAC X
        LAC (200000)
        LMQ
        LAC (000000)
        DIVS
X,      0
        DAC Y
        LACQ
        DAC Z
        SNL
        DZM LINK
        JMP I (037777)
Y,      0
Z,      0
LINK,   77
START BEGIN
```

```
FA

/TEST
/TEST

'

Y/      1
Z,      1
LINK,   77
```

III results ⟷

```
FA

/TEST
/TEST

'

Y/      0
Z,      400000
LINK,   0
```

```
FA

/TEST
/TEST

'

Y/      0
Z,      200000
LINK,   0
```

```
>Y/     LAW -2  Z/      3       LINK/   77  >Y/     0       Z/      LAC     LINK/   0
```

⟵ PDP-15 Results ⟶