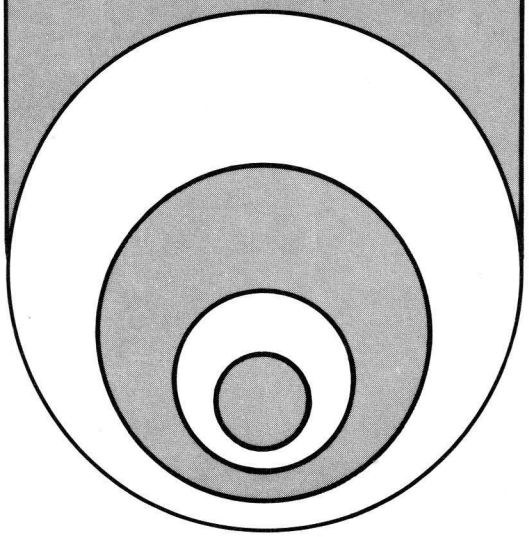
signetics

MOS
MICROPROCESSOR



BINARY ARITHMETIC ROUTINES.....AS53

INTRODUCTION

Binary arithmetic routines, like addition, subtraction, multiplication, and division, are often used in microprocessor-based systems. This applications memo provides several suggested examples for processing binary arithmetic routines on the 2650 microprocessor. These examples include:

- SIGNED BINARY ADDITION/SUBTRACTION
  Two-byte operands giving a two-byte result.
- UNSIGNED BINARY MULTIPLICATION
  One-byte operands giving a two-byte result.
  Two-byte operands giving a four-byte result.
- SIGNED BINARY MULTIPLICATION
  One-byte operands giving a two-byte result.
  Two-byte operands giving a four-byte result.
- BINARY DIVISION – UNSIGNED AND SIGNED
  Two-byte dividend and quotient with one-byte divisor and remainder.

In these examples, emphasis is placed on minimizing program memory requirements rather than on processing speed. The different branch instructions and the indexing features of the Signetics 2650 proved useful in minimizing memory requirements.

	HARDWARE AFFECTED						
REGISTERS	R0 X	R1 X	R2	R3	R1'	R2'	R3'
PSU	F	II	SP				
PSL	CC X	IDC X	RS	WC X	OVF X	COM	C X

RAM REQUIRED (BYTES): 6

ROM REQUIRED (BYTES): 45

EXECUTION TIME: Variable

MAXIMUM SUBROUTINE NESTING LEVELS: None

ASSEMBLER/COMPILER USED: PIPHASM

1. BINARY ADDITION/SUBTRACTION FOR TWO-BYTE SIGNED INTEGERS

FUNCTION:

Performs the addition or subtraction of two 2-byte signed integers giving a two-byte result.
(OPR1, OPR1 + 1) +/– (OPR2, OPR2 + 1) ⟶
RSLT, RSLT + 1

PARAMETERS:

Input: OPR1, OPR1 + 1 contains augend/subtrahend
OPR2, OPR2 + 1 contains addend/minuend
COM-flag in PSL indicates addition/subtraction:
COM = 0 addition
COM = 1 subtraction

Output: RSLT, RSLT + 1 contains sum/difference.
The condition code CC is set to the proper value of the two byte result.
OPR1, OPR2 and RSLT are MS-bytes.

SPECIAL REQUIREMENTS

None

Refer to Figures 1.1 and 1.2 for flowchart and program listing.

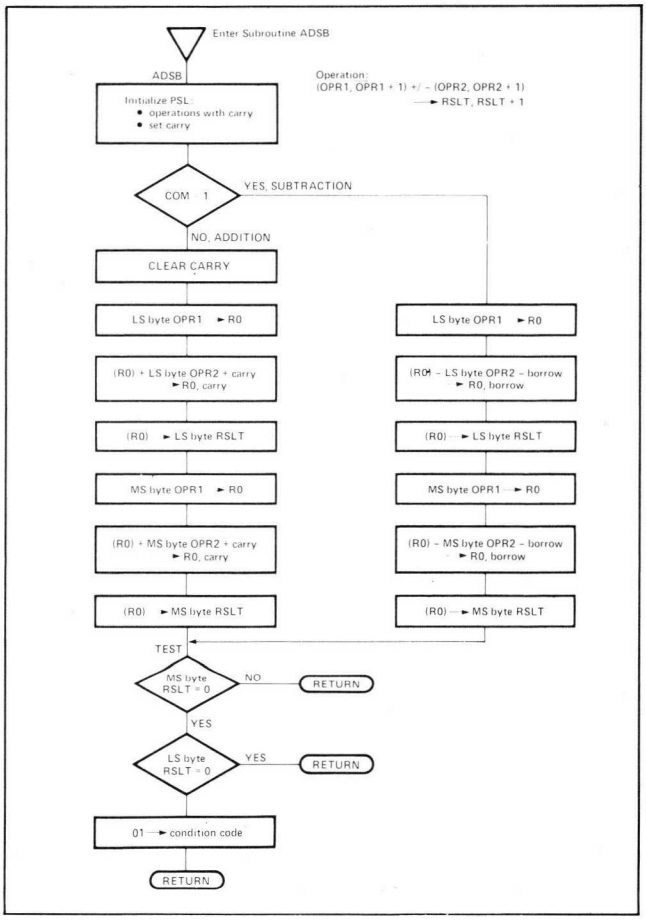


FIGURE 1-1 Flowchart for Double Precision Addition/Subtraction

```
1                                       * PD760010
2                                       ************************************************************
3                                       * BINARY DOUBLE PRECISION ADDITION/SUBTRACTION
4                                       ************************************************************
5                                       * OPERATION:
6                                       *    (OPR1,OPR1+1)+/-(OPR2,OPR2+1)-->RSLT,RSLT+1
7                                       * OPR1,OPR2,RSLT ARE MOST SIG BYTES
8                                       * COM IN PSL IS USED AS ADD/SUB FLAG
9                                       *    COM=0 IS ADD; COM=1 IS SUBTRACT
10                                      * AFTER ADD/SUB THE CC,OVF,AND C BITS IN PSL
11                                      *    ARE VALID FOR THE RESULT
12                                      *
13                                      * DEFINITION OF SYMBOLS
14                                      *
15      0000                            R0   EQU     0          PROCESSOR REGISTERS
16      0001                            R1   EQU     1
17      0002                            R2   EQU     2
18      0003                            R3   EQU     3
19      0080                            CC1  EQU     H'80'      PSL: MSB OF CONDITION CODE
20      0040                            CC0  EQU     H'40'           LSB OF CONDITION CODE
21      0008                            WC   EQU     H'08'           1=WITH,0=WITHOUT CARRY
22      0002                            COM  EQU     H'02'           1=LOGICAL,0=ARITH COMP
23      0001                            C    EQU     H'01'           CARRY/BORROW
24      0000                            Z    EQU     0          BRANCH COND: ZERO
25      0003                            UN   EQU     3                       UNCONDITIONAL
26      0000                            ON   EQU     0                       ALL BITS ARE 1
27                                      *
28                                           ORG     H'500'     START OF SUBROUTINE
29                                      *
30  0500  0500  77 09                  ADSB PPSL    WC+C       ARITH WITH CARRY;SET CARRY
31  0502        05 02                       LODI,R1 2          LOAD INDEX REGISTER
32  0504        B5 02                       TPSL    COM
33  0506        18 0F                       BCTR,ON LPSB       BRANCH IF SUBTRACTION
34  0508        75 01                       CPSL    C          ADDITION,CLEAR CARRY
35  050A  050A  0D 45 2D               LPAD LODA,R0 OPR1,R1,-  BYTE OF FIRST OPERAND TO R0
36  050D        8D 65 2F                    ADDA,R0 OPR2,R1    ADD BYTE OF SECOND OPERAND
37  0510        CD 65 31                    STRA,R0 RSLT,R1    STORE RESULT
38  0513        59 75                       BRNR,R1 LPAD       BRANCH IF NOT DONE
39  0515        1B 0B                       BCTR,UN TEST
40  0517  0517  0D 45 2D               LPSB LODA,R0 OPR1,R1,-  BYTE OF FIRST OPERAND TO R0
41  051A        AD 65 2F                    SUBA,R0 OPR2,R1    SUB BYTE OF SECOND OPERAND
42  051D        CD 65 31                    STRA,R0 RSLT,R1    STORE RESULT
43  0520        59 75                       BRNR,R1 LPSB       BRANCH IF NOT DONE
44  0522  0522  98 08                  TEST BCFR,Z  RTRN       RETURN IF MS BYTE NOT ZERO
45  0524        0C 05 32                    LODA,R0 RSLT+1
46  0527        14                          RETC,Z             RETURN IF LS BYTE ALSO ZER0
47  0528        75 80                       CPSL    CC1        SET CC. TO 01 (POSITIVE)
48  052A        77 40                       PPSL    CC0
49  052C  052C  17                     RTRN RETC,UN
50                                      *
51      052D                            OPR1 RES     2          LOCATION OF: FIRST  OPERAND
52      052F                            OPR2 RES     2                       SECOND OPERAND
53      0531                            RSLT RES     2                       RESULT
54                                           END
```

FIGURE 1-2

2. BINARY MULTIPLICATION FOR ONE-BYTE UNSIGNED INTEGERS

FUNCTION:

One byte by one byte multiplication for unsigned integers, giving a two-byte result.
(OPR1) X (OPR2) ⟶ RSLT, RSLT + 1

PARAMETERS:

Input OPR1 contains multiplier
OPR2 contains multiplicand

Output: RSLT contains high-order product-byte.
RSLT + 1 contains low-order product-byte.

SPECIAL REQUIREMENTS:

None

Refer to Figures 2.1 and 2.2 for flowchart and program listing.

	HARDWARE AFFECTED						
REGISTERS	R0	R1	R2	R3	R1'	R2'	R3'
	X	X	X	X			
PSU	F	II	SP				
PSL	CC	IDC	RS	WC	OVF	COM	C
	X	X		X	X		X

RAM REQUIRED (BYTES): 4

ROM REQUIRED (BYTES): 29

EXECUTION TIME: Variable

MAXIMUM SUBROUTINE NESTING LEVELS: None

ASSEMBLER/COMPILER USED: PIPHASM

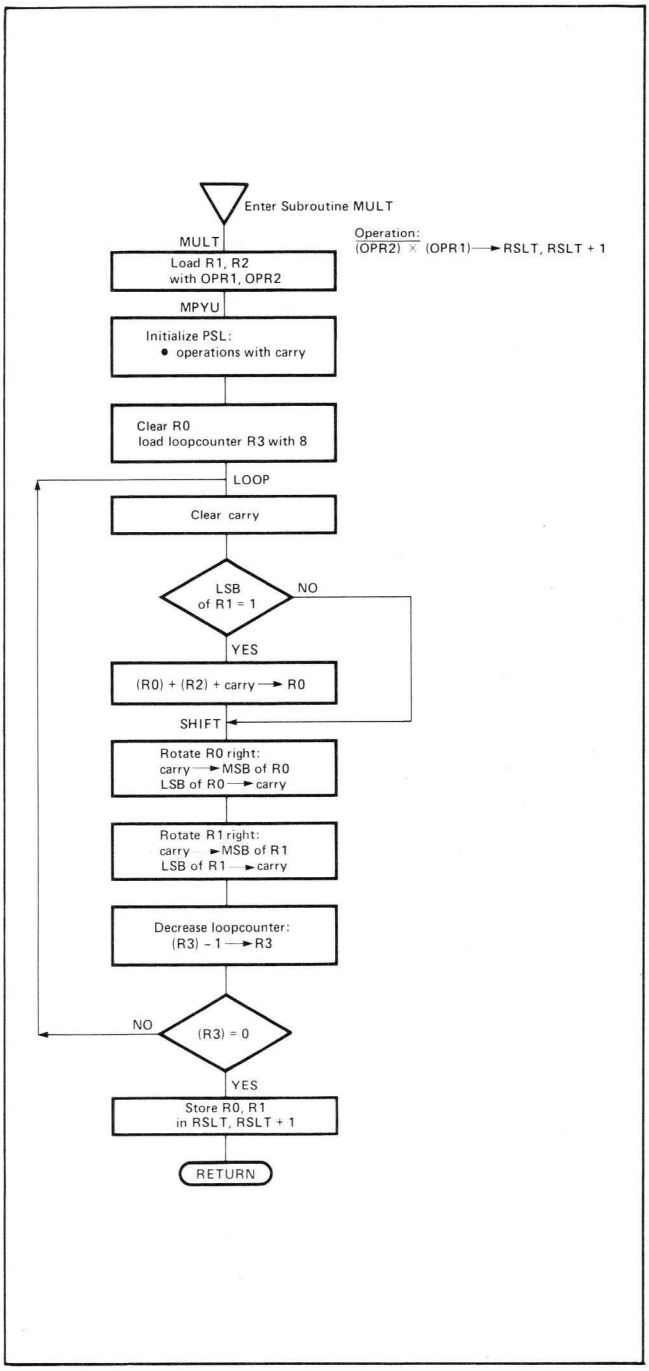


FIGURE 2-1 Flowchart for Unsigned Multiplication (One-Byte Operands; Two-Byte Result)

```
1                                       *       PD76ØØ3Ø                                        *
2                                       ********************************************************************
3                                       *     BINARY MULTIPLICATION FOR 2 UNSIGNED INTEGERS
4                                       ********************************************************************
5                                       *
6                                       *     MULTIPLIER IS IN OPR1
7                                       *     MULTIPLICAND IS IN OPR2
8                                       *     RESULT WILL BE STORED IN RSLT,RSLT+1 (RSLT = MS BYTE)
9                                       *
10                                      *
11                                      *
12                                      *
13                                      *           SYMBOL DEFINITIONS
14          ØØØØ                        R0        EQU       Ø
15          ØØØ1                        R1        EQU       1
16          ØØØ2                        R2        EQU       2
17          ØØØ3                        R3        EQU       3
18          ØØØ1                        R4        EQU       1
19          ØØØ2                        R5        EQU       2
20          ØØØ3                        R6        EQU       3
21          ØØØ3                        UN        EQU       3             UNCONDITIONAL BRANCHING
22          ØØØØ                        ON        EQU       Ø
23          ØØØ2                        LT        EQU       2
24          ØØØØ                        Z         EQU       Ø
25          ØØØ1                        P         EQU       1
26          ØØØ2                        N         EQU       2
27          ØØØ8                        WC        EQU       8
28          ØØØ1                        C         EQU       1
29          ØØ4Ø                        F         EQU       H'4Ø'
30          ØØØ4                        OVF       EQU       4
31          ØØØ2                        COM       EQU       2
32                                      *
33                                      * R/W MEMORY
34                                      *
35                                            ORG   H'5ØØ'
36          Ø5ØØ                        OPR1      RES       2
37          Ø5Ø2                        OPR2      RES       2
38          Ø5Ø4                        RSLT      RES       4
39                                      *
40                                      *
41                                      *
42                                              ORG  H'6ØØ'
43  Ø6ØØ    Ø6ØØ  ØD Ø5 ØØ              MULT      LODA,R1   OPR1          GET OPERAND IN R1
44  Ø6Ø3          ØE Ø5 Ø2                        LODA,R2   OPR2          GET OPERAND IN R2
45  Ø6Ø6    Ø6Ø6  77 Ø8                 MPYU      PPSL      WC            ARITH
46  Ø6Ø8          2Ø                              EORZ      RØ            CLEAR RØ
47  Ø6Ø9          Ø7 Ø8                           LODI,R3   8             LOAD LOOP COUNTER R3
48  Ø6ØB    Ø6ØB  75 Ø1                 LOOP      CPSL      C             CLEAR CARRY
49  Ø6ØD          F5 Ø1                           TMI,R1    H'Ø1'
50  Ø6ØF          98 Ø1                           BCFR,ON   SHFT          SKIP ADDITION IF LSB R1=Ø
51  Ø611          82                              ADDZ      R2            ADD MULTIPLICAND TO PARTIAL PROD
52  Ø612    Ø612  5Ø                   SHFT      RRR,RØ                  ROTATE PARTIAL PROD AND MULTIPLIER
53  Ø613          51                              RRR,R1
54  Ø614          FB 75                           BDRR,R3   LOOP          BRANCH TO LOOP IF NOT READY
55  Ø616          CC Ø5 Ø4                        STRA,RØ   RSLT          SAVE RESULT IN RESULT AREA
56  Ø619          CD Ø5 Ø4                        STRA,R1   RSLT+1        SAVE RESULT IN RESULT AREA
57  Ø61C          17                              RETC,UN                 RETURN TO MAIN PROGRAM
```

FIGURE 2-2

3. BINARY MULTIPLICATION FOR TWO-BYTE UNSIGNED INTEGERS

FUNCTION:

Two byte by two byte multiplication for unsigned integers, giving a four byte result.

(OPR2, OPR2 + 1) X (OPR1, OPR1 + 1) ⟶
RSLT, RSLT + 1, RSLT + 2, RSLT + 3

PARAMETERS:

Input: (OPR1, OPR1 + 1) contains multiplier
(OPR2, OPR2 + 1) contains multiplicand

Output: RSLT, RSLT + 1, RSLT + 2, RSLT + 3 contains product.
OPR1, OPR2, and RSLT are most-significant bytes.

SPECIAL REQUIREMENTS:

None

Refer to Figures 3.1 and 3.2 for flowchart and program listing.

	HARDWARE AFFECTED						
REGISTERS	R0 X	R1 X	R2	R3 X	R1'	R2'	R3'
PSU	F	II	SP				
PSL	CC X	IDC X	RS	WC X	OVF X	COM	C X

RAM REQUIRED (BYTES): 8

ROM REQUIRED (BYTES): 57

EXECUTION TIME: Variable

MAXIMUM SUBROUTINE NESTING LEVELS: None

ASSEMBLER/COMPILER USED: PIPHASM

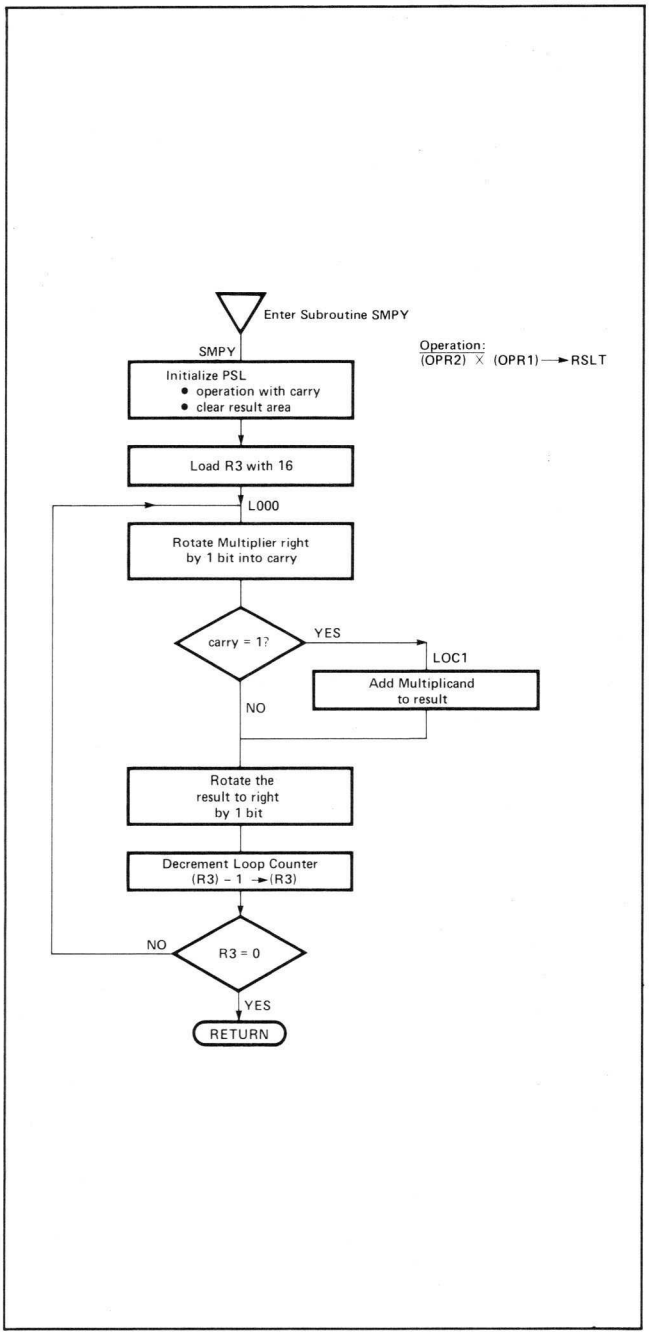


FIGURE 3-1 Flowchart for Unsigned Multiplication (Two-Byte Operands; Four-Byte Result)

```
                          *    PD760031                                                  *
                          ***********************************************************************
                          *  BINARY MULTIPLICATION FOR 2 TWO-BYTE INTEGERS
                          ***********************************************************************
                          *
                          * MULTIPLIER IS IN OPR1 , OPR1+1
                          * MULTIPLICAND IS IN OPR2 ,OPR2+1
                          * RESULT WILL BE IN RSLT ,RSLT+1 ,RSLT+2 ,RSLT+3
                               ORG    H'790'
                          *
0790  0790  77 08         SMPY      PPSL      WC              SET MODE
0792        20                      EORZ      R0
0793        CC 05 04                STRA,R0   RSLT            CLEAR  RESULT
0796        CC 05 05                STRA,R0   RSLT+1          CLEAR RESULT +1
0799        07 10                   LODI,R3   16              LOAD COUNT
079B  079B  05 FE         LOOO      LODI,R1   -2              TO GET 254
079D        75 01                   CPSL      C               CLEAR CARRY
079F  079F  0D 64 02      LOCO      LODA,R0   OPR1-256+2,R1   FOR INDEXING INTO OPR1
07A2        50                      RRR,R0                    ROTATE RIGHT WITH C
07A3        CD 64 02                STRA,R0   OPR1-256+2,R1
07A6        D9 77                   BIRR,R1   LOCO            ROTATE 2ND TIME
                          *                    THIS ROTATES MULTIPLIER BY 1 BIT TO GET THE LSB
                          *                    INTO CARRY
07A8        20                      EORZ      R0              CLEAR R0
07A9        D0                      RRL,R0                    GET CARRY INTO LSB
07AA        F8 02                   BDRR,R0   LOC1
07AC        1B 0D                   BCTR,UN   LOC4
                          *
07AE  07AE  05 02         LOC1      LODI,R1   2               GET INDEX
07B0  07B0  0D 65 03      LOC2      LODA,R0   RSLT-1,R1       ADD MULTIPLICAND TO PRODUCT
07B3        8D 65 01                ADDA,R0   OPR2-1,R1
07B6        CD 65 03                STRA,R0   RSLT-1,R1
07B9        F9 75                   BDRR,R1   LOC2            FINISH THE ADD
                          *
                          *
07BB  07BB  05 FC         LOC4      LODI,R1   -4              ROTATE THE PRODUCT TO RIGHT
07BD  07BD  0D 64 08      LOC5      LODA,R0   RSLT-256+4,R1
07C0        50                      RRR,R0                    ROTATE RESULT
07C1        CD 64 08                STRA,R0   RSLT-256+4,R1
07C4        D9 77                   BIRR,R1   LOC5
07C6        FB 53                   BDRR,R3   LOOO            FINISH THE LOOP
07C8        17                      RETC,UN
```

FIGURE 3-2

4. BINARY MULTIPLICATION FOR ONE-BYTE SIGNED INTEGERS

FUNCTION:

One byte by one byte multiplication for signed integers giving a two-byte result.
(OPR1) X (OPR2) ⟶ RSLT, RSLT + 1
The Booth algorithm is used (see Figure 4.1).

PARAMETERS:

Input: OPR1 contains multiplier
OPR2 contains multiplicand

Output: RSLT contains high-order product byte.
RSLT + 1 contains low-order product byte.

SPECIAL REQUIREMENTS:

None

Refer to Figures 4.1 and 4.2 for flowcharts and to Figure 4.3 for program listing.

	HARDWARE AFFECTED						
REGISTERS	R0	R1	R2	R3	R1′	R2′	R3′
	X	X	X	X			
PSU	F	II	SP				
PSL	CC	IDC	RS	WC	OVF	COM	C
	X	X		X	X		X

RAM REQUIRED (BYTES): 4

ROM REQUIRED (BYTES): 51

EXECUTION TIME: Variable

MAXIMUM SUBROUTINE NESTING LEVELS: None

ASSEMBLER/COMPILER USED: PIPHASM

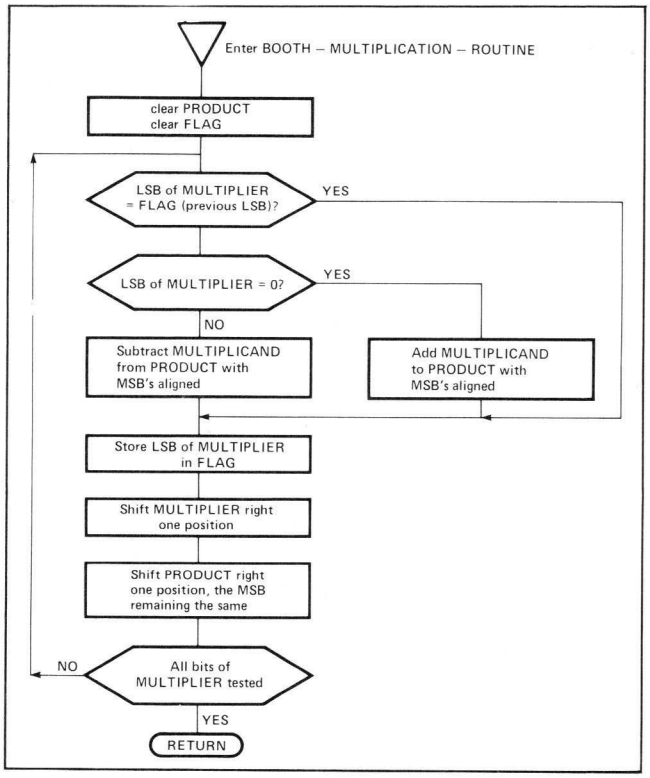


FIGURE 4-1 Flowchart of Booth Algorithm
Multiplicand X Multiplier → Product

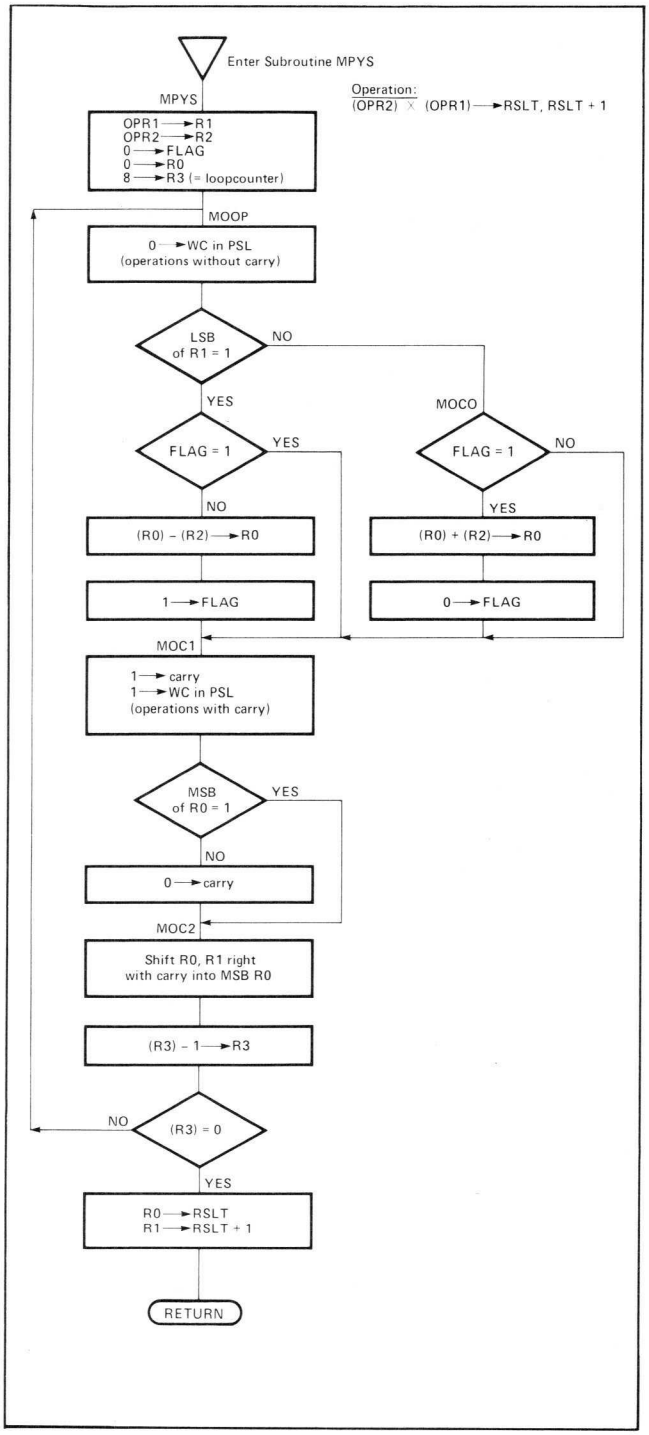


FIGURE 4-2 Flowchart for Signed Multiplication Using Booth Algorithm (One-Byte Operands; Two-Byte Result)

```
100                                 *       PD760032
101                                 ********************************************************************
102                                 *  BINARY MULTIPLICATION USING BOOTH-ALGORITHM
103                                 *  FOR 2 ONE-BYTE SIGNED  INTEGERS.
104                                 ********************************************************************
105                                 *  FIRST OPERAND IS IN OPR1
106                                 *  SECOND OPERAND IS IN OPR2 (OPR2) ≠ H'80'
107                                 *  PRODUCT WILL BE IN RSLT,RSLT+1
108                                 *
109                                         ORG   H'800'
110  0800  0800  74 40              MPYS     CPSU     F           CLEAR FLAG IN PSU
111  0802        0D 05 00                    LODA,R1  OPR1        GET 1ST OPERAND
112  0805        0E 05 02                    LODA,R2  OPR2        GET 2ND OPERAND
113  0808        07 08                       LODI,R3  8           LOAD LOOP COUNTER R3
114  080A        20                          EORZ     R0          CLEAR R0
115  080B  080B  75 08              MOOP     CPSL     WC          CLEAR WC IN PSL
116  080D        F5 01                       TMI,R1   H'01'
117  080F        98 09                       BCFR,ON  MOC0        LSB OF R1 SET?
118  0811        B4 40                       TPSU     F           YES
119  0813        18 0C                       BCTR,ON  MOC1        FLAG =1?
120  0815        A2                          SUBZ     R2          NO,SUBTRACT WITHOUT BORROW
121  0816        76 40                       PPSU     F           SET FLAG
122  0818        1B 07                       BCTR,UN  MOC1        BRANCH TO DOUBLE SHIFT
123  081A  081A  B4 40              MOC0     TPSU     F           LSB OF R1 WAS 0
124  081C        98 03                       BCFR,ON  MOC1        FLAG =1?
125  081E        82                          ADDZ     R2          YES,ADD WITHOUT CARRY
126  081F        74 40                       CPSU     F           CLEAR FLAG
127  0821  0821  77 09              MOC1     PPSL     WC+C        SET C AND WC
128  0823        60                          IORZ     R0
129  0824        1A 02                       BCTR,N   MOC2        MSB OF R0 SET?
130  0826        75 01                       CPSL     C           NO,CLEAR CARRY
131  0828  0828  50                MOC2     RRR,R0               SHIFT R0 R1 RIGHT
132  0829        51                          RRR,R1               MSB OF R0 IS SAME
133  082A        FB 5F                       BDRR,R3  MOOP        BRANCH TO LOOP IF NOT READY
134  082C        CC 05 04                    STRA,R0  RSLT        STORE RESULT
135  082F        CD 05 05                    STRA,R1  RSLT+1
136  0832        17                          RETC,UN              EXIT SUBROUTINE MPYS
137                                 *
```

FIGURE 4-3

5. BINARY MULTIPLICATION FOR TWO-BYTE SIGNED INTEGERS

FUNCTION:

Two byte by two byte multiplication for signed integers giving a four byte result.
(OPR1, OPR1 + 1) X (OPR2, OPR2 + 1)
⟶ RSLT, RSLT + 1, RSLT + 2, RSLT + 3.
The Booth algorithm (Figure 4.1) is used.

PARAMETERS:

Input: OPR1, OPR1 + 1 contains multiplicand
OPR2, OPR2 + 1 contains multiplier

Output: RSLT, RSLT + 1, RSLT + 2, RSLT + 3 contains product.
OPR1, OPR2, and RSLT are most-significant bytes.

SPECIAL REQUIREMENTS

None

Refer to Figure 5.1 for flowchart and to Figure 5.2 for program listing.

	HARDWARE AFFECTED						
REGISTERS	R0	R1	R2	R3	R1'	R2'	R3'
	X	X	X	X			
PSU	F	II	SP				
PSL	CC	IDC	RS	WC	OVF	COM	C
	X	X		X	X		X

RAM REQUIRED (BYTES): 8

ROM REQUIRED (BYTES): 71

EXECUTION TIME: Variable

MAXIMUM SUBROUTINE NESTING LEVELS: None

ASSEMBLER/COMPILER USED: PIPHASM

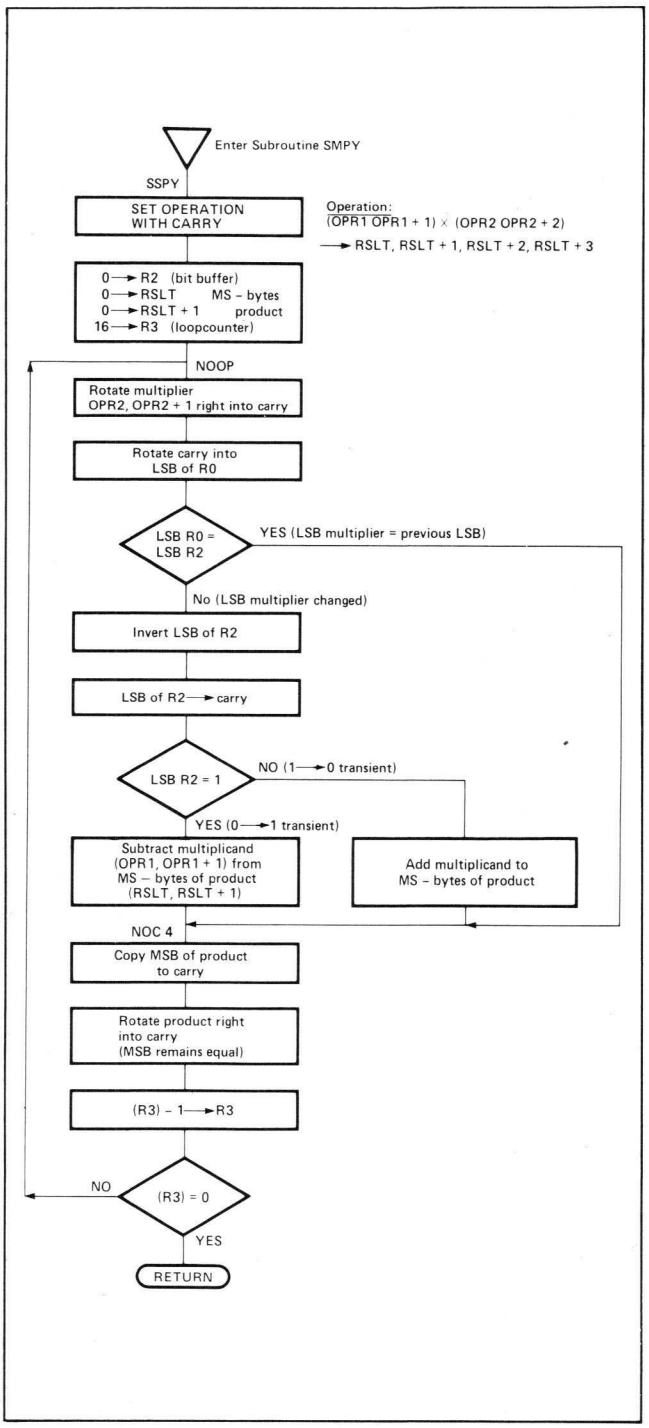


FIGURE 5-1 Flowchart for Signed Multiplication Using Booth Algorithm (Two-Byte Operands; Four-Byte Result)

```
                             *    PD760033
                             ***********************************************************************
                             * BINARY MULTIPLICATION FOR TWO BYTE SIGNED INTEGERS
                             ***********************************************************************
                             * MULTIPLICAND IS IN LOCATIONS OPR1,OPR1+1
                             * MULTIPLIER IS IN LOCATIONS OPR2,OPR2+1
                             *
                             * RESULT WILL BE STORED IN RSLT,RSLT+1,RSLT+2,RSLT+3
                             *
                             *   AFTER MULTIPLICATION THE MULTIPLICAND IS UNCHANGED
                             *   THE MULTIPLIER IS DESTROYED
                             *   THE MULTIPLICAND MUST BE UNEQUAL H'8000'
                             ***********************************************************************
0833 0833 77 08              SSPY  PPSL      WC             ARITH AND ROTATE WITH C
0835      20                       EORZ      R0             CLEAR R0
0836      C2                       STRZ      R2             CLEAR R2
0837      CC 05 04                 STRA,R0   RSLT           CLEAR 2 MSBYTES OF PRODUCT
083A      CC 05 05                 STRA,R0   RSLT+1
083D      07 10                    LODI,R3   16             LOAD LOOP COUNTER R3
083F 083F 05 FE              NOOP  LODI,R1   -2             LOAD INDEX REG WITH 254
0841 0841 0D 64 04           NOC0  LODA,R0   OPR2-256+2,R1  ROTATE MULTIPLIER
0844      50                       RRR,R0                   INTO CARRY
0845      CD 64 04                 STRA,R0   OPR2-256+2,R1
0848      D9 77                    BIRR,R1   NOC0           BRANCH IF NOT DONE
084A      20                       EORZ      R0             CLEAR R0
084B      D0                       RRL,R0                   ROTATE CARRY IN LSB OF R0
084C      22                       EORZ      R2             LSB OF R0 BECOMES 1 FOR CHANGE
084D      18 19                    BCTR,Z    NOC4           BRANCH IF NO CHANGE
084F      22                       EORZ      R2             INVERT LSB OF R2
0850      C2                       STRZ      R2             RESTORE NEW R2
0851      50                       RRR,R0                   LSB OF R2 INTO CARRY OR BORROW
0852      05 02                    LODI,R1   2              LOAD INDEX
0854 0854 0D 45 04           NOC1  LODA,R0   RSLT,R1,-      LOAD BYTE OF RSLT IN R0
0857      F6 01                    TMI,R2    1
0859      18 05                    BCTR,ON   NOC2           BRANCH TO SUBTRACT IF LSB R2=1
085B      8D 65 00                 ADDA,R0   OPR1,R1        ADD BYTE MPLCND TO RSLT
085E      1B 03                    BCTR,UN   NOC3
0860 0860 AD 65 00           NOC2  SUBA,R0   OPR1,R1        SUB BYTE MPLCND FROM RSLT
0863 0863 CD 65 04           NOC3  STRA,R0   RSLT,R1        RESTORE INTERMEDIATE RSLT
0866      59 6C                    BRNR,R1   NOC1           BRANCH IF ADD SUBTRACT NOT READY
                             *
0868 0868 0C 05 04           NOC4  LODA,R0   RSLT
086B      D0                       RRL,R0
086C      04 FC                    LODI,R0   -4             LOAD INDEX
086E 086E 0D 64 08           NOC5  LODA,R0   RSLT-256+4,R1  FETCH MS BYTE PRODUCT
0871      50                       RRR,R0                   ROTATE RSLT,PROD+1 ETC TO RIGHT
0872      CD 64 08                 STRA,R0   RSLT-256+4,R1  KEEPING MSB SAME
0875      D9 77                    BIRR,R1   NOC5           BRANCH IF NOT DONE
0877      FB 46                    BDRR,R3   NOOP           BRANCH IF LOOP NOT READY
0879      17                       RETC,UN                  RETURN TO MAIN PROGRAM
                                    END
```

FIGURE 5-2

6. BINARY DIVISION

A. UNSIGNED INTEGERS TWO-BYTE DIVIDEND; ONE-BYTE DIVISOR

FUNCTION:

Division of a two byte dividend by a one byte divisor, resulting in a two-byte quotient and a one-byte remainder.

$$\frac{(DVDN,\ DVDN+1)}{(DVSR)} \longrightarrow \begin{cases} DVDN,\ DVDN+1 & \text{(quotient)} \\ R1 & \text{(remainder)} \end{cases}$$

PARAMETERS:

Input: DVDN, DVDN + 1 contains dividend
DVSR contains divisor
DVDN is most-significant byte

Output: DVDN, DVDN + 1 contains quotient
R1 contains remainder
DVDN is most-significant byte.
Dividend is destroyed after execution of division.

SPECIAL REQUIREMENTS:

None

Refer to Figure 6.1 for flowchart and to Figure 6.2 for program listing.

	HARDWARE AFFECTED						
REGISTERS	R0	R1	R2	R3	R1'	R2'	R3'
	X	X	X	X			
PSU	F	II	SP				
PSL	CC	IDC	RS	WC	OVF	COM	C
	X	X		X	X	X	X

RAM REQUIRED (BYTES): 3

ROM REQUIRED (BYTES): 45

EXECUTION TIME: Variable

MAXIMUM SUBROUTINE NESTING LEVELS: None

ASSEMBLER/COMPILER USED: PIPHASM

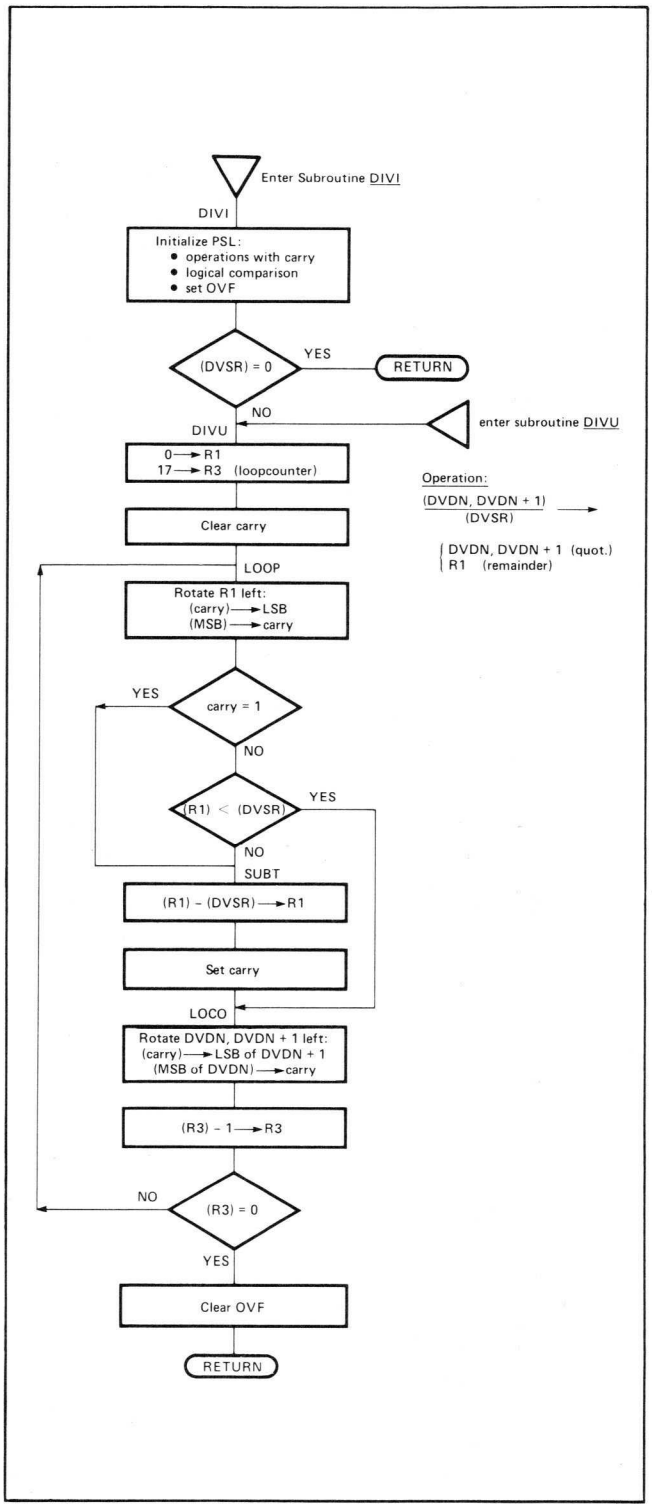


FIGURE 6-1 Flowchart for Unsigned Division (Dividend or Quotient: Two-Bytes; Divisor or Remainder: One-Byte)

```
                                *       PD760040                                        *
                                ********************************************************************************
                                *     BINARY DIVISIONS FOR INTEGERS
                                ********************************************************************************
                                * DIVIDEND IS IN DVDN,DVDN+1 16 BITS
                                * DIVISOR IS IN  DVSR        8 BITS
                                * QUOTIENT WILL BE IN DVDN,DVDN+1 16 BITS
                                * AFTER DIVISION, DIVIDEND WILL BE DESTROYED
                                * R1 WILL HOLD REMAINDER
                                * OVF=1 IMPLIES OVERFLOW
                                *
                                *
                                *          SYMBOL DEFINITIONS
        0000                    R0      EQU     0
        0001                    R1      EQU     1
        0002                    R2      EQU     2
        0003                    R3      EQU     3
        0001                    R4      EQU     1
        0002                    R5      EQU     2
        0003                    R6      EQU     3
        0003                    UN      EQU     3               UNCONDITIONAL BRANCHING
        0001                    C       EQU     1
        0000                    ON      EQU     0
        0002                    LT      EQU     2
        0000                    Z       EQU     0
        0000                    EQ      EQU     0
        0001                    P       EQU     1
        0002                    N       EQU     2
        0008                    WC      EQU     8
        0004                    OVF     EQU     4
        0002                    COM     EQU     2
                                *
                                        ORG     H'500'          UNSIGNED DIVISION SUBROUTINE
                                *
0500  0500  77 0E               DIVI    PPSL    WC+OVF+COM      ARITH ROTATE WITH CARRY
0502        0C 06 02                    LODA,R0 DVSR            FETCH DIVISOR
0505        14                          RETC,Z                  RETURN WITH OVF =1 IF DVSR =0
                                *
0506  0506  05 00               DIVU    LODI,R1 0               CLR R1
0508        07 11                       LODI,R3 17              LOAD LOOP COUNTER R3
050A        75 01                       CPSL    C               CLEAR CARRY
050C  050C  D1                  LOOP    RRL,R1                  ROTATE CARRY IN LSB OF R1
050D        B5 01                       TPSL    C
050F        18 05                       BCTR,ON SUBT            GO TO SUBTRACT IF CARRY =1
0511        ED 06 02                    COMA,R1 DVSR
0514        1A 07                       BCTR,LT LOC0            IF R1<DVSR,NO SUBTRACTION
0516  0516  77 01               SUBT    PPSL    C               CLR BORROW
0518        AD 06 02                    SUBA,R1 DVSR            SUBTR DVSR FROM REMAINDER
051B        77 01                       PPSL    C               SET CARRY
051D  051D  06 02               LOC0    LODI,R2 2               LOAD INDEX REGISTERR
051F  051F  0E 46 00            LOC1    LODA,R0 DVDN,R2,-       ROTATE QUOTIENT BIT
0522        D0                          RRL,R0                  DVDN,DVDN+1 AND MSB OF
0523        CE 66 00                    STRA,R0 DVDN,R2         DVDN INTO CARRY
0526        5A 77                       BRNR,R2 LOC1            BRANCH IF ROTATE NOT READY
0528        FB 62                       BDRR,R3 LOOP            BRANCH IF DIVISION NOT READY
052A        75 04                       CPSL    OVF             CLEAR OVF IN PSL
052C        17                          RETC,UN                 RETURN TO MAIN PROGRAM
                                *
```

FIGURE 6-2

B. SIGNED INTEGERS TWO-BYTE DIVIDEND; ONE-BYTE DIVISOR

FUNCTION:

Division of a two-byte dividend by a one-byte divisor, resulting in a two-byte quotient and a one-byte remainder.

$$\frac{(\text{DVDN, DVDN + 1})}{(\text{DVSR})} \longrightarrow \begin{cases} \text{DVDN, DVDN + 1} & \text{(quotient)} \\ \text{R1} & \text{(remainder)} \end{cases}$$

PARAMETERS:

Input: DVDN, DVDN + 1 contains dividend
DVSR contains divisor
DVDN is most-significant byte.

Output: DVDN, DVDN + 1 contains quotient
R1 contains remainder
DVDN is most-significant byte.
Dividend is destroyed after execution of division; negative divisor becomes positive

SPECIAL REQUIREMENTS:

Software: Unsigned division subroutine

Refer to Figure 6.3 for flowchart and to Figure 6.4 for program listing.

	HARDWARE AFFECTED						
REGISTERS	R0 X	R1 X	R2 X	R3 X	R1'	R2'	R3'
PSU	F	II	SP				
PSL	CC X	IDC X	RS	WC X	OVF X	COM X	C X

RAM REQUIRED (BYTES): 4

ROM REQUIRED (BYTES): 61

EXECUTION TIME: Variable

MAXIMUM SUBROUTINE NESTING LEVELS: 1

ASSEMBLER/COMPILER USED: PIPHASM

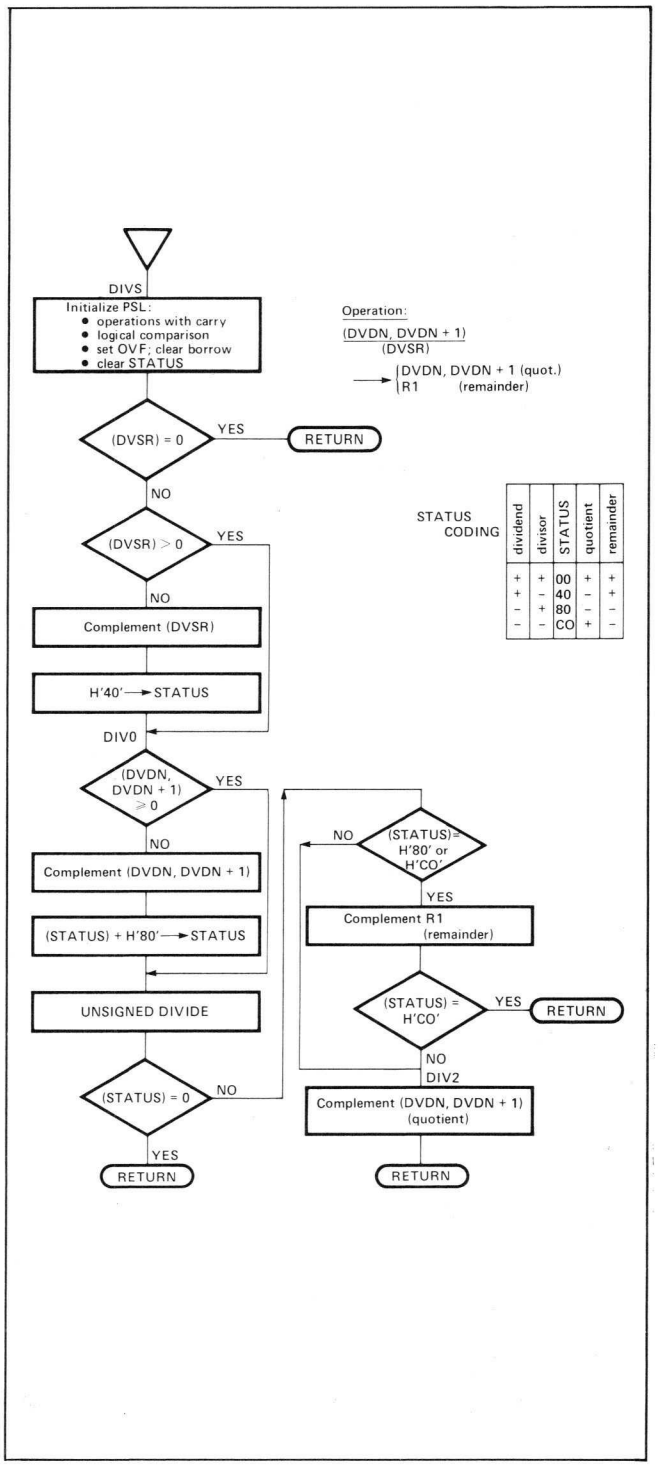


FIGURE 6-3 Flowchart for Signed Division (Dividend & Quotient: 2 Bytes; Divisor & Remainder: 1 Byte)

```
                                *       PD760041
                                ********************************************************************
                                *     SIGNED DIVISION
                                ********************************************************************
                                *
                                * NEGATIVE DIVIDEND AND OR DIVISOR ARE COMPLEMENTED
                                * PRIOR TO EXECUTION OF DIVISION
                                *
                                * SIGNS ARE CODED IN STATUS:
                                *   STATUS CODING:DVDN DVSR STAT QUOT RMDR
                                *                  +    +   00    +    +
                                *                  +    -   40    -    +
                                *                  -    +   80    -    -
                                *                  -    -   C0    +    -
                                * DIVIDEND MUST BE UNEQUAL H'8000' (NO CORRECT OVF)
                                * NEGATIVE SIGN OF DIVISOR IS LOST AFTER EXECUTION.
0052D  0052D  77 0D             DIVS    PPSL    WC+OVF+C        ARITH ROTATE WITH CARRY ETC
0052F         20                        EORZ    R0
00530         C1                        STRZ    R1              CLEAR R1
00531         0E 06 02                  LODA,R2 DVSR            FETCH DIVISOR IN R2
00534         14                        RETC,Z                  RETURN WITH OVF SET IF DVSR= 0
00535         19 06                     BCTR,P  DIV0            BRANCH IF DIVISOR >0
00537         A2                        SUBZ    R2              TAKE 2S COMPLEMENT OF DVSR
00538         CC 06 02                  STRA,R0 DVSR            RESTORE DIVISOR
0053B         05 40                     LODI,R1 H'40'           LOAD STATUS IN R1
0053D  0053D  0E 06 00          DIV0    LODA,R2 DVDN            FETCH MS BYTE OF DIVIDEND
00540         9A 04                     BCFR,N  DIV1            BRANCH IF DIVIDEND NOT<0
00542         3B 18                     BSTR,UN CMPL            TAKE 2S COMPLEMENT OF DIVIDEND
00544         85 80                     ADDI,R1 H'80'           UPDATE STATUS
00546  0546   CD 06 03         DIV1    STRA,R1 STAT            SAVE STATUS
00549         3F 05 06                  BSTA,UN DIVU            CALL UNSIGNED DIVISION
0054C         0F 06 03                  LODA,R3 STAT            LOAD STATUS IN R3
0054F         14                        RETC,Z                  RETURN IF BOTH DVDN AND DVSR NOT<0
00550         19 07                     BCTR,P  DIV2            BRANCH IF DVDN WAS NOT <0 AND DVSR<0
00552         77 01                     PPSL    C               CLEAR BORROW
00554         20                        EORZ    R0              CLEAR R0
00555         A1                        SUBZ    R1              TAKE 2 S COMPLEMENT OF REMAINDER
00556         C1                        STRZ    R1              RESTORE REMAINDER IN R1
00557         D3                        RRL,R3                  SHIFT R3 LEFT
00558         16                        RETC,N                  RETURN IF BOTH DVDN,DVSR<0
00559  0559   3B 01          DIV2    BSTR,UN CMPL            TAKES 2S COMPL. OF QUOTIENT
0055B         17                        RETC,UN                 RETURN TO MAINPROGRAM
                                *
                                *
                                *  SUBROUTINE TO TAKE 2S COMPL
                                *    OF (DVDN,DVDN+1)
                                *
0055C  0055C  77 01             CMPL    PPSL    C               CLEAR BORROW
0055E         07 02                     LODI,R3 2               LOAD INDEX REG
00560  0560   20                CMP0    EORZ    R0              CLR R0
00561         AF 46 00                  SUBA,R0 DVDN,R3,-       COMPLEMENT BYTE
00564         CF 66 00                  STRA,R0 DVDN,R3         RESTORE RESULT
00567         5B 77                     BRNR,R3 CMP0            BRANCH IF NOT DONE
00569         17                        RETC,UN
                                        ORG     H'600'
              0600              DVDN    RES     2               DIVIDEND AND QUOTIENT
              0602              DVSR    RES     1               DIVISOR
              0603              STAT    RES     1               STATUS REG
                                    END
```

FIGURE 6-4

Signetics 2650 Microprocessor application memos currently available:

AS50	Serial Input/Output
AS51	Bit and Byte Testing Procedures
AS52	General Delay Routines
AS54	Conversion Routines
SP50	2650 Evaluation Printed Circuit Board Level System (PC1001)
SP51	2650 Demo Systems
SP52	Support Software for use with the NCSS Timesharing System
SP53	Simulator, Version 1.2
SP54	Support Software for use with the General Electric Mark III Timesharing System
SS50	PIPBUG
SS51	Absolute Object Format (Revision 1)
MP51	2650 Initialization
MP52	Low Cost Clock Generator Circuits