PROGRAMMING THE 1401 DIVIDE FEATURE*

When the beginning programmer is told that his organization has the Multiply-Divide feature on its IBM 1401, he is inclined to think that this means it is going to be easy to multiply and divide. He is shocked from this false sense of security when he is first exposed to the rules for programming a division with the use of the feature. After he is on the job, he is likely to continue to be annoyed by the complexity of programming a division and frustrated by the common occurence of programming errors in one part or another of his division routines.

It is something of a paradox that programming a division may actually be easier at an institution that does not have the Multiply-Divide feature. At that institution, the programmers first write themselves a division subroutine that is then available for use in all subsequent programs. With the typical subroutine of this sort, one moves the dividend and divisor to a particular place, branches to the subroutine, and gets the quotient. With such a subroutine, division becomes, in the words of a programmer at the Fed in Philadelphia, a snap.

For some unknown reason, on the other hand, programmers for machines with the special feature have generally been left to their own resources. This note is intended to remedy this injustice. It discusses the computer instructions that must be given to perform a division correctly, and at the same time presents a form (division worksheet) that can be used to program a 100 per cent accurate division the first time every time.

To write a division operation, a programmer always starts with at least seven items of information. He must know the name or core location of the dividend and divisor; the length of the dividend and divisor fields; the number of decimal places, if any, in the dividend and divisor; and the number of decimal places, if any, desired in the quotient. Suppose we take a typical problem as an example. We want to divide a 5-digit number with 2 decimal places by a 10-digit number with no decimal places, and get the answer rounded to 4 decimal places. The dividend is the change in deposits at banks and so can be either positive or negative, which means that the quotient can also be either positive or negative.

The items of information that the programmer has in this case can be listed as follows:

	Name of Field	Length of Field	Decimal Places
Dividend	DVD	5	2
Divisor	DIV	10	0
Quotient			4

*Prepared by Emanuel Melichar, Economist, Division of Research and Statistics, Board of Governors. With appropriate modifications in mnemonic operation codes, this discussion is also applicable to the IBM 1410.

The first step in using the division worksheet is to write these items into the places provided at the top of the page, as has been done in the excerpt from the worksheet that is shown below. Note that the five items of numerical information have each been assigned a letter of the alphabet by which they may be more easily referred to later in the worksheet as well as in this discussion. Thus, for instance, the length of the dividend field will be referred to as A, the length of the divisor field as B, the number of decimal places in the dividend as C, and so forth.

	Name of Field	Length of Field	Decimal Places
Dividend	DVD	A = 5	C = 2
Divisor	DIV	B = 10	D = 0
Quotient			E = 4

With this information recorded on the worksheet, the sheet can become a useful part of the documentation of the program for which the division routine is being written. Note that space is also provided for filling in the name and length of the field into which the programmer will move the quotient. Often, however, such information is not pertinent because the quotient is moved directly into an output area.

The next part of the division worksheet, labeled "Computation of Character Adjustments," consists of a set of simple calculations which yields answers to such questions as how big an area is needed for the field in which the division will take place, where the dividend should be placed in this field, which digit of this field should be addressed by the B-operand of the divide instruction, and so forth. All of these calculations, however, have been reduced to simple and foolproof formulas.

The formulas as they appear on the worksheet are shown below. The values to be used in the calculations, and also the answers, are referred to by letters of the alphabet.

Computation of Character Adjustments

G = D + E - C = _____

H = _____ (If G is greater than 0, enter that value for H. If not, enter 0.)

F = A + B = _____

J = F + H + 2 = _____

K = B + 1 = _____

L = A + G = _____

M = A + H + 1 = _____

N = L - 1 = _____

The first two formulas call for computing the values of G and H by following the directions. To get the value for G, one must add the values of D and E and subtract the value of C. In our example, D is zero, E is 4, and C is 2. Adding D to E and subtracting C therefore yields 2, which is the answer for G that is filled into the blank space provided on the worksheet.

The directions for filling in the value of H state that H is to be the same as G if G is greater than zero. This is the case in our example, so we fill in the value of 2 in the space provided. If G had been zero or negative, we would have filled in a zero for H, according to the directions on the worksheet.

The values of the other items are similarly calculated according to the formulas given. After this is done, the results for our example appear as follows:

Computation of Character Adjustments

G = D + E - C = 2

H = 2 (If G is greater than 0, enter that value for H. If not, enter 0.)

F = A + B = 15

J = F + H + 2 = 19

K = B + 1 = 11

L = A + G = 7

M = A + H + 1 = 8

N = L - 1 = 6

The next part of the worksheet contains instructions for setting up the division area and the constants that are needed. The field in which the division will take place will be referred to by the label BF given to its high order position, and the length of this field must be at least J+1, which is 20 in our example. The worksheet shows one way in which such a field can be set up in either SPS or Autocoder, namely, by establishing a one-digit DCW labeled BF, followed by a DC with a count or length at least as great as J, which is 19 in our example. This part of the worksheet is shown below with the count for the DC filled in as 19:

Required Constants

Count	Label	OP Code	Contents	Comments
1	BF	DCW		
19		DC		Count must be at least as great as J.
1	ZERO	DCW	0	
1	FIVE	DCW	5	

The length of the field can be greater if this is convenient, but it must be at least as great as the value calculated for J+1 for any division that is to be performed in that field. The only other requirements are that the field have a word-mark in the high order position and that this position be labeled BF. These requirements are automatically met by use of the DCW command as indicated above.

Two constants are also required, a zero and a five. Chances are that these constants have already been set up for use elsewhere in the program. If not, they must be established for use in the division routine. Programmers

writing in SPS can follow the example in the worksheet, while those writing in Autocoder will undoubtedly prefer to use numeric literals rather than the DCW method to get the constants.

The final step is writing the seven instructions required to perform the division. This is done by filling in the blanks in the instructions printed on the worksheet and then copying these instructions onto the coding sheet for the program. This part of the worksheet appears as follows:

Instructions Number	OP Code	A-Address	B-Address
1	ZA	ZERO	BF + _____ (J)
2	ZA	_____ (Dividend)	BF + _____ (F)
3	MZ	BF + _____ (F)	BF + _____ (J)
4	MZ	ZERO	BF + _____ (F)
5	D	_____ (Divisor)	BF + _____ (K)
6	A	FIVE	BF + _____ (L)
7	MZ	BF + _____ (M)	BF + _____ (N)

The Quotient is located at BF +N, is signed, and in L digits in length.

Note that worksheet makes the writing of the instructions utterly simple because all that the programmer now has to do is to fill in the blanks with the appropriate numerical values previously calculated. These values become character adjustments in the instructions. For instance, in our example the B-operand of the first instruction becomes BF with a character adjustment of +019 if written in SPS. In Autocoder it is BF+19. In either case, the writing of the instruction has been "automated" and the chance of making an error has been drastically reduced. The same procedure is followed in writing the remaining instructions, with corresponding benefits. The complete set of instructions in our example, ready for copying to a coding sheet, is as follows:

Instructions Number	OP Code	A-Address	B-Address
1	ZA	ZERO	BF + 19 (J)
2	ZA	DVD (Dividend)	BF + 15 (F)
3	MZ	BF + 15 (F)	BF + 19 (J)
4	MZ	ZERO	BF + 15 (F)
5	D	DIV (Divisor)	BF + 11 (K)
6	A	FIVE	BF + 7 (L)
7	MZ	BF + 8 (M)	BF + 6 (N)

The Quotient is located at BF+N, is signed, and in L digits in length.

If one stops to consider the procedures by which this set of instructions have been obtained, one realizes that it has been completely unnecessary for the programmer to know anything about the complicated rules for doing a division. This is as it should be, because then temporarily forgetting one of the rules does not result in an error in the program.

A brief discussion of the function of each of the instructions, however, may be informative to neophyte programmers. Instruction (1) puts zeros into the field in which the division will take place and places AB bits in the low order position of this field. It thus prepares for the division operation.

Instruction (2) moves the dividend into the correct location in the division area. After it is moved in, the number of spaces remaining at the high order end of the division area will always be equal to the length of the divisor field plus 1, as is required. At the low order end of the division area, spaces are automatically allowed for the development of extra digits in the quotient, if any are required (two were needed in our example), and two additional spaces have also been allowed. One of these will be used in rounding the quotient and the other is used to retain the sign of the quotient during the rounding operation. In addition, the zero and add op code of instruction (2) has insured that the low order position of the dividend in the division area is now signed with either a B-bit or AB-bits.

Instruction (3) now moves these zone bits to the low order position of the division area. It is necessary that there be either a B-bit (if the dividend is negative) or AB-bits (if the dividend is positive) in the low order position of the division area because the computer looks for these bits during the execution of the divide instruction in order to discover when the division is finished. It does <u>not</u> look for a word-mark to find out this fact. This has two implications. First, as previously noted, the length of the area used for division can be longer than J+1 as long as there is a word-mark in the high order digit and the high order end of the area is used for the division. Thus the same area can be used for several division routines involving dividends and divisors with different lengths. Second, it is important to ensure that there will be zone bits in the low order position of the division area and nowhere else in the area, which is invariably accomplished by instructions (1) through (4). The lack of zone bits to stop the division operation correctly can cause a very puzzling kind of error stop.

Instruction (4) removes the zone bits from the low order position of the dividend in the division area. Failure to remove these bits would cause the division operation to stop too early.

Instruction (5) is the actual division operation. As required, the B-operand of this instruction addresses the high order position of the dividend in the division area.

Instruction (6) rounds off (half-adjusts) the quotient by adding 5 to the proper location. As noted above, an extra digit has been developed so that the sign of the quotient will not be stored in the location to which the 5 is added, because then negative quotients would be rounded in the wrong direction. This avoids an apparently common error condition that is hard to detect during testing and that has even been observed to be present in supposedly debugged programs.

Instruction (7) moves the zone bits to the low order position of the properly rounded quotient. The quotient will therefore always be signed with either a B-bit or AB-bits.

The complete division worksheet is reproduced on the following page. At the botton of the sheet, there are instructions to cover special cases in which the number of instructions in the division routine can be reduced, including detailed instructions for the common special case in which the dividend field is unsigned (no zone bits) and it is known that the quotient will always be positive.

Additional copies of the division worksheet are available from the author.

401 Subroutine
Melichar

DIVISION WORKSHEET

For 1401 Multiply-Divide Feature

Program _____ Date _____

Routine _____

Name of Field	Length of Field	Decimal Places
Dividend _____	A = _____	C = _____
Divisor _____	B = _____	D = _____
Quotient _____		E = _____

Computation of Character Adjustments

G = D + E - C = _____

H = _____ (If G is greater than 0, enter that value for H. If not, enter 0.

F = A + B = _____

J = F + H + 2 = _____

K = B + 1 = _____

L = A + G = _____

M = A + H + 1 = _____

N = L - 1 = _____

Required Constants

Count	Label	OP Code	Contents	Comments
1	BF	DCW		
___		DC		Count must be at least as great as J.
1	ZERO	DCW	0	
1	FIVE	DCW	5	

Instructions

Number	OP Code	A-Address	B-Address
1	ZA	ZERO	BF + _____ (J)
2	ZA	_____ (Dividend)	BF + _____ (F)
3	MZ	BF + _____ (F)	BF + _____ (J)
4	MZ	ZERO	BF + _____ (F)
5	D	_____ (Divisor)	BF + _____ (K)
6	A	FIVE	BF + _____ (L)
7	MZ	BF + _____ (M)	BF + _____ (N)

The Quotient is located at BF +N, is signed, and in L digits in length.

Special Cases

1) If the Dividend is known to be unsigned, you may use MCW for OP code of Instruction (2) and omit Instructions (3) and (4).
2) If the Dividend is known to be signed and positive, you may omit Instruction (3)
3) If the Quotient is known to be positive, you may omit Instruction (7). If you do this, the Quotient will always be unsigned.

Instructions for the common combination of special cases (1) and (3):

ZA	ZERO	BF + _____ (J)
MCW	_____ (Dividend)	BF + _____ (F)
D	_____ (Divisor)	BF + _____ (K)
A	FIVE	BF + _____ (L)

The Quotient is located at BF +N, is unsigned, and is L digits in length.