The eighth in a series of tutorials for the beginner to intermediate machine language programmer

Machine Language Made BASIC: Part VIII: And More Math

By William P. Nee

Because the computer uses the Base 2 system, math problems involving multiplying or dividing by two are very easy. Shifts are a quick way to multiply or divide registers A, B, or D by two. The LSR (logical shift right) command will shift each bit in registers A or B to the right, effectively dividing either by two; however, any remainder is lost. As the following example illustrates, the right bit (Bit 0) of the register goes to the *carry bit* of the CC register, and the left bit (Bit 7) becomes a zero:

Register A	=	10101010	=	170
LSRA	=	01010101	=	85
Carry Bit	=	0		

Notice that the carry bit of the CC register is now a zero. When signed numbers are used, the ASR (arithmetic shift right) works the same way as LSR except that Bit 7 (the sign bit) stays the same, so the sign of the number will remain unchanged.

An LSL (logical shift left) will multiply register A or B by two. This time, Bit 0 will become a zero and Bit 7 will be stored in the carry bit of the CC register. An ASL (arithmetic shift left) does exactly the same function as an LSL; neither will retain the sign bit. Instead, it is stored in the carry bit, as shown by the following:

Register B	=	01010101	=	85
LSLB	=	10101010	=	170
Carry Bit	=	0		

The following two branch commands are useful if you want to check the carry bit:

BCC — branch if the carry bit is clear (=0)

BCS — branch if the carry bit is set (=1)

There is no specific command to shift Register D. Therefore, if your number uses both registers A and B, to shift Register D you must use both shift and rotate functions. Rotating (ROR, ROL) either register A or B will shift all of the bits one space. The empty bit will be filled by the value in the carry bit, and the bit that is lost will go to the carry bit. To perform a right shift on Register D, you must complete the following functions:

(For Unsigned Numbers)	(For Signed Numbers)
LSRA	ASRA
RORB	RORB

Let's try an example. If we load Register D with #938, our computer will follow these instructions:

	Register	Carry Bit:	Register
	00000011		10101010
LSRA	00000001	1	10101010
RORB	00000001		11010101

The one in Bit 0 of Register A went to the carry bit after the LSRA operation and then to Bit 7 of Register B after the RORB function. These two operations have divided the number in Register D by two. Repeating will continue to divide the number by two. To multiply Register D by two, perform a logical shift left on Register B (LSLB), and then rotate Register A to the left (ROLA).

These two routines perform the function opposite that of the routines in the division example. Continued operations will continue doubling the number. Of course, in either multiplication or division, if you keep repeating the same routine, you will get an incorrect result. Not only can registers A, B and D be shifted and rotated, but memory locations can be shifted or rotated to the right or left as well.

You may also use the floating point format (FP1) to multiply or divide. Location $4F is the exponent of the number in FP1, and changing this will change the number. Adding one to $4F is the same as multiplying that number by two to the first power; adding an eight would multiply the number by two to the eighth power, or 256. Subtracting two from $4F would divide the number by four. You can do these operations by performing the following operations:

```
LDA $4F     exponent of the number
            in FP1
ADDA #8     multiply by 256
STA $4F     new number is in FP1
```

Bill Nee bucked the "snowbird" trend by retiring to Wisconsin from a banking career in Florida. He spends the long, cold winters writing programs for his CoCo.

There is no division command in either BASIC or machine language that does not use *FP1*. The program at the end of this article will divide a one-byte unsigned number by another unsigned one-byte number. The result will be a two-byte number in Register D. Register A will hold the whole number, and Register B will hold the decimal. Remember, they both are Hex numbers. A .8 in Hex is 8/16 (or .5 in Base 10). A .C is 12/16 in Hex (or .75 in Base 10).

We will use the second half of this program in a future article to compute the slope of a line. Load Register A (the dividend) and Register B (the divisor) with different numbers to make sure you understand the results.

(Questions and comments concerning this tutorial may be directed to the author at Route 2, Box 216 C, Mason, WI 54846-9302. Please include a self-addressed, stamped-envelope when requesting a reply.) □

The Listing: SHIFTS

```
3ØØØ                ØØ1ØØ          ORG    $3ØØØ
3ØØØ 7F     3Ø53    ØØ11Ø START    CLR    WHOLE
3ØØ3 86     FF      ØØ12Ø          LDA    #255     DIVIDEND EXAMPLE
3ØØ5 C6     7F      ØØ13Ø          LDB    #127     DIVISOR EXAMPLE
3ØØ7 F7     3Ø54    ØØ14Ø          STB    DIVSR    SAVE THE DIVISOR
3ØØA B1     3Ø54    ØØ15Ø          CMPA   DIVSR
3ØØD 25     23      ØØ16Ø          BLO    DIVID2   IT'S A FRACTION
3ØØF 27     3D      ØØ17Ø          BEQ    SAME     IT'S = 1
3Ø11 C6     Ø8      ØØ18Ø DIVID1   LDB    #8       DO IT 8 TIMES
3Ø13 F7     3Ø52    ØØ19Ø          STB    COUNT
3Ø16 1F     89      ØØ2ØØ          TFR    A,B      PUT DIVIDEND IN REGISTER B
3Ø18 4F             ØØ21Ø          CLRA
3Ø19 58             ØØ22Ø LOOP1    ASLB            SHIFT REGISTER B TO THE LEFT
3Ø1A 49             ØØ23Ø          ROLA            SHIFT REGISTER A TO THE LEFT
3Ø1B B1     3Ø54    ØØ24Ø          CMPA   DIVSR
3Ø1E 25     Ø4      ØØ25Ø          BLO    CONT1
3Ø2Ø BØ     3Ø54    ØØ26Ø          SUBA   DIVSR
3Ø23 5C             ØØ27Ø          INCB            INCREASE THE QUOTIENT
3Ø24 7A     3Ø52    ØØ28Ø CONT1    DEC    COUNT    FILLED THE BYTE YET?
3Ø27 26     FØ      ØØ29Ø          BNE    LOOP1
3Ø29 F7     3Ø53    ØØ3ØØ          STB    WHOLE    SAVE THE WHOLE NUMBER
3Ø2C 4D             ØØ31Ø          TSTA            ANY REMAINDER?
3Ø2D 26     Ø3      ØØ32Ø          BNE    DIVID2   IF SO, COUNTINUE DIVIDING
3Ø2F 1E     89      ØØ33Ø          EXG    A,B      REGISTER D HAS THE RESULTS
3Ø31 3F             ØØ34Ø          SWI
3Ø32 C6     Ø8      ØØ35Ø DIVID2   LDB    #8       DO IT 8 MORE TIMES
3Ø34 F7     3Ø52    ØØ36Ø          STB    COUNT
3Ø37 5F             ØØ37Ø          CLRB            LEAVE ROOM FOR THE QUOTIENT
3Ø38 58             ØØ38Ø LOOP2    ASLB            MOVE QUOTIENT OVER
3Ø39 49             ØØ39Ø          ROLA            SHIFT REGISTER A TO THE LEFT
3Ø3A 25     Ø5      ØØ4ØØ          BCS    LOOP3    BRANCH IF THERE'S A CARRY
3Ø3C B1     3Ø54    ØØ41Ø          CMPA   DIVSR
3Ø3F 25     Ø4      ØØ42Ø          BLO    LOOP4
3Ø41 BØ     3Ø54    ØØ43Ø LOOP3    SUBA   DIVSR
3Ø44 5C             ØØ44Ø          INCB            INCREASE THE QUOTIENT
3Ø45 7A     3Ø52    ØØ45Ø LOOP4    DEC    COUNT    FINISHED DIVIDING YET?
3Ø48 26     EE      ØØ46Ø          BNE    LOOP2
3Ø4A B6     3Ø53    ØØ47Ø          LDA    WHOLE    GET THE WHOLE NUMBER
3Ø4D 3F             ØØ48Ø          SWI             REGISTER D HAS THE RESULTS
3Ø4E CC     Ø1ØØ    ØØ49Ø SAME     LDD    #$Ø1ØØ
3Ø51 3F             ØØ5ØØ          SWI
3Ø52                ØØ51Ø COUNT    RMB    1
3Ø53                ØØ52Ø WHOLE    RMB    1
3Ø54                ØØ53Ø DIVSR    RMB    1
            3ØØØ    ØØ54Ø          END    START

ØØØØØ TOTAL ERRORS
```