1974

AN-738
Application Note

NBCD SIGN AND MAGNITUDE ADDER/SUBTRACTER

Prepared by
Joe Roy
Industrial Logic
Applications Engineering

This note describes a parallel sign and magnitude adder/subtracter for natural binary coded decimal (NBCD) numbers. The design is implemented with CMOS MSI functions: the MC14560 NBCD Adder and MC14561 9's Complementer. Decimal number representations, complement arithmetic, and adder/subtracters for unsigned numbers are also discussed.

MOTOROLA *Semiconductor Products Inc.*

NBCD SIGN AND MAGNITUDE ADDER/SUBTRACTER

INTRODUCTION

Frequently in small digital systems, simple decimal arithmetic is performed. Decimal data enters and leaves the system arithmetic unit in a binary coded decimal (BCD) format. The adder/subtracter in the arithmetic unit may be required to accept sign as well as magnitude, and generate sign, magnitude, and overflow. In the past, it has been cumbersome to build sign and magnitude adder/subtracters. Now, using Motorola's MSI CMOS functions, the MC14560 NBCD Adders and MC14561 9's Complementers, NBCD adder/subtracters may be built economically, with surprisingly low package count and moderate speed.

Some background information on BCD arithmetic is presented here, followed by simple circuits for unsigned adder/subtracters. The final circuit discussed is an adder/subtracter for signed numbers with complete overflow and sign correction logic.

DECIMAL NUMBER REPRESENTATION

Because logic elements are binary or two-state devices, decimal digits are generally represented as a group of bits in a weighted format. There are many possible binary codes which can be used to represent a decimal number. One of the most popular codes using 4 binary bits to represent 0 thru 9 is Natural Binary Coded Decimal (NBCD or 8-4-2-1 code).

NBCD is a weighted code. If a value of "0" or "1" is assigned to each of the bit positions, where the rightmost position is 2^0 and the leftmost is 2^3, and the values are summed for a given code, the result is equal to the decimal digit represented by the code. Thus, 0110 equals $0 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 = 4 + 2 = 6$. The 1010, 1011, 1100, 1101, 1110, and 1111 binary codes are not used. Because of these illegal states, the addition and subtraction of NBCD numbers is more complex than similar calculations on straight binary numbers.

ADDITION OF UNSIGNED NBCD NUMBERS

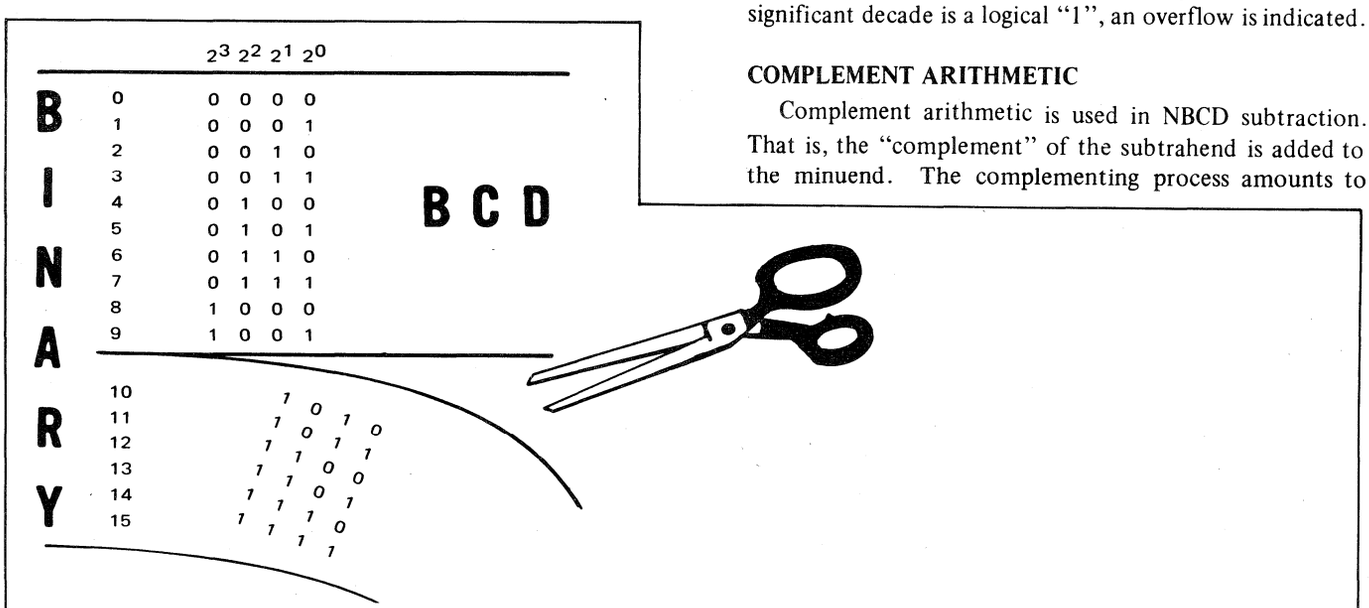
When 2 NBCD digits, A and B, and a possible carry, C, are added, a total of 20 digit sums (A + B + C) are possible as shown in Table 1.

The binary representations for the digit sums 10 thru 19 are offset by 6, the number of unused binary states, and are not correct. An algorithm for obtaining the correct sum is shown in Figure 1. A conventional method of implementing the BCD addition algorithm is shown in Figure 2(a). The NBCD digits, A and B, are summed by a 4 bit binary full adder. The resultant (sum and carry) is input to a binary/BCD code converter which generates the correct BCD code and carry.

An NBCD adder block which performs the above function is available in a single CMOS package (MC14560). Figure 2(b) shows n decades cascaded for addition of n digit unsigned NBCD numbers. Add time is typically 0.1 + 0.2n μs for n decades. When the carry out of the most significant decade is a logical "1", an overflow is indicated.

COMPLEMENT ARITHMETIC

Complement arithmetic is used in NBCD subtraction. That is, the "complement" of the subtrahend is added to the minuend. The complementing process amounts to



Binary code is often used in modified form. A 4-bit binary word has 16 states: 0 through 15. However, in the commonly-used decimal system, the basic digits are 0 through 9. Binary code is converted to BCD (binary coded decimal) by "shearing off" the illegal states 10 through 15 as shown. Then each decimal digit is represented by 4 bits.

biasing the subtrahend such that all possible sums are positive. Consider the subtraction of the NBCD numbers, A and B:

$$R = A - B$$

where R is the result. Now bias both sides of the equation by $10^N - 1$ where N is the number of digits in A and B.

$$R + 10^N - 1 = A - B + 10^N - 1$$

Rearranging,

$$R + 10^N - 1 = A + (10^N - 1 - B)$$

The term $(10^N - 1 - B)$, -B biased by $10^N - 1$, is known as the 9's complement of B. When $A > B$, $R + 10^N - 1 > 10^N - 1$; thus R is a positive number. To obtain R, 1 is added to $R + 10^N - 1$, and the carry term, 10^N, is dropped. The addition of 1 is called End Around Carry (EAC).

When $A < B$, $R + 10^N - 1 < 10^N - 1$, no EAC results and R is a negative number biased by $10^N - 1$; thus $R + 10^N - 1$ is the 9's complement of R.

TABLE 1 – Sum = A + B + C

Binary Sums	Decimal Number	Corrected Binary Sums
0000	0	0000
0001	1	0001
0010	2	0010
0011	3	0011
0100	4	0100
0101	5	0101
0110	6	0110
0111	7	0111
1000	8	1000
1001	9	1001
1010	10	0000 + Carry
1011	11	0001 + Carry
1100 Non valid	12	0010 + Carry
1101 BCD	13	0011 + Carry
1110 representation	14	0100 + Carry
1111	15	0101 + Carry
0000 + Carry	16	0110 + Carry
0001 + Carry	17	0111 + Carry
0010 + Carry	18	1000 + Carry
0011 + Carry	19	1001 + Carry

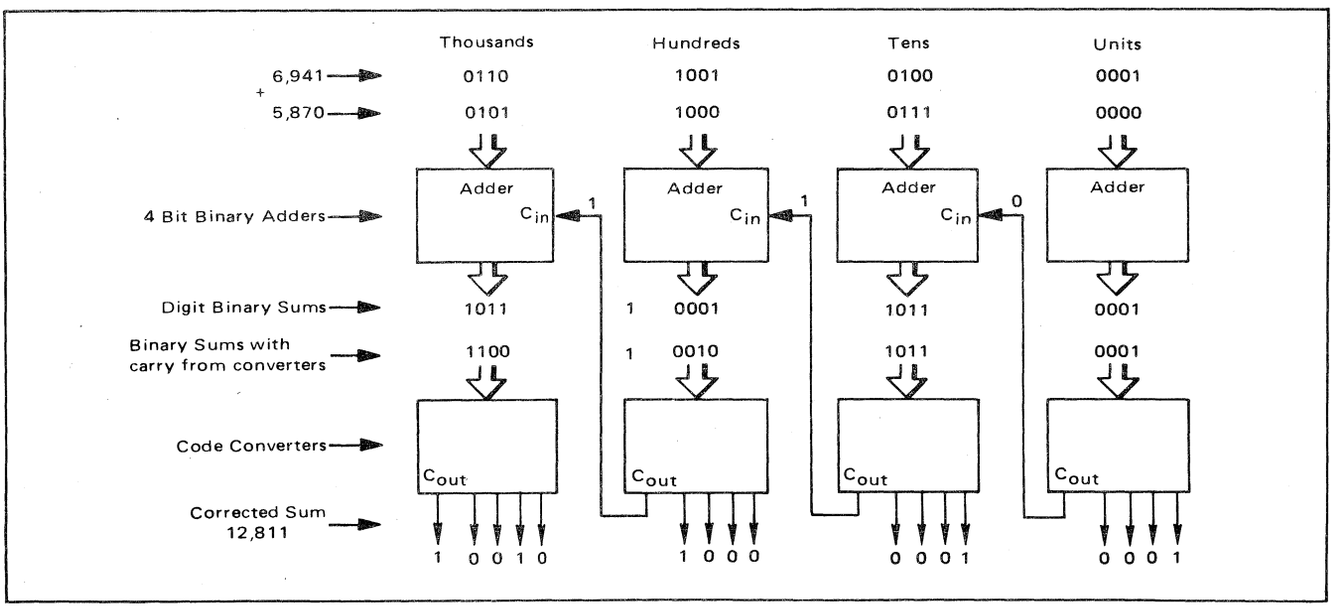


FIGURE 1 – Unsigned NBCD Addition Algorithm

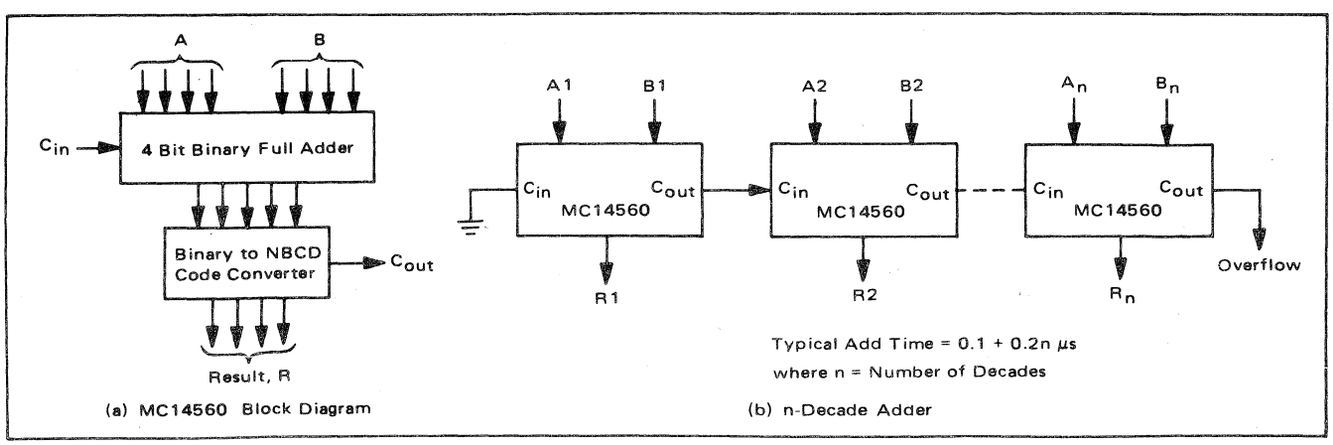


FIGURE 2 – Addition of Unsigned NBCD Numbers

SUBTRACTION OF UNSIGNED NBCD NUMBERS

Nine's complement arithmetic requires an element to perform the complementing function. An NBCD 9's complementer may be implemented using a 4 bit binary adder and 4 inverters, or with combinatorial logic. The Motorola MC14561 9's complementer is available in a single package. It has true and inverted complement disable, which allow straight-through or complement modes of operation. A "zero" line forces the outputs to "0". Figure 3 shows an NBCD subtracter block using the MC14560 and MC14561. Also shown are n cascaded blocks for subtraction of n digit unsigned numbers. Subtract time is 0.6 + 0.4n μs for n stages. Underflow (borrow) is indicated by a logical "0" on the carry output of the most significant digit. A "0" carry also indicates that the difference is a negative number in 9's complement form. If the result is input to a 9's complementer, as shown, and its mode controlled by the carry out of the most significant digit, the output of the complementer will be the correct negative magnitude. Note that the carry out of the most significant digit (MSD) is the input to carry in of the least significant digit (LSD). This End Around Carry is required because subtraction is done in 9's complement arithmetic.

By controlling the complement and overflow logic with an add/subtract line, both addition and subtraction are performed using the basic subtracter blocks (Figure 4).

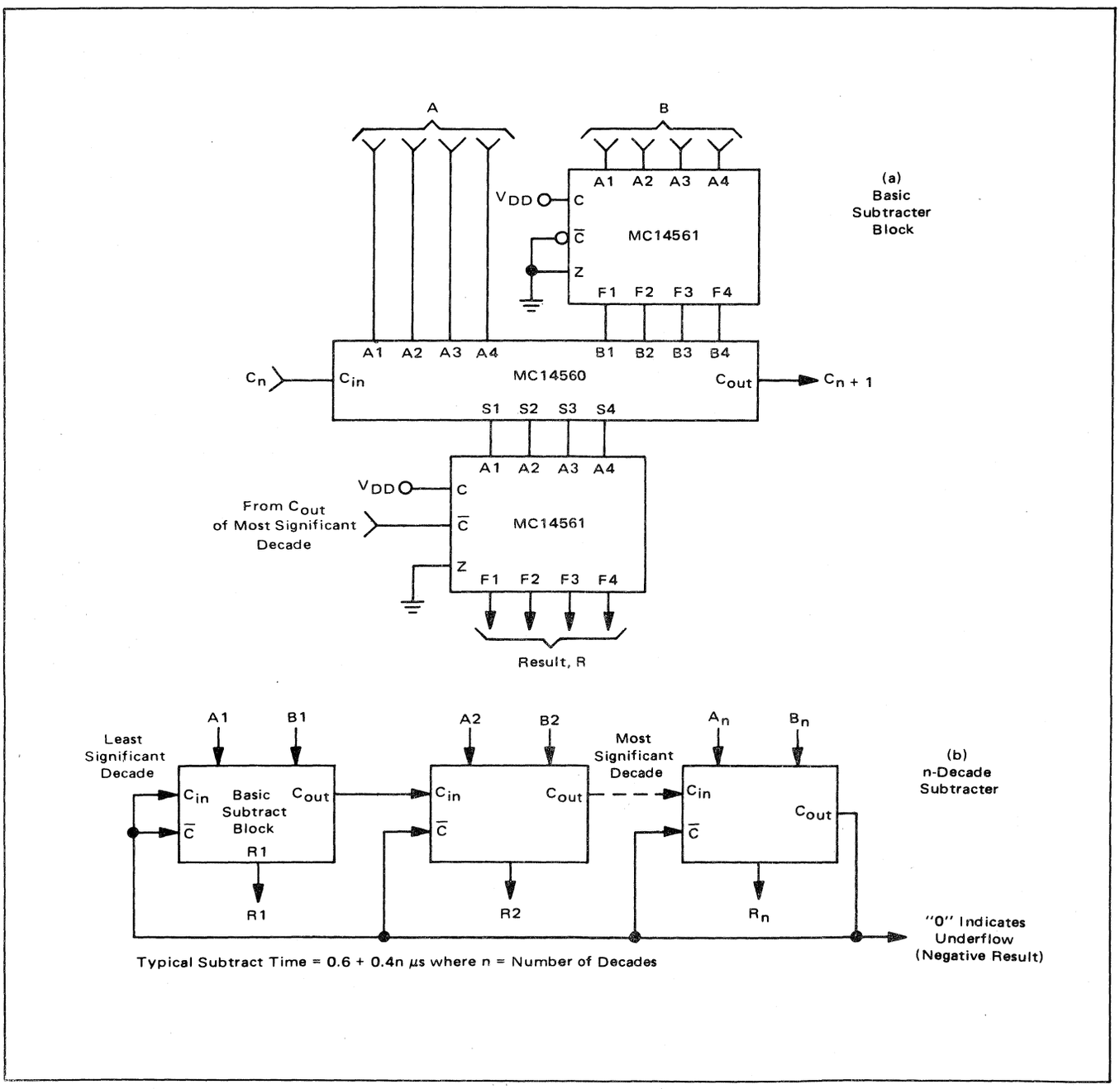


FIGURE 3 – Subtraction of Unsigned NBCD Numbers

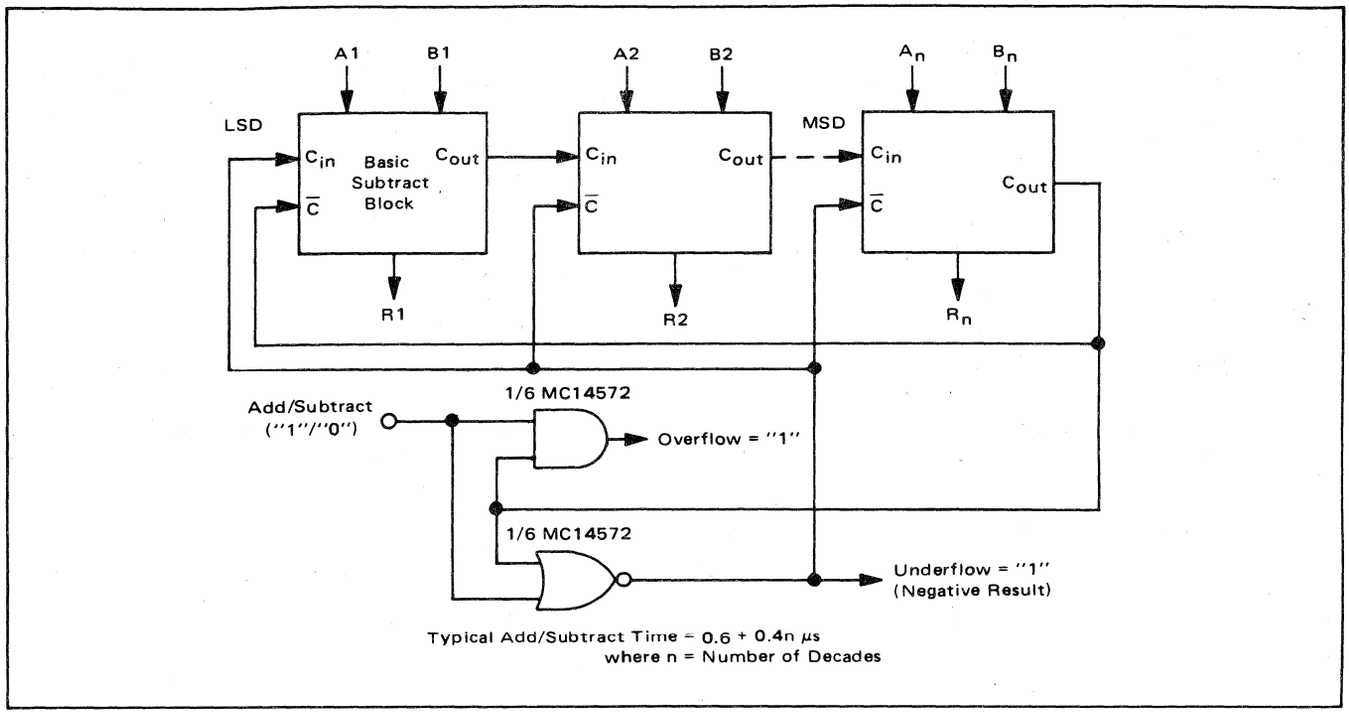


FIGURE 4 – Adder/Subtracter for Unsigned NBCD Numbers

ADDITION AND SUBTRACTION OF SIGNED NBCD NUMBERS

Using MC14560 NBCD Adders and MC14561 9's Complementers, a sign and magnitude adder/subtracter can be configured (Figure 5). Inputs A and B are signed positive (A_S, B_S = "0") or negative (A_S, B_S = "1"). B is added to or subtracted from A under control of an Add/Sub line (subtraction = "1"). The result, R, of the operation is positive signed, positive signed with overflow, negative signed, or negative signed with overflow. Add/subtract time is typically 0.6 + 0.4n μs for n decades.

An exclusive-OR of Add/Sub line and B_S produces B_S', which controls the B complementers. If B_S, the sign of B, is a logical "1" (B is negative) and the Add/Sub line is a "0" (add B to A), then the output of the exclusive-OR (B_S') is a logical "1" and B is complemented. If B_S = "1" and Add/Sub = "1", B is not complemented since subtracting a negative number is the same as adding a positive number. When Add/Sub is a "1" and B_S = "0", B_S' is a "1" and B is complemented. The A complementer is controlled by the A sign bit, A_S. When A_S = "1", A is complemented.

The truth table and Karnaugh maps for sign, overflow, and End Around Carry are shown in Figures 6 and 7. Note the use of B_S' from the exclusive-OR of Add/Sub and B_S. B_S' eliminates Add/Sub as a variable in the truth table. As an example of truth table generation, consider an n decade adder/subtracter where A_S = "0", B_S = "1", and Add/Sub = "0". B is in 9's complement form, $10^N - 1 - B$. Thus $A + (10^N - 1 - B) = 10^N - 1 + (A - B)$. There is no carry when $A \leq B$, and the sign is negative (sign = "1"). When A_S and B_S are opposite states and Add/Sub is a "0" (add mode), no overflow can occur (overflow = "0"). The other output states are determined in a similar manner (see Figure 6).

From the Karnaugh maps it is apparent that End Around Carry is composed of the two symmetrical functions S2 and S3 of three variables with $\overline{A_S}\ \overline{B_S'}\ \overline{C}_{out}$ as the center of symmetry. This is the definition of the majority logic function $M_3(ABC)$. Similarly the Sign is composed of the symmetrical functions S2(3) and S3(3) but with the center of symmetry translated to $A_S B_S'\ \overline{C}_{out}$. This is equivalent to the majority function $M_3(A_S B_S'\ \overline{C}_{out})$. Further evaluation of the maps and truth table reveal that Overflow can be generated by the exclusive-OR function of End Around Carry and Carry Out. This analysis results in a minimum device count consisting of one exclusive-OR package and one dual Majority Logic package to implement B_S', EAC, Sign and Overflow. The logic connections of these devices are shown in Figure 5.

The output sign, R_S, complements the result of the add/subtract operation when R_S = "1". This is required because the adder performs 9's complement arithmetic. Complementing, when R_S indicates the result is negative, restores sign and magnitude convention.

Several variations of the adder/subtracter are possible. For example, 9's complement is available at the output of the NBCD adders, and output complementers are eliminated if sign and magnitude output is not required.

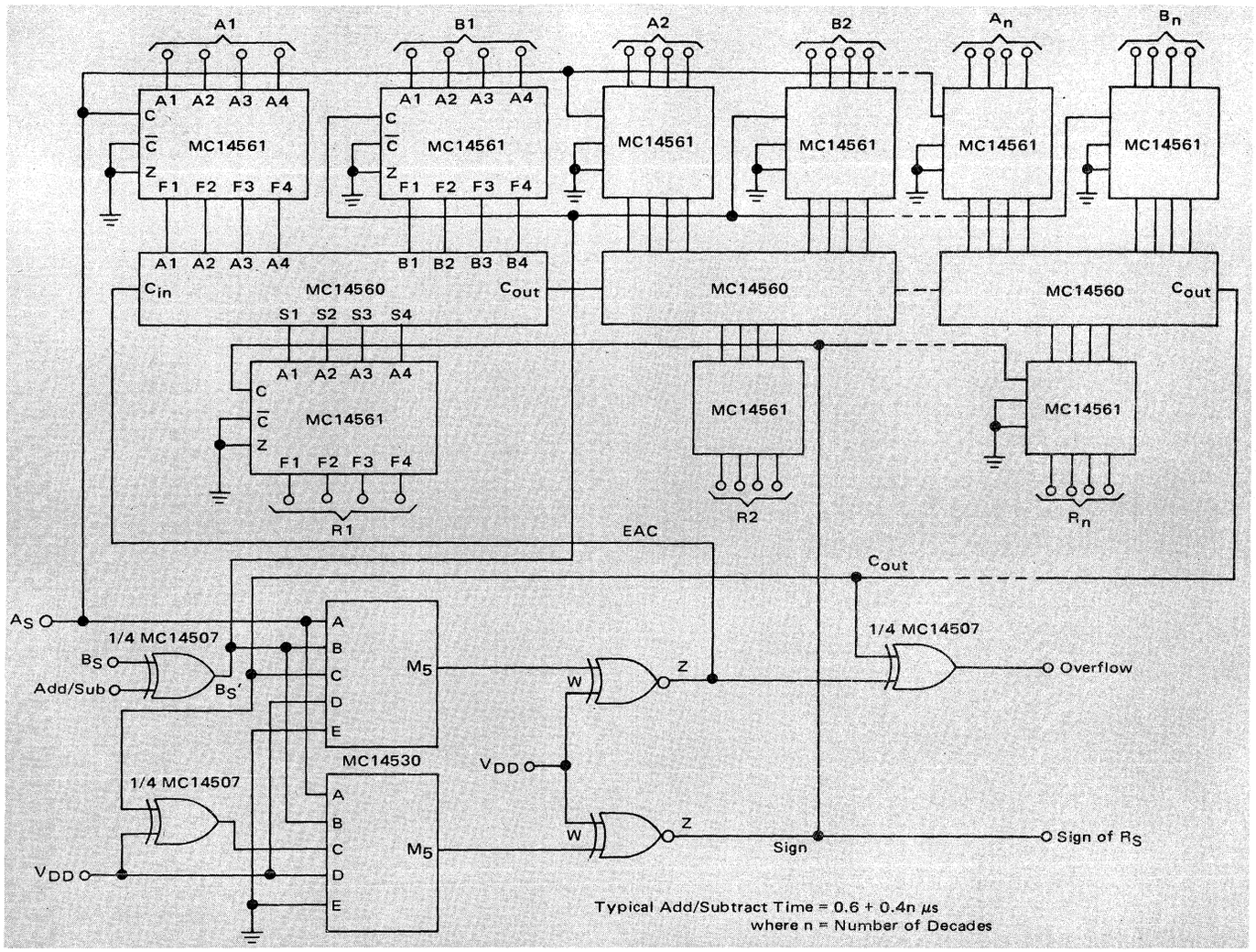


FIGURE 5 – Sign and Magnitude Adder/Subtracter with Overflow

INPUTS			Arithmetic Expression for R* (Result) (N = Number of Digits, 10^N = Modulus A, B, R are Positive Magnitudes)	OUTPUTS		
A_S "1" = Neg	B_S' "1" = Neg	C_{out} "1" = Carry		End Around Carry (EAC) "1" = EAC	Sign of R "1" = Negative	Overflow "1" = Overflow
0	0	0	R = A + B	No EAC ("0") because R is correct result.	Since A and B are positive signed, R is positive signed ("0").	When C_{out} = "0", there is no carry ($R < 10^N$) and thus no overflow ("0").
0	0	1				When C_{out} = "1", there is a carry ($R \geq 10^N$) and thus overflow ("1")
0	1	0	R = A – B = A + (10^N – 1 – B) = A – B + 10^N – 1	No EAC ("0") because 9's complement expression for R is correct result.	A ≤ B when C_{out} = "0"; thus sign of R must be negative ("1").	There is never an overflow when numbers of opposite sign are added.
0	1	1		EAC = "1" because expression for R is in error by 1.	A > B when C_{out} = "1"; thus sign of R must be positive ("0").	
1	0	0	R = B – A = B + (10^N – 1 – A) = B – A + 10^N – 1	No EAC ("0") because 9's complement expression for R is correct result.	B ≤ A when C_{out} = "0"; thus sign of R must be negative ("1").	
1	0	1		EAC = "1" because expression for R is in error by 1.	B > A when C_{out} = "1"; thus sign of R must be positive ("0").	
1	1	0	R = – A – B = (10^N – 1 – A) + (10^N – 1 – B) = – (A + B) + 2 x 10^N – 2	EAC = "1" because 9's complement expression for R is in error by 1.	Since A and B are negative signed, R is negative signed ("1").	When C_{out} = "0", there is no carry ($R < 10^N$) and (A + B) > 10^N – 1 indicating overflow ("1").
1	1	1				When C_{out} = "1", there is a carry ($R \geq 10^N$) and (A + B) ≤ 10^N – 1 indicating no overflow ("0").

*Output of Adders

FIGURE 6 – Truth Table Generation for EAC, Sign, and Overflow Logic

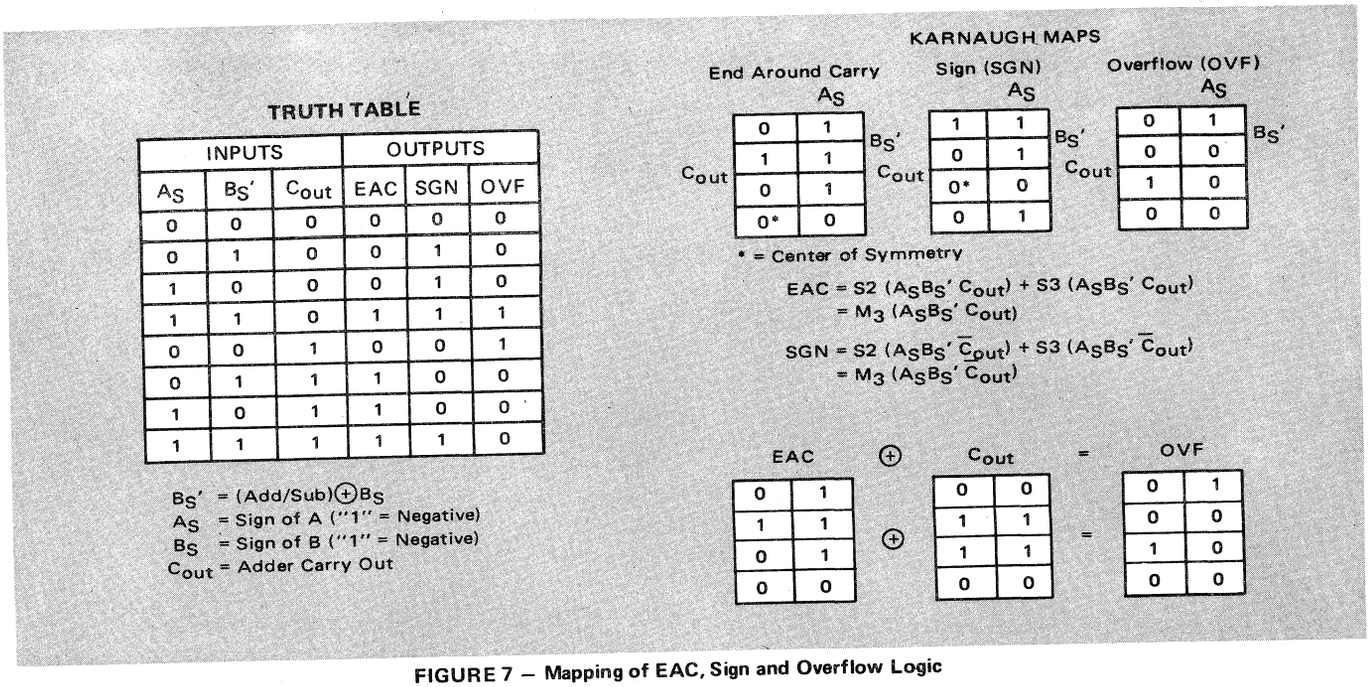


FIGURE 7 — Mapping of EAC, Sign and Overflow Logic

SUMMARY

The concepts of binary code representations for decimal numbers, addition, and complement subtraction were discussed in detail. Using the basic Adder and Complementer MSI blocks, adder/subtracters for both signed and unsigned numbers were illustrated with examples.

REFERENCES

1. Chu, Y.: *Digital Computer Design Fundamentals,* New York, McGraw-Hill, 1962.
2. *McMOS Handbook,* Motorola Inc., 1st Edition.
3. Beuscher, H.: *Electronic Switching Theory and Circuits,* New York, Van Nostrand Reinhold, 1971.
4. Garrett, L.: CMOS May Help Majority Logic Win Designer's Vote, *Electronics,* July 19, 1973.
5. Richards, R.: *Digital Design,* New York, Wiley-Interscience, 1971.

MOTOROLA Semiconductor Products Inc.

Printed in Switzerland