

Technical Newsletter No. 5

APPLIED SCIENCE DIVISION



IBM

APPLIED SCIENCE DIVISION

Technical Newsletter No. 5

August 1953

We wish to extend thanks to the authors of these papers for their cooperation in contributing to this interchange of technical information.

CONTENTS

1. A Model II Four Address Floating-Decimal Coding System 5
Dora W. Sweeney
Los Alamos, New Mexico

2. Boolean Notation for Selector Networks 25
Eric V. Hankam
Watson Scientific Computing Laboratory

3. Automatic Inversion of Fifteen-Station Weissinger Circulation Matrices . . 39
Murray L. Lesser
Northrop Aircraft, Incorporated

A MODEL II FOUR ADDRESS FLOATING-DECIMAL CODING SYSTEM

Dora W. Sweeney
Box 1663
Los Alamos, New Mexico

The control panels are wired for general-purpose use for all problems which can be reduced to a chain of arithmetic operations. Since the CPC operates normally at the rate of 150 cards per minute with a fixed amount of calculate-time between cards, the problem has been to design control panels which make maximum use of this calculate-time and still retain a simple and flexible coding system. The best use of the available calculate-time is made when there are two operations performed, one of them being addition. A single operation does not use all the calculate-time, and the time required for a double operation in which the operations are combinations of multiply and divide will in the majority of cases exceed the calculate-time.

A three-address coding system is a system in which the coder may address two numbers to read into the calculating unit and store the result of an operation between the two. A four-address coding system allows the coder to read three numbers into the calculating unit and store the result of the calculation, which now is an operation between two of the numbers and a further operation between the third number and the result of the first operation. Both systems, including several transcendental functions and some special operations, are now available to the coder in this set-up. The three factors are delivered over three channels called A, B, and X. The result is returned over channel C. Some operations on these factors, such as absolute values and changing sign before operations, are done in the accounting machine.

Floating-Decimal Calculations:

A floating-decimal system represents all numbers in "scientific" notation; as $x = x_0 \cdot 10^p$, where x_0 consists of an eight digit number* in the range, $1 \leq x_0 < 10$; the number is considered as an integer and seven decimals; p , the exponent of 10 is represented as $50 + p$ where p has the range $-50 \leq p < 50$ so that $100\pi = 3.1415927 \cdot 10^2$ is represented as 3141592752, and $\frac{1}{\pi} = 3.1830989 \cdot 10^{-1}$ is represented as 3183098949.

This system has a range of p large enough to handle most problems encountered. The

NOTE: Complete information for wiring the 605 is to be found in the several pages at the end of this article.

*A true floating decimal number contains a digit not zero in the high-order (left-hand) position.

ease of coding for this system rather than a fixed decimal system is apparent, as the coder does not need to know the magnitudes of numbers during the calculation for scaling purposes as the machine automatically scales the result at the end of each operation.

In multiplication; the x_n 's are multiplied and the exponents added. In division; the x_n 's are divided and the exponents differenced. In addition; the number with the smaller exponent is shifted to the right the same number of places as the difference of the exponents before it is added to the number with the larger exponent. There is normally no loss of figures in multiplication or division, but the addition of two similar numbers with opposite sign may occur with a consequent loss of the high order significant digits. The machine in this case will shift the remaining digits to the left and reduce the exponent to again put the number in floating decimal notation. The now right-hand digits will be zero. In the case where two identical numbers are subtracted causing a complete figure loss the result will be all zeros including the exponent. Rounding is automatic. The CPC design is such that the result of a division by zero is zero.

If the machine operates upon numbers which are not in true floating-decimal form correct results will be obtained except in division. If there are more high-order zeros in the denominator than in the numerator, the result will be zero.

Operations:

The only possible calculational operations are shown in Table I. One card time is defined as .4 seconds.

The arithmetic operations are virtually self-explanatory except for the design in which the order of the operations in column 3 of the instruction field take place. The three address codes perform only a single operation such as $A + B$, $A \cdot B$, or $A \div B$, and require only a single punch in the operations column. The four address codes require multiple punches as they perform two operations. The design is such that a 3 punch takes precedence over a 4 or a 1 punch, and a 4 punch takes precedence over a 1 punch. A Y punch will interchange the order of precedence of the operations, a 2 punch will interchange the order in which B and X are used in the calculation, and a 0 and a 2 punch will interchange the order in which A and X are used in the calculation. The wiring code gives the order of operations such that a capital letter means the first operation and a small letter stands for the second operation. A capital letter does not necessarily imply a second operation, but a small letter definitely means a first operation has preceded it. For example:

A means add the first number to the second number:

$A + B$, $A + X$ if I, or $X + B$ if i.

M means multiply the first number by the second number:

AB , AX if I, or XB if i.

D means divide the first number by the second number:

$A \div B$, $A \div X$ if I, or $X \div B$ if i.

a means add the result of the first operation to the third number.

m means multiply the result of the first operation by the third number.

d means divide the result of the first operation by the third number.

An example of the coding is as follows:

<u>Code Punch</u>	<u>Wiring Code</u>	<u>Operation</u>
1	A	$A + B$
3	M	AB
1, 3	M, a	$(AB) + X$
1, 2, 3	M, a, I	$(AX) + B$
0, 1, .	M, a, i	$(XB) + A$
Y, 1, 3	A, m	$(A + B) X$
Y, 1, 2, 3	A, m, I	$(A + X) B$
Y, 0, 1, 2, 3	A, m, i	$(X + B) A$

B and X interchange is coded by the symbol I, the code punch is a 2.

A and X interchange is coded by the symbol i, the code punch is a 0 and a 2.

There is no operation A, a.

The special operations which may be used only with the arithmetic operations are an X punch in column 3 and an X punch in column 7 of the instruction field. These instructions are respectively "Do Not Shift Left" and "Do Not Round", their wiring codes are N and n.

Do Not Shift Left (N)

This operation is useful in a variety of ways. It can be used to convert floating decimal numbers to a fixed decimal system. For example: if the coder punches the operation A and N, addresses one of the factors of the addition to read-in as 0000000051, and the other factor as the number to be converted to fixed decimal, all of the channel C results will be arranged as two integers and six decimals with the exponent 51. If any factor has an exponent greater than or equal to 51 the number will read-out on channel C unchanged. If any factor has an exponent less than 51 it will be shifted to the right until it is in the form of two integers and six decimals. If the numbers are shifted to the right they will be properly rounded.

The operation is also useful in checking the answers from the solution of several simultaneous equations. If the solutions are fed into the left-hand side of the equation and the operation N is used, and that result as well as the original right-hand side of the equations are printed together, the coder can tell at a glance the accuracy of the solution.

Another use of this order which may speed-up the rate of calculation is in the cross-product and sum of two columns of numbers. If the N operation is coded for all operations except the last, and if a loss of figures might occur in the individual sums, the number will not shift left with the consequent time loss, and the machine will only shift to the left after the last operation. This result will have the exponent of the largest cross-product.

Do Not Round (n)

This operation is used mainly for finding the integral part of a number. If the coder punches the operation A and n, and addresses one of the factors to read-in as 0000000057, and reads-in the other factor as the number of which the integral part is desired, the machine will shift off to the right the decimal part of the number and then shift the integral part back to the left without rounding. If the exponent is greater than or equal to 57 the number will be unchanged.

Transcendental Operations:

The transcendental operations are coded as 5, 6, 7, 8, 0 and 8, or 9 and will calculate respectively the square root of the absolute value of the number addressed on channel A, the exponential of the number addressed on channel B, the natural logarithm (\log_e) of the absolute value of the number addressed on channel B, the sine of the number addressed on channel B, the cosine of the number addressed on channel B, and the arctangent of the number addressed on channel B. Other numbers must be addressed to read-in on channels A and X of the last five operations. They are listed under the proper functions below.

Square Root of |A| :

Operation code punch 5; wiring code R.

The range of A is $10^{-50} \leq |A| < 10^{50}$.

The operation requires approximately three card times.

The error is not more than one or two in the eighth figure.

The coder does not have to address other numbers to read-in on channels B or X.

The method of computation is as follows. The number A is considered as $|x_0| \cdot 10^p$. If p is even the result on channel C is $\sqrt{|x_0|} \cdot 10^{p/2}$. If p is odd the result on channel C is $\sqrt{10|x_0|} \cdot 10^{\frac{p-1}{2}}$. The method of getting the square root is the Newton iteration technique:

$$y_{n+1} = \frac{1}{2} \left[\frac{z}{y_n} + y_n \right] \text{ where } z \text{ is } |x_0| \text{ or } 10|x_0|.$$

When y_{n+1} equals y_n , that result and the proper exponent is read-out onto channel C. The starting value for y_0 is generated in the machine as 99999999.

Exponential of B:

Operation code punch 6; wiring code E.

The range of B is $|B| < 50 \ln 10 = 115.12925$.

The operation requires approximately six card times.

The error is not more than one in the seventh figure.

The coder must address $\ln 10 = 2302585150$ to read-in on channel A and $\frac{1}{2} \ln 10 = 1151292550$ to read-in on channel X.

If the range of B is exceeded the exponent will be out of the range $-50 \leq p \leq 50$. The result on channel C will be correct except for the wrong exponent. For example, $e^{-150} = 7.1750960^{-68}$ will read-out as 7175095984 and $e^{150} = 1.3937096 \cdot 10^{65}$ will read-out as 1393709715.

The method of computation is as follows. B is positioned so that a division by $\ln 10$ will give only the integral part as the quotient. Define this quotient as i and the remainder as r. This splits B into two numbers such that $B = i \ln 10 + r$. Since $e^{i \ln 10} = 10^i$, then $e^B = e^{i \ln 10 + r} = 10^i \cdot e^r$. A further reduction is now made. The absolute value of r is now tested to see whether it is less than or greater than $\frac{1}{2} \ln 10$.

$$\text{If } |r| \leq \frac{1}{2} \ln 10, e^B = 10^i \cdot e^r. \text{ If } |r| > \frac{1}{2} \ln 10, e^B = 10^{i + \frac{1}{2}} \cdot e^{r - \frac{1}{2} \ln 10}$$

The argument of either of the above exponentials is now less than $\frac{1}{2} \ln 10$ in absolute value, and the series calculated is:

$$\sum_{n=0}^{17} \frac{x^n}{n!}$$

Logarithm of |B| :

Operation code punch 7; wiring code L.

The range of B is $10^{-43} < |B| < 10^{43}$.

The operation requires approximately nine card times.

The error is not more than one in the seventh figure.

The coder must address $\sqrt{10} = 3162277750$ to read-in on channel A and $\ln 10 = 2302585150$ to read-in on channel X.

If the range of B is exceeded the number will be calculated properly, but since the machine can only sense two integers, and any number greater than $43 \ln 10$ is greater than 100, the high order digit will be discarded, and the next eight digits will be read-out. For example, $\ln 9.8765432 \cdot 10^{47} = 110.511662$ will read-out as 1051166251.

The method of computation is as follows. Consider B as $x_0 \cdot 10^p$. Then:

if $x_0 \leq \sqrt{10}$

$$\ln (x_0 \cdot 10^p) = \ln x_0 + p \ln 10$$

$$\text{where } \ln x_0 = 2 \sum_{n=0}^{13} \frac{y^{2n+1}}{2n+1}, y = \frac{x_0 - 1}{x_0 + 1},$$

or if $x_0 > \sqrt{10}$

$$\mathcal{L}n(x_0 \cdot 10^p) = \mathcal{L}n\left(\frac{x_0}{\sqrt{10}}\right) + (p + \frac{1}{2})\mathcal{L}n10,$$

where

$$\mathcal{L}n \frac{x_0}{\sqrt{10}} = 2 \sum_{n=0}^{13} \frac{y^{2n+1}}{2n+1}, \quad y = \frac{x_0 - \sqrt{10}}{x_0 + \sqrt{10}}$$

Sine of B:

Operation code punch 8; wiring code S

The range of B is $10^{-7} < |B| < 10^5$. B in radians.

The operation requires approximately six card times.

The error is not more than one in the seventh decimal.

The coder must address $\pi = 3141592750$ to read-in on channel A and $\frac{1}{2}\pi = 1570596350$ to read-in on channel X.

If B is in the range $10^{-50} \leq |B| < 10^{-7}$, the result will be all zeros including the exponent. If B is in the range $10^5 \leq |B| < 10^{50}$, the machine will calculate, but the result will bear no relationship to $\sin B$.

The method of computation is as follows. B is positioned properly in the range of five integers and seven decimals according to the exponent of B. (If $|B| \leq 10^{-7}$, B is shifted out of the machine to the right so that all zeros result). Now a division by π will give only the integral part as the quotient. Define this quotient as i and the remainder as r. This splits B into two numbers such that $B = i\pi + r$. A further reduction is now made. The absolute value of r is tested to see whether it is less than or greater than $\frac{1}{2}\pi$ and i is tested to see whether it is even or odd. These tests determine in which quadrant B lies, and the argument y, obtained from r is:

First quadrant:	i even,	$ r \leq \frac{1}{2}\pi;$	$y = r.$
Second quadrant:	i even,	$ r > \frac{1}{2}\pi;$	$y = -r + \frac{ r }{r}\pi.$
Third quadrant:	i odd,	$ r \leq \frac{1}{2}\pi;$	$y = -r.$
Fourth quadrant:	i odd,	$ r > \frac{1}{2}\pi;$	$y = r - \frac{ r }{r}\pi.$

The argument, y, is now less than $\frac{1}{2}\pi$, and the series calculated is:

$$\sin B = \sum_{n=0}^8 (-)^n \frac{x^{2n+1}}{(2n+1)!}$$

Cosine of B:

Operation code punch 0 and 8; wiring code C.

The range of B is $10^{-7} < |B| < 10^5$, B is in radians.

The operation requires approximately six card times.

The error is not more than one in the seventh decimal.

The coder must address $\pi = 3141592750$ to read-in on channel A and $\frac{1}{2}\pi = 1570796350$ to read-in on channel X.

The method of computation is the same as for $\sin B$ except $\frac{1}{2}\pi$ is added to B before the division by π .

Arctangent of B:

Operation code punch 9; wiring code T.

The range of B is $10^{-7} < |B| < 10^{50}$.

The operation requires approximately nine card times.

The error is not more than one in the seventh decimal.

The coder must address $0.1 = 1000000049$ to read-in on channel A and $\frac{5}{2}\pi = 7853981650$ to read-in on channel X.

If B is in the range $10^{-50} \leq |B| < 10^{-7}$, the result will be all zeros including the exponent. If B is in the range $10^7 < |B| < 10^{50}$ the result will be $\frac{1}{2}\pi$.

The method of computation is as follows. The reciprocal of B is found. B and its reciprocal are tested to see which is less than or equal to 1.0 in absolute value. The smaller value is now shifted into the form of one integer and seven decimals. This value is now tested to determine whether it is less than or greater than 0.4 in absolute value. This will place B in one of four ranges, and the function calculated is as follows:

$$\begin{aligned}
 10^{-50} \leq |B| < 0.4 & \quad \arctan B = \frac{|B|}{B} \arctan |B|, \\
 0.4 < |B| \leq 1.0 & \quad \arctan B = \frac{|B|}{B} \left[\frac{1}{4}\pi + \arctan \frac{|B| - 1}{|B| + 1} \right], \\
 1.0 < |B| \leq 2.5 & \quad \arctan B = \frac{|B|}{B} \left[\frac{1}{4}\pi - \arctan \frac{1}{\frac{|B| - 1}{|B| + 1}} \right], \text{ and} \\
 2.5 < |B| \leq 10^{50} & \quad \arctan B = \frac{|B|}{B} \left[\frac{1}{2}\pi - \arctan \frac{1}{|B|} \right]
 \end{aligned}$$

The arguments above are now less than 0.43 and the series calculated is:

$$\sum_{n=0}^{13} (-)^n \frac{x^{2n+1}}{2n+1}$$

Other transcendental functions can be card programmed from the previous functions by making use of the following relationships:

The exponential and hyperbolic functions:

$$\begin{aligned}
 a^b &= e^{b \ell_{na}} \\
 \sinh y &= \frac{1}{2} [e^y - e^{-y}].
 \end{aligned}$$

$$\cosh y = \frac{1}{2} [e^y + e^{-y}] .$$

$$\tanh y = [e^{2y} - 1] / [e^{2y} + 1] .$$

The logarithmic and arc-hyperbolic functions:

$$\operatorname{arcsinh} y = \mathcal{L}n [y + \sqrt{y^2 + 1}] .$$

$$\operatorname{arccosh} y = \pm \mathcal{L}n [y + \sqrt{y^2 - 1}] .$$

$$\operatorname{artanh} y = \frac{1}{2} \mathcal{L}n [(1 + y) / (1 - y)] .$$

$$\mathcal{L}n(a + ib) = \frac{1}{2} \mathcal{L}n (a^2 + b^2) + i \operatorname{arc} \tan (b/a) .$$

The circular functions:

$$\tan y = \sin y / \cos y .$$

$$\cot y = \cos y / \sin y .$$

$$\sec y = 1 / \cos y .$$

$$\csc y = 1 / \sin y .$$

The arc-circular functions:

$$\operatorname{arcsin} y = \operatorname{arctan} [y / \sqrt{1 - y^2}] .$$

$$\operatorname{arccos} y = \operatorname{arctan} [\sqrt{1 - y^2} / y] .$$

$$\operatorname{arccot} y = \pi/2 - \operatorname{arctan} y .$$

TABLE I
Arithmetic Operation

(The operation in parentheses is the first operation)

Code Punch Col. 3	Operation	Wiring Code	Calculate Selector
1	$A + B$	A	4
3	AB	M	6
4	A/B	D	7
1, 3	$(AB) + X$	M, a	6, 12
1, 2, 3	$(AX) + B$	M, a, I	6, 12, 5, 11
0, 1, 2, 3	$(XB) + A$	M, a, i	6, 12, 2, 3
Y, 1, 3	$(A + B)X$	A, m	4, 13
Y, 1, 2, 3	$(A + X)B$	A, m, I	4, 13, 5, 11
Y, 0, 1, 2, 3	$(X + B)A$	A, m, i	4, 13, 2, 3
1, 4	$(A/B) + X$	D, a	7, 12
1, 2, 4	$(A/X) + B$	D, a, I	7, 12, 5, 11
0, 1, 2, 4	$(X/B) + A$	D, a, i	7, 12, 2, 3
Y, 1, 4	$(A + B)/X$	A, d	4, 14
Y, 1, 2, 4	$(A + X)/B$	A, d, I	4, 14, 5, 11
Y, 0, 1, 2, 4	$(X + B)/A$	A, d, i	4, 14, 2, 3

The above operations require one card time if there is no loss of figures necessitating a shift to the left.

Code Punch Col. 3	Operation	Wiring Code	Calculate Selector
3, 4	$(AB)/X$	M, d	6, 14
2, 3, 4	$(AX)/B$	M, d, I	6, 14, 5, 11
0, 2, 3, 4	$(XB)/A$	M, d, i	6, 14, 2, 3
Y, 3, 4	$(A/B) X$	D, m	7, 13
Y, 2, 3, 4	$(A/X) B$	D, m, I	7, 13, 5, 11
Y, 0, 2, 3, 4	$(X/B) A$	D, m, i	7, 13, 2, 3
0, 3	$(AB) X$	M, m	6, 13
0, 4	$(A/B)/X$	D, d	7, 14
0, 2, 4	$(X/B)/A$	D, d, i	7, 14, 2, 3

These operations will in the majority of cases require two card times; one card time otherwise.

(Table 1 continued)

Transcendental and Special Operations

Code Punch Col. 3	Operation	Requirements and Usage	Wiring Code	Calculate Selector
5	$\sqrt{ A }$	No read-in on B necessary	R	1, 8, 9
6	e^B	Read-in $\ell n 10$ on A and $1/2 \ell n 10$ on X	E	12, 15, 16
7	$\ell n B $	Read-in $\sqrt{10}$ on A and $\ell n 10$ on X	L	10, 13
8	$\sin B$	Read-in π on A and $\pi/2$ on X	S	11, 15, 16
0, 8	$\cos B$	Read-in π on A and $\pi/2$ on X	C	2, 11, 15, 16
9	$\arctan B$	Read-in $1/10$ on A and $5\pi/2$ on X	T	7, 17, 18
X	Do not shift left	Used only with arithmetic operations	N	16, 18
X in Col. 7	Do not round	Used only with arithmetic operations	n	Channel shift

CPC MODEL II 605 PLANNING CHART

The following pages show the planning charts for the 605 Control Panel. It should be specifically noted that the 605 must have "Modified Group Suppress" and also must be made to operate as if the electronic units located at I-5-U and I-7-T were removed.

Factor A is read into GS 1 & 2, GS4; B into FS 1 & 2, GS3; and X into FS 3 & 4, MQ. The eight significant digits of Factor A are read into GS 1 & 2 and the two digits representing the exponent of A are read into GS4. Similarly the eight significant digits of Factor B are read into FS 1 & 2, the exponent digits of B into GS3, the eight significant digits of X into FS 3 & 4 and its digits representing the exponent into MQ. Factor C is developed in the low order positions of the counter and read out from the counter. The exponents of A, B, and X are always read in positively.

The following abbreviations are used in the charts:

P - Plus Balance.
M - Minus Balance.
Z - Zero Balance.
NZ - Non-zero Balance.
Sup - Suppress without test.
GS#1APU & #1A - Group Suppress #1, A side.
GS#1BPU & #1B - Group Suppress #1, B side.
Similarly for Group Suppress #2, #3, and #4.

A - Add first.
M - Multiply first.
D - Divide first.
a - Add second.
m - Multiply second.
d - Divide second.
I - Interchange B and X.
i - Interchange A and X.
N - Do not shift left.
n - Do not round.
s - Shift left.
R - $\sqrt{|A|}$.
E - e^B .
L - $\ln B$.
S - Sin B.
C - Cosin B.
T - Arctan B.
t - Arctan B, second sweep
of reduction.
Series - Series calculation.
sls - Series last sweep

PANEL A

	READ OUT	READ IN	SHIFT	PROGRAM		COMMENT
	EXIT 1	EXIT 2	EXIT 3	SUPP	PU	
1	GS4(53)	Ctrl+	RptDelPU(1)	#1A	MDTR	
2	emit 5	Ctrl+(2)	in 2	#1A		
3	GS3(4)	Ctrl+(6)		#1A		
4	MQ(52)	GS3		#5	MDTR	
5	R&R(7)	GS4		#1A		
6	GS1&2(58)	Ctrl+	(8)	#1A		
7	FS1&2(9)	Divide			DT	
8	R&R	GS1&2(55)				
9	GS1&2(54)	Ctrl+	in 6			
10	MQ	GS1&2(55)	in 3		DT	
11	FS1&2(9)	Divide	Reset			
12	MQ	Ctrl+	in 2			
13	GS1&2(54)	Ctrl+	(8)	#7	MDTR	
14	emit 5	Half-add (62)	in 2	#7		
15	RO	GS1&2(55)	out 3	#7		
16	emit 5	Half-add (62)	in 2	#7	MDTR	
17	emit 4	Half-add (62)	in 3	#7		
18	RO	FS1&2(11)	out 4	#7		
19	GS1&2(54)	Ctrl-	in 3	#7	MDTR	
20	emit 1	ZT, Reset (59)	in 4	#1A		
21	FS1&2(9)	GS1&2(55)		#9		
22	GS4	Ctrl+			a	
23	emit 1	Ctrl+(6)		#10		
24	RO	GS4				
25	GS3	Ctrl-, BT			a	
26	GS3	GS4		P		
27	FS3&4(12)	FS1&2(10)				
28	GS1&2(54)	FS1&2		P	a	
29	FS3&4(56)	GS1&2(55)		P		
30	R&R	MQ				
31	MQ	Half-add	GS#3APU		a	
32	emit 4	Ctrl-, BT				
33	emit 4	Ctrl+	GS#2APU	P		
34	emit 3	Ctrl-, BT			a	
35	emit 3	Ctrl+	GS#1APU	P		
36	emit 1	Ctrl-, BT				
37	emit 1	ZT, Reset			a	
38	FS1&2	Ctrl+	in 3	P		
39	FS1&2	Ctrl+	in 2	M, NZ		
40	FS1&2	Ctrl+		M, Z	a	
41	R&R	FS1&2	out 4	#1A		
42	FS1&2	Ctrl+		#1A		
43	R&R	FS1&2	out 5	#2A	a	
44	FS1&2	Ctrl+		#2A		
45	GS1&2(54)	Ctrl+	in 3			
46	emit 5	Half-add (62)	in 2		AMDs	
47	RO	GS1&2	out 3			
48	emit 5	Half-add (62)	in 2			
49	emit 4	Half-add (62)	in 3		AMDs	
50	RO	FS1&2	out 4			
51	emit 1	FS3&4	in 6			
52	FS3&4	ZT, Reset	in 6		AMDs	
53	FS1&2	GS1&2	GS#3BPU	Z		
54	GS1&2	Half-add	BT			
55	emit 1	Ctrl-, BT	GS#1BPU	NZ	AMDs	
56	FS3 & 4	ZT, Reset	in 3			
57	RptDelPU(13)	Prog Rpt	RS#1PU	#12		
58	GS1&2	Ctrl+	in 3(14)	#13	AMDs	
59	GS4	Half-add		M		
60	emit 1	Half-add		#3A		

PANEL B

	READ OUT	READ IN	SHIFT	PROGRAM		COMMENT
	EXIT 1	EXIT 2	EXIT 3	SUPP	PU	
1	GS4(15)	Ctrl+	GS#2BPU	#16	Series sls	
2	emit 4(16)	Ctrl-, BT		#14		
3	emit 3(17)	Ctrl+	(18)			
4	R&R	GS4	Prog Rpt		Series sls	
5	GS1&2	Ctrl-(19)	GS#1APU	#17		
6	GS1&2	Ctrl+	GS#2APU	#18		
7	RO	MQ			M Series sls	
8	R&R	GS1&2(55)	out 6			
9	FS1&2(9)	Multiply			M Series sls	
10	GS1&2(54)	MQ	(20)			
11	R&R	GS1&2(55)	out 6		Series sls	
12	FS1&2(9)	Multiply				
13	GS1&2	Ctrl+	(20)		Series sls	
14	R&R	GS1&2	out 3 (21)	#22		
15	GS4	MQ	in 4	#22	Series sls	
16	GS1&2	Multiply		#23		
17	GS1&2	Ctrl-(22)	in 4	#22		
18	MQ	GS1&2	in 4	#22	Series sls	
19	emit 1	Ctrl-	GS#3APU	#1A		
20	R&R	GS4		#1A	s	
21	GS1&2	Ctrl+	in 4	#1A		
22	GS1&2(54)	Ctrl+	in 5	#24	A If not a, then M or D.	
23	emit 5	Half-add (62)	in 4	#24		
24	RO	GS1&2(55)	out 5	#24	A If not a, then M, or D.	
25	emit 5	Half-add (62)	in 4	#24		
26	emit 4	Half-add (62)	in 5	#24	A If not a, then M, or D.	
27	RO	FS1&2(10)	out 6	#24		
28	GS1&2(54)	Ctrl-	in 5	#24	A If not a, then M or D.	
29	emit 1	ZT, Reset	in 6	#24		
30	FS1&2(9)	GS1&2(55)		#25	A If not a, then M or D.	
31	GS4(57)	Ctrl+	GS#3APU			
32	emit 1	Ctrl+(6)		#10	A If not a, then M or D.	
33	GS3(24)	Ctrl+(25)		#29		
34	emit 5	Ctrl+(26)	in 2	#29	A If not a, then M or D.	
35	emit 1	Ctrl-		#28		
36	R&R	GS4			If not d, then A. If not a or d, then M or D.	
37	GS1&2(54)	Ctrl+		#29		
38	RO	MQ		#29	If not d, then A If not a or d, then M or D.	
39	R&R	GS1&2(55)	out 6	#29		
40	FS3&4(12)	Multiply		#29	If not d, then A If not a or d, then M or D.	
41	GS1&2(54)	MQ		#29		
42	R&R	GS1&2(55)	out 6	#29	If not d, then A If not A or d, then M or D.	
43	FS3&4(12)	Multiply		#29		
44	GS1&2(54)	Ctrl+		#29	R, sls	
45	GS1&2(54)	Ctrl+	in 3	#30		
46	R&R	GS1&2	out 4		Series	
47		RptDelDO	GS#1BPU	#2A		
48	RS#9&10PU	RptDelPU	RS#7&8PU	P		
49	R&R(43)	FS3&4(44)	out 3 (45)		R, sls	
50	emit 2	MQ	in 3	#20		
51	emit 1(46)	MQ	in 4 (21)	#11	R, sls	
52	GS3	MQ	in 2	#39		
53	FS3&4	Multiply	(3)	#19	R, sls	
54	FS1&2	Multiply		#22		
55	MQ	GS1&2	in 5	#15	R, sls	
56	RO(47)	FS3&4(48)	out 5(23)	#16		
57	FS1&2	Ctrl-, BT, Reset	in 5	#22	R, sls	
58	FS3&4	FS1&2	Prog Rpt	#21		
59	FS3&4	Ctrl+	in 3	#42	R, sls	
60	GS3	Ctrl+		#42		

PANEL C

	READ OUT	READ IN	SHIFT	PROGRAM		COMMENT
	EXIT 1	EXIT 2	EXIT 3	SUPP	PU	
1	GS4(53)	Ctrl+	GS#3APU			
2	GS3(4)	Ctrl-, BT	RptDelPU		ASet	
3	GS3(28)	Ctrl+(29)	(30)	#31		
4	R&R	GS3(31)				
5	GS1&2(54)	Ctrl+	GS#3BPU	#8	ASet	
6	FS1&2(9)	GS1&2(55)		#8		
7	R&R	FS1&2(10)		#8		
8	GS3(4)	Half-add			ASet	
9	emit 4	Ctrl-, BT				
10	emit 4	Ctrl+	GS#2APU	P		
11	emit 3	Ctrl-, BT			ASet	
12	emit 3	Ctrl+	GS#1APU	P		
13	emit 1	Ctrl-, BT				
14	emit 1	ZT, Reset			ASet	
15	FS1&2(9)	Ctrl+	in 5	P		
16	FS1&2(9)	Ctrl+	in 4	M, NZ		
17	FS1&2(9)	Ctrl+	in 3	M, Z	ASet	
18	R&R	FS1&2(10)	out 5	#1A		
19	FS1&2(9)	Ctrl+	in 2	#1A		
20	R&R	FS1&2(10)	out 5	#2A	ASet	
21	FS1&2(9)	Ctrl+		#2A		
22	R&R(32)	FS1&2	out 5	#33		
23	FS1&2	Ctrl+(33)	(34)	#32	SELt	
24	emit 4(49)	GS1&2(50)	in 6(51)	#35		
25	GS1&2	Divide	Prog Rpt	#34		
26	GS1&2	ZT, Reset	(34)	#19	SELt	
27	emit 1	MQ	in 5	#36		
28	MQ	GS1&2	in 4	#36		
29	GS3	Ctrl+	in 2	#39	SELt	
30	emit 5	Ctrl-	in 3	#39		
31	emit 5	Ctrl+	GS#4APU	#37		
32	R&R	GS3	RptDelPU	#39	SELt	
33	FS1&2	Half-add	in 5	#19		
34	GS1&2	FS1&2	Prog Rpt	#19		
35	GS1&2	Ctrl+	in 5	#40	SELt	
36	RO	FS1&2	out 6(35)	#40		
37	GS1&2	Ctrl-(60)	in 5(61)	#40		
38	GS1&2(54)	Ctrl+(36)	in 5	#41	DRLt Series	
39	FS1&2(37)	Divide	GS#1APU			
40	R&R	GS1&2(39)	(27)			
41	GS1&2(40)	Ctrl+	in 6(5)		DRLt Series	
42	MQ	GS1&2(39)	in 3			
43	FS1&2(38)	Divide	Reset			
44	MQ	Ctrl+	in 2		DRLt Series	
45	GS1&2(40)	Ctrl+	in 5			
46	RO	FS1&2		#34		
47	FS3&4	ZT, Reset		#34	SELt	
48	FS1&2	Ctrl+	in 5	#15		
49	MQ	Half-add	GS#3BPU	#34		
50	emit 1	Half-add	GS#3APU	#43	SELt	
51	emit 2	Divide		#44		
52	emit 1	Ctrl-, BT, Reset		#44		
53	RO	GS3		#15	SELt	
54	FS1&2	Ctrl+	in 5	#26		
55	FS1&2	Ctrl-	in 5	#27		
56	GS1&2	ZT	in 5	#38	SELt	
57	R&R	FS1&2	out 5(41)			
58	emit 2(42)	Ctrl+	in 2			
59	emit 7	Ctrl+	RS#5&6PU		SELt	
60	R&R	GS4	GS1&2RI			

CALCULATE SELECTORS

1 R	<ul style="list-style-type: none"> ● FS1&2 RI ● (54) ● (58) 	<ul style="list-style-type: none"> ● Ctr- ● ZT, Reset ● (59) 	<ul style="list-style-type: none"> ● Ctr+ ● Ctr- ● (60) 	<ul style="list-style-type: none"> ● in 4 ● in 5 ● (61) 	<ul style="list-style-type: none"> ○ ○ ○
2 C,i	<ul style="list-style-type: none"> ● FS3&4 RO ● emit 4 ● (49) 	<ul style="list-style-type: none"> ● Ctr+ ● GS1&2 RI ● (50) 	<ul style="list-style-type: none"> ○ ● in 6 ● (51) 	<ul style="list-style-type: none"> ○ ● Sup ● RS3-2C 	<ul style="list-style-type: none"> ● GS4 RO ● MQ RO ● (52)
3 i	<ul style="list-style-type: none"> ● MQ RO ● GS4 RO ● (53) 	<ul style="list-style-type: none"> ● MQ RI ● GS4 RI ● CS4-4T 	<ul style="list-style-type: none"> ● FS3&4 RO ● GS1&2 RO ● (54) 	<ul style="list-style-type: none"> ● FS3&4 RI ● GS1&2 RI ● (55) 	<ul style="list-style-type: none"> ● GS1&2 RO ● FS3&4 RO ● (56)
4 A	<ul style="list-style-type: none"> ○ ● Sup ● #24 	<ul style="list-style-type: none"> ● (53) ● GS4 RO ● (57) 	<ul style="list-style-type: none"> ● CS5-5C ● GS3 RO ● (24) 	<ul style="list-style-type: none"> ● CS3-2C ● Ctr+ ● (29) 	<ul style="list-style-type: none"> ● PSF#4&5 ● CS6-5C ● CS7-4N
5 I	<ul style="list-style-type: none"> ● MQ RO ● GS3 RO ● CS8-2N 	<ul style="list-style-type: none"> ● FS3&4 RO ● FS1&2 RO ● (9) 	<ul style="list-style-type: none"> ● FS3&4 RI ● FS1&2 RI ● (10) 	<ul style="list-style-type: none"> ● MQ RI ● GS3 RI ● (31) 	<ul style="list-style-type: none"> ● GS3 RO ● (52) ● CS4-3T
6 M	<ul style="list-style-type: none"> ● Ctr- ● Ctr+ ● (2) 	<ul style="list-style-type: none"> ○ ○ ○ 	<ul style="list-style-type: none"> ● PSF#1 ● CS10-3C ● 	<ul style="list-style-type: none"> ● <u>7-12B</u> 4-way wire ○ ● 	<ul style="list-style-type: none"> ● PSF#3 CS12-5C ○ ● CS4-5N
7 D,T	<ul style="list-style-type: none"> ● Ctr- ● Ctr+ ● CS10-5N CS8-3N 	<ul style="list-style-type: none"> ● in 5 ○ ● (8) 	<ul style="list-style-type: none"> ● NZ ● Z ● #10 	<ul style="list-style-type: none"> ● PSF#2 CS17-4C ● CS4-5C ● CS15-5N 	<ul style="list-style-type: none"> ● out 4 ● out 3 ● (21)
8 R	<ul style="list-style-type: none"> ○ ● RptDelPU ● (1) 	<ul style="list-style-type: none"> ● emit 2 ● CS5-1C ● (4) 	<ul style="list-style-type: none"> ● Divide ● CS10-5N CS7-1C ● (6) 	<ul style="list-style-type: none"> ● PX#5 ● R&R ● (7) 	<ul style="list-style-type: none"> ● NZ ○ CS14-3C ● #40
9 R	<ul style="list-style-type: none"> ● #7 ○ ● Sup 	<ul style="list-style-type: none"> ● Z ● RS5-2C ● #41 	<ul style="list-style-type: none"> ● FS3&4 RI ● (55) ● (39) 	<ul style="list-style-type: none"> ● FS3&4 RO ● (54) ● (40) 	<ul style="list-style-type: none"> ● PSF#6&7 ○ ● Prog Source

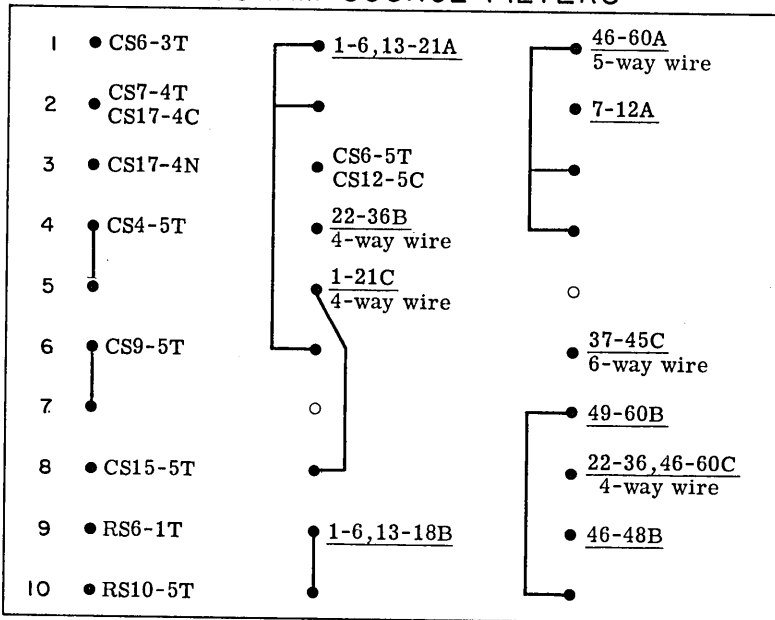
CALCULATE SELECTORS

10 L	● SF#7 RS7-2T	● Ctr+	● <u>22-36,46-60C</u> 4-way wire	● <u>37-45C</u> 6-way wire	● Ctr-
	● #39	● Ctr-	○	○	● CS7-1C CS8-3N
	● Sup	● CS12-2N	● CS6-3N	● RS10-3N	● (22)
11 I,S	● Sup	● FS1&2 RO	○	● P	● M
	● #1A	● (56)	● #44	● Sup	● Sup
	● #5	● (12)	● Sup	● #26	● #27
12 E,a	○	● Ctr+	○	● Sup	● <u>22-45A</u>
	● #15	● CS10-2C	● #1B	● #1A	● <u>22-36B</u> 4-way wire
	● Sup	● (19)	● RS5-2T ● RS8-2N	● RS8-3N	● CS6-5T ● PSF#3
13 L,m	● Ctr-	● #30	● NZ	● Z	○
	● Ctr+	● #29	● #19	● #19	○
	● (26)	● #28	● #36	● #37	○
14 d	● Ctr-	○	● Sup	● (12)	● <u>37-45C</u> 6-way wire
	● Ctr+	● #28	● RS5-1C	● FS1&2 RO	● <u>37-45B</u>
	● (25)	● Sup	● CS8-5N	● RS5-5N	● <u>22-36B</u> 4-way wire
15 S,E	● #19	● Sup	● emit 4	● #3B	● PSF#8
	● #34	○ P	● (4)	○	CS7-4C
	● Sup	● #8	● RS2-1N	● #32	● RS10-5N
16 S,E,N	● (18)	● emit 1	● Z	● Sup	○
	○	● emit 2	● Sup	● M,NZ	● Z
	● (20)	● (42)	● #43	● CS18-4T	CS18-5T
17 T	● PX#2	● Sup	● PX#10	○	● #3B
	● (10)	● Z	● in 3	● PSF#3	○
	● (11)	● SF#5	● (14)	● PSF#2 ● CS7-4T	● RS8-3T
18 T,N	● #3A	● #4A	● (3)	● CS16-4C	● CS16-5C
	● #19	○	● RS8-5T	● M,NZ	● Z
	● #20	● #11	● PX#3	● #12	● #13

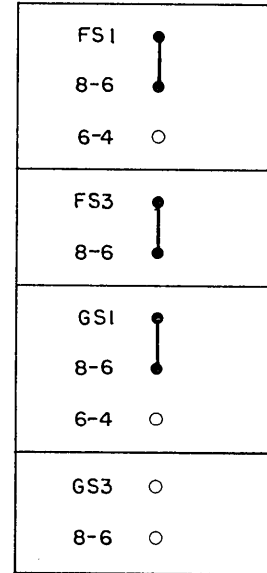
REPEAT SELECTORS

1 s	<ul style="list-style-type: none"> ● <u>19-21B</u> ● RS4-2C ● Prog Source 	<ul style="list-style-type: none"> ● RptDelDO ● RptDelPU ● (13) 	<ul style="list-style-type: none"> ● <u>46-60A</u> 5-way wire ● RS4-3C ● Prog Source 	<ul style="list-style-type: none"> ○ ○ ○ 	<ul style="list-style-type: none"> ○ ○ ○
2 t	<ul style="list-style-type: none"> ● emit 1 ● CS15-3C ● (28) 	<ul style="list-style-type: none"> ○ ● GS#3BPU ● (30) 	<ul style="list-style-type: none"> ○ ● P ● #31 	<ul style="list-style-type: none"> ● RO ● R&R ● (32) 	<ul style="list-style-type: none"> ● GS3 RI ● Ctr+ ● (33)
3 t	<ul style="list-style-type: none"> ● in 6 ○ ● (34) 	<ul style="list-style-type: none"> ○ ● #35 ● CS2-4C 	<ul style="list-style-type: none"> ● Z ● #19 ● RS5-2N 	<ul style="list-style-type: none"> ● PX#9 ● out 6 ● (35) 	<ul style="list-style-type: none"> ● Ctr- ● (26) ● RS5-3N
4 t	<ul style="list-style-type: none"> ● out 3 ● out 5 ● (41) 	<ul style="list-style-type: none"> ● <u>1-21C</u> 4-way wire ● RS6-1C ● RS1-1N 	<ul style="list-style-type: none"> ● <u>37-45C</u> 6-way wire ● RS6-4C ● RS1-3N 	<ul style="list-style-type: none"> ● <u>22-36, 46-60C</u> 4-way wire ○ ○ ● Prog Source 	<ul style="list-style-type: none"> ○ ○ ○
5 Series	<ul style="list-style-type: none"> ● Sup ● CS14-3N 	<ul style="list-style-type: none"> ● CS12-3C RS8-2N ● RS3-3C ● CS9-2N 	<ul style="list-style-type: none"> ● Ctr+ ● RS3-5C ● (36) 	<ul style="list-style-type: none"> ● GS1&2 RO ● (37) 	<ul style="list-style-type: none"> ● GS4 RO ● CS14-4C ● (38)
6 Series	<ul style="list-style-type: none"> ● PSF#9 ● RS10-5C ● RS4-2N 	<ul style="list-style-type: none"> ● out 2 ○ ● (27) 	<ul style="list-style-type: none"> ○ ● in 6 ● (5) 	<ul style="list-style-type: none"> ● <u>7-12B</u> 4-way wire ● RS10-3C ● RS4-3N 	<ul style="list-style-type: none"> ● <u>37-45C</u> 6-way wire ○ ● Prog Source
Series Last 7 Sweep sls	<ul style="list-style-type: none"> ● GS3 RO ● GS4 RO ● (15) 	<ul style="list-style-type: none"> ● SF#7 CS10-1T ○ ● #14 	<ul style="list-style-type: none"> ● emit 1 ● emit 4 ● (16) 	<ul style="list-style-type: none"> ● #15 ○ ● #16 	<ul style="list-style-type: none"> ● emit 5 ● emit 3 ● (17)
8 sls	<ul style="list-style-type: none"> ● in 2 ○ ● (18) 	<ul style="list-style-type: none"> ● #20 ● CS12-3C RS5-2T ● #17 	<ul style="list-style-type: none"> ● CS17-5C ● CS12-4C ● #18 	<ul style="list-style-type: none"> ● #22 ○ ● Sup 	<ul style="list-style-type: none"> ● CS18-3N ● R&R ● (43)
9 sls	<ul style="list-style-type: none"> ● MQ RI ● FS3&4 RI ● (44) 	<ul style="list-style-type: none"> ● RS#1PU ● out 3 ● (45) 	<ul style="list-style-type: none"> ● emit 1 ● emit 5 ● (46) 	<ul style="list-style-type: none"> ● GS1&2 RO ● RO ● (47) 	<ul style="list-style-type: none"> ● Ctr+ ● FS3&4 RI ● (48)
10 sls	<ul style="list-style-type: none"> ● in 4 ● out 5 ● (23) 	<ul style="list-style-type: none"> ● Sup ● P ● #21 	<ul style="list-style-type: none"> ● <u>7-12B</u> 4-way wire ● CS10-4C ● RS6-4N 	<ul style="list-style-type: none"> ● Sup ● M ● #42 	<ul style="list-style-type: none"> ● PSF#10 ● CS15-5C ● RS6-1N

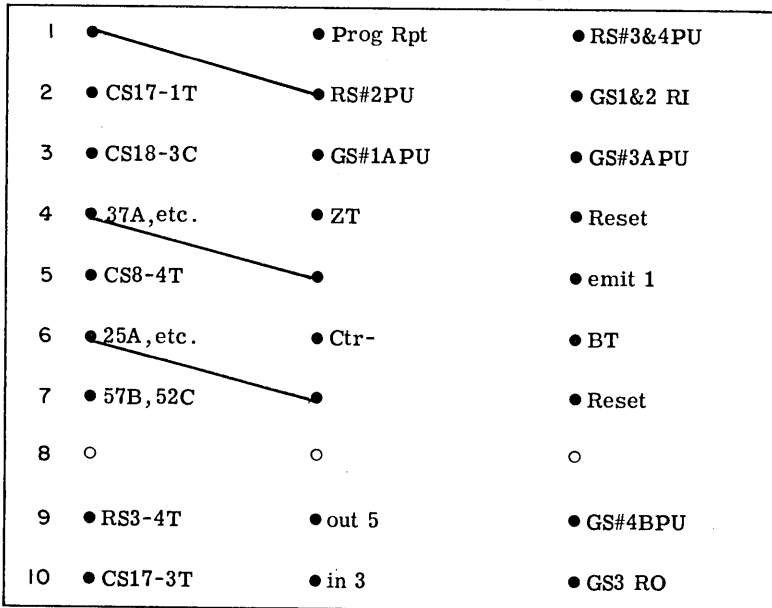
PROGRAM SOURCE FILTERS



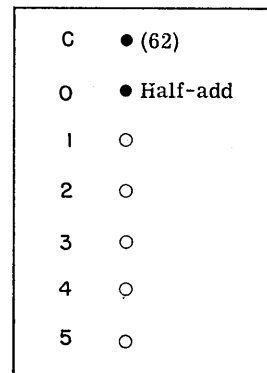
STORAGE ASSIGNMENT



PROGRAM EXPANSION



CHANNEL SHIFT



SUPPRESSION FILTERS

1	2	3	4	5
IN • NZ	• Z	○	○	• CS17-2C
IN • M	• M	○	○	• #7
OUT • 39A, etc.	• 40A, etc.	○	○	• #9
6	7	8	9	10
IN • #22	• RS7-2T • CS10-1T	• Z	○	• #34
IN • #19	• #32	• #24	○	• #32
OUT • #23	• #33	• #25	○	• #38

	CS	RS	SELECTED OPERATION
(1)	8		none(R), RptDelPU other.
(2)	6		Ctr-(M), Ctr+ other.
(3)	18		PX#3(T), none other.
(4)	8		emit 2(R), MQ RO(I), GS3 RO other.
(5)		6	none(Series), in 6 other.
(6)	8		Divide(R), Ctr-(D), Ctr-(T), Ctr+ other.
(7)	8		PX#5(R), R&R other.
(8)	7		in 5(D), in 5(T), none other.
(9)	5		FS3&4 RO(I), FS1&2 RO other.
(10)	5		FS3&4 RI(I), FS1 & 2 RI other.
(11)	17		PX #2 (T) (10) other
(12)	11		FS1&2 RO(I), GS1&2 RO(i), FS3&4 RO other.
(13)		1	RptDelDO(s), RptDelPU other.
(14)	17		PX#10(T), in 3 other.
(15)		7	GS3 RO(sls), GS4 RO other.
(16)		7	emit 1(sls), emit 4 other.
(17)		7	emit 5(sls), emit 3 other.
(18)		8	in 2(sls), none other.
(19)	12		Ctr+(L), Ctr+(E), Ctr- other.
(20)	16		(18) (S), (18) (E), none other.
(21)	7		out 4(T), out 3 other.
(22)	10		Ctr-(T), Ctr-(L), Ctr+ other.
(23)		10	in 4(sls), out 5 other.
(24)	4		MQ RO (A), GS3 RO(I), GS4 RO(i), GS3 RO other.
(25)	14		Ctr-(d), Ctr+ other.
(26)	13		Ctr-(m), Ctr-(1), Ctr+ other.
(27)		6	out 2(Series), none other.
(28)		2	emit 4(S), emit 4(E), emit 1(t), (4) other.
(29)	4		GS4 RI(A), MQ RI(i), Ctr+ other.
(30)		2	none(t), GS#3 BPU other.
(31)	5		MQ RI(I), GS3 RI other.
(32)		2	RO(t), R&R other.
(33)		2	GS3 RI(t) Ctr+ other.
(34)		3	in 6(t), none other.
(35)		3	PX#9(t), out 6 other.
(36)		5	Ctr-(L), Ctr-(t), Ctr+(Series), Ctr+ other.
(37)		5	GS1&2 RO(Series), (12)(d), FS1&2 RO other.
(38)		5	GS4 RO(Series), (12)(d), FS1&2 RO other.
(39)	9		FS3&4 RI(R), FS3&4 RI(i), GS1&2 RI other.
(40)	9		FS3&4 RO(R), FS3&4 RO(i), GS1&2 RO other.
(41)		4	out 3(t), out 5 other.
(42)	16		emit 1(S), emit 1(E), emit 2 other.
(43)		8	PX#3(sls), none(T), R&R other.
(44)		9	MQ RI(sls), FS3&4 RI other.
(45)		9	RS#1PU(sls), out 3 other.
(46)		9	emit 1(sls), emit 5 other.
(47)		9	GS1&2 RO(sls), RO other.
(48)		9	Ctr+(sls), FS3&4 RI other.
(49)	2		FS3&4 RO(C), emit 4 other.
(50)	2		Ctr+(C), GS1&2 RI other.
(51)	2		none(C), in 6 other.
(52)	2		GS4 RO(i), MQ RO other.
(53)	3		MQ RO(i), GS4 RO other.
(54)	3		FS3&4 RO(i), GS1&2 RO other.
(55)	3		FS3&4 RI(i), GS1&2 RI other.
(56)	3		GS1&2 RO(i), FS3&4 RO other.
(57)	4		(53)(A), GS4 RO other.
(58)	1		FS1&2 RI(R), GS1&2 RO other.
(59)	1		Ctr-(R), ZT, Reset other.
(60)	1		Ctr+(R), Ctr- other.
(61)	1		in 4(R), in 5 other.
(62)			none(n), Half-add other. (See Channel Shift)

A - Add first.
M - Multiply first.
D - Divide first.
a - Add second.
m - Multiply second.
d - Divide second.
I - Interchange B and X.
i - Interchange A and X.
N - Do not shift left.
n - Do not round.
s - Shift left.
R - $\sqrt{|A|}$.
E - e^B .
L - $\ln B$.
S - Sin B.
C - Cosin B.
T - Arctan B.
t - Arctan B, second sweep of reduction.
Series - Series calculation.
sls - Series last sweep

The Calculate or Repeat Selector referred to above is the source of the operation. It may be wired to the hubs of other selectors.

	CS	RS	SF	SUPPRESSION TYPES
#5	11			Sup(I), #1A other.
#6	8			NZ(R), none other.
#7	9			Sup(R), none other.
#8	15			Sup(S), Sup(E), P other.
#9			5	#7 & Sup(T), Z other.
#10	7			NZ(D), Z other.
#11	18			#4A(T), none other.
#12	18			Sup(N), M, NZ other.
#13	18			none(N), Z other.
#14		7		none; except if sls, Sup(L), none other.
#15	12			none(E), Sup other.
#16		7		#15(sls), none other.
#17		8		#20(sls), none(E), #1B other.
#18		8		Sup(E), #1A other; except if sls, #3B(T), none other.
#19	15			Sup(S), Sup(E), none other.
#20	18			#3A(T), #19 other.
#21		10		Sup(sls), P other.
#22		8		Sup(sls), none other.
#23			6	#22 & #19.
#24	4			none(A), Sup other.
#25			8	Z & #24.
#26	11			P(S), Sup other.
#27	11			M(S), Sup other.
#28	14			none(d), Sup other.
#29	13			none(m), #28 other.
#30	13			Sup(m), none other.
#31		2		none(t), P other.
#32	15			#3B(S), #3B(E), none other.
#33			7	#32 & Sup(L), none other.
#34	15			none(S), none(E), Sup other.
#35		3		none(C), none(t), Sup other.
#36	13			NZ(L), #19 other.
#37	13			Z(L), #19 other.
#38			10	#32 & #34.
#39	10			none(L), Sup other.
#40	8			Sup(d), Sup(Series), Z(t), NZ(R), #19 other.
#41	9			Z(R), Z(t), #19 other; except if Series, none (E), #1B other.
#42		10		Sup(sls), M other.
#43	16			Z(S), Z(E), Sup other.
#44	11			none(S), Sup other.

The Calculate or Repeat Selector or Suppression Filter referred to above is the source of the suppression. It may be wired to the hubs of other selectors.

P - Plus Balance.

M - Minus Balance.

Z - Zero Balance.

NZ - Non-zero Balance.

Sup - Suppress without test.

GS#1APU & #1A - Group Suppress #1, A side.

GS#1BPU & #1B - Group Suppress #1, B side.

Similarly for Group Suppress #2, #3, and #4.

BOOLEAN NOTATION FOR SELECTOR NETWORKS

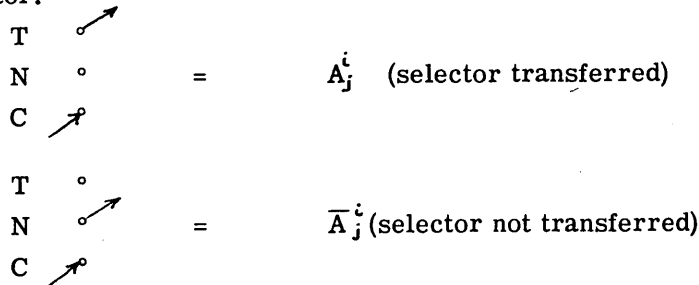
Eric V. Hankam

Watson Scientific Computing Laboratory
International Business Machines Corporation





Let A, B, C, ... represent types of selectors, such as Pilot selectors, Co-selectors, Calculator selectors, etc. Denote by A_j^i the j^{th} position (point) of selector number i of type A.

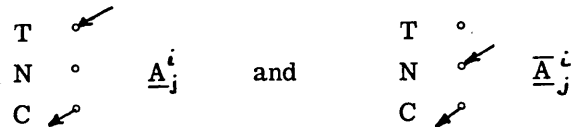
1. Single Selector

We adopt the following notation to indicate the "transferred" and "normal" condition of a selector.



In Logic \bar{A} denotes the negation of A; other notations for \bar{A} are A' or $\sim A$.

The above representation, while sufficient to outline a problem involving selectors, does not describe a wiring diagram uniquely, since  is equivalent to  and  is equivalent to . Whenever desirable to differentiate between these two cases, we can denote circuits where the direction of the impulse is toward the C-hub of the selector by:



where, logically, $\underline{A}_j^i = A_j^i$ and $\bar{\underline{A}}_j^i = \bar{A}_j^i$.

It should be noted that the numbering (i) of the selectors as well as the numbering (j) of the individual points of a multiple position selector is arbitrary. For example, two 5-position Co-selectors coupled to a 2-position Pilot selector form a twelve position selector group which we may call A^1 . The numbering (j) of the twelve positions is then arbitrary.

II Combination of Selectors

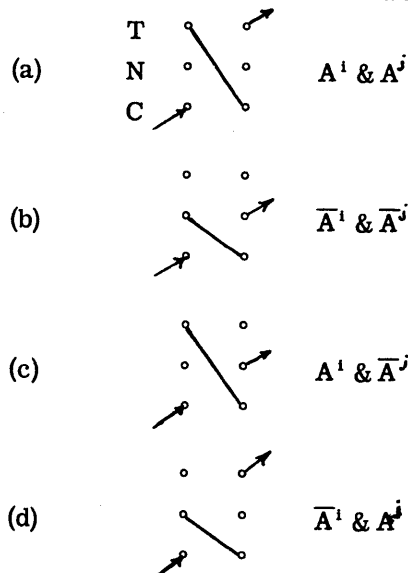
Consider two selectors A^i and A^j . For simplicity we shall omit subscripts whenever dealing with single position selectors. Furthermore, whenever convenient, we shall write A, B, C, ... to distinguish between different selectors even though the selectors may be of the same type.

The path of each impulse through a selector network can be traced by means of the logical connectives "not", "and" and "or". We shall employ the following symbolic notation:

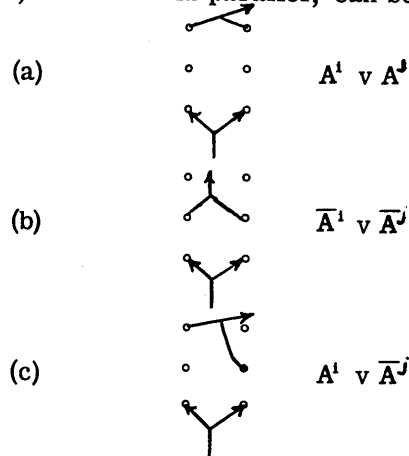
\bar{A}	denotes	"not" A
A & B	denotes	A "and" B
A v B	denotes	A "or" B

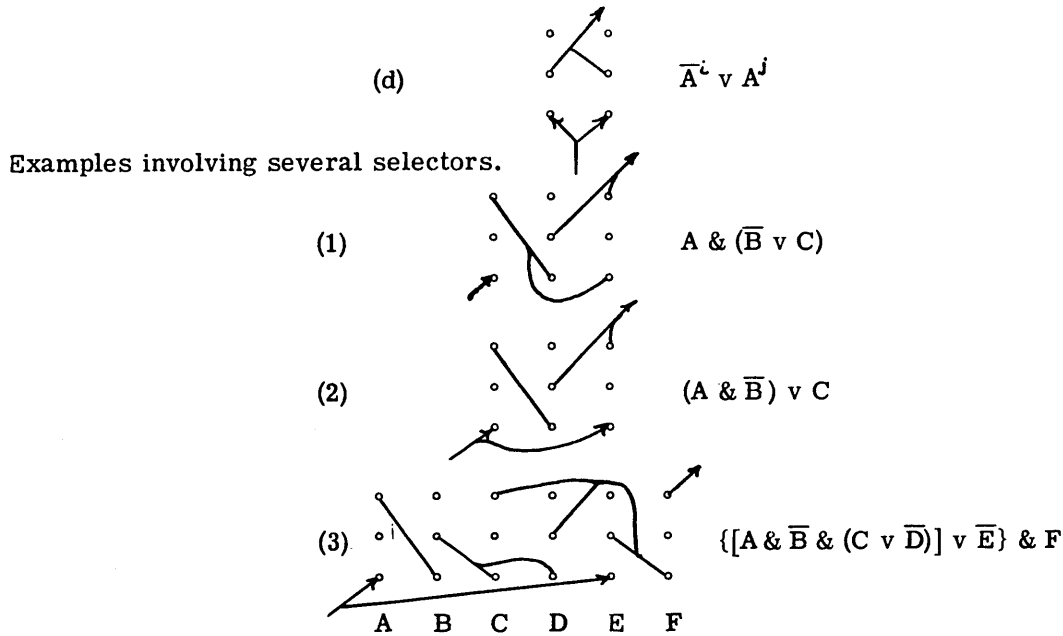
Any combination of the above expressions, such as $(A \& B) v C$, is called a Boolean polynomial. $A = B$ means that A and B are logically equivalent.

The symbolic representation for two selectors A^i and A^j , connected in series, is:

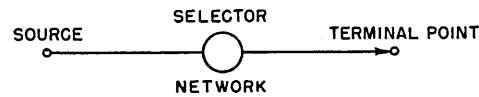


Two selectors A^i and A^j , connected in parallel, can be described as:

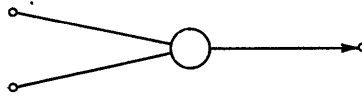




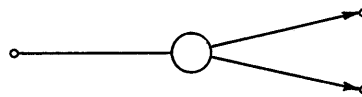
It should be observed that in the above notation a Boolean polynomial corresponds to the path of a single impulse through a chain of selectors; i. e. a single source pulse is entered into the selector (indicated by an arrow pointing toward a selector hub), wired through the selector network, and is available as a single terminal pulse out of the selector (indicated by an arrow pointing away from a selector hub).



In case several source pulses are wired through a selector network into one terminal point,



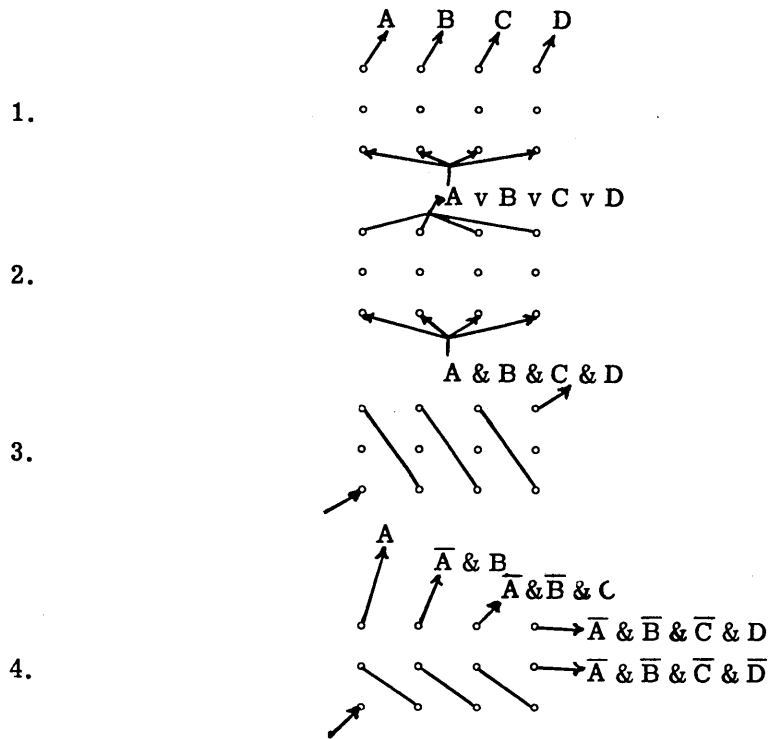
or one source pulse is wired through a selector network into several terminal points,



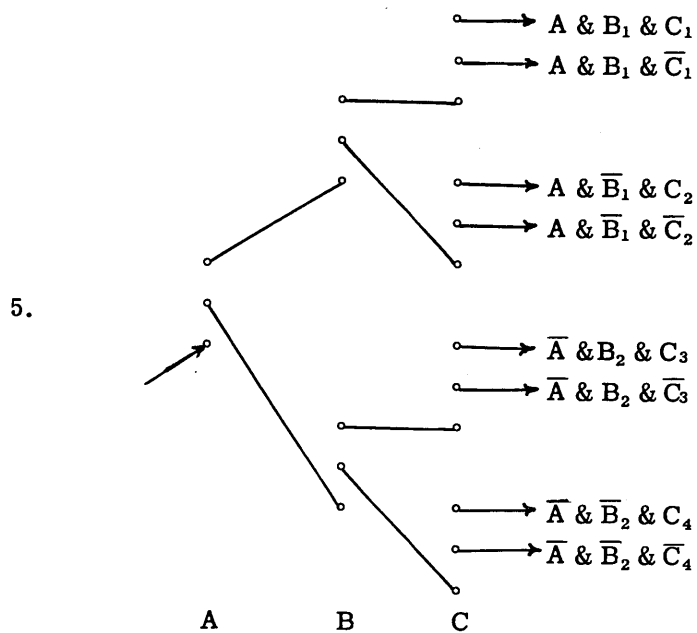
or several source pulses are wired through a selector network into several terminal points, each impulse has to be traced through separately. Thus, each path is described separately by a Boolean polynomial.

III Examples

In this section the method of Boolean notation will be further illustrated by a selection of examples that occur frequently in wiring diagrams.

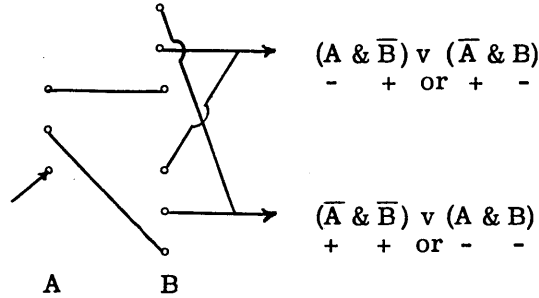


In the subsequent examples a selector may have several positions. In accordance with previous notation, subscripts will be used for selectors with more than one position.

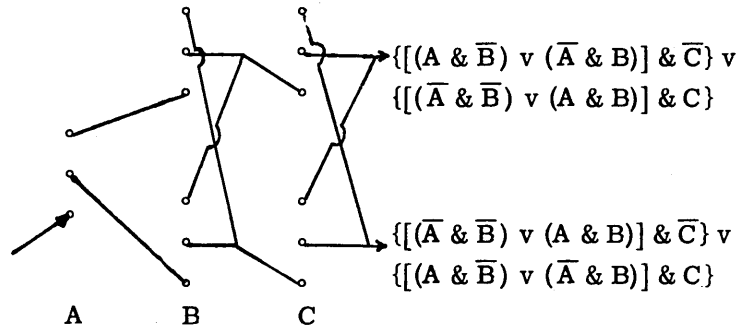


Note: Subscripts can be omitted when outlining a problem, but must be included when outlining a wiring diagram. A^1 , A^2 and A^3 should be used in place of A, B and C whenever the selectors are of the same type.

6. Algebraic signs of two quantities.

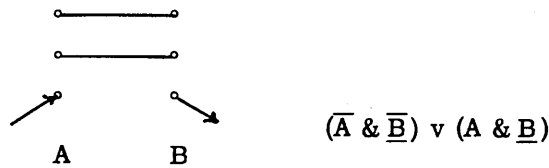


7. Algebraic signs of three quantities.

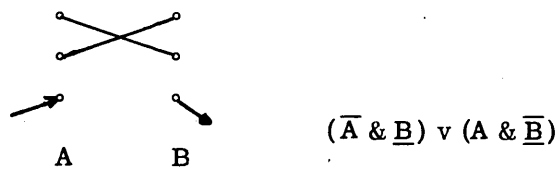


Examples in logic.

8. Equality (A if and only if B)

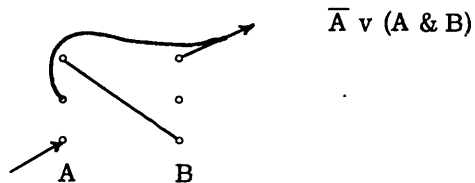


9. Inequality (A or B but not both)

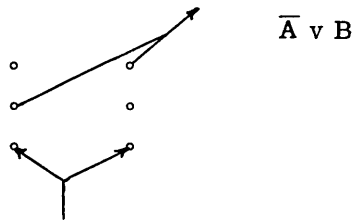


Note: Algebraic signs can also be determined by means of examples 8 and 9.

10. Implication (If A then B)



11. By Boolean algebra* $\bar{A} \vee (A \& B) = (\bar{A} \vee A) \& (\bar{A} \vee B) = I \& (\bar{A} \vee B) = \bar{A} \vee B$. Therefore, example 10 can be represented by the equivalent circuit:



Since a familiarity with Boolean algebra will aid the reader with the analysis of selector circuits, the fundamental laws of that algebra have been compiled in the next section.

IV Boolean Algebra

This section contains a list of definitions, axioms and theorems encountered in Boolean algebra.

Reference: Birkhoff and MacLane, *A Survey of Modern Algebra*, Chapter XI, *Algebra of Classes*.

Consider sets of elements A, B, C, \dots . Let I be the set that consists of the totality of elements contained in A, B, C, \dots , and denote the empty set, which contains no element, by O .

Definitions:

1. Inclusion relation

$A \subset B$ (A is contained in B) means that every element of A is also in B . In particular, A may be equal to B , i. e. A and B may consist of the same elements.

2. Binary operations

a) $A \& B$ (the intersection of A and B) is defined as the set of all elements contained in both A and B .

b) $A \vee B$ (the union of A and B) is defined as the set of all elements in either A or B or both.

3. Unary operation

A' (the complement of A) is defined as the set of all elements not in A . In sections I to III, A' was denoted by \bar{A} . However, in this section it will be more convenient to use primes in expressions involving parentheses.

The symbols $\&$ and \vee will be called "cap" and "cup", respectively.

*applying the distributive, complementarity and intersection laws in that order; see section IV.

The following laws are stated without proof. For proofs of any of the theorems, consult the reference on the preceding page.

Laws of the inclusion relation

Reflexive:	For all A, $A \subset A$
Anti-symmetric:	If $A \subset B$ and $B \subset A$, then $A = B$
Transitive:	If $A \subset B$ and $B \subset C$, then $A \subset C$

Laws of the operations of intersection and union

Idempotent:	$A \& A = A$ and $A \vee A = A$
Commutative:	$A \& B = B \& A$ and $A \vee B = B \vee A$
Associative:	$A \& (B \& C) = (A \& B) \& C$ and $A \vee (B \vee C) = (A \vee B) \vee C$
Distributive:	$A \& (B \vee C) = (A \& B) \vee (A \& C)$ and $A \vee (B \& C) = (A \vee B) \& (A \vee C)$

Relation between intersection, union and inclusion

Consistency:	The three conditions $A \subset B$, $A \& B = A$, and $A \vee B = B$ are mutually equivalent.
--------------	---

Properties of O and I

Universal bounds:	$O \subset A \subset I$ for all A
Intersection:	$O \& A = O$ and $I \& A = A$
Union:	$O \vee A = A$ and $I \vee A = I$

Relation between intersection, union and complementation

Complementarity:	$A \& A' = O$ and $A \vee A' = I$
Dualization:	$(A \& B)' = A' \vee B'$ and $(A \vee B)' = A' \& B'$
Involution (double negation)	$(A')' = A$

The generalized dualization law:

To find the complement of an expression (Boolean polynomial) built up from primed and unprimed letters by cups and caps (but not using primed parentheses), interchange cups and caps throughout, prime each unprimed letter and unprime each primed letter.

As a result of this law and the involution law, any expression enclosed by a primed parenthesis can be replaced by the equivalent dual expression where all primes have been moved inside the parenthesis.

By applying the distributive law, any cap standing outside a parenthesis containing a cup, can be moved inside the parenthesis. Thus, any expression can be transformed into a new equivalent expression which is a union of unprimed terms, where each term is the intersection of primed and unprimed letters.

Furthermore, if a letter appears twice in a term, one occurrence may be omitted,

for $A \& A = A$. If the letter appears in the same term both primed and unprimed, the whole term is zero, since $A \& A' = 0$.

In Logic, A, B, C, ... represent statements (propositions) which can be either true (I) or false (O). Similar to sets, these statements can be combined by the logical connectives "and", "or", and "not" to give new statements. We write:

$A \& B$ for A "and" B
 $A \vee B$ for A "or" B or both
 A' for "not" A

The truth-value (true or false) of these basic statements is given in the following table.

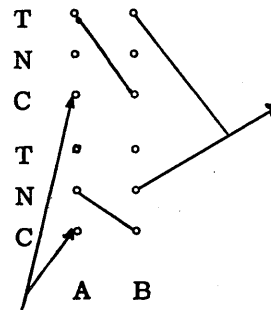
A	B	A & B	A ∨ B	A'
I	I	I	I	O
O	I	O	I	I
I	O	O	I	
O	O	O	O	

Statements formed as a combination of these basic statements can be either true or false, depending on the truth value of the individual components (statements). A statement, such as $A \vee A' = I$, which is always true (regardless of the truth-value of the components) is called a tautology.

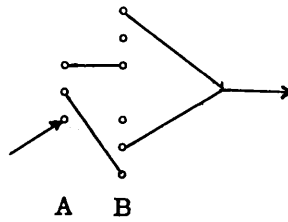
It can be shown that the algebra of logic satisfies all the basic properties listed for the algebra of sets. Any abstract algebraic system satisfying these laws is called a Boolean algebra.

The system of Boolean algebra is equivalent to Selector algebra, but is independent of the number of positions in a selector; i. e. independent of the subscripts. It is also independent of the direction of the impulse wired through a selector, A or A.

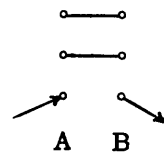
For example, $(A_1 \& B_1) \vee (\bar{A}_2 \& \bar{B}_2)$ can be represented as:



which is equivalent to $(A \& B_1) \vee (\bar{A} \& \bar{B}_2)$

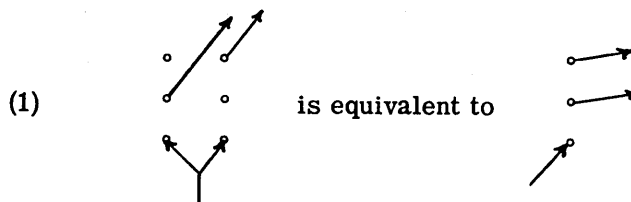


which in turn is equivalent to
 $(A \ \& \ B) \vee (\bar{A} \ \& \ \bar{B})$

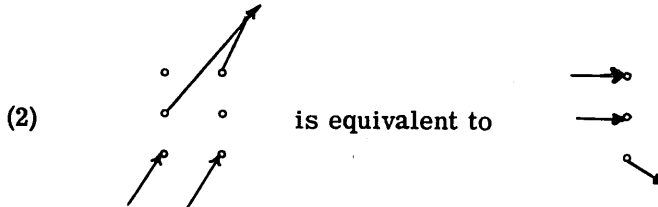


Thus, while Boolean algebra may assist us in finding the minimum number of selectors, it does not indicate how to find the minimum number of selector positions. An extension of the algebraic system is required for that purpose.

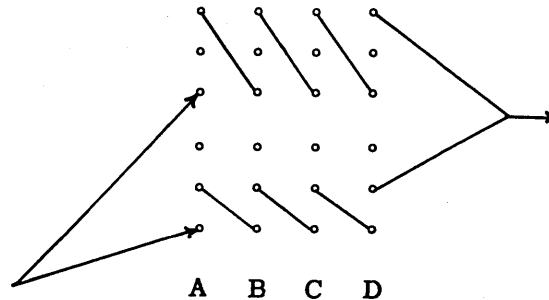
As we saw in the above example, one double position selector with one input and two outputs,



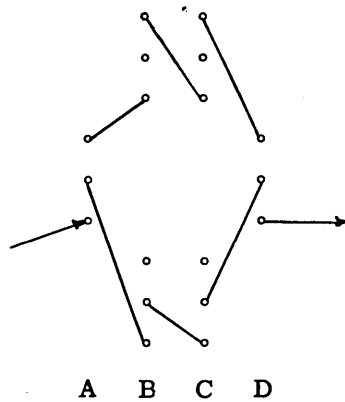
Likewise, a double position selector with two inputs and one output



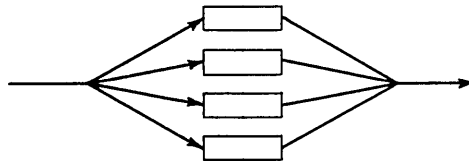
Generalizing the above example to 4 selectors, we see that $(A_1 \ \& \ B_1 \ \& \ C_1 \ \& \ D_1) \vee (\bar{A}_2 \ \& \ \bar{B}_2 \ \& \ \bar{C}_2 \ \& \ \bar{D}_2)$



is equivalent to $(A \ \& \ B_1 \ \& \ C_1 \ \& \ D) \vee (\bar{A} \ \& \ \bar{B}_2 \ \& \ \bar{C}_2 \ \& \ \bar{D})$



where substitutions (1) and (2) have been made for selectors A and D respectively. That is, we have reduced the number of positions of selectors A and D, appearing at the beginning and at the end of the parenthesized expressions, respectively, once with and once without the negation bar. This can be generalized further. We have shown in section IV that each Boolean polynomial can be expressed as a union of parenthesized terms where each term consists of an intersection of primed and unprimed letters, no two of which appear in the same term. For such Boolean polynomials, expressed as unions of intersections, the selector circuits can be represented symbolically by



where each represents the wiring circuit of a parenthesized term consisting of intersections of primed and unprimed letters. Since each letter appears at most once in a parenthesis, a will require for each letter at most a single position of the corresponding selector, (i. e. the selector identified by that letter). However, the number of selector positions can be reduced by successive substitutions (1) and (2) [see preceding page] in such a way that a single selector position can be utilized by more than one . Applying the commutative and associative laws, we can move any letter to the leftmost or rightmost position of a parenthesis. Moving the same letter, say A, (primed or unprimed*) to the left of each parenthesis, we see that a single position selector will suffice for all . This follows from the fact that due to the distributive and associative laws,

$$(A \& B \& C) \vee (A \& D \& E) = A \& [(B \& C) \vee (D \& E)]$$

*note that we use primes and bars interchangeably

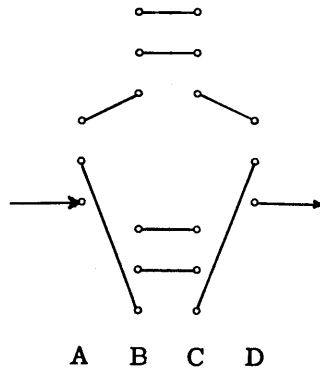
Applying this to the case of selectors, we obtain an expression of the type:

$$\{A \& [() v () v \dots v ()]\} v \{A' \& [() v () v \dots v ()]\} v \{() v () v \dots v ()\}$$

where A and A' have been factored out of all terms they appeared in, and each () represents a term now devoid of A or A'. The third { } represents terms that did not contain A or A' to begin with*. The letter that appears the greatest number of times in the parentheses should be chosen as the letter to be moved left (factored out) first. Next, we move another letter, say B (again selecting the letter that appears the greatest number of times in the () to the left, adjacent to A. We continue in this manner, and also proceed similarly by moving letters to the right until half of the letters have been moved into either direction. Thus, the polynomial,

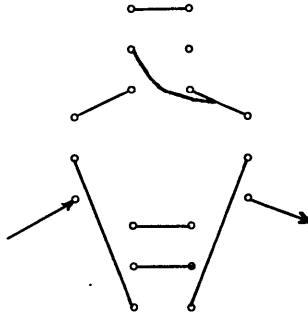
$$(A \& B \& C \& D) v (A \& \bar{B} \& \bar{C} \& D) v (\bar{A} \& B \& C \& \bar{D}) v (\bar{A} \& \bar{B} \& \bar{C} \& \bar{D})$$

can be represented by,

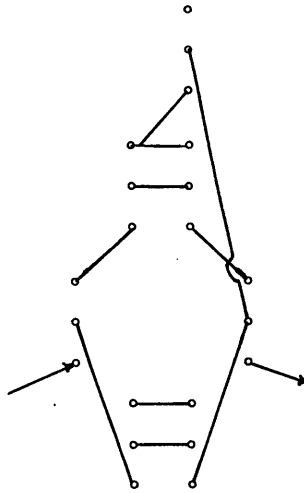


It will be observed, of course, that this expression is symmetric with respect to the left and right pairs of letters. For any letter omitted in any of the parentheses, an empty hub will appear in the diagram. For example, omitting \bar{C} in the second parenthesized term, the diagram becomes:

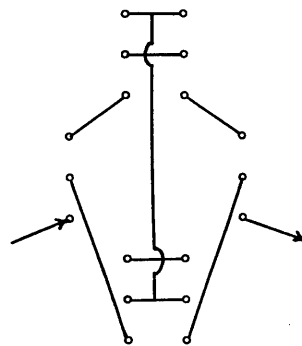
* It can be shown (Birkhoff and MacLane, op. cit., paragraph 5, pp 320 - 323) that by adjoining the factor $A v A' = I$ by means of an & to the entire polynomial, whenever a term does not contain A or A', each parenthesized term of the resulting polynomial will contain each letter (primed or unprimed) once and only once. Identical parenthesized terms (connected by a v) can be omitted, since $A v A = A$. Thus, we have found a unique expression for any Boolean polynomial as a union of intersections.



For any additional parenthesized term (no fifth letter can be added to any parenthesis, because each letter can appear only once, primed or unprimed) additional selector positions may be required. Suppose we add the term $(A \& B \& \bar{C} \& \bar{D})$ to the above expression. Then the diagram will be:



The split wire,

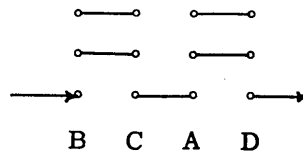


would not only add the term $(A \& B \& \bar{C} \& \bar{D})$ but also its symmetric dual term $(\bar{A} \& \bar{B} \& C \& D)$.

Another approach is to factor out a combination of letters of the original polynomial. Thus, we can write:

$$\begin{aligned}
 & (A \& B \& C \& D) \vee (A \& \bar{B} \& \bar{C} \& D) \vee (\bar{A} \& B \& C \& \bar{D}) \vee (\bar{A} \& \bar{B} \& \bar{C} \& \bar{D}) = \\
 & [(A \& D \& B \& C) \vee (A \& D \& \bar{B} \& \bar{C})] \vee [(\bar{A} \& \bar{D} \& B \& C) \vee (\bar{A} \& \bar{D} \& \bar{B} \& \bar{C})] = \\
 & \{(A \& D) \& [(B \& C) \vee (\bar{B} \& \bar{C})]\} \vee \{(\bar{A} \& \bar{D}) \& [(B \& C) \vee (\bar{B} \& \bar{C})]\} = \\
 & [(B \& C) \vee (\bar{B} \& \bar{C})] \& [(A \& D) \vee (\bar{A} \& \bar{D})]
 \end{aligned}$$

which is represented by the simple diagram:



The total number of selector positions in case of n selectors should never exceed $2^0 + 2^1 + 2^2 + \dots + 2^{n-1}$ as demonstrated in example 5, (section III) which illustrates all possible combinations of 3 selectors. 2^n is the number of choices that can be obtained by a combination of n selectors. That arrangement yields the maximum number of choices with the minimum number of selectors. It is used, in general, when a single source pulse is to be distributed through a selector network into several distinct, mutually exclusive conditions. Each of these conditions (i. e. impulses represented by arrows pointing away from the selector hubs in example 5) corresponds to a polynomial which is an intersection of letters and in which each letter is represented once, and only once (primed or unprimed). Furthermore, we notice that each possible combination of letters is represented once, and only once, by such a "Christmas tree" arrangement of selectors. Whenever any of these basic polynomials are to be combined by unions, the arrows corresponding to these polynomials would be split-wired. This is a systematic way to obtain any polynomial consisting of unions of intersections. It may then be possible to reduce the number of selector positions by applying any of the preceding transformations and substitutions.

This article is not intended to show any specific method to determine the wiring that requires the minimum number of selector positions. It merely attempts to analyze methods to detect the equivalence between two selector networks. This analysis seems to indicate that the number of selector positions can be reduced by a suitable factorization of the Boolean polynomial associated with the particular selector network.

AUTOMATIC INVERSION OF FIFTEEN-STATION
WEISSINGER CIRCULATION MATRICES

Murray L. Lesser

Northrop Aircraft, Incorporated
Hawthorne, California

Introduction

The use of the fifteen-station Weissinger method to calculate the spanwise airload distribution on a tapered, swept wing requires the inversion of an eighth or seventh-order matrix (for the symmetrical or anti-symmetrical case, respectively) for each case investigated. As this method is used extensively, it is desirable to reduce the time and effort required to invert these matrices to a practical minimum. Consequently, control panels for the IBM CPC, Model II, have been designed to perform this inversion by automatic programming in about seven minutes (for an 8X8 matrix) using, as input information, one punched card for each row of the matrix. As implied in the term "automatic programming", the machine generates its own instructions as to the manipulation of the data entered on the cards.

The mathematical procedure used is equivalent to that of transforming the given matrix and a unit matrix simultaneously, essentially by elimination, until the original matrix is replaced by a unit matrix and the unit matrix by the desired inverse. A check column is carried, and the check sum for each row is computed and printed automatically at the end of operating on each row.

The original matrix is fed into the machine, one row per card, and the first column is eliminated, simultaneously adding a column of the transformed unit matrix. Operation is by row, and a single summary card is punched carrying the equivalent row of the transformed matrix. The eight summary cards are then fed back into the machine, producing the second transformed matrix, etc. This process is repeated "n" times (for an nth-order matrix), with the desired inversion appearing on the last set of summary cards. The only card handling required is to remove the summary cards from the punch, place the first one at the end of the group, add a blank card to the back of the stack, and reinsert into the 418 card feed. No auxiliary card-handling machines are required.

Mathematical Operations

The eighth- and seventh-order circulation matrices obtained during the application of the fifteen-station Weissinger method have several characteristics leading toward ease of inversion. In particular, there is always a predominant main diagonal with the maximum absolute value term as the leading term (having a value of the order of ± 40) and decreasing (on the diagonal) to a value of about ± 8 . The terms on the diagonal are all of the same sign, being negative for the symmetric case ($n=8$) and positive for the anti-symmetric case ($n=7$). Since the matrix represents a set of non-singular physical quantities, (as long as the original analysis was performed for a set of physical conditions within the limitations of the theory) there are no singularities, and the whole process of inversion is relatively well-behaved.

Figure 1

COL. 1	COL. 2	COL. 3	COL. 4	COL. 5	COL. 6	COL. 7	COL. 8	CHECK COL.	CONTROL "X"	ROW NO.
00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
1 2 3 4 5 6 7 8	9 10 11 12 13 14 15 16	17 18 19 20 21 22 23 24	25 26 27 28 29 30 31 32	33 34 35 36 37 38 39 40	41 42 43 44 45 46 47 48	49 50 51 52 53 54 55 56	57 58 59 60 61 62 63 64	65 66 67 68 69 70 71 72	73 74 75 76 77 78 79 80	
11111111	11111111	11111111	11111111	11111111	11111111	11111111	11111111	11111111	11111111	11111111
22222222	22222222	22222222	22222222	22222222	22222222	22222222	22222222	22222222	22222222	22222222
33333333	33333333	33333333	33333333	33333333	33333333	33333333	33333333	33333333	33333333	33333333
44444444	44444444	44444444	44444444	44444444	44444444	44444444	44444444	44444444	44444444	44444444
55555555	55555555	55555555	55555555	55555555	55555555	55555555	55555555	55555555	55555555	55555555
66666666	66666666	66666666	66666666	66666666	66666666	66666666	66666666	66666666	66666666	66666666
77777777	77777777	77777777	77777777	77777777	77777777	77777777	77777777	77777777	77777777	77777777
88888888	88888888	88888888	88888888	88888888	88888888	88888888	88888888	88888888	88888888	88888888
99999999	99999999	99999999	99999999	99999999	99999999	99999999	99999999	99999999	99999999	99999999
1 2 3 4 5 6 7 8	9 10 11 12 13 14 15 16	17 18 19 20 21 22 23 24	25 26 27 28 29 30 31 32	33 34 35 36 37 38 39 40	41 42 43 44 45 46 47 48	49 50 51 52 53 54 55 56	57 58 59 60 61 62 63 64	65 66 67 68 69 70 71 72	73 74 75 76 77 78 79 80	IBM SOBI

NOTE: Negative quantities indicated by an "X" (11) overpunch in low-order card column of number field (i.e. if matrix column 1 term were negative, there would be an "X" overpunch in card column 8, etc.)

The matrices are obtained directly from a card-programmed CPC run as summary cards. There is one card for each row (Fig. 1) containing, in addition to the values of each column of the matrix for that row, a "check" column computed by the expression:

$$M_{ic} = -1 - \sum_{j=1}^n M_{ij} \quad (1)$$

where the M_{ij} are the j terms of the i th row of matrix M of order n , and M_{ic} is the check column term of the i th row. During the transformation process, use is made of the fact that for any row of the transformed matrix M'

$$S' = M'_{ic} + \sum_{j=1}^n M'_{ij} = -1 \quad (2)$$

Thus, the check sum S' , is computed and printed simultaneously with the transformation of each row as a visual indication to the operator that no machine error has occurred. (Note: it is not believed that the variation of S' from -1.000000 has any great significance as to the error involved in the inversion, as the process appears to be

somewhat self-healing. The usual variations from -1 involved in using the procedure seem to have maximums of the order of ± 0.000006 . Six decimal places are carried throughout.)

The elimination process consists of setting up each term of the diagonal, in turn, as the leading term of the matrix, and introducing, in a combined matrix, the first column of the unit matrix. Thus, effectively, before each transformation the matrix appears as follows:

$$\begin{array}{cccc|ccc}
 M_{11} & M_{12} & \dots & M_{1j} & \dots & M_{1n} & M_{1c} & 1 \\
 M_{21} & M_{22} & & & & & \cdot & \cdot \\
 \cdot & & & & & & & \cdot \\
 \cdot & & & & & & & \cdot \\
 \cdot & & & & & & & \cdot \\
 M_{11} & \dots & \dots & M_{1j} & \dots & M_{1n} & M_{1c} & 0 \\
 \cdot & & & & & & \cdot & \cdot \\
 \cdot & & & & & & \cdot & \cdot \\
 M_{n1} & \dots & \dots & M_{nj} & \dots & M_{nn} & M_{nc} & 0
 \end{array}$$

Only the columns 1 to n, and c appear on the card. The unit column is manufactured by the machine during the computation process.

The first row is operated on by dividing the remaining terms in the row by M_{11} , and storing the $n + 1$ terms in the storage registers of the machine, as well as printing and summary punching the terms. Thus,

$$M'_{1j} = M_{1j} / M_{11} \tag{3}$$

where M'_{1j} is the transformed term. At the same time, the check sum is being accumulated, and is printed immediately following the printing of the row.

The remaining rows are operated on to eliminate column 1. Thus:

$$M'_{ij} = M_{ij} - M_{i1} M'_{1j} \tag{4}$$

The transformed matrix then becomes

$$\begin{array}{c|cccc|cc|cc}
 1 & M'_{12} & \dots & M'_{1j} & \dots & M'_{1n} & M'_{1c} & & M'_{1u} & \\
 0 & M'_{22} & \dots & M'_{2j} & \dots & M'_{2n} & M'_{2c} & & M'_{2u} & \\
 \cdot & \cdot & & & & \cdot & \cdot & & \cdot & \\
 \cdot & \cdot & & & & \cdot & \cdot & & \cdot & \\
 \cdot & \cdot & & & & \cdot & \cdot & & \cdot & \\
 \cdot & \cdot & & & & \cdot & \cdot & & \cdot & \\
 0 & M'_{i2} & \dots & M'_{ij} & \dots & M'_{in} & M'_{ic} & & M'_{iu} & \\
 \cdot & \cdot & & & & \cdot & \cdot & & \cdot & \\
 \cdot & \cdot & & & & \cdot & \cdot & & \cdot & \\
 \cdot & \cdot & & & & \cdot & \cdot & & \cdot & \\
 \cdot & \cdot & & & & \cdot & \cdot & & \cdot & \\
 0 & M'_{n2} & \dots & M'_{nj} & \dots & M'_{nn} & M'_{nc} & & M'_{nu} &
 \end{array}$$

No operations are actually performed on the first column and it does not appear in the results. On the summary cards, the transformed column 2 is located in the column 1 field, column j in the (j-1) field, and column u in the column 8 field. The rows are renumbered (automatically) so that the transformed row 1 becomes row 8, and all other row numbers are decreased by one. Thus, the process is ready to be repeated if the new row 8 card is removed from the front of the summary stack and placed at the end.

Table I shows a list of a typical eighth-order matrix obtained during the Weissinger process. Immediately below this is shown the first transformed matrix. The third listing in Table I is a list of the last set of cards, showing the inverse of the original matrix.

Table I

Sample Tape Forms

Original Matrix List:

Col. 1	Col. 2	Col. 3	Col. 4	Col. 5	Col. 6	Col. 7	Col. 8	Check Col.
41 1044529-	14718078	72994-	11147255	50862-	343894	34581-	114039	231879700
7 499326	20973278-	8 075579	70746-	757535	44829-	291394	17132-	3482151
32996-	5 552690	14 483469-	5671070	65755-	599947	42492-	161449	1639556
309628	51379-	4 442222	11 402103-	4513025	62708-	567429	22864-	706750
17090-	337690	59647-	3 822841	9 708967-	3 918715	63225-	377008	392675
69053	26401-	348335	64642-	3 512059	8 746887-	3 772544	35296-	171235
10113-	107795	34797-	394257	72143-	3 538326	8 249226-	3 241631	84270
41665	19376-	169428	47516-	606754	88128-	6 345103	8 046575-	38645

First Transformed Matrix:

Check Col.	Col. 2	Col. 3	Col. 4	Col. 5	Col. 6	Col. 7	Col. 8	Unit Col.
581800-	358588-	1778	27951-	1239	8379-	843	2778-	24364-
7 845259	18 284110-	8 062245	138868	748243	18008	285072	3701	1 000000-
1 620359	5 540858	14 483410-	5 670148	65714-	599671	42464-	161357	182714
886892	59650	4 441671	11 393449-	4 512641	60114-	567168	22004-	1 000000-
382732	331562	59617-	3 822363	9 708946-	3 918572	63211-	376961	804-
211410	1639-	348212	62712-	3 511973	8 746308-	3 772486	35104-	999999-
78386	104169	34779-	393974	72130-	3 538241	8 249217-	3 241603	7544
62886	4435-	169354	46351-	606702	87779-	6 345068	8 046459-	1 000001-

Check sums

Inverted Matrix List:

Col. 1	Col. 2	Col. 3	Col. 4	Col. 5	Col. 6	Col. 7	Col. 8
29327-	26342-	20747-	19065-	16621-	15612-	14673-	7343-
13407-	71693-	52951-	40174-	36124-	32012-	30837-	14961-
7208-	36288-	114508-	78460-	58857-	53127-	48484-	24156-
5119-	21297-	61108-	155071-	103378-	78644-	72945-	34698-
3701-	15939-	38199-	86897-	193866-	130234-	103404-	50707-
3056-	12386-	30339-	58215-	115726-	234100-	165447-	71507-
2642-	11009-	25477-	49855-	84883-	154054-	286229-	309220-
2570-	10349-	24668-	45966-	80913-	129393-	232282-	266655-
							118994-
							221455-
							850266-
							707840-
							578910-
							467737-
							377050-
							309220-
							266655-
							252401-

NORTHROP AIRCRAFT 605 PLANNING CHART

FIG. 2

ENGINEER M.L. LESSER PROBLEM WEISSINGER MATRIX INVERSION DATE _____

REMARKS	CALC. SEL. PICK UP	ELECTRONIC SUPPRES	CALC. SEL. SUPPRES	PROGRAM NUMBER	FACTOR STORAGE ASSIGNMENT				MULT. QUOT.	COUNTER	GENERAL STORAGE ASSIGNMENT			
					6-4 (8-6)		6-9				6-4 (8-6)		6-9	
					1	2	3	4			1	2	3	4
				R										
				1				RO		RI +, BT				
				2						RES				
				3				RO		RI -				
				4						RR		RI		
				5		RO				RI +, BT, RES				
				6		RO				RI -, TO 3				
				7		RO				RI +, TO 3				
				8						RI +, BT, RES		RO		
EMIT 1				9						RI -, BT NBS1				
				10				RO		RI +, TO 3				
				11		RO				-				
				12						RR		RI		
				13						RI +, TO 6		RO		
				14					RO			RI		
				15		RO				÷, RES				
				16					RO	RI +				
				17						RI +, TO 6		RO		
				18						1/2 ADJ				
				19						RR, FR 2		RI		
				20						RI +		RO		
				21						RI +				RO
				22						RR				RI
				23						RI +		RO		
EMIT 8				24								RI		
				25						RI +, TO 3		RO		
				26						RO, FR 6		RI		
				27					RI	RR				
				28		RO				MPLY -				
				29					RI			RO		
				30						RR, FR 6		RI		
				31						RI +		RO		
				32		RO				MPLY -				
				33				RO		RI +, TO 4				
				34						1/2 ADJ, TO 3				
				35						RO, FR 4		RI		
				36				RI		RR, FR 6				
				37						RI +		RO		
				38						RI +				RO
				39						RR				RI
				40				RO		RI +				
				41				RI		RR, FR 4				
				42				RO		RI +				
				43						O TEST, TO 4, RES				
EMIT 1				44						RI -, BT NBS1				
				45						RI +		RO		
EMIT CODE*				46								RI		
				47										
				48						RI +				RO
				49										
				50										
				51										
				52										
*SEE FIG. 7-B				53										
				54										
				55										
				56										
				57										
				58										
				59										
				60										
				P										

Table II presents the results of taking the matrix product $[M M^{-1}]$. As can be seen, the maximum "error" is 0.000067. The magnitude of this error is largely due, it is believed, to the effect of the large difference in number of significant figures involved in the leading terms of the first row of the original matrix and in any column of M^{-1} . (The total error could be explained by an error of about $1\frac{1}{2}$ units in the low order digit of any term in M^{-1} that multiplies M_{11}). As a rough check on this, a few terms of the matrix product $[M^{-1}M]$ were calculated, and the maximum error involved appeared in column 1 and was of the order of 0.000040 (not shown). This general order of accuracy is sufficient for the Weissinger process. However, an increase in accuracy could be obtained if desired (with slight wiring changes on the 605 panel) by shifting the decimal point in the original matrix such that all terms were divided by 10. This would produce an inversion with one more significant figure in each term, with no change in the operating procedure or time for inversion. (The same number of digits--a maximum of eight--would be handled in either case. The differences would be in the internal shifts in the 605).

Table II
Matrix Product $[M M^{-1}]$

0.999976	-0.000021	0.000043	-0.000067	-0.000008	-0.000025	0.000010	-0.000003
0.000017	1.000000	-0.000006	0.000009	-0.000019	0.000014	-0.000002	0.000008
-0.000001	-0.000006	0.999999	-0.000005	0.000014	0.000002	-0.000003	-0.000007
0.000005	-0.000008	0.000004	1.000004	0.000000	-0.000010	0.000003	0.000004
-0.000010	-0.000002	-0.000009	0.000002	1.000002	0.000006	0.000003	-0.000001
0.000005	0.000004	0.000001	-0.000007	-0.000009	0.999993	0.000000	0.000001
0.000001	-0.000003	0.000002	0.000000	0.000003	0.000005	1.000002	-0.000002
0.000000	-0.000003	-0.000001	-0.000002	-0.000002	-0.000002	-0.000001	1.000000

Machine Wiring - (605 Panel)

The 605 is only required to perform two basic operations. These are: to compute M'_{1j} (first row terms), and to compute M'_{ij} (remaining row terms). In addition, the check sum is accumulated and stored within the 605, to be brought out when needed, and provision is made to generate the correct row number to appear on the summary card. Due to the well-behaved nature of the Weissinger matrices, it is not really necessary to worry about overflows due to effective singularities. However, since it may be desirable to invert other matrices of order 8 or less with the same set-up, and since there was no cost in operating time, checks were inserted to stop the machine if overflow occurs.

All numbers are carried in eight-digit form (six decimal places). The three-digit electronic storages were assigned 8-6, and all were coupled into 8-digit groups (FS 1-2, FS 3-4, GS 1-2, and GS 3-4). The 605 planning chart is shown in Fig. 2. The convention

adopted for the three columns to the left of the program numbers is as follows: column one refers to the program levels picked up by a "Program Source" pulse through the transferred points of the "Calc" selectors indicated. Where two numbers are shown, pick up is through either selector (wired in parallel). As there were a surplus of points available on the Calc selectors, "Program Source Filters" were not used. Column two signifies impulses from the test-result hubs indicated are carried directly to the suppress hub shown (no selectors involved). Column three entries signify "Suppress on no test" impulses carried through the transferred points of the Calc selectors noted to the suppress hubs. Only Panel A of the board is used for this set-up.

The storage RI and RO inverter hubs are coupled into groups, and the pulses shown refer to both storages in the group. The notation "BT" refers to "Balance Test for Step Suppress", while "BT NBS 1" refers to "Balance Test for Selector Pick-up, Negative Balance Selector No. 1". The remaining notation is obvious. ("Program Expansion" is used where necessary).

Calculate selector No. 1 is used to compute the transformed terms of matrix row 1. Program levels 1-9 check the relative magnitudes of M_{11} and M_{1j} (FS 1-2, and FS 3-4 respectively) to assure that the quotient $M'_{1j} \leq 99.999999$. If this is not so (effectively dividing by zero), Negative Balance Selector No. 1 is transferred, thus stopping the machine (see discussion of 418 wiring). Matrix column 1 is entered directly from the card into FS 1-2 and is held in the 605 throughout the remaining cycles of that row. Matrix column 2 is entered directly from the card into FS 3-4, and is replaced by the remaining terms of the matrix, in turn, that have been stored in the 418 counters. The check sum is accumulated in GS 3-4 (program levels 21-23), which was cleared on the card-feed cycle initiating the row. The transformed term is left in the Electronic Counter (level 23) for transfer to the Storage Register (941), and to 418 counters or to punch and type magnets. The transformed row number, "8", is emitted into GS 2 on level 24 to be available for read-out during the punching cycle. Previous M'_{1j} terms (from earlier transformations) are read into GS 1-2 on each cycle, but are not used. (Note: It is possible to use GS 2 for both Read In and Punch Out in this manner, as the two operations are not occurring at the same time).

The transformed terms of the remaining rows are computed on Calc selector 2. The high-order digits of the transformed term are stored in FS 3-4 after computation, to check for possible multiply-overflow (levels 36, 40-44). The check sum is accumulated as before on levels 37-39. The transformed row number is emitted into GS 2 on level 46 (on operation 2 only) through a chain of Calc selectors under control of Pilot selectors on the 418 panel (see 418 wiring section and Fig. 7-B).

Calculate selector 3 is used to transfer the check sum to the Electronic counter for return to the 418 type bars.

Machine Wiring (418 Panel):

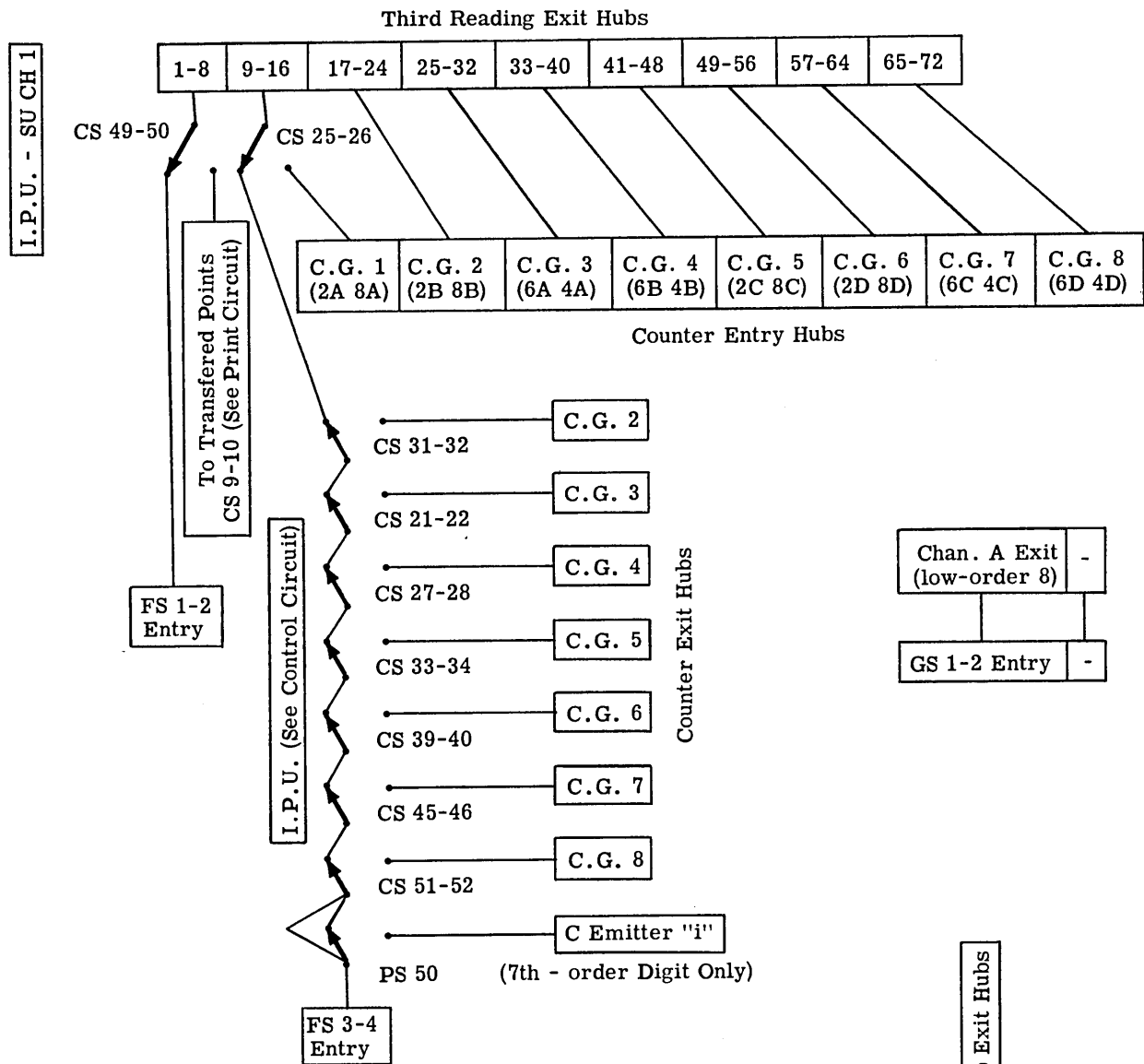
The 418 serves as the control center for the entire operation. It also serves to store the terms of the original matrix row until needed in the 605, and the transformed terms until they are cleared out of the counters by summary punching and printing. The CPC is made to generate its own program through the use of the "Special Program" feature and the Field Selectors. Control from information on the read-in cards is maintained through the automatic (or CB) cycles by virtue of the fact that Pilot Selectors, when picked by "X" or "D" pick-up, will remain transferred through the following card cycle--irrespective of how many CB cycles intervene. Over-all control is maintained by Field Selector No. 2, rather than by use of the "Program level exits" provided on the 418 panel. The 418 panel wiring is shown schematically in Figs. 3-7.

The terms of a row of the matrix are entered into the machine as shown in Fig. 3. The column 1 term is entered directly into the 605 FS 1-2, and remains there during the following CB cycles. The matrix column 2 term is entered into FS 3-4, and the remaining terms on the card are entered directly into 418 counters. Sign control is shown in Fig. 4. On successive CB cycles (operations on the remaining terms in the row), the remaining terms of the row are entered into the 605 through the entry selector chain shown in Fig. 3. Transfer of any selector-pair in this chain connects "Counter Exit" to the 605 FS 3-4 Entry, Counter "Negative Balance Test Exit" to the sign entry hubs of both FS 3 and FS 4, and an "All-Cycles" pulse to the Counter Total Hub; thus reading-out and resetting the counter group. Conversion is prevented by the wiring shown at the bottom of Fig. 4. (Note: The term "Counter" as used herein actually refers to a counter group. The counters are coupled into eight ten-digit groups as shown in Fig. 3. The groups, carrying eight digits and sign, are considered as individual counters for purposes of discussion.)

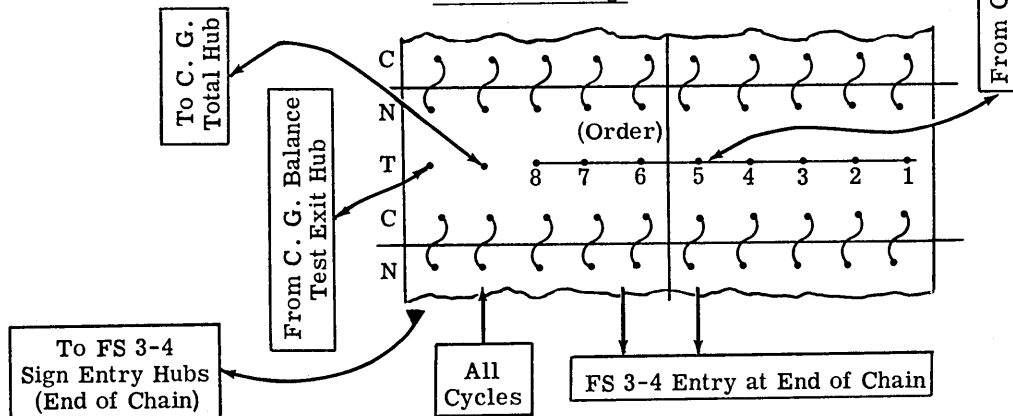
The returning transformed matrix term is transferred from the "Electronic Counter Exit" to the "Channel C Shift Entry" as shown in Fig. 5. The shift feature is not used--the wiring is such that the number appears as the low-order eight digits at the Channel C hubs with no shift. Wiring to the various counter entries is through Field Selector No. 1, as shown. During row 1 operations only, the number is also transferred to 941 storage by means of Channel C (see Fig. 7-A). The transformed row 1 terms, used in the calculation of the remaining rows, are brought to GS 1-2 entry from Channel A (Fig. 3).

Print and punch wiring is shown in Fig. 6. The counter-exits are wired directly to type-bars, with one exception, as shown. This allows simultaneous conversion, summary-punching and printing of all terms in the row at the proper time, the unit column value being printed directly from the 605 counter. The selector arrangement (picked up by "Set Up Change Switch #1") shown in Figs. 3 and 6 allows listing of any set of matrix cards directly without performing any operations thereon, and without initiating any program (CB) cycles (see Table I and Fig. 7-A). Negative signs, for

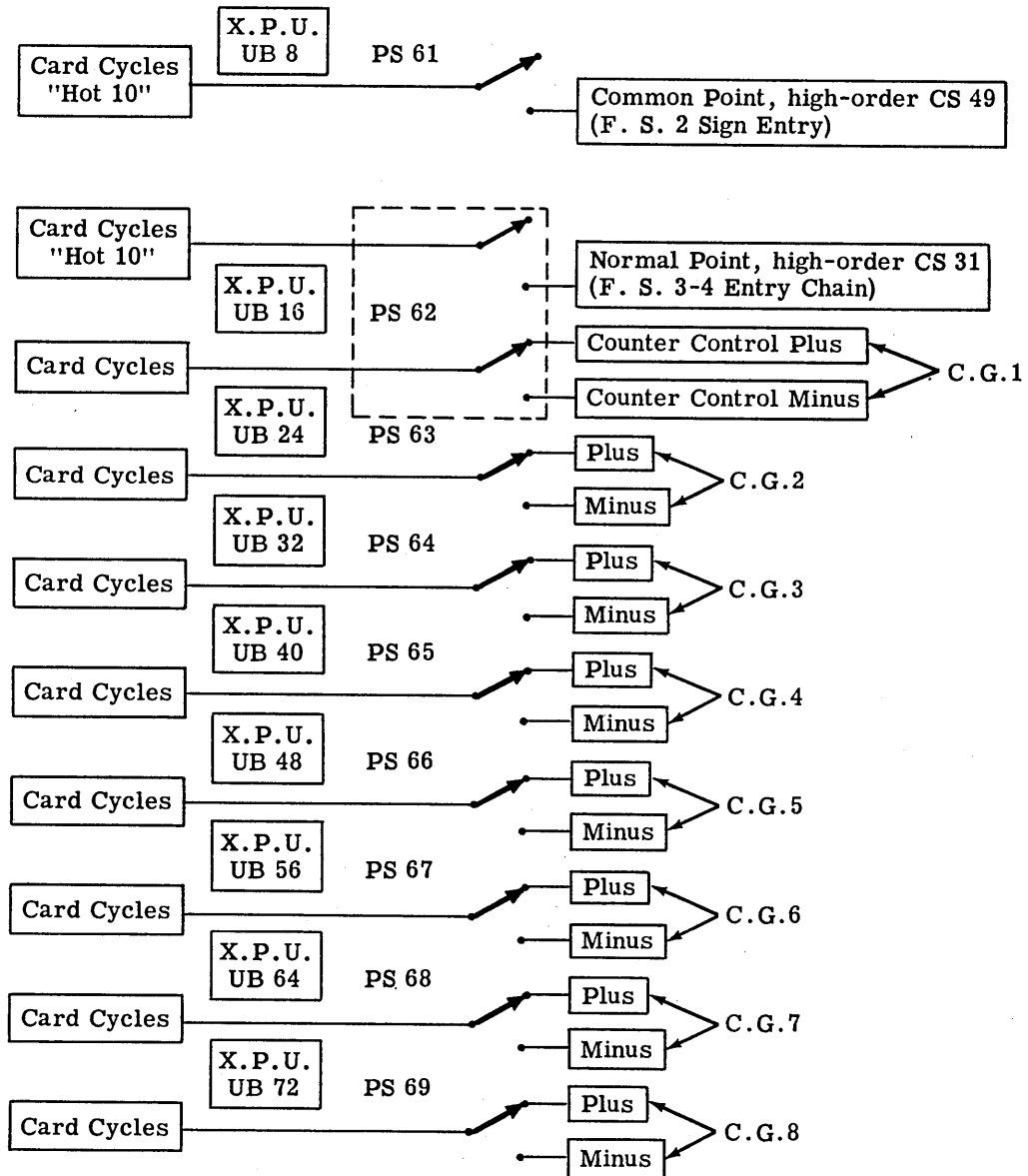
Data Entry Circuits:



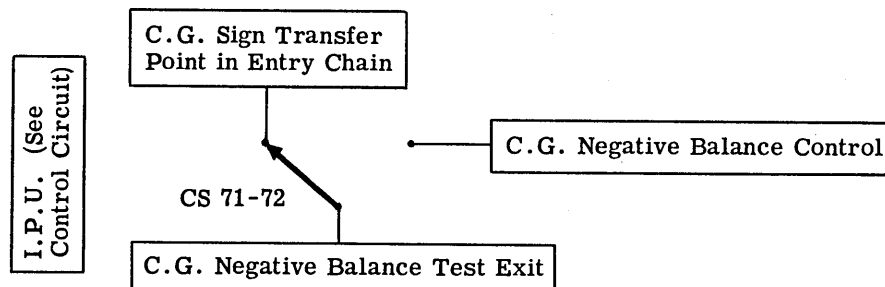
Typical Entry Chain Selector Wiring:



Card Entry:



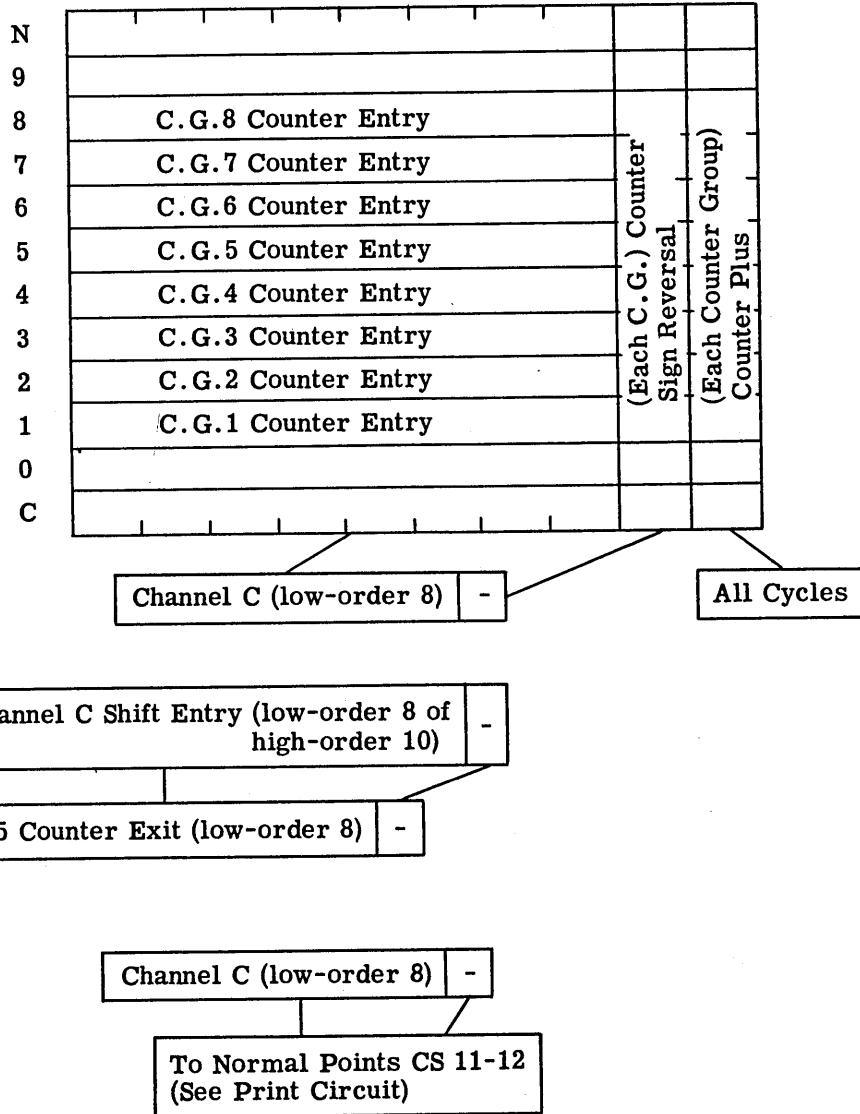
418 Counter to 605:



Note: Each Counter Group wired through individual points of CS 71-72

Data Return (from 605) Circuits:

Field Selector No. 1 (See Control Circuits for PU)



summary punching, are carried from the counter "Transfer and SP Control Minus" to the 527 through the "SP Control" entry hubs. Use was made of corresponding numbering as a memory aid. As may be noted from Figs. 4, 6, and 7-B, the "Negative Balance Test Exits" are wired to the "Negative Balance Control" hubs only during the punch-print cycle. Thus, there is no conversion before reading out to the 605.

All control circuits are shown in Figs. 7-A and 7-B. Primary control is through Field Selector No. 2 (Note: The sequence is Normal (CF cycle) followed by levels 1 through 9 in turn). The left-hand column on the Field Selector distributes "All Cycles" pulses to the elements of the entry selector chain shown in Fig. 3. It should be noted that at level 8, there is a further control through PS 10---this latter selector being transferred only during row 1 CB cycles (and the CF cycle that loads row 2). Thus, the 01.000000 for the unit column is entered only for row 1, and zero is entered into FS 3-4 for the unit-column term on the remaining rows. The entry-chain selectors are transferred, by I. P. U., on the cycle that the factor is needed in the 605.

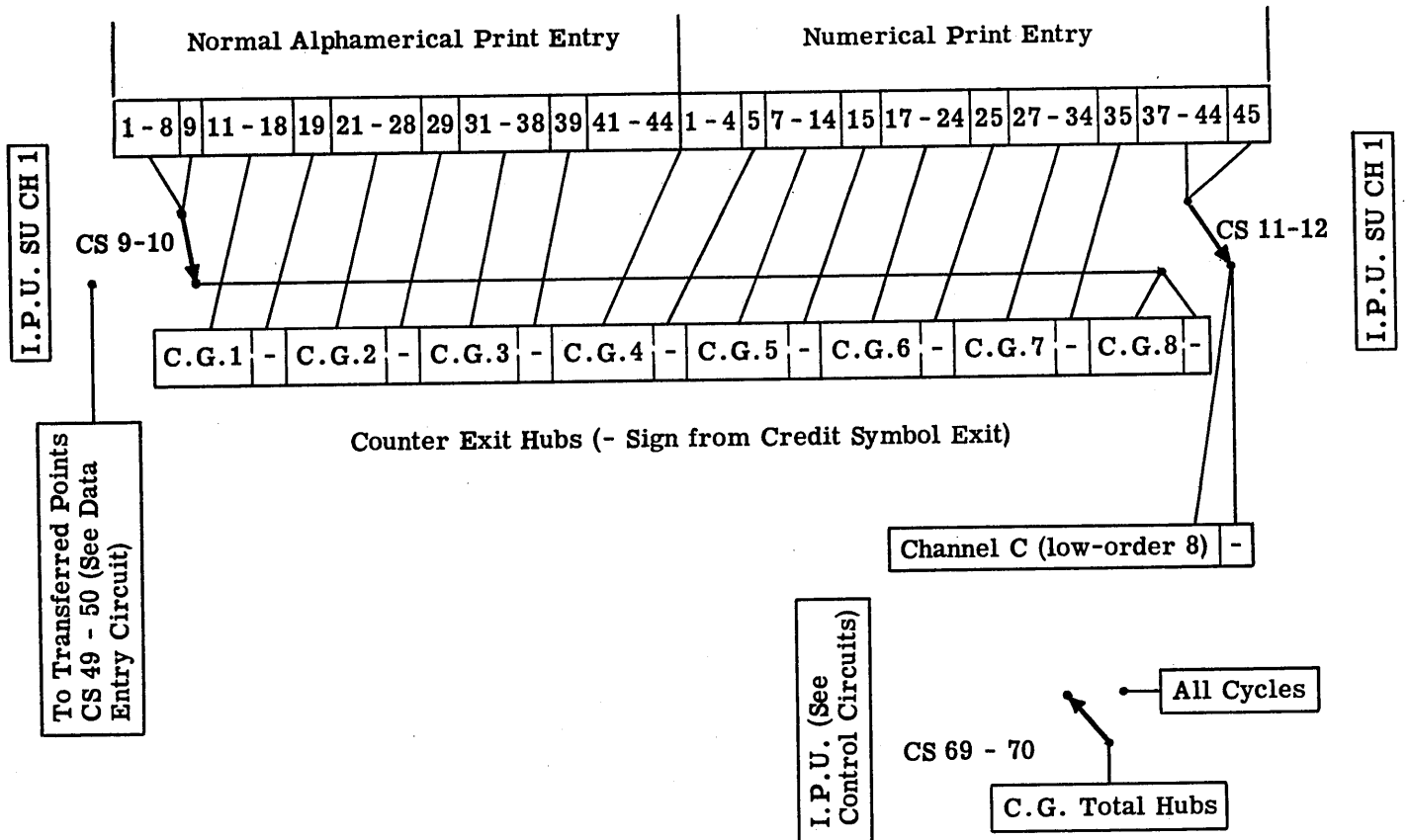
The next two columns of the Field Selector carry emitted digits (from the "C Emitter") to form the Channel C Control to load the transformed terms of the first row into the 941 storage. In accordance with the timing requirements of the CPC, these instructions are made available to the Channel C Control on the cycle during which the factors are sent to the 605 (one cycle before the answer returns). The selector arrangement shown between the Field Selector Common and the Channel C Control is required to allow instructions to reach Channel C Control only for the row 1 terms. Control for these selectors is shown in the lower left corner of Fig. 7-A.

The next two Field-Selector columns supply emitted digits for Channel A Control. The fact that Channel A receives instructions (and thereby supplies whatever numbers happen to be standing in the corresponding 941 positions) during row 1 operation is not important, as no use is made of the values read into GS 1-2 during these operations. In accordance with the timing requirements, the instructions are made available to the Channel A Control one cycle before the factor is required.

Column six of the Field Selector controls the Field Selector advance. Advance is initiated by the "Card Count" pulse (a "one") that also starts the program, carried through the transferred points of PS 6 (picked by the upper brush--"Second Reading"--control "X" in card-column 73). Successive advances of the Field Selector are made from the digits emitted at each level. The Field Selector pick-up pulse is carried through the normal points of Negative Balance Selector No. 1 and CS 65-66 (matrix list) to stop advance if an overflow occurs, or prevent advance during the "Matrix List" operation. The pick-up for the two Field Selectors are wired together, so that the data return circuit in Field Selector No. 1 (Fig. 5) is synchronized with the control circuits. (It should be noted that Channel C is connected to counter entry during the cycle on which the answer returns from the 605.)

Fig. 6

Print and Punch Circuits:



Note: Each Counter Group
Wired through Individual
Points of CS 69 - 70.

Summary Punch Sign Control:

"Transfer and SP control minus" hub for each counter group wired to
"SP control entry" hub having the same number as the counter group.
(See Control Circuits for Summary Punch PU.)

Since, during "Special Programming", a list stroke will normally be taken on each CB cycle, and on the first CF cycle following a CB cycle, printing must be inhibited during CB cycles 1 through 8 when the transformed values are being returned to the 418 counters. This is done as shown in Field Selector column 7 in Fig. 7-A. An All-Cycles pulse is carried through the B side of Toggle-Switch 1 (and through CS 65-66) from levels 1 through 8, to Non-Print. In addition, a Card-Cycles pulse is carried to all "Counter Exit Suppress" hubs to prevent detail-listing the input data from the cards during the cycle the check sum is being printed. If it is desired to see all the intermediate results (for checking), this is accomplished by throwing Toggle-Switch 1 to "A". The Card-Cycle to List connection is required to detail-list the row 1 card. Printing and punching from the 418 counters is controlled by the level-9 All-Cycles pulse as shown on the SP PU bus diagram on Fig. 7-B.

Calculate Selector pick-up is controlled by Field-Selector column eight (Fig. 7-A). Levels N and 1 through 8 pick up either Calc selector 1 or 2, as controlled by selectors PS 56 and PS 57 (through CS 65-66). A 9-common pulse is used to pick up these selectors. Calculate selector 3, and "Program Stop" are controlled from level 9. The method of interposing an absolute stop if an overflow occurs (transferring Negative Balance Selector No. 1) is shown in Fig. 7-A, and also in Fig. 7-B where the coupling exit is shown wired to machine stop.

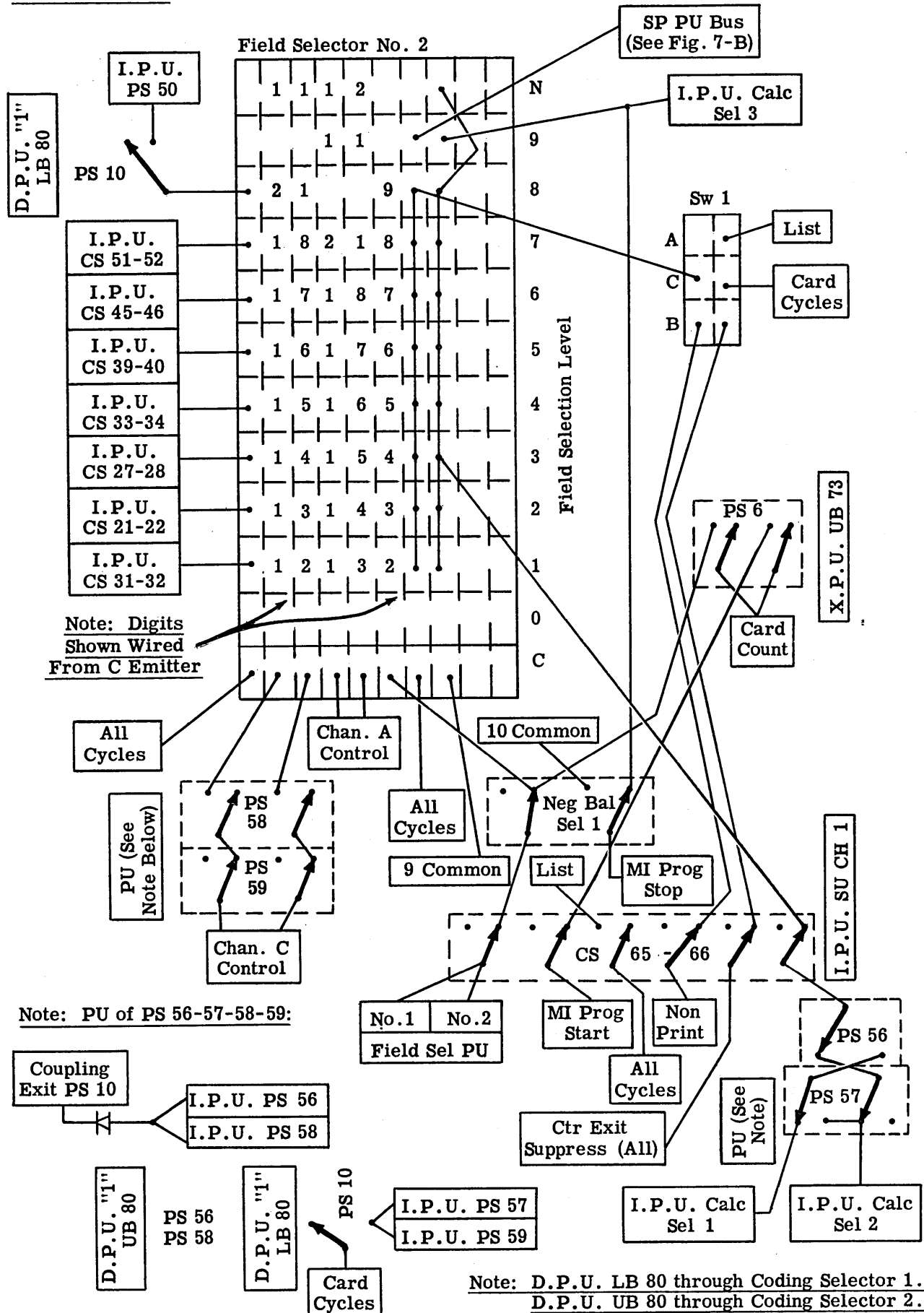
The 605 Entry and Exit Read In and Read Out controls are shown in Fig. 7-B, as well as the method used to emit the correct row number in the 605. Here, again, use is made of the fact that Pilot Selectors picked on "X" or "D" P. U. will hold through the following CF cycle.

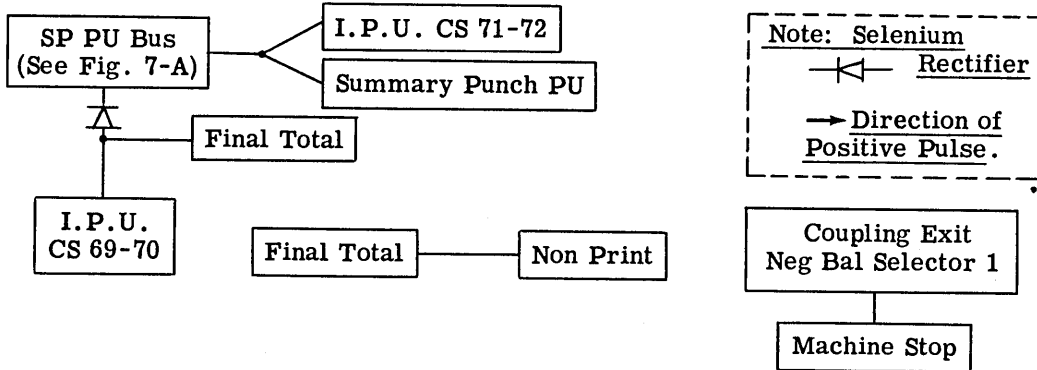
The 527 (punch) wiring is straightforward, and is shown in Fig. 8.

Operating Procedure

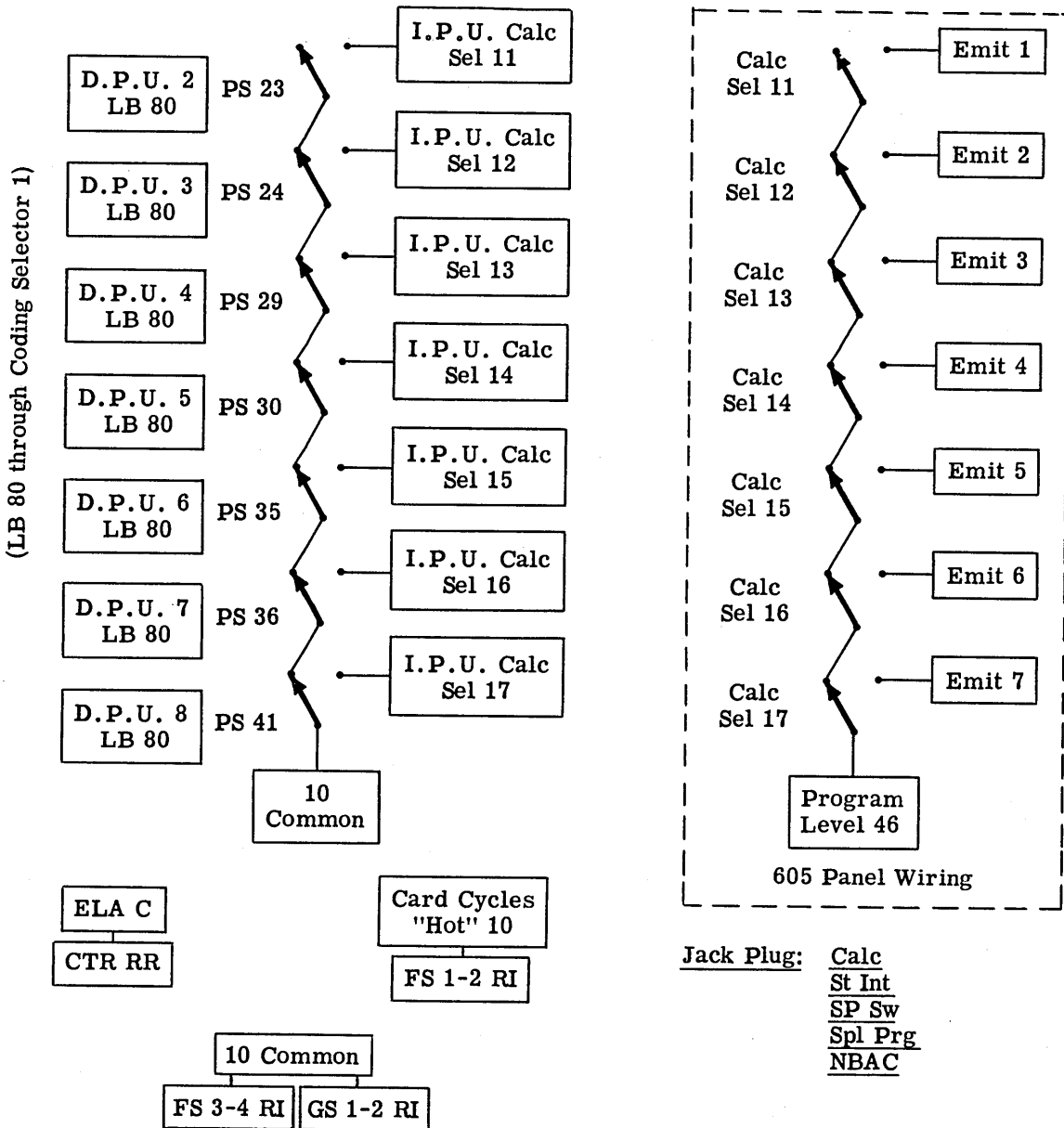
The operating procedure is fairly obvious from the foregoing discussion of the mathematical treatment and the wiring. To recapitulate: A Final Total is taken. Matrices, one row per card in the form of Fig. 1, are fed into the 418, producing a set of summary cards containing the rows of a transformed matrix. (One blank card must follow the row-8 card). The summary cards are put in proper row order by removing the row 8 card from the front of the stack and placing it at the back, to be followed by a blank card (discarding the three extra cards one gets from the punch each time.) These cards are then fed into the 418, repeating the process n times in all to invert an n-order matrix. Any matrix set may be listed directly by turning on Set Up Change No. 1 (such a list must be followed by a "Final Total".) If the check-sums show an error was made on a particular row, the process may be stopped at the end of the next row by holding down the "stop" switch, the cards cleared by use of "non-print runout", a Final Total taken, and the particular transformation rerun. If the error persists, it is probably due to a punch error on the previous transformation, so the process should be

Control Circuits:



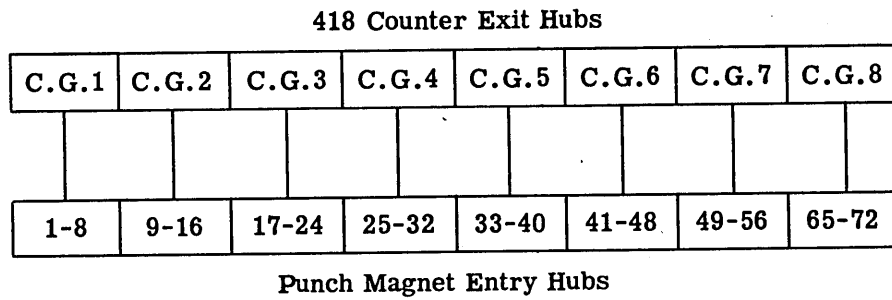


Code Control (To operate emitter on 605 Step 46):

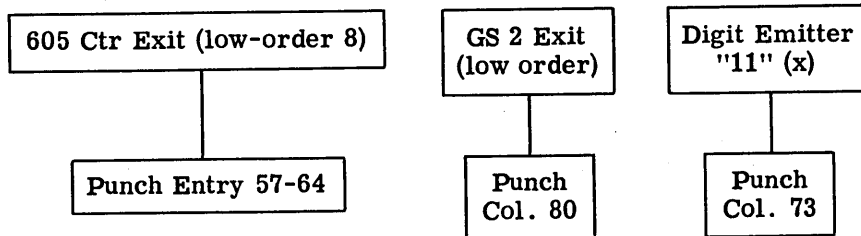
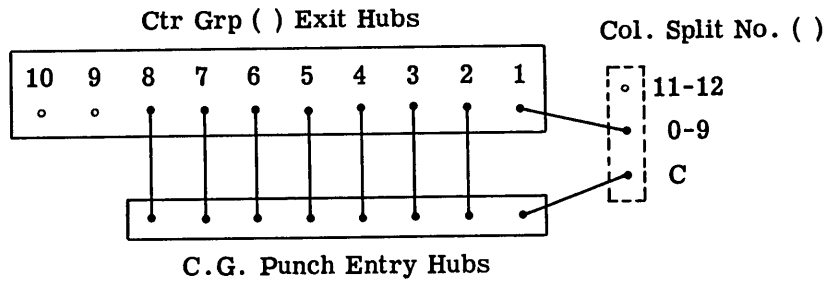


Note: SU CH 1 "On" Lists Matrix
Toggle Sw 1 "A" All-Cycles List

Summary Punch (527 Panel) Circuits:



Typical Counter Group Punch Wiring:



Wire: Calc "N"

"Card Cycles" to "Ctr RO" & "GS 2 RO"

repeated going back one set.

If seventh-order matrices are being run, the matrix column 8 Field (Fig. 1) is left blank on the original cards. As the inversion process proceeds, the blank field moves back to the left, one column at a time. At the end of the seventh pass (inverse matrix), the column-one field will be the blank field.

WARNING: Run with "Last Card Auto Total" OFF.

Concluding Remarks

The method of matrix inversion on the IBM CPC, Model II, described herein has been found to be extremely effective for the use intended, i. e., inverting the Weissinger Circulation Matrices. Its advantages are that it is nearly as rapid as is theoretically possible (only one machine cycle per transformation per term plus a minimum of conversion and punch cycles are used), it requires no extensive card handling and no auxiliary equipment other than the CPC, and it provides a running, visual check for machine error as the inversion proceeds. The loading cards (original matrix) may be obtained directly from a summary-punched CPC pass in which they were computed, requiring no intermediate handling, or they may be key-punched.

Without modification of the control circuits, an appreciable loss in computing time occurs (as compared to the optimum) if the order of the matrix is much less than 7, but such changes may be easily made if desired. It is also possible to extend the procedure to higher-order matrices by carrying as many terms as possible per card (more than one card per row), and modifying the control circuits accordingly. The theoretical limit would be of order one less than the number of 941-register positions available, or even larger matrices might be inverted by parts. However, there is need for a "round-off error" analysis before the procedure is extended indefinitely.

INTERNATIONAL BUSINESS MACHINES CORPORATION

590 Madison Avenue, New York 22, N. Y.

Form No. 34-6099-0-3M-PP

Litho in USA