Feb. 9, 1960 D. E. ECKDAHL ET AL 2,924,381

DIGITAL DIFFERENTIAL ANALYZER

Filed April 22, 1952 48 Sheets-Sheet 1

Fig. 1

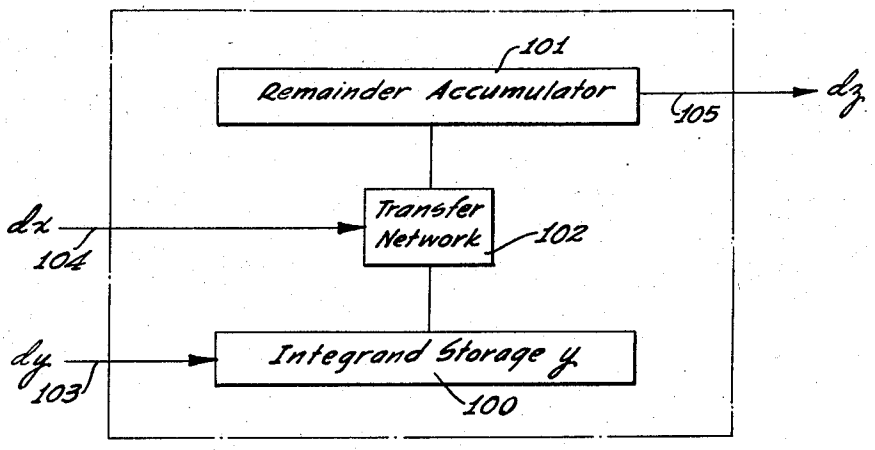
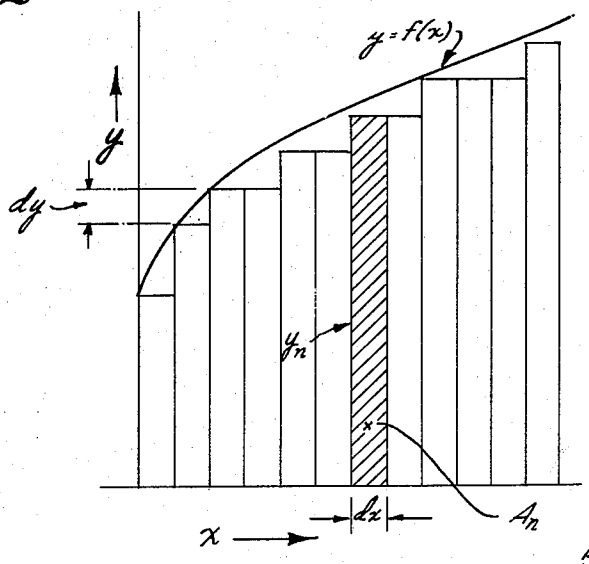


Fig. 2



INVENTORS:
DONALD E. ECKDAHL
JOHN F. DONAN
ALBERT E. WOLFE, JR.
RICHARD E. SPRAGUE
HRANT H. SARKISSIAN
CARL L. ISBORN
BERNARD T. WILSON

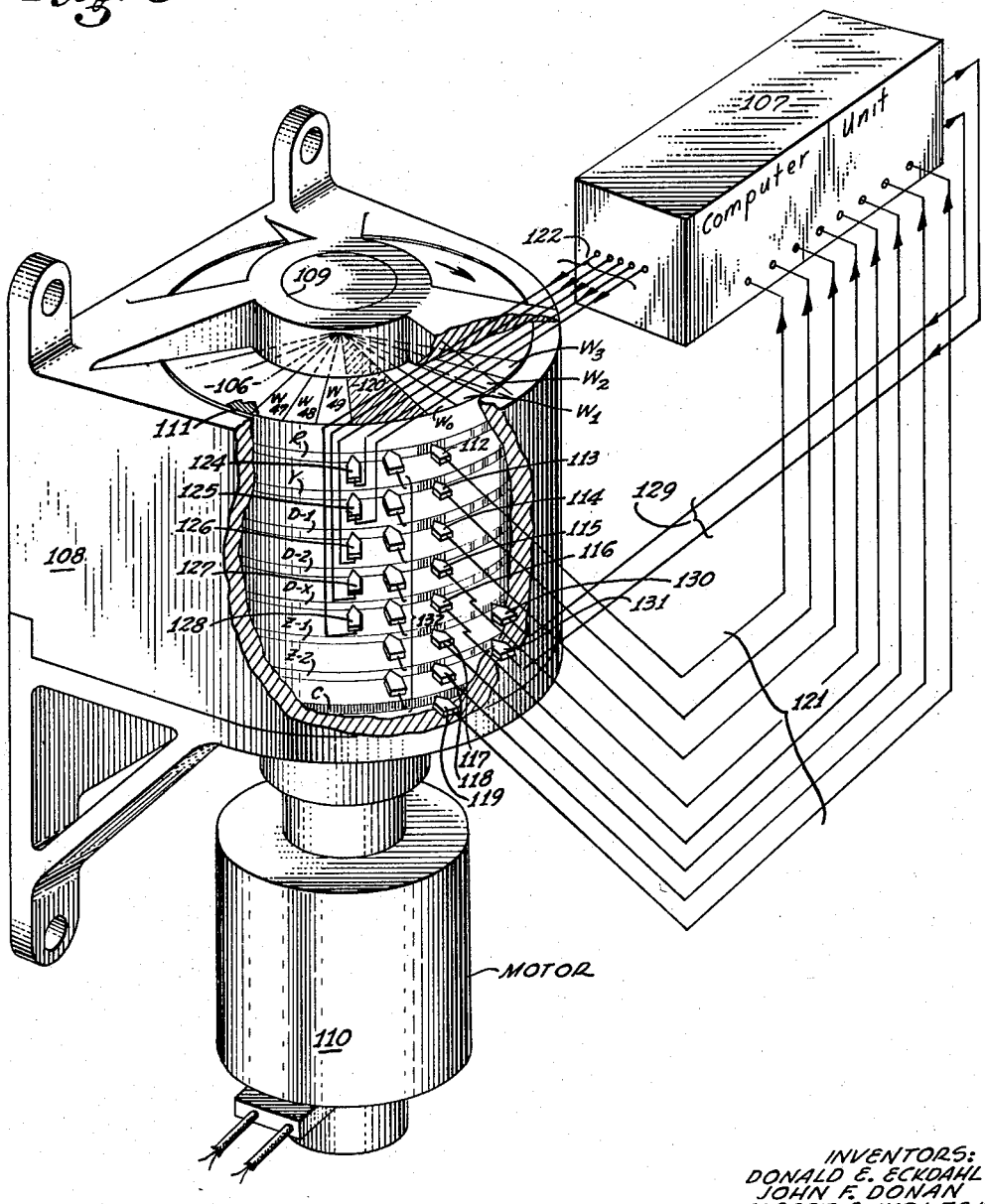
By Mason, Mason & Sheridan
PATENT ATTORNEYS

Feb. 9, 1960 D. E. ECKDAHL ET AL 2,924,381

DIGITAL DIFFERENTIAL ANALYZER

Filed April 22, 1952 48 Sheets-Sheet 2

Fig. 3



INVENTORS:
DONALD E. ECKDAHL
JOHN F. DONAN
ALBERT E. WOLFE, JR.
RICHARD E. SPRAGUE
HRANT H. SARKISSIAN
CARL L. ISBORN
BERNARD T. WILSON

By Mason, Mason & Sheridan
PATENT ATTORNEYS

Feb. 9, 1960 D. E. ECKDAHL ET AL 2,924,381

DIGITAL DIFFERENTIAL ANALYZER

Filed April 22, 1952 48 Sheets-Sheet 3

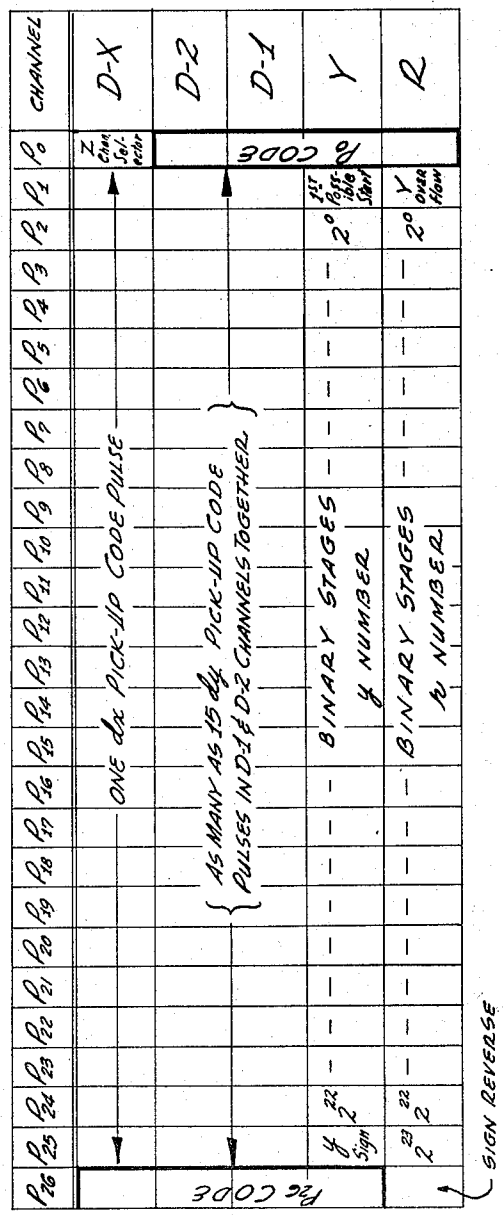


Fig. 4

INVENTORS:
DONALD E. ECKDAHL
JOHN F. DONAN
ALBERT E. WOLFE, JR.
RICHARD E. SPRAGUE
HRANT H. SARKISSIAN
CARL L. ISBORN
BERNARD T. WILSON

By Mason, Mason & Sheridan
PATENT ATTORNEYS

Feb. 9, 1960 D. E. ECKDAHL ET AL 2,924,381

DIGITAL DIFFERENTIAL ANALYZER

Filed April 22, 1952 48 Sheets-Sheet 4

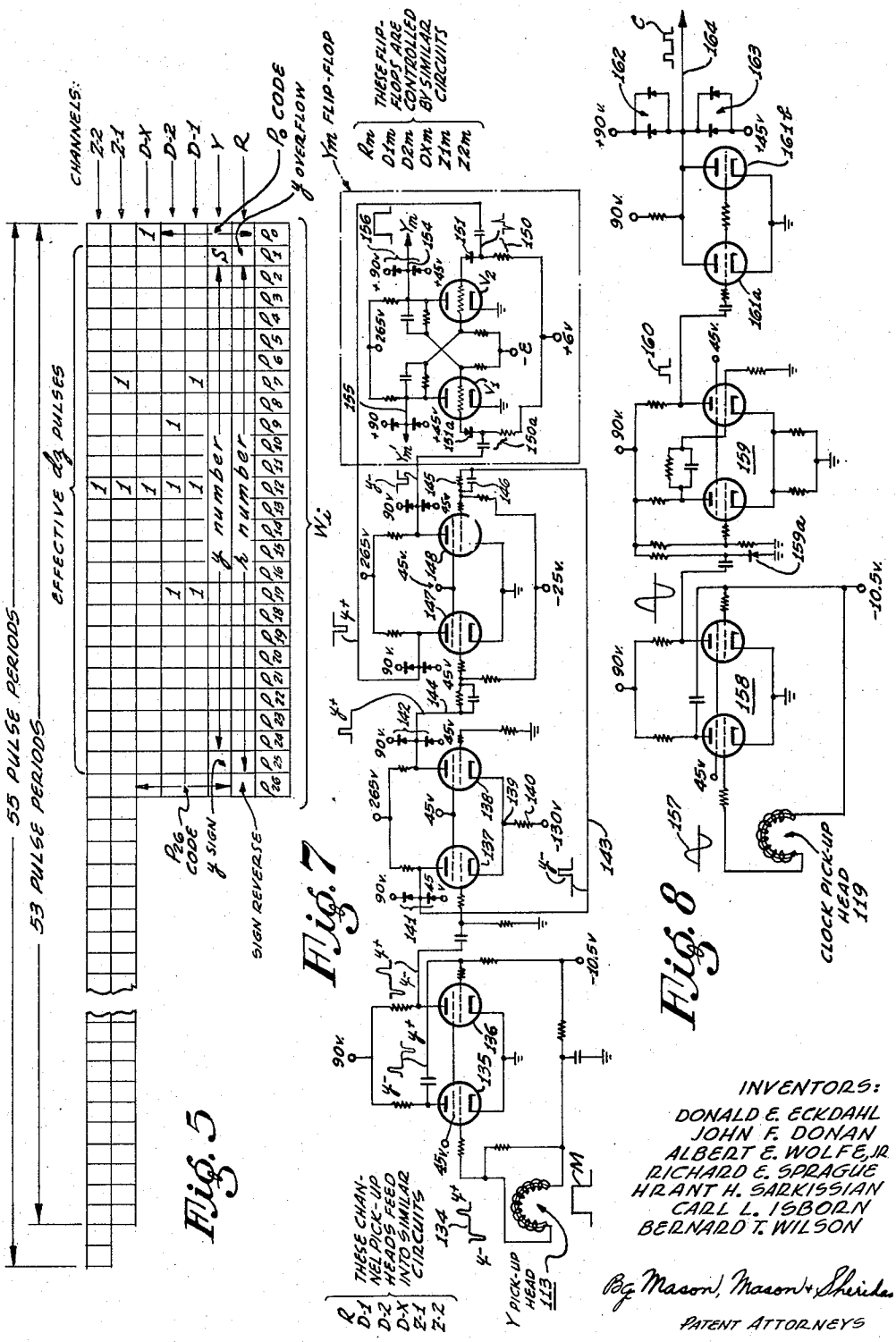


Fig. 5

Fig. 7

Fig. 8

INVENTORS:
DONALD E. ECKDAHL
JOHN F. DONAN
ALBERT E. WOLFE, JR.
RICHARD E. SPRAGUE
HRANT H. SARKISSIAN
CARL L. ISBORN
BERNARD T. WILSON

By Mason, Mason & Sheridan
PATENT ATTORNEYS

DIGITAL DIFFERENTIAL ANALYZER

Filed April 22, 1952 48 Sheets-Sheet 5

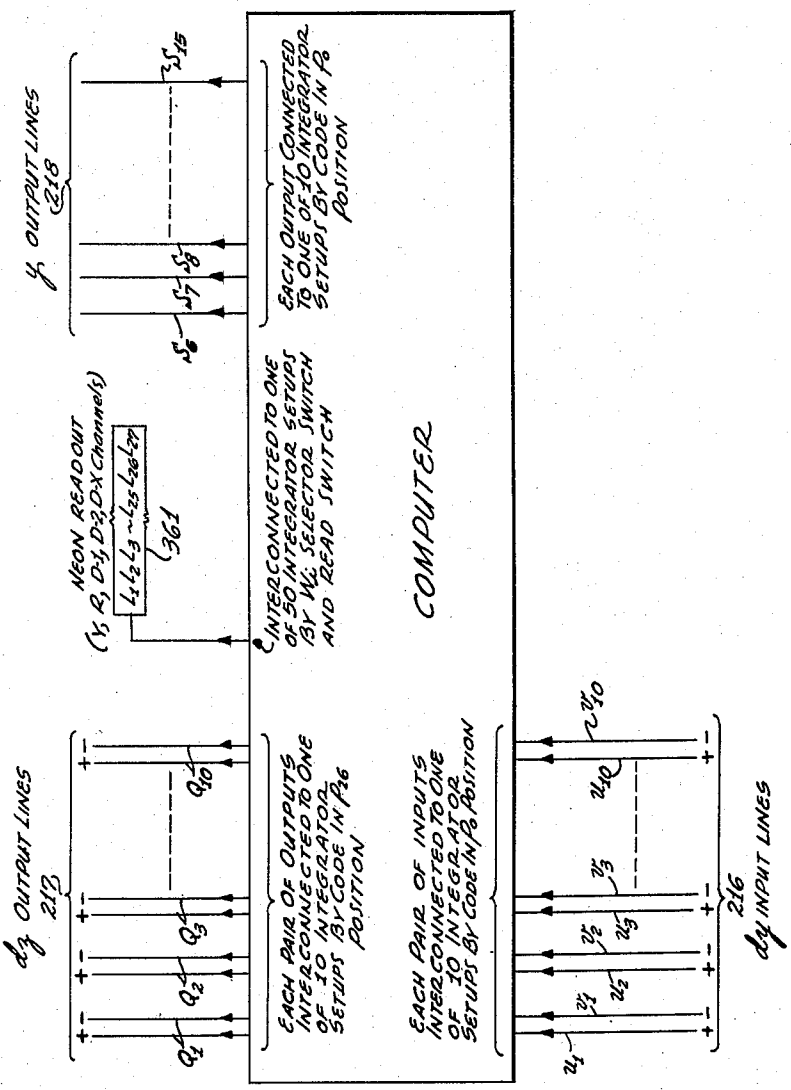


Fig. 6

INVENTORS:
DONALD E. ECKDAHL
JOHN F. DONAN
ALBERT E. WOLFE, JR.
RICHARD E. SPRAGUE
HRANT H. SARKISSIAN
CARL L. ISBORN
BERNARD T. WILSON

By Mason, Mason & Sheridan
PATENT ATTORNEYS

Feb. 9, 1960 D. E. ECKDAHL ET AL 2,924,381

DIGITAL DIFFERENTIAL ANALYZER

Filed April 22, 1952 48 Sheets-Sheet 6

Fig. 11

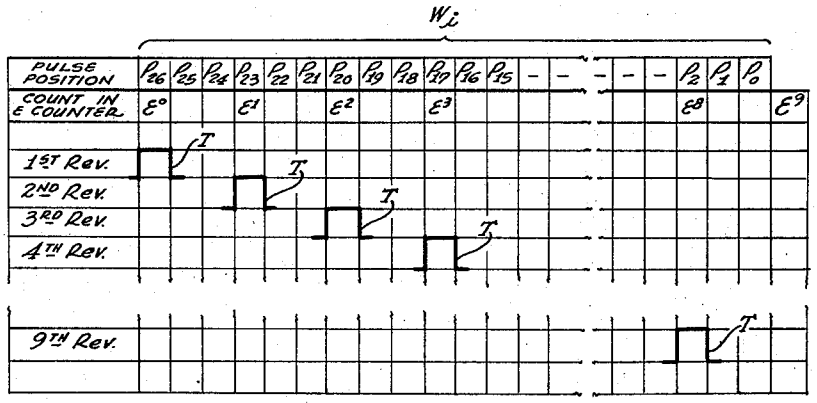


Fig. 10

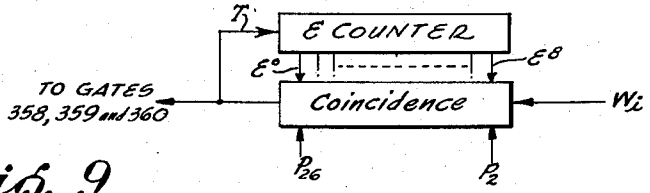
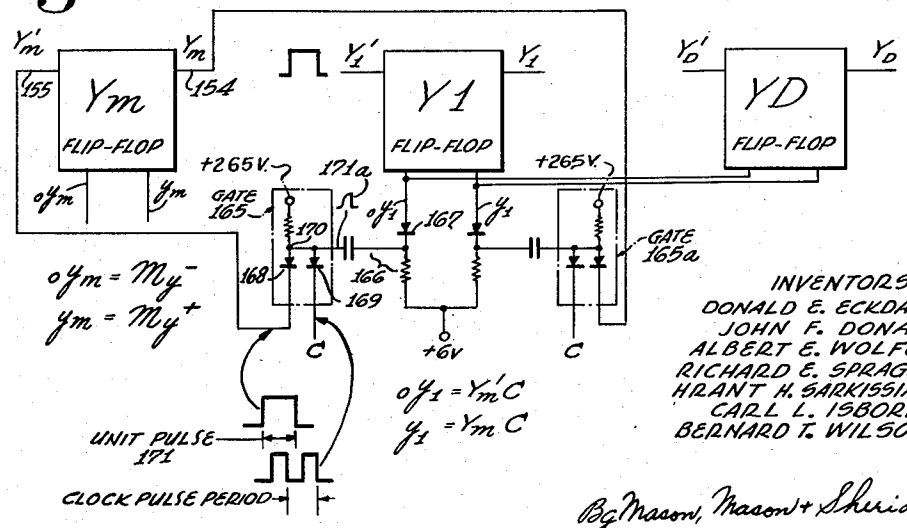


Fig. 9



INVENTORS:
DONALD E. ECKDAHL
JOHN F. DONAN
ALBERT E. WOLFE, JR.
RICHARD E. SPRAGUE
HRANT H. SARKISSIAN
CARL L. ISBORN
BERNARD T. WILSON

By Mason, Mason & Sheridan
PATENT ATTORNEYS

Fig. 13

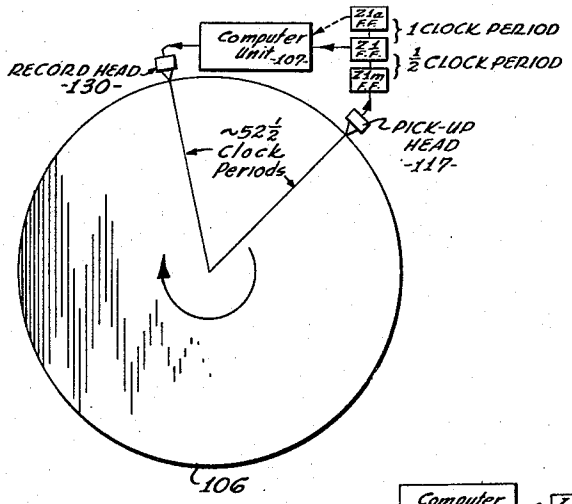


Fig. 14

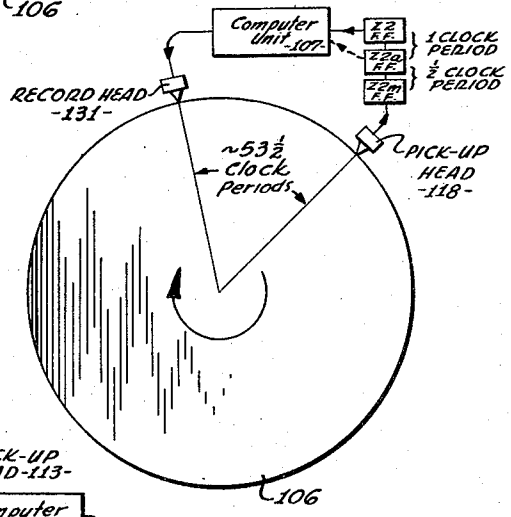
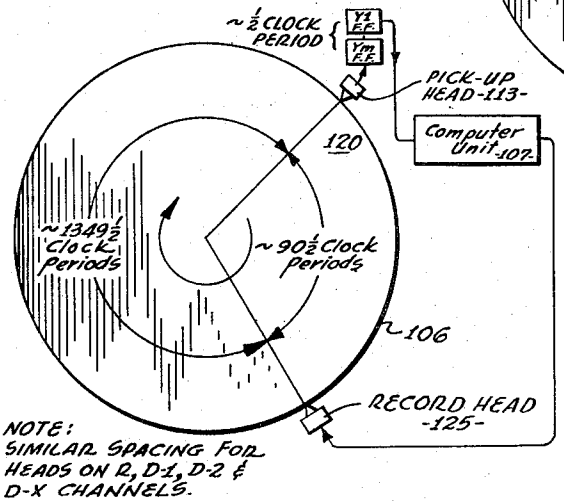


Fig. 12



NOTE:
SIMILAR SPACING FOR HEADS ON R, D-1, D-2 & D-X CHANNELS.

INVENTORS:
DONALD E. ECKDAHL
JOHN F. DONAN
ALBERT E. WOLFE, JR.
RICHARD E. SPRAGUE
HRANT H. SARKISSIAN
CARL L. ISBORN
BERNARD T. WILSON

By Mason, Mason & Sheridan
PATENT ATTORNEYS

Feb. 9, 1960 D. E. ECKDAHL ET AL 2,924,381

DIGITAL DIFFERENTIAL ANALYZER

Filed April 22, 1952 48 Sheets-Sheet 8

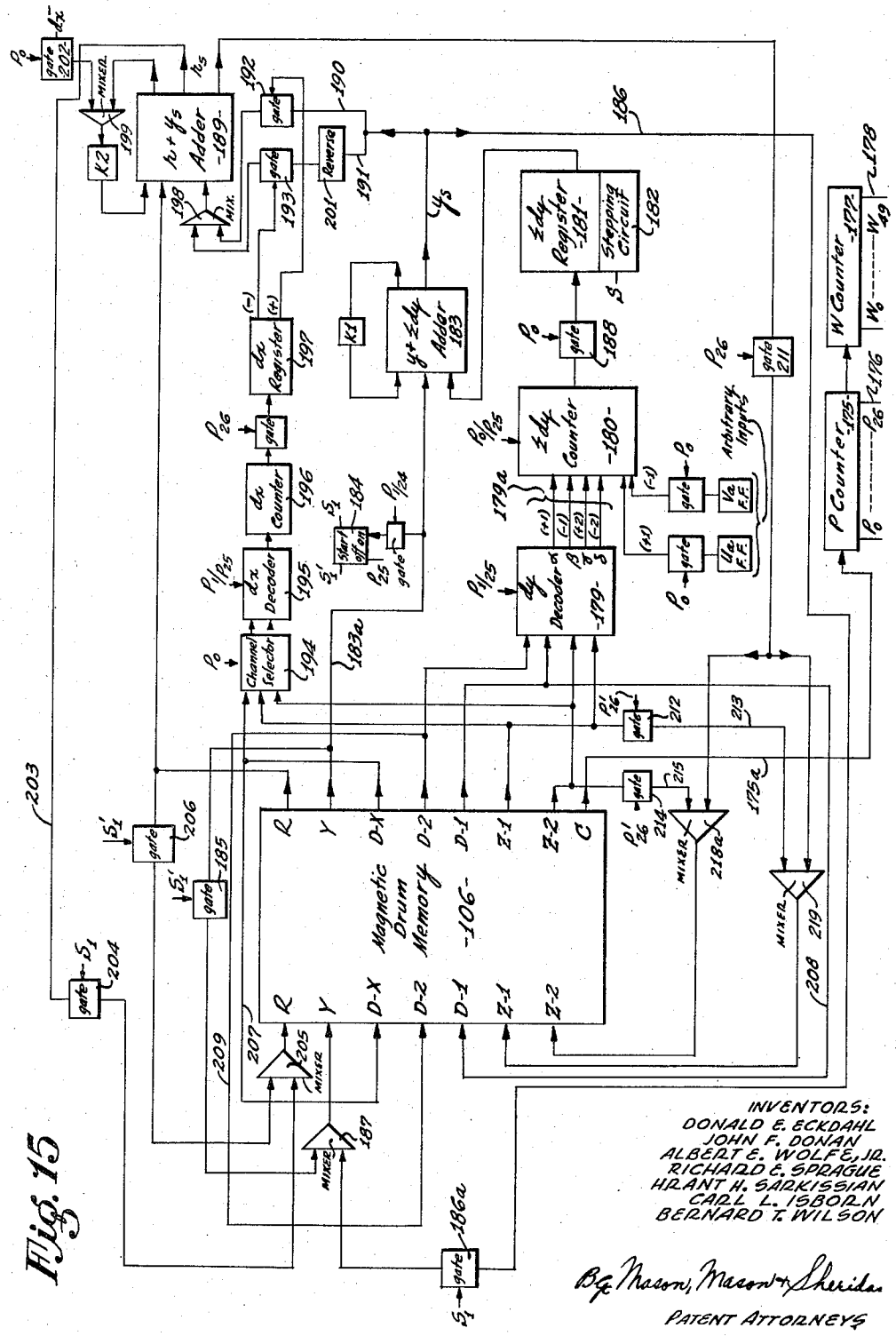


Fig. 15

INVENTORS:
DONALD E. ECKDAHL
JOHN F. DONAN
ALBERT E. WOLFE, JR.
RICHARD E. SPRAGUE
HRANT H. SARKISSIAN
CARL L. ISBORN
BERNARD T. WILSON

By Mason, Mason & Sheridan
PATENT ATTORNEYS

DIGITAL DIFFERENTIAL ANALYZER

Filed April 22, 1952 48 Sheets-Sheet 9

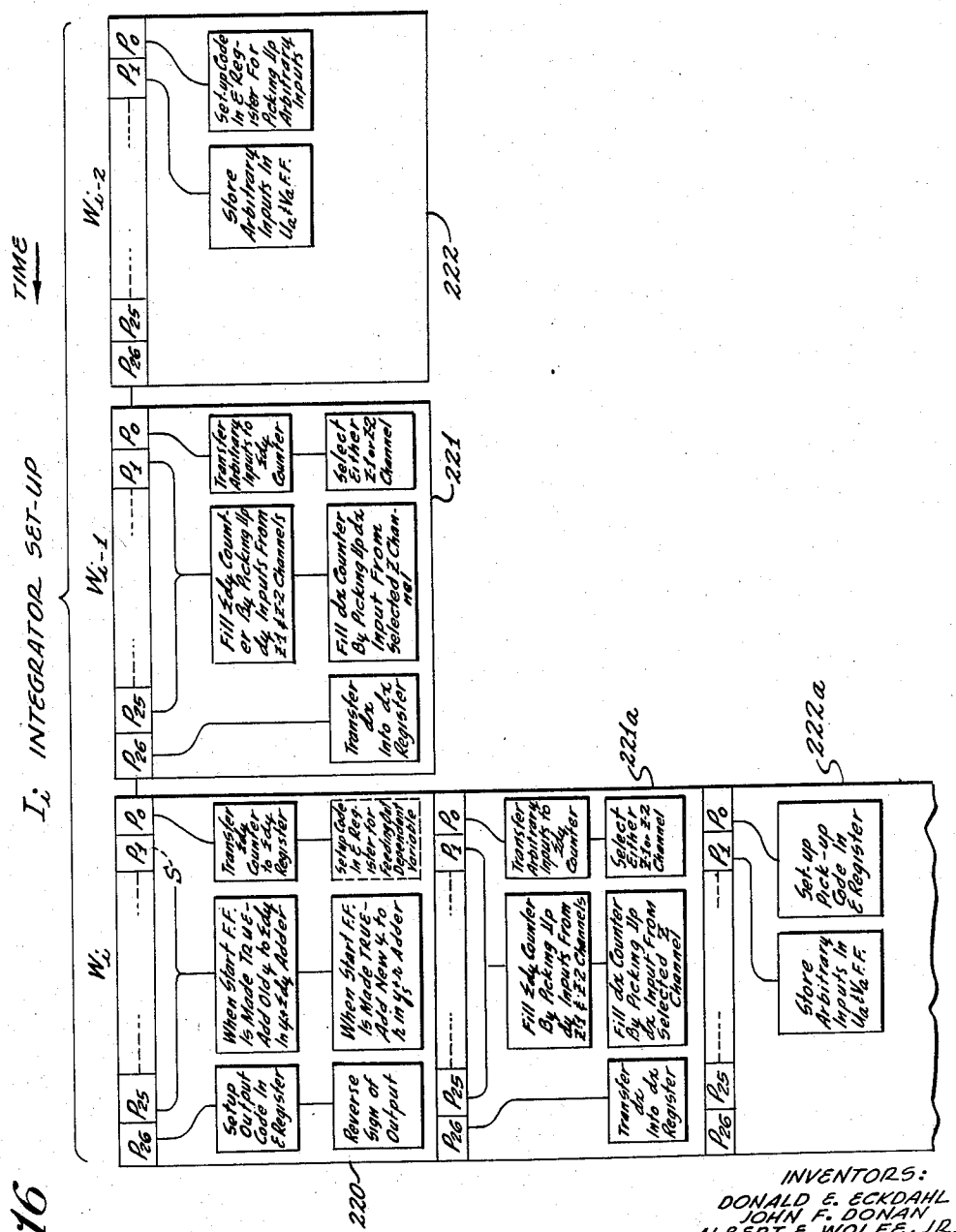


Fig. 16

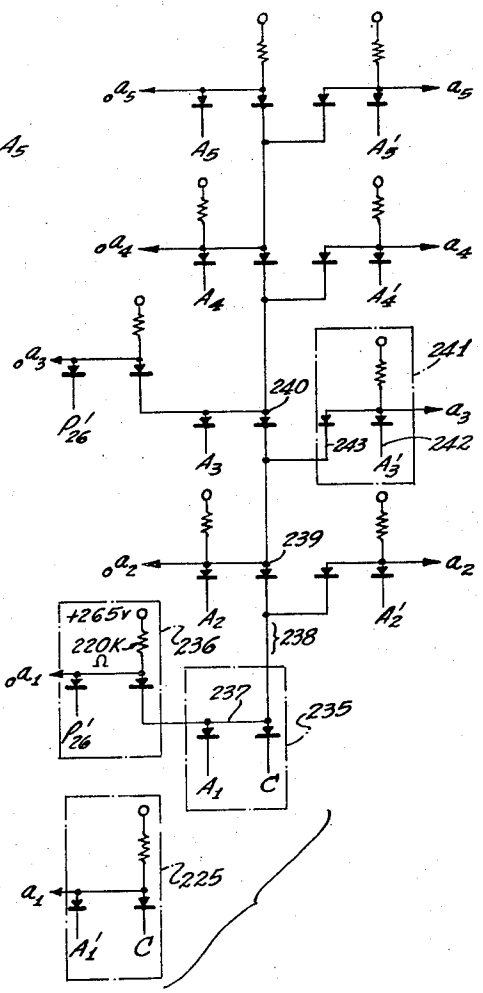
INVENTORS:
DONALD E. ECKDAHL
JOHN F. DONAN
ALBERT E. WOLFE, JR.
RICHARD E. SPRAGUE
HRANT H. SARKISSIAN
CARL L. ISBORN
BERNARD T. WILSON

By Mason, Mason & Sheridan
PATENT ATTORNEYS

Fig. 17

	STAGES					
	A1	A2	A3	A4	A5	
P_0	1	0	1	0	0	$P_0 = A_1 A_2' A_3 A_4' A_5'$
P_1	0	1	1	0	0	$P_1 = A_1' A_2 A_3 A_4' A_5'$
P_2	1	1	1	0	0	$P_2' = A_1' + A_2' + A_3' + A_4 + A_5$
P_3	0	0	0	1	0	
P_4	1	0	0	1	0	
P_5	0	1	0	1	0	$P_5 = A_1' A_2 A_3' A_4 A_5'$
P_6	1	1	0	1	0	
P_7	0	0	1	1	0	
P_8	1	0	1	1	0	$P_8 = A_1 A_2' A_3 A_4 A_5'$
P_9	0	1	1	1	0	
P_{10}	1	1	1	1	0	
P_{11}	0	0	0	0	1	$P_{11} = A_1' A_2' A_3' A_4' A_5$
P_{12}	1	0	0	0	1	
P_{13}	0	1	0	0	1	
P_{14}	1	1	0	0	1	$P_{14} = A_1 A_2 A_3' A_4' A_5$
P_{15}	0	0	1	0	1	
P_{16}	1	0	1	0	1	
P_{17}	0	1	1	0	1	$P_{17} = A_1' A_2 A_3 A_4' A_5$
P_{18}	1	1	1	0	1	
P_{19}	0	0	0	1	1	
P_{20}	1	0	0	1	1	$P_{20} = A_1 A_2' A_3' A_4 A_5$
P_{21}	0	1	0	1	1	
P_{22}	1	1	0	1	1	
P_{23}	0	0	1	1	1	$P_{23} = A_1' A_2' A_3 A_4 A_5$
P_{24}	1	0	1	1	1	
P_{25}	0	1	1	1	1	$P_{25} = A_1' A_2 A_3 A_4 A_5$
P_{26}	1	1	1	1	1	$P_{26} = A_1 A_2 A_3 A_4 A_5$

Fig. 21



INVENTORS:
DONALD E. ECKDAHL
JOHN F. DONAN
ALBERT E. WOLFE, JR.
RICHARD E. SPRAGUE
HRANT H. SARKISSIAN
CARL L. ISBORN
BERNARD T. WILSON

By Mason, Mason & Sheridan
PATENT ATTORNEYS

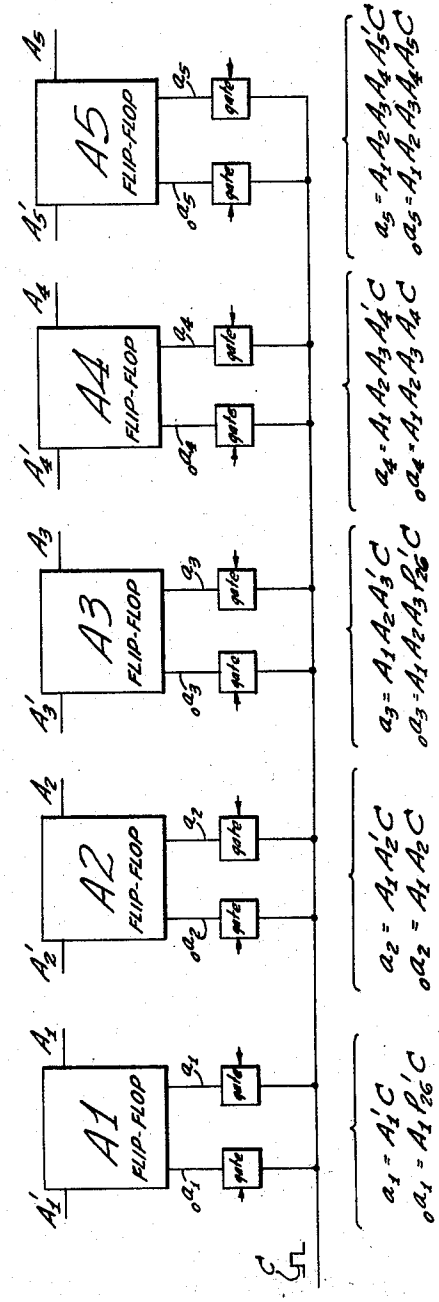


Fig. 18

	S11	S12	S13
	A11	A12	A13
0	0	0	0
1	1	0	0
2	0	1	0
3	1	1	0
4	0	0	1

DECIMAL EQUIVALENT

Fig. 24

SWITCHES FLIP-FLOPS	S7	S8	S9	S10
DECIMAL EQUIVALENT	A7	A8	A9	A10
0	0	0	0	0
1	1	0	0	0
2	0	1	0	0
3	1	1	0	0
4	0	0	1	0
5	1	0	1	0
6	0	1	1	0
7	1	1	1	0
8	0	0	0	1
9	1	0	0	1

$W_0 = A_7' A_8' A_9' A_{10}' A_{11}' A_{12}' A_{13}'$
$W_{39} = A_7 A_8 A_9 A_{10} A_{11} A_{12} A_{13}$

Fig. 23

INVENTORS:
DONALD E. ECKDAHL
JOHN F. DONIAN
ALBERT E. WOLFE, JR.
RICHARD E. SPRAGUE
HRANT H. SARKISSIAN
CARL L. ISBORN
BERNARD T. WILSON

By Mason, Mason & Sheridan
PATENT ATTORNEYS

Fig. 19

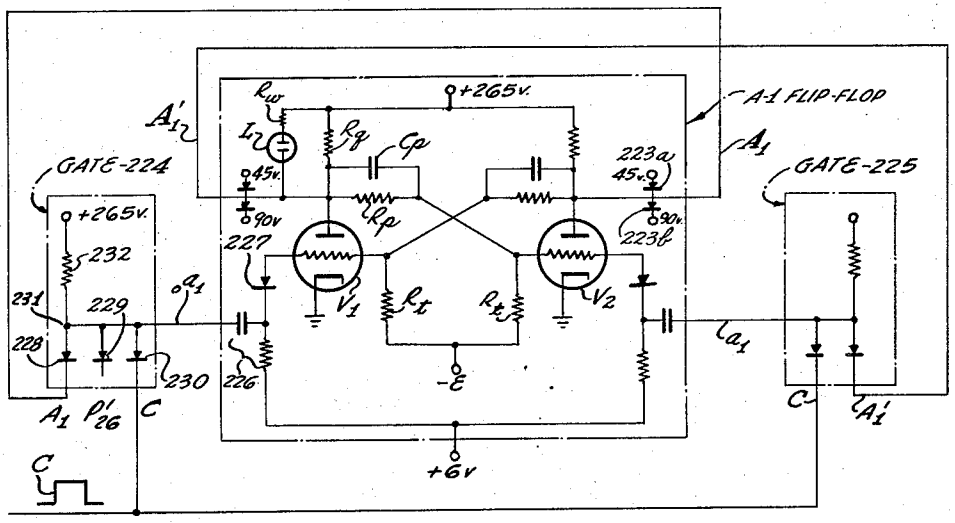
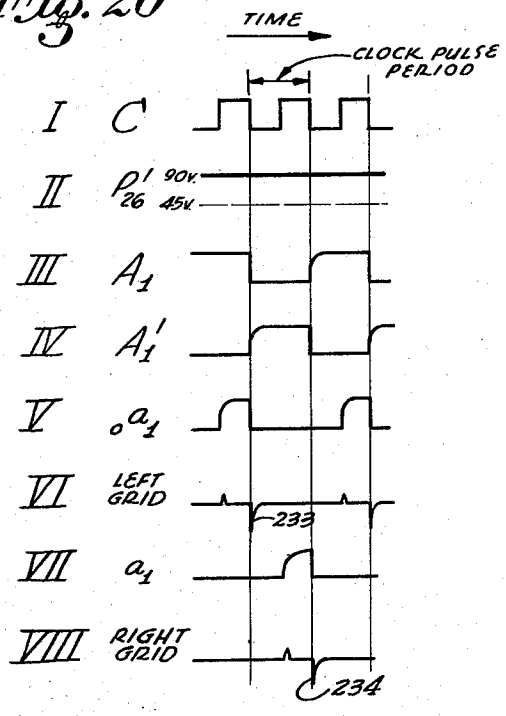


Fig. 20



INVENTORS:
DONALD E. ECKDAHL
JOHN F. DONAN
ALBERT E. WOLFE, JR.
RICHARD E. SPRAGUE
HRANT H. SARKISSIAN
CARL L. ISBORN
BERNARD T. WILSON

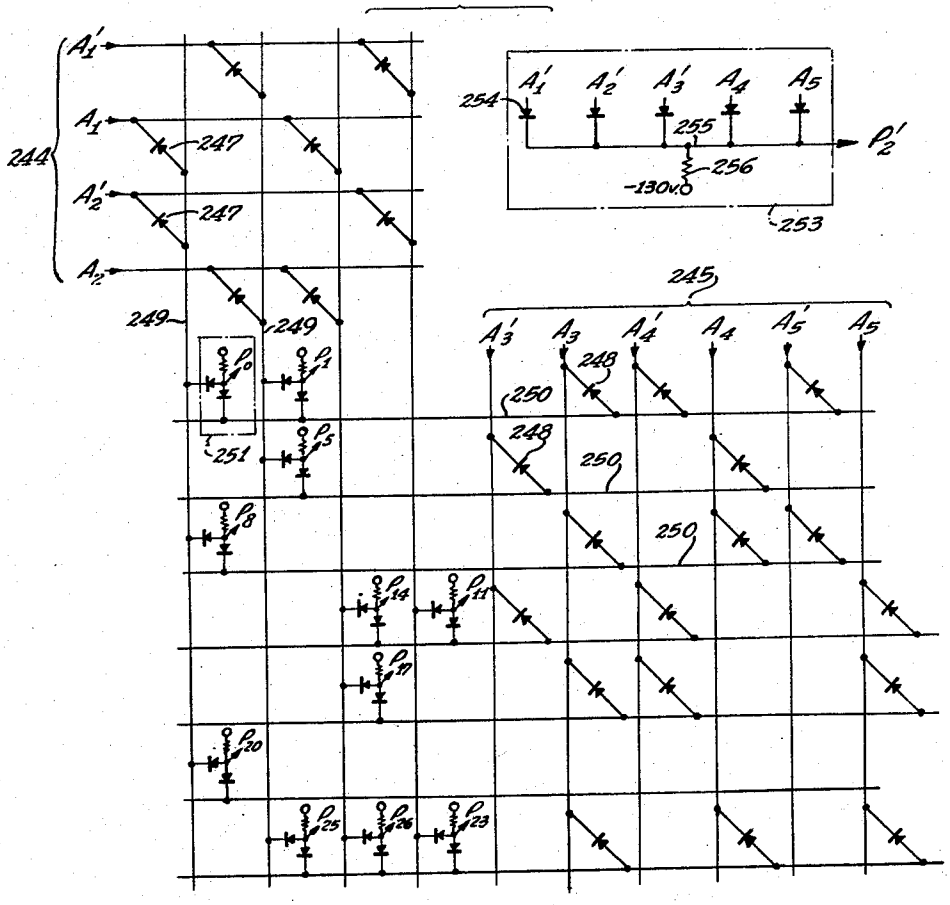
By Mason, Mason & Sheridan
PATENT ATTORNEYS

Feb. 9, 1960 D. E. ECKDAHL ET AL 2,924,381

DIGITAL DIFFERENTIAL ANALYZER

Filed April 22, 1952 48 Sheets-Sheet 13

Fig. 22



INVENTORS:
DONALD E. ECKDAHL
JOHN F. DONAN
ALBERT E. WOLFE, JR.
RICHARD E. SPRAGUE
HRANT H. SARKISSIAN
CARL L. ISBORN
BERNARD T. WILSON

Bg Mason, Mason & Sheridan
PATENT ATTORNEYS

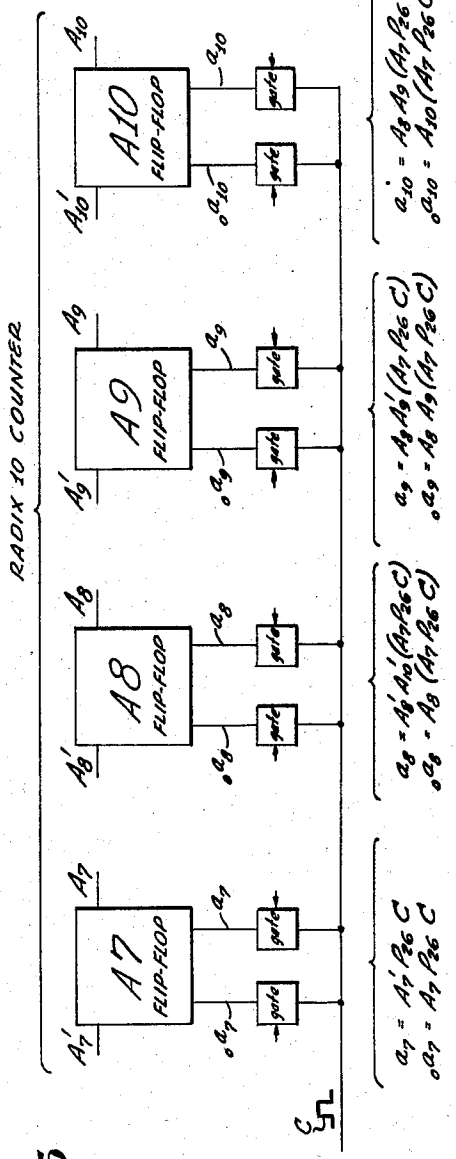


Fig. 25

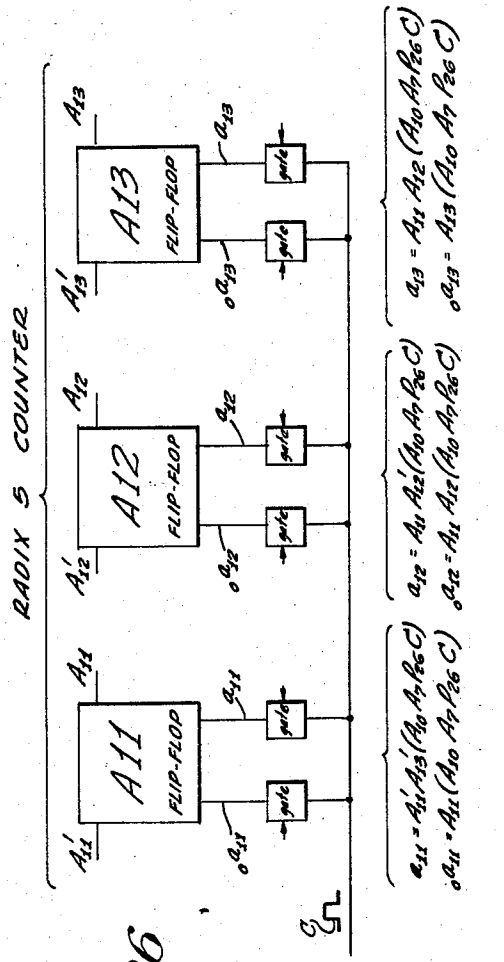


Fig. 26

INVENTORS:
DONALD E. ECKDAHL
JOHN F. DONAN
ALBERT E. WOLFE, JR.
RICHARD E. SPRAGUE
HRANT H. SARKISSIAN
CARL L. ISBORN
BERNARD T. WILSON

By Mason, Mason & Sheridan
PATENT ATTORNEYS

DIGITAL DIFFERENTIAL ANALYZER

Filed April 22, 1952 48 Sheets-Sheet 15

Fig. 27

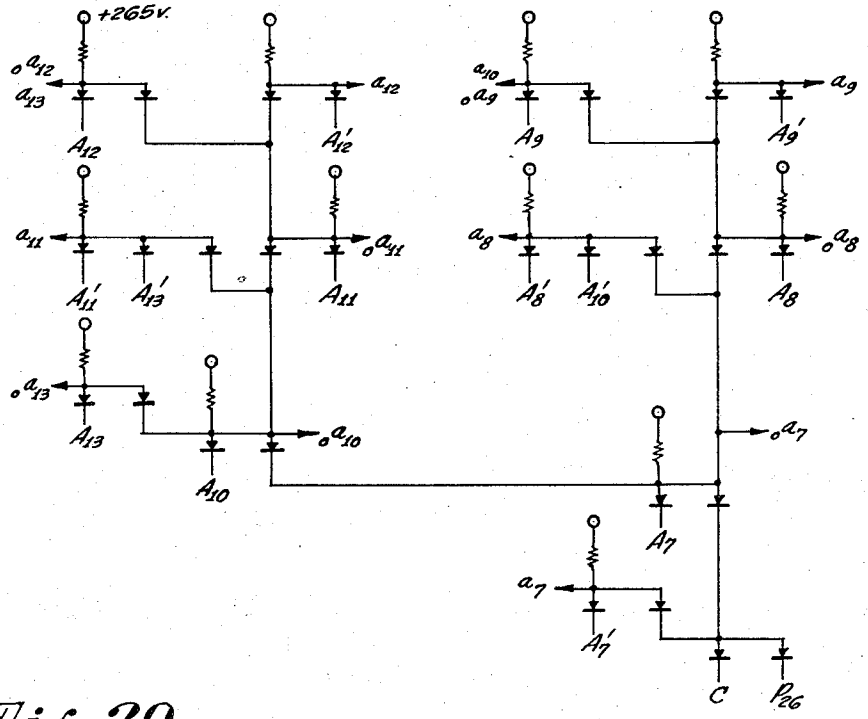
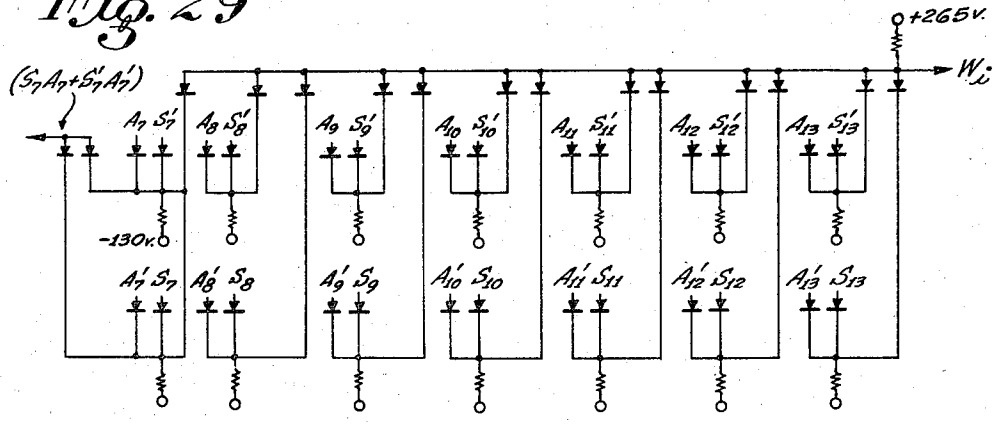


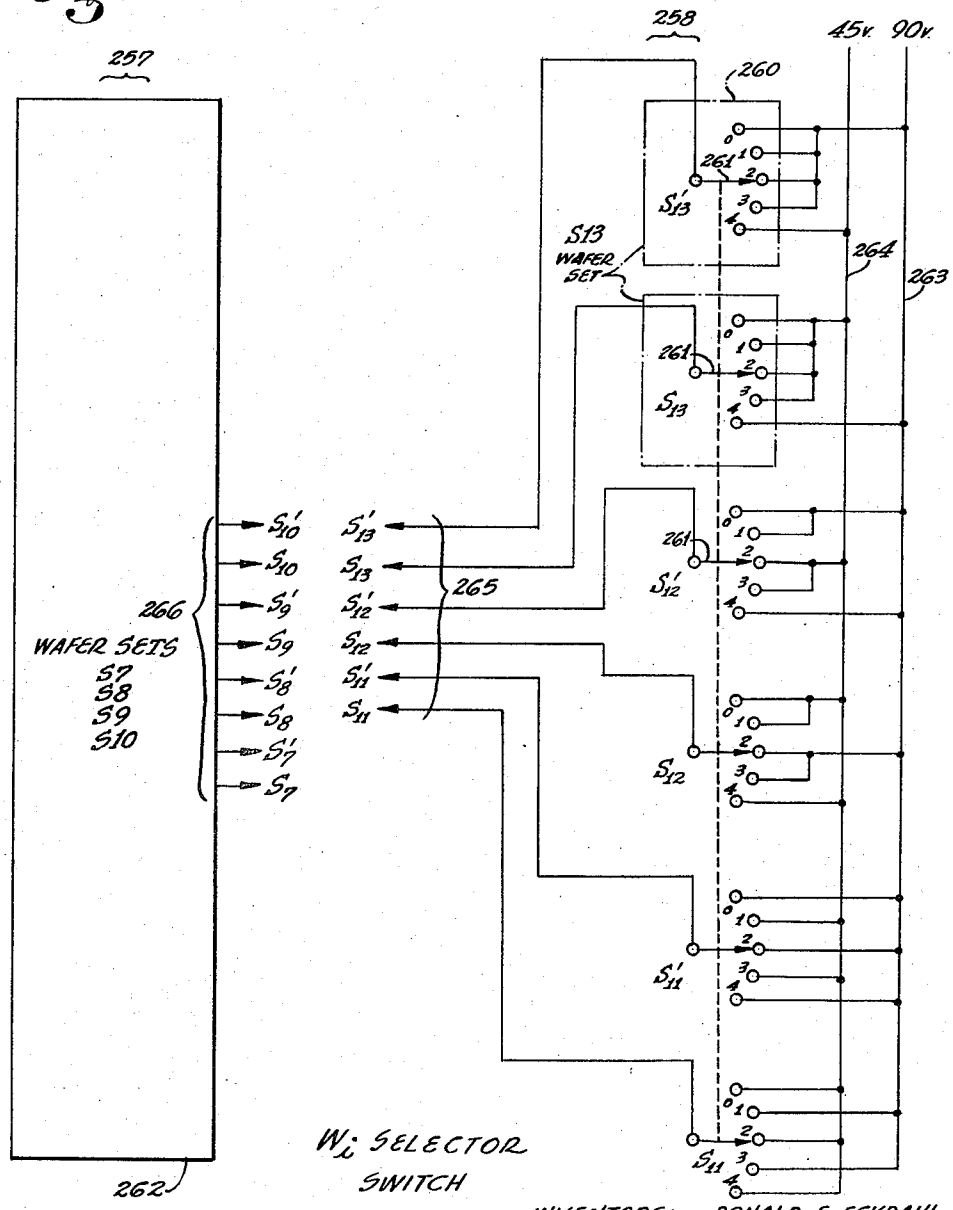
Fig. 29



INVENTORS: DONALD E. ECKDAHL
JOHN F. DONAN
ALBERT E. WOLFE, JR.
RICHARD E. SPRAGUE
HRANT H. SARKISSIAN
CARL L. ISBORN
BERNARD T. WILSON

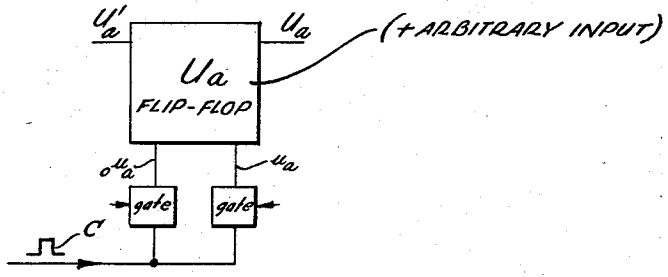
By Mason, Mason & Sheridan
PATENT ATTORNEYS

Fig. 28



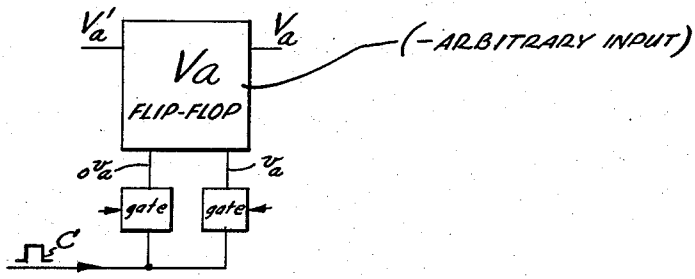
INVENTORS: DONALD E. ECKDAHL
JOHN F. DONAN
ALBERT E. WOLFE, JR.
RICHARD E. SPRAGUE
HRANT H. SARKISSIAN
CARL L. ISBORN
BERNARD T. WILSON
By Mason, Mason & Sheridan
PATENT ATTORNEYS

Fig. 34



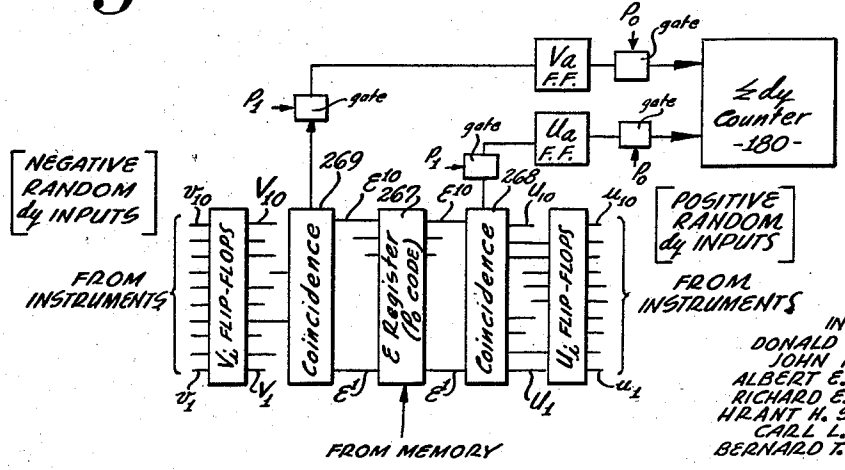
$$U_a = P_1 (U_1 \varepsilon^1 + U_2 \varepsilon^2 + U_3 \varepsilon^3 + U_4 \varepsilon^4 + U_5 \varepsilon^5 + U_6 \varepsilon^6 + U_7 \varepsilon^7 + U_8 \varepsilon^8 + U_9 \varepsilon^9 + U_{10} \varepsilon^{10}) C$$
$$_0U_a = P_0 C$$

Fig. 33



$$V_a = P_1 (V_1 \varepsilon^1 + V_2 \varepsilon^2 + V_3 \varepsilon^3 + V_4 \varepsilon^4 + V_5 \varepsilon^5 + V_6 \varepsilon^6 + V_7 \varepsilon^7 + V_8 \varepsilon^8 + V_9 \varepsilon^9 + V_{10} \varepsilon^{10}) C$$
$$_0V_a = P_0 C$$

Fig. 30



INVENTORS:
DONALD E. ECKDAHL
JOHN F. DONAN
ALBERT E. WOLFE, JR.
RICHARD E. SPRAGUE
HRANT H. SARKISSIAN
CARL L. ISBORN
BERNARD T. WILSON

By Mason, Mason & Sheridan
PATENT ATTORNEYS

DIGITAL DIFFERENTIAL ANALYZER

Filed April 22, 1952 48 Sheets-Sheet 18

Fig. 35

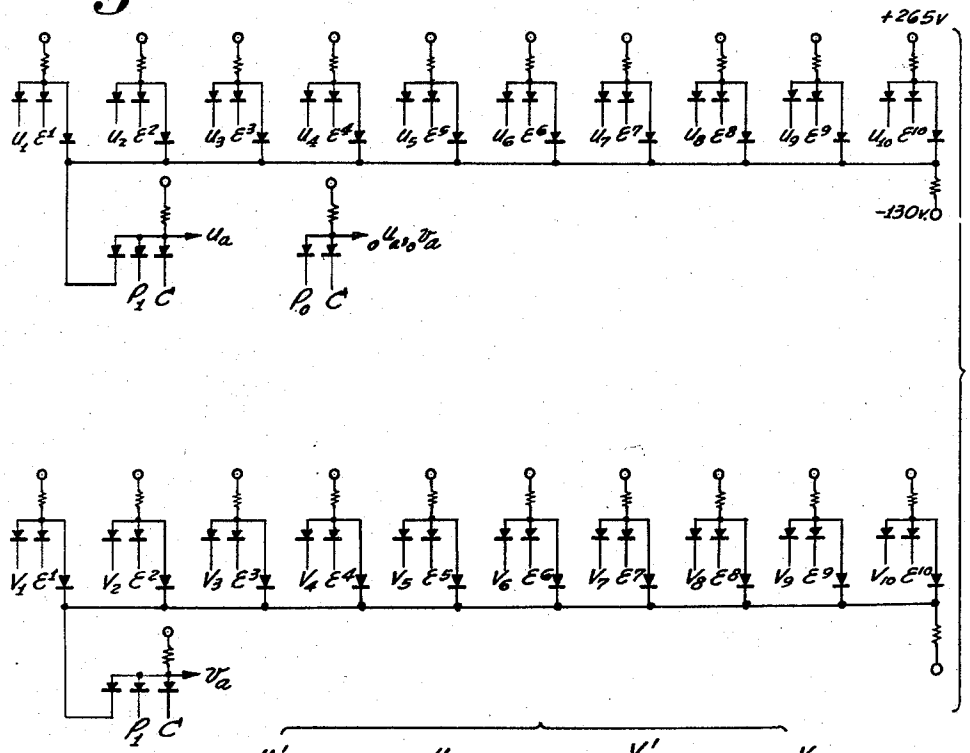
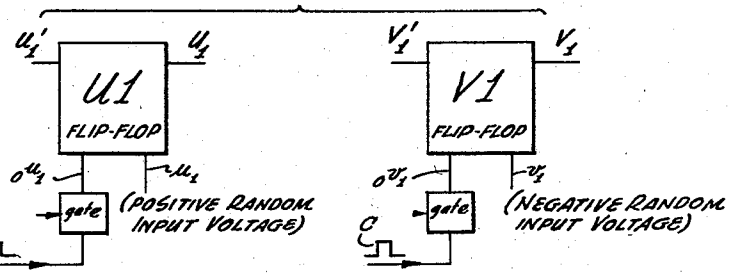


Fig. 31



$U_1 = 1^+$ $U_2 = 2^+$ $U_3 = 3^+$ $U_4 = 4^+$ $U_5 = 5^+$ $U_6 = 6^+$ $U_7 = 7^+$ $U_8 = 8^+$ $U_9 = 9^+$ $U_{10} = 10^+$

$P_2 U_a E^1 C$ $P_2 U_a E^2 C$ $P_2 U_a E^3 C$ $P_2 U_a E^4 C$ $P_2 U_a E^5 C$ $P_2 U_a E^6 C$ $P_2 U_a E^7 C$ $P_2 U_a E^8 C$ $P_2 U_a E^9 C$ $P_2 U_a E^{10} C$

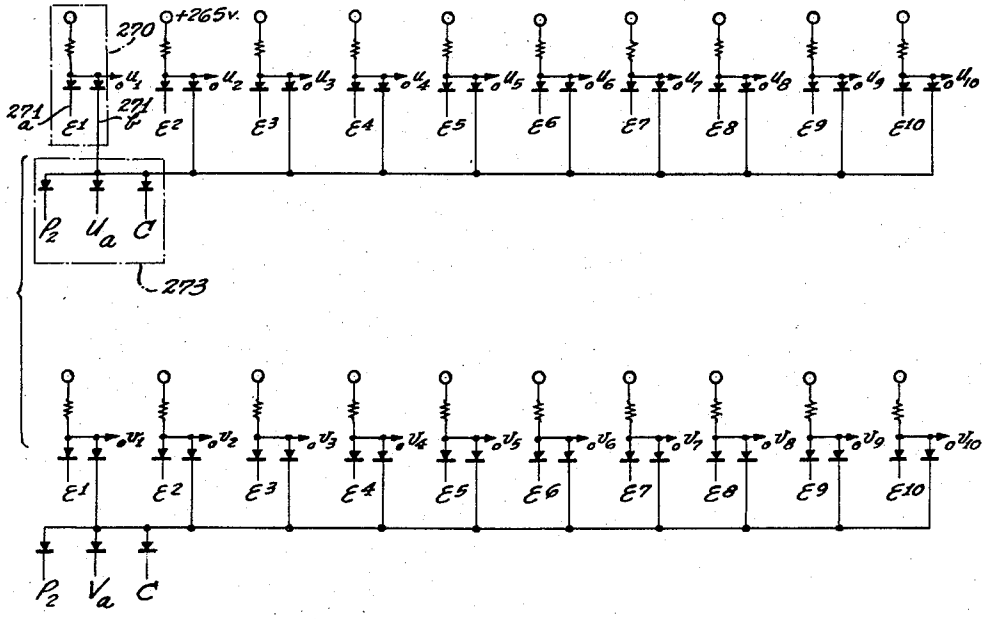
$V_1 = 1^-$ $V_2 = 2^-$ $V_3 = 3^-$ $V_4 = 4^-$ $V_5 = 5^-$ $V_6 = 6^-$ $V_7 = 7^-$ $V_8 = 8^-$ $V_9 = 9^-$ $V_{10} = 10^-$

$P_2 V_a E^1 C$ $P_2 V_a E^2 C$ $P_2 V_a E^3 C$ $P_2 V_a E^4 C$ $P_2 V_a E^5 C$ $P_2 V_a E^6 C$ $P_2 V_a E^7 C$ $P_2 V_a E^8 C$ $P_2 V_a E^9 C$ $P_2 V_a E^{10} C$

INVENTORS: DONALD E. ECKDAHL
JOHN F. DONAN
ALBERT E. WOLFE, JR.
RICHARD E. SPRAGUE
HRANT H. SARKISSIAN
CARL L. ISBORN
BERNARD T. WILSON

By Mason, Mason & Sheridan
PATENT ATTORNEYS

Fig. 32



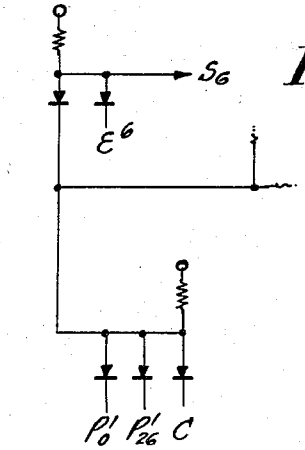


Fig. 32a

$S_6 = E^6 P_0' P_{26}' C$

INVENTORS:
DONALD E. ECKDAHL
JOHN F. DONAN
ALBERT E. WOLFE, JR.
RICHARD E. SPRAGUE
HRANT H. SARKISSIAN
CARL L. IGBORN
BERNARD T. WILSON

By Mason, Mason & Sheridan
PATENT ATTORNEYS

Fig. 36

Edy COUNTER

DECIMAL EQUIVALENT	2^0 B1	2^1 B2	2^2 B3	2^3 B4	Sign B5
+15	1	1	1	1	0
+14	0	1	1	1	0
+13	1	0	1	1	0
+12	0	0	1	1	0
+11	1	1	0	1	0
+10	0	1	0	1	0
+9	1	0	0	1	0
+8	0	0	0	1	0
+7	1	1	1	0	0
+6	0	1	1	0	0
+5	1	0	1	0	0
+4	0	0	1	0	0
+3	1	1	0	0	0
+2	0	1	0	0	0
+1	1	0	0	0	0
±0	0	0	0	0	0
-1	1	1	1	1	1
-2	0	1	1	1	1
-3	1	0	1	1	1
-4	0	0	1	1	1
-5	1	1	0	1	1
-6	0	1	0	1	1
-7	1	0	0	1	1
-8	0	0	0	1	1
-9	1	1	1	0	1
-10	0	1	1	0	1
-11	1	0	1	0	1
-12	0	0	1	0	1
-13	1	1	0	0	1
-14	0	1	0	0	1
-15	1	0	0	1	1

Fig. 37

Y NUMBER

Pulse Position / Stage	P_1	P_2 2^0	P_3 2^1	P_4 2^2	P_5 2^3	–	–	SIGN POSITION P_{24} –	P_{25} –	P_{26} –
n										
+8		0	0	0	1			0	1	
+7		1	1	1	0			0	1	
+6		0	1	1	0			0	1	
+5		1	0	1	0			0	1	
+4		0	0	1	0			0	1	
+3	START PULSE	1	1	0	0			0	1	BLANK
+2		0	1	0	0			0	1	
+1		1	0	0	0			0	1	
0		0	0	0	0			0	1	
-1		1	1	1	1			1	0	
-2		0	1	1	1			1	0	
-3		1	0	1	1			1	0	
-4		0	0	1	1			1	0	
-5		1	1	0	1			1	0	
-6		0	1	0	1			1	0	
-7		1	0	0	1			1	0	
-8		0	0	0	1			1	0	
n										

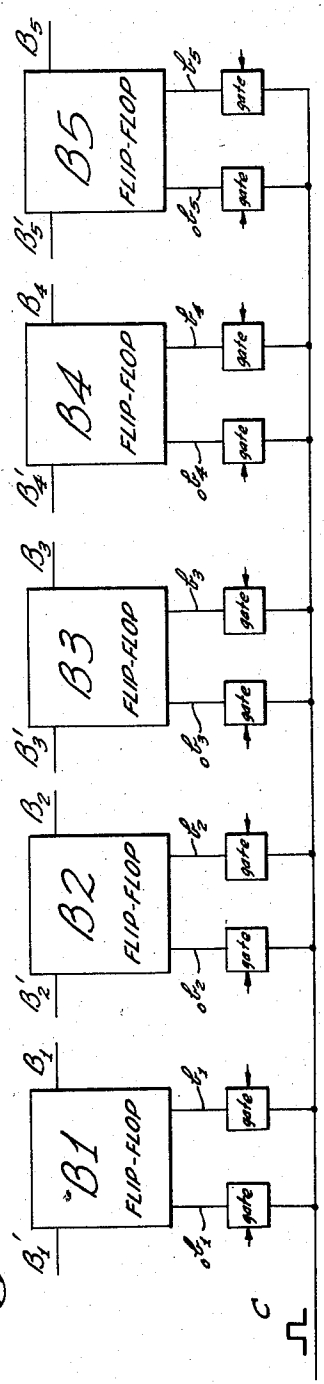
INVENTORS:
DONALD E. ECKDAHL
JOHN F. DONAN
ALBERT E. WOLFE, JR.
RICHARD E. SPRAGUE
HRANT H. SARKISSIAN
CARL L. ISBORN
BERNARD T. WILSON

By Mason, Mason & Sheridan
PATENT ATTORNEYS

DIGITAL DIFFERENTIAL ANALYZER

Filed April 22, 1952 48 Sheets-Sheet 21

Fig. 38



Positive Unit Counting	$b_1 = \alpha B_1' C$ $\circ b_1 = \alpha B_1 C$	$b_2 = \alpha B_1 B_2' C$ $\circ b_2 = \alpha B_1 B_2 C$	$b_3 = \alpha B_1 B_2 B_3' C$ $\circ b_3 = \alpha B_1 B_2 B_3 C$	$b_4 = \alpha B_1 B_2 B_3 B_4' C$ $\circ b_4 = \alpha B_1 B_2 B_3 B_4 C$	$b_5 = \alpha B_1 B_2 B_3 B_4 B_5' C$ $\circ b_5 = \alpha B_1 B_2 B_3 B_4 B_5 C$
Negative Unit Counting	$b_1 = \beta B_1' C$ $\circ b_1 = \beta B_1 C$	$b_2 = \beta B_1' B_2' C$ $\circ b_2 = \beta B_1' B_2 C$	$b_3 = \beta B_1' B_2' B_3' C$ $\circ b_3 = \beta B_1' B_2' B_3 C$	$b_4 = \beta B_1' B_2' B_3' B_4' C$ $\circ b_4 = \beta B_1' B_2' B_3' B_4 C$	$b_5 = \beta B_1' B_2' B_3' B_4' B_5' C$ $\circ b_5 = \beta B_1' B_2' B_3' B_4' B_5 C$
Positive Two Unit Counting	$b_1 = 0$ $\circ b_1 = 0$	$b_2 = \gamma B_2' C$ $\circ b_2 = \gamma B_2 C$	$b_3 = \gamma B_2 B_3' C$ $\circ b_3 = \gamma B_2 B_3 C$	$b_4 = \gamma B_2 B_3 B_4' C$ $\circ b_4 = \gamma B_2 B_3 B_4 C$	$b_5 = \gamma B_2 B_3 B_4 B_5' C$ $\circ b_5 = \gamma B_2 B_3 B_4 B_5 C$
Negative Two Unit Counting	$b_1 = 0$ $\circ b_1 = 0$	$b_2 = \delta B_2' C$ $\circ b_2 = \delta B_2 C$	$b_3 = \delta B_2' B_3' C$ $\circ b_3 = \delta B_2' B_3 C$	$b_4 = \delta B_2' B_3' B_4' C$ $\circ b_4 = \delta B_2' B_3' B_4 C$	$b_5 = \delta B_2' B_3' B_4' B_5' C$ $\circ b_5 = \delta B_2' B_3' B_4' B_5 C$
(Reset) for (Add) Input (Carry) Input	$b_1 = u_1 V_1' P_0 C$ $\circ b_1 = (u_1' V_1' + u_1 V_1) P_0 C$ $t_1 = u_1' V_1 P_0 C$	$\circ b_2 = u_2 P_0 C$ $\circ b_2 = V_2' P_0 C$ $t_2 = u_2' V_2 P_0 C$	$\circ b_3 = u_3 P_0 C$ $\circ b_3 = V_3' P_0 C$ $t_3 = u_3' V_3 P_0 C$	$\circ b_4 = u_4 P_0 C$ $\circ b_4 = V_4' P_0 C$ $t_4 = u_4' V_4 P_0 C$	$\circ b_5 = u_5 P_0 C$ $\circ b_5 = V_5' P_0 C$ $t_5 = u_5' V_5 P_0 C$

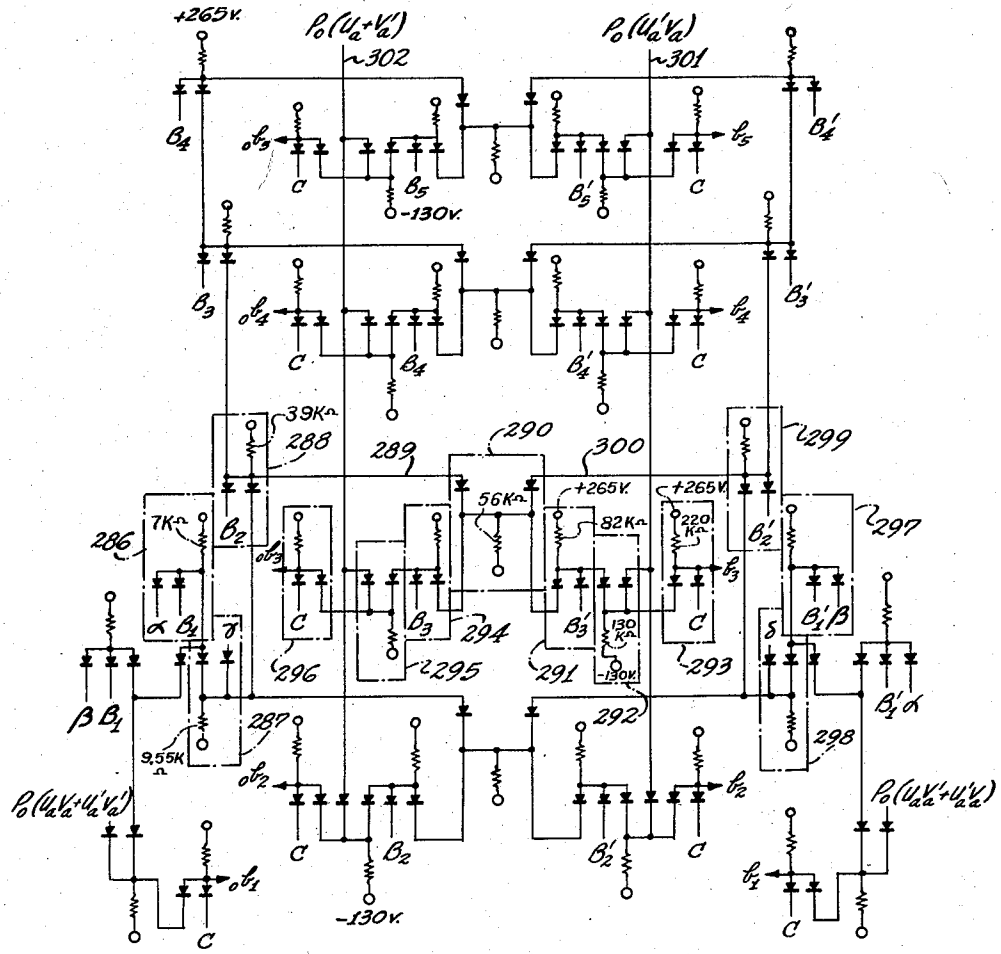
INVENTORS:
DONALD E. ECKDAHL
JOHN F. DONAN
ALBERT E. WOLFE, JR.
RICHARD E. SPRAGUE
HIRANT H. SARKISSIAN
CARL L. ISBORN
BERNARD T. WILSON

By Mason, Mason & Sheridan
PATENT ATTORNEYS

DIGITAL DIFFERENTIAL ANALYZER

Filed April 22, 1952 48 Sheets-Sheet 22

Fig. 39



INVENTORS:
DONALD E. ECKDAHL
JOHN F. DONAN
ALBERT E. WOLFE, JR.
RICHARD E. SPRAGUE
HRANT H. SARKISSIAN
CARL L. ISBORN
BERNARD T. WILSON

By Mason, Mason & Sheridan
PATENT ATTORNEYS

DIGITAL DIFFERENTIAL ANALYZER

Filed April 22, 1952 48 Sheets-Sheet 23

Fig. 40

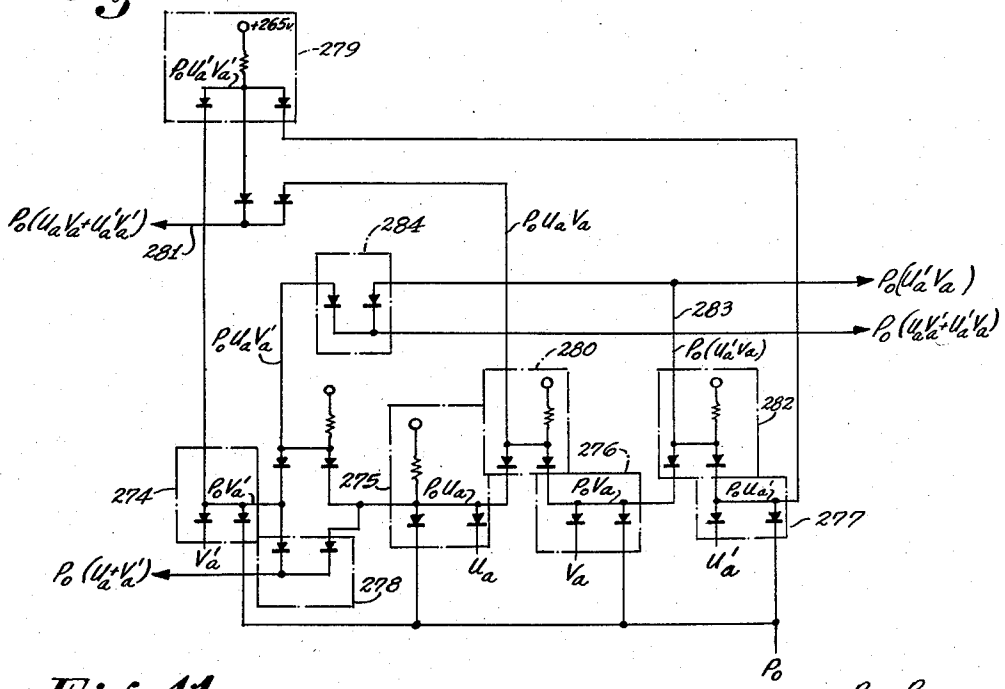
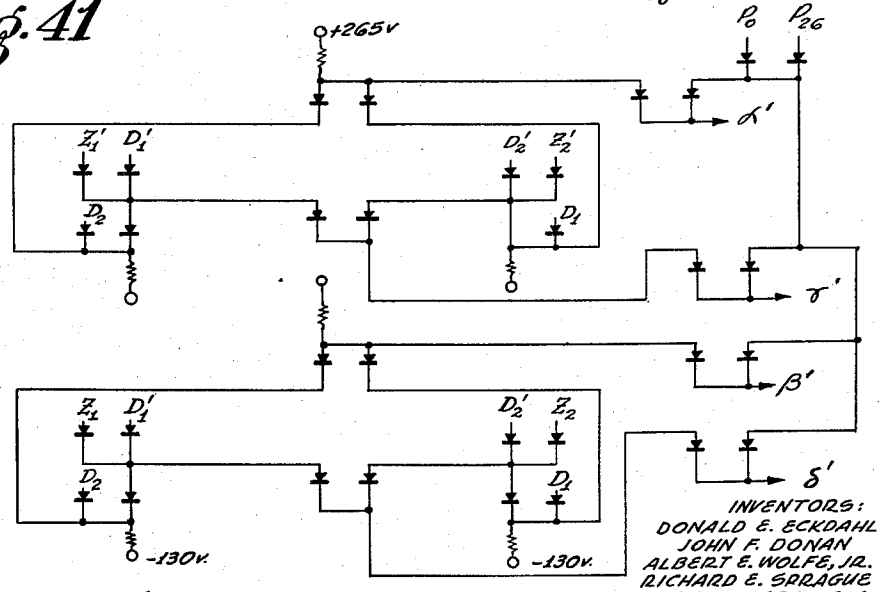


Fig. 41



$\alpha' = (Z_1' + D_1' + D_2)(D_1 + Z_2' + D_2') + P_0 + P_{26}$

$\beta' = (Z_1 + D_1' + D_2)(D_1 + Z_2 + D_2') + P_0 + P_{26}$

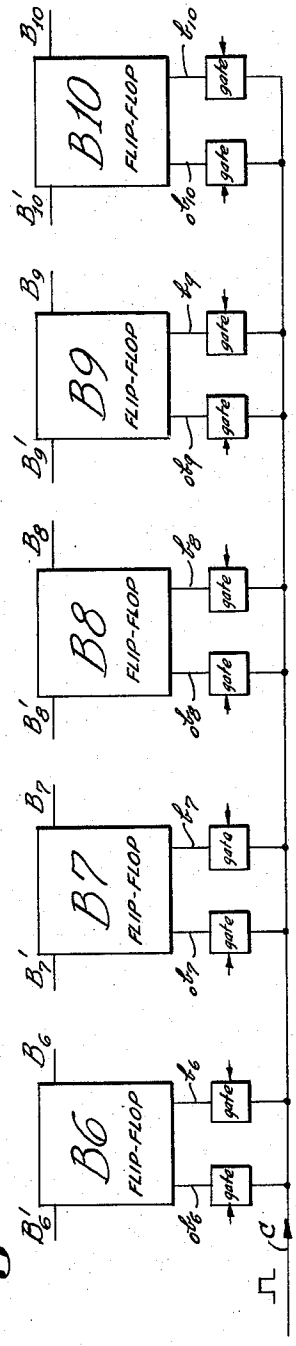
$\gamma' = Z_1' + D_1' + Z_2' + D_2' + P_0 + P_{26}$

$\delta' = Z_1 + D_1' + Z_2 + D_2' + P_0 + P_{26}$

INVENTORS:
DONALD E. ECKDAHL
JOHN F. DONAN
ALBERT E. WOLFE, JR.
RICHARD E. SPRAGUE
HRANT H. SARKISSIAN
CARL L. ISBORN
BERNARD T. WILSON

By Mason, Mason & Sheridan
PATENT ATTORNEYS

Fig. 42



	B_6	B_7	B_8	B_9	B_{10}
TRANSFER	$t_6 = B_1 P_0 G_3 C$ $\circ t_6 = B_1' P_0 G_3 C$	$t_7 = B_2 P_0 G_3 C$ $\circ t_7 = B_2' P_0 G_3 C$	$t_8 = B_3 P_0 G_3 C$ $\circ t_8 = B_3' P_0 G_3 C$	$t_9 = B_4 P_0 G_3 C$ $\circ t_9 = B_4' P_0 G_3 C$	$t_{10} = B_5 P_0 G_3 C$ $\circ t_{10} = B_5' P_0 G_3 C$
STEP	$t_6 = B_7' S_1 C$ $\circ t_6 = B_7' S_1 C$	$t_7 = B_8' S_1 C$ $\circ t_7 = B_8' S_1 C$	$t_8 = B_9' S_1 C$ $\circ t_8 = B_9' S_1 C$	$t_9 = B_0' S_1 C$ $\circ t_9 = B_0' S_1 C$	$t_{10} = 0$ $\circ t_{10} = 0$
READ OUT	$t_6 = G_3'(X_1 R_1 + R_1 R_2 + D_1 R_{01} + D_2 R_{02} + D_x R_{0x}) C$ $\circ t_6 = G_3'(Y_1 R_1 + R_1 R_2 + D_1' R_{01} + D_2' R_{02} + D_x' R_{0x}) C$	$t_7 = G_3' B_6' C$ $\circ t_7 = G_3' B_6' C$	$t_8 = G_3' T(X_1 R_1 + R_1 R_2 + D_1 R_{01} + D_2 R_{02} + D_x R_{0x}) C$ $\circ t_8 = G_3' T(Y_1 R_1 + R_1 R_2 + D_1' R_{01} + D_2' R_{02} + D_x' R_{0x}) C$	$t_9 = G_3' T B_6' C$ $\circ t_9 = G_3' T B_6' C$	$t_{10} = G_3' T B_7' C$ $\circ t_{10} = G_3' T B_7' C$

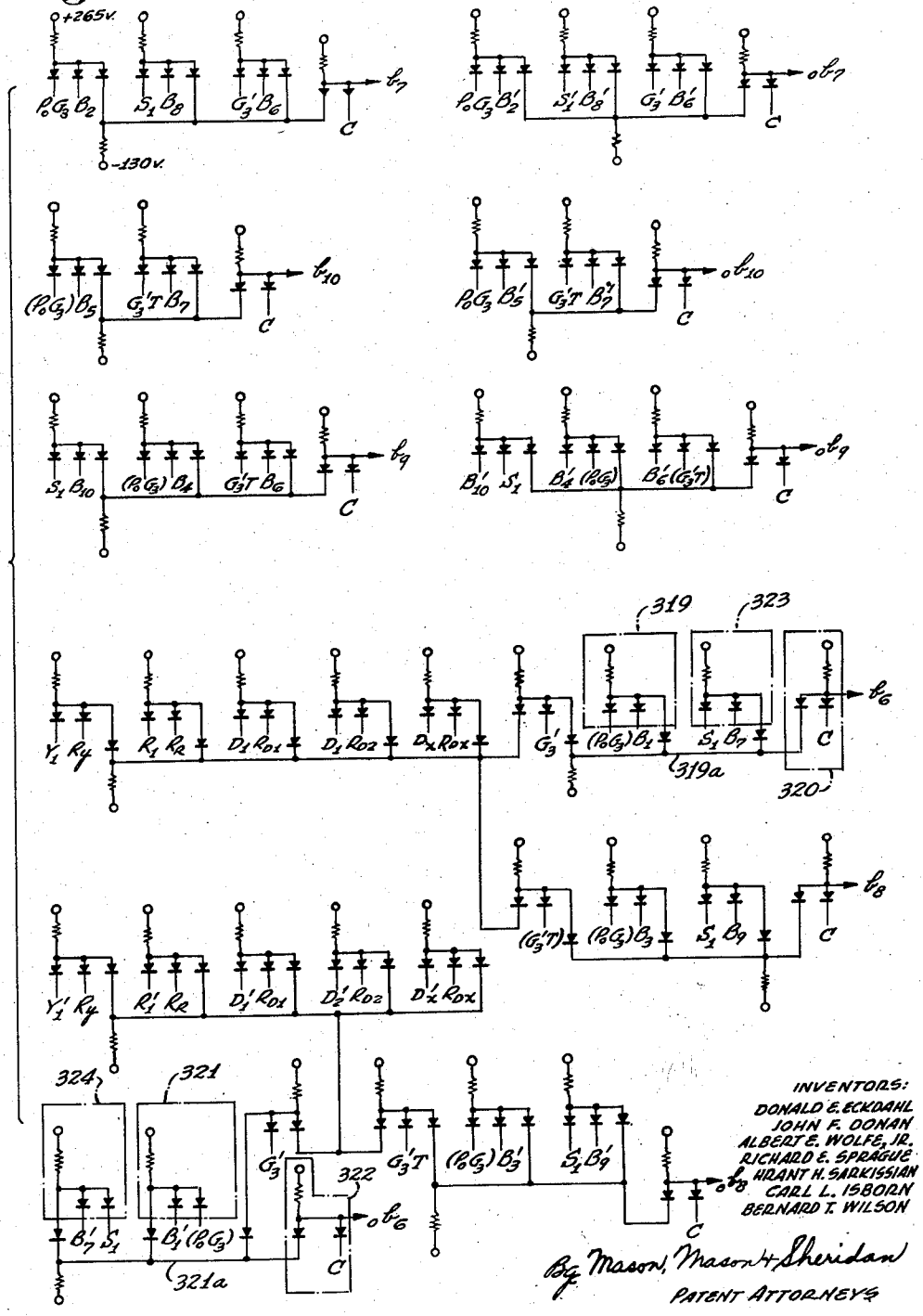
INVENTORS:
DONALD E. ECKDAHL
JOHN F. DONAN
ALBERT E. WOLFE, JR.
RICHARD E. SPRAGUE
HRANT H. SARKISSIAN
CARL L. ISBORN
BERNARD T. WILSON

By Mason, Mason & Sheridan
PATENT ATTORNEYS

DIGITAL DIFFERENTIAL ANALYZER

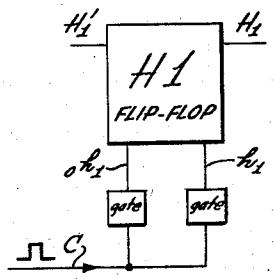
Filed April 22, 1952 48 Sheets-Sheet 25

Fig. 43



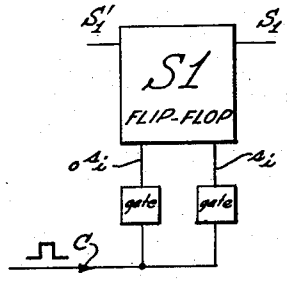
INVENTORS:
DONALD E. ECKDAHL
JOHN F. DONAN
ALBERT E. WOLFE, JR.
RICHARD E. SPRAGUE
HRANT H. SARKISSIAN
CARL L. ISBORN
BERNARD T. WILSON
By Mason, Mason & Sheridan
PATENT ATTORNEYS

Fig. 46



$h_1 = R_1 P_{26} W_0 G_3 C_2 C$
$_0h_1 = P_{26} W_{49} C$

Fig. 44



$A_1 = Y_1 S_1' P_0' P_{26}' G_3 (C_1+C_2) C$
$_0A_1 = P_{25} C$

Fig. 47

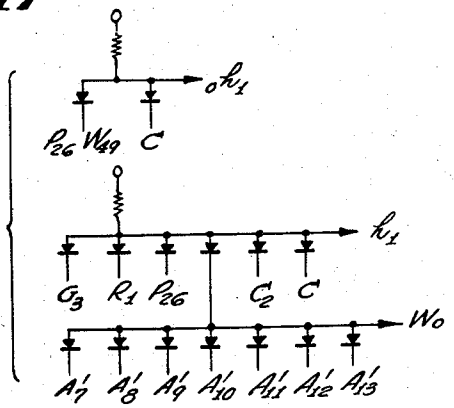
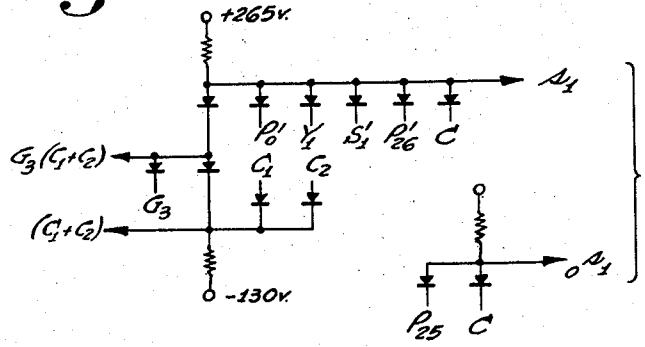


Fig. 45



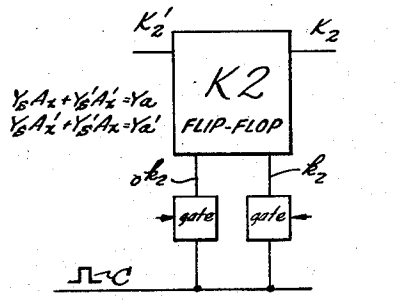
INVENTORS:
DONALD E. ECKDAHL
JOHN F. DONAN
ALBERT E. WOLFE, JR.
RICHARD E. SPRAGUE
HRANT H. SARKISSIAN
CARL L. ISBORN
BERNARD T. WILSON

By Mason, Mason & Sheridan
PATENT ATTORNEYS

DIGITAL DIFFERENTIAL ANALYZER

Filed April 22, 1952 48 Sheets-Sheet 27

Fig. 51



$Y_S A_x + Y_S' A_x' = Y_a$
$Y_S A_x' + Y_S' A_x = Y_a'$

$k_2 = [(Y_S A_x + Y_S' A_x') S_1 R_1 + P_0 A_x'] C$
$_0k_2 = [(Y_S A_x' + Y_S' A_x) S_1' R_1' + P_0 A_x] C$

Fig. 50

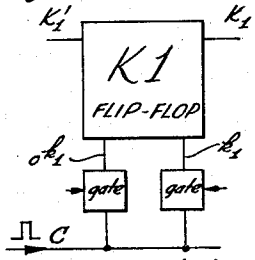
Adder Inputs

Ya	R1	K2	SUM (316)	CARRY (317)	(Sum)' (318)	Δ 0→1 (328)	Δ 1→0 (329)
0	0	0	0	0	1	0	0
0	0	1	1	0	0	0	1
0	1	0	1	0	0	0	0
0	1	1	0	1	1	0	0
1	0	0	1	0	0	0	0
1	0	1	0	1	1	0	0
1	1	0	0	1	1	1	0
1	1	1	1	1	0	0	0

$0 \xrightarrow{\Delta} 1 = Y_a R_1$
$1 \xrightarrow{\Delta} 0 = Y_a' R_1'$

$R_S = Y_a' R_1' K_2 + Y_a' R_1 K_2' + Y_a R_1' K_2' + Y_a R_1 K_2$
$= (R_1' K_2 + R_1 K_2') Y_a' + (R_1' K_2' + R_1 K_2) Y_a$
$R_S' = (R_1 K_2 + R_1' K_2') Y_a' + (R_1' K_2 + R_1 K_2') Y_a$

Fig. 49



$k_1 = Y_1 B_6 S_1 C$
$_0k_1 = (Y_1' B_6' K_1 S_1 + P_0) C$

Fig. 48

Adder Inputs

Y1	B6	K1	Sum (304)	Carry (305)	Δ 0→1 (306)	Δ 1→0 (307)
0	0	0	0	0	0	0
0	0	1	1	0	0	1
0	1	0	1	0	0	0
0	1	1	0	1	0	0
1	0	0	1	0	0	0
1	0	1	0	1	0	0
1	1	0	0	1	1	0
1	1	1	1	1	0	0

$0 \xrightarrow{\Delta} 1 = Y_1 B_6$
$1 \xrightarrow{\Delta} 0 = Y_1' B_6'$

$Sum = Y_S = Y_1 B_6 K_1 + Y_1 B_6' K_1' + Y_1 B_6 K_1' + Y_1' B_6' K_1$

INVENTORS:
DONALD E. ECKDAHL
JOHN F. DONAN
ALBERT E. WOLFE, JR.
RICHARD E. SPRAGUE
HRANT H. SARKISSIAN
CARL L. ISBORN
BERNARD T. WILSON

By Mason, Mason & Sheridan
PATENT ATTORNEYS

DIGITAL DIFFERENTIAL ANALYZER

Filed April 22, 1952 48 Sheets-Sheet 28

Fig. 52

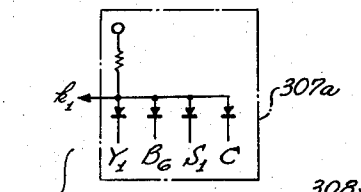
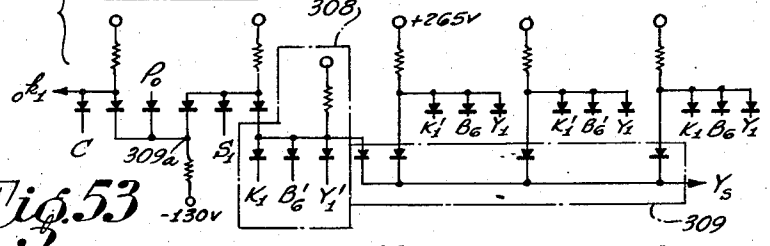


Fig. 53



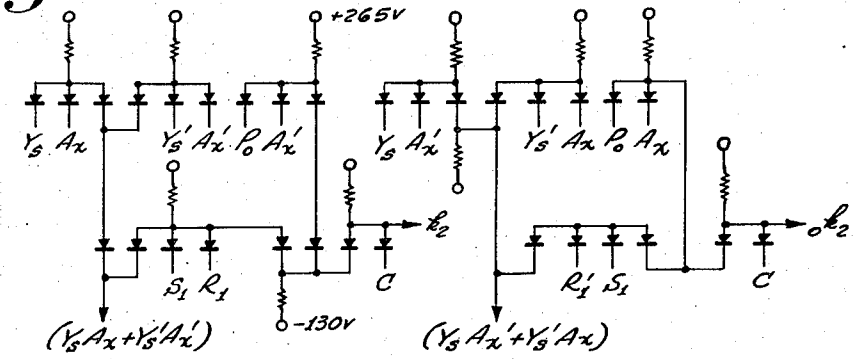
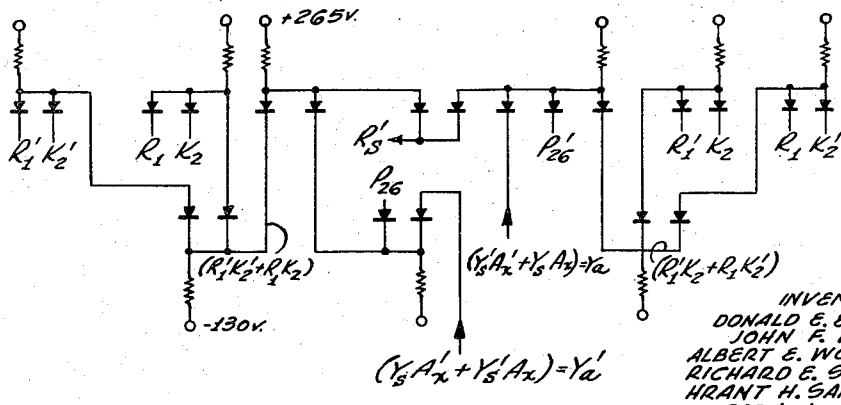


Fig. 54

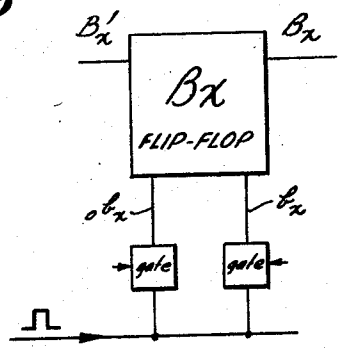


$$R_S' = (Y_a' + P_{26})(R_1' K_2' + R_1 K_2) + Y_a P_{26}' (R_1' K_2 + R_1 K_2')$$

INVENTORS:
DONALD E. ECKDAHL
JOHN F. DONAN
ALBERT E. WOLFE, JR.
RICHARD E. SPRAGUE
HRANT H. SARKISSIAN
CARL L. ISBORN
BERNARD T. WILSON

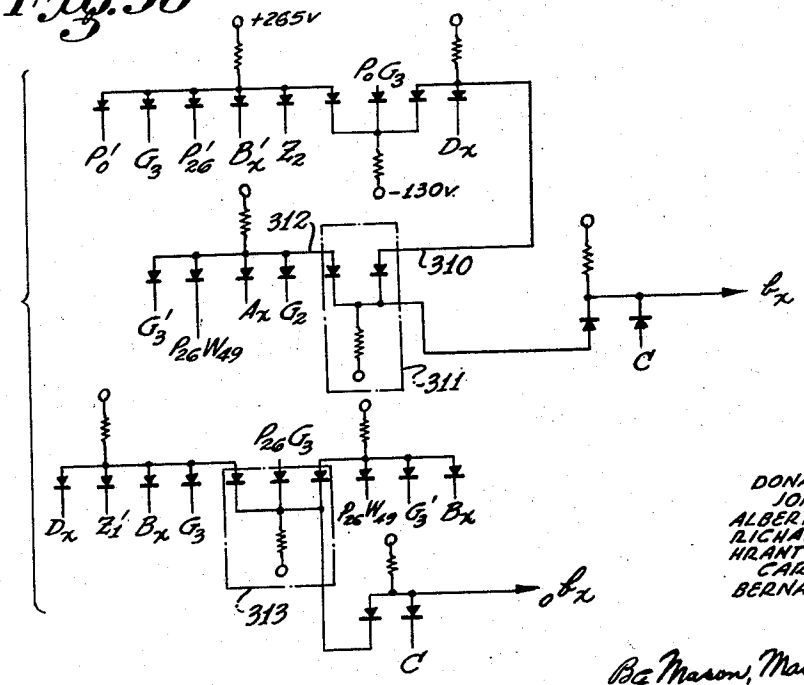
By Mason, Mason & Sheridan
PATENT ATTORNEYS

Fig. 55



d_x COUNTER	$+d_z$ FROM Z-1 CHANNEL	$b_x = P_0 G_3 D_x C$
	$+d_z$ FROM Z-2 CHANNEL	$b_x = Z_2 B'_x P'_0 P'_{26} G_3 D_x C$
	$-d_z$ FROM Z-2 CHANNEL	$_0b_x = P_{26} G_3 C$
	$-d_z$ FROM Z-1 CHANNEL	$_0b_x = D_x Z'_1 B_x G_3 C$
FILL REFERENCE		$b_x = G_2 P_{26} W_{49} A_x G'_3 C$
		$_0b_x = P_{26} W_{49} B_x G'_3 C$

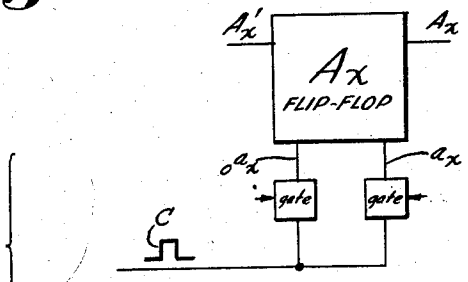
Fig. 56



INVENTORS:
DONALD E. ECKDAHL
JOHN F. DONAN
ALBERT E. WOLFE, JR.
RICHARD E. SPRAGUE
HRANT H. SARKISSIAN
CARL L. ISBORN
BERNARD T. WILSON

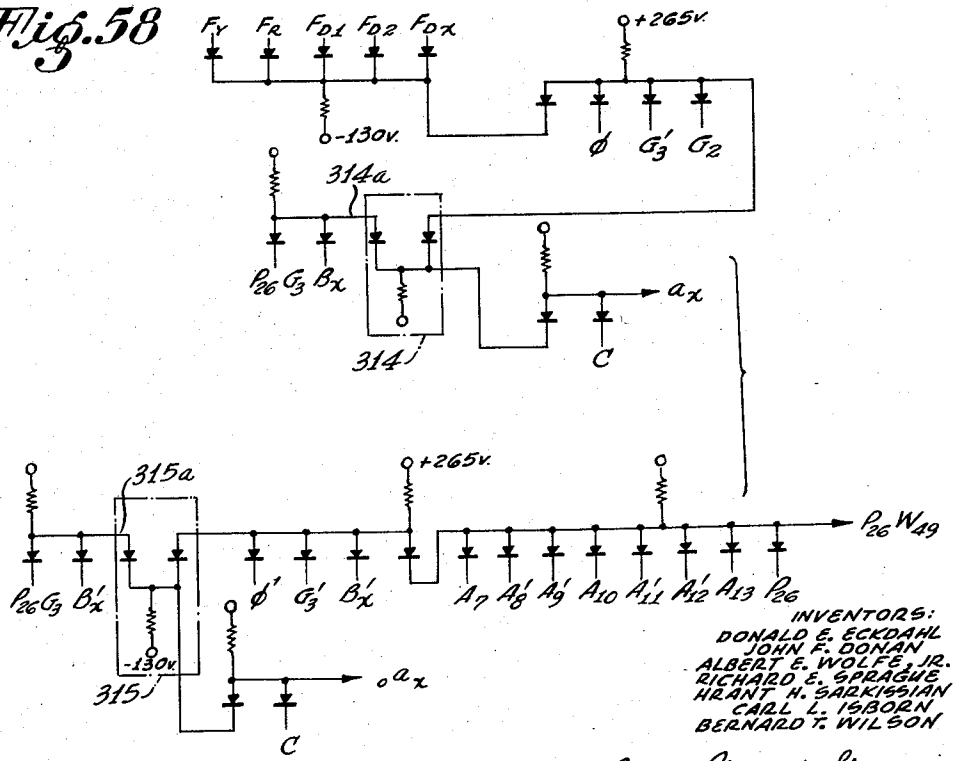
By Mason, Mason & Sheridan
PATENT ATTORNEYS

Fig. 57



d_x REGISTER	$a_x = P_{26} B_x G_3 C$ $_0a_x = P_{26} B_x' G_3 C$
FILL REFERENCE	$a_x = \phi G_3' (F_R + F_Y + F_{D1} + F_{D2} + F_{Dx}) G_2 C$ $_0a_x = P_{26} W_{49} \phi' G_3' B_x' C$

Fig. 58



INVENTORS:
DONALD E. ECKDAHL
JOHN F. DONAN
ALBERT E. WOLFE, JR.
RICHARD E. SPRAGUE
HRANT H. SARKISSIAN
CARL L. ISBORN
BERNARD T. WILSON

By Mason, Mason & Sheridan

PATENT ATTORNEYS

Fig. 59

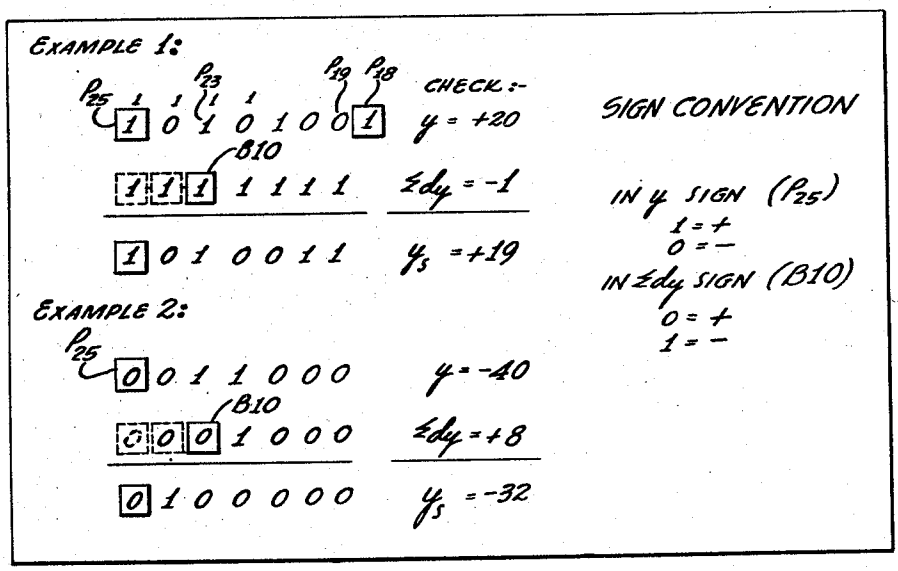
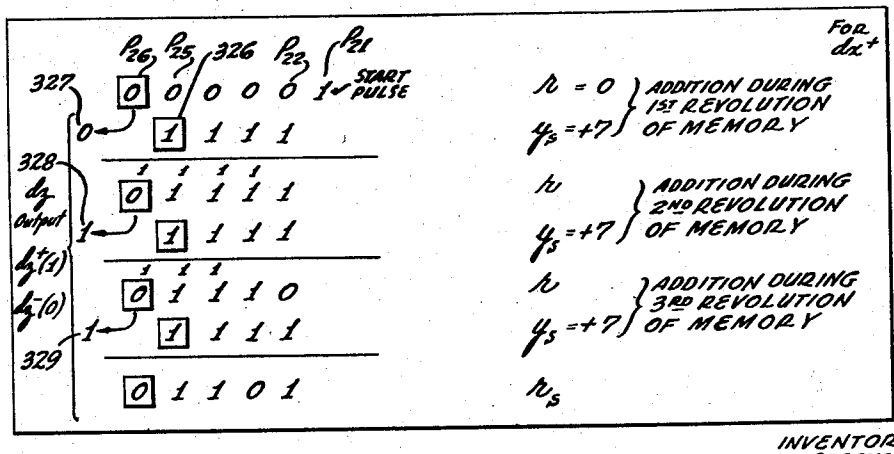


Fig. 60



INVENTORS:
DONALD E. ECKDAHL
JOHN F. DONAN
ALBERT E. WOLFE, JR.
RICHARD E. SPRAGUE
HRANT H. SARKISSIAN
CARL L. ISBORN
BERNARD T. WILSON

By Mason, Mason & Sheridan
PATENT ATTORNEYS

DIGITAL DIFFERENTIAL ANALYZER

Filed April 22, 1952 48 Sheets-Sheet 32

Fig. 61

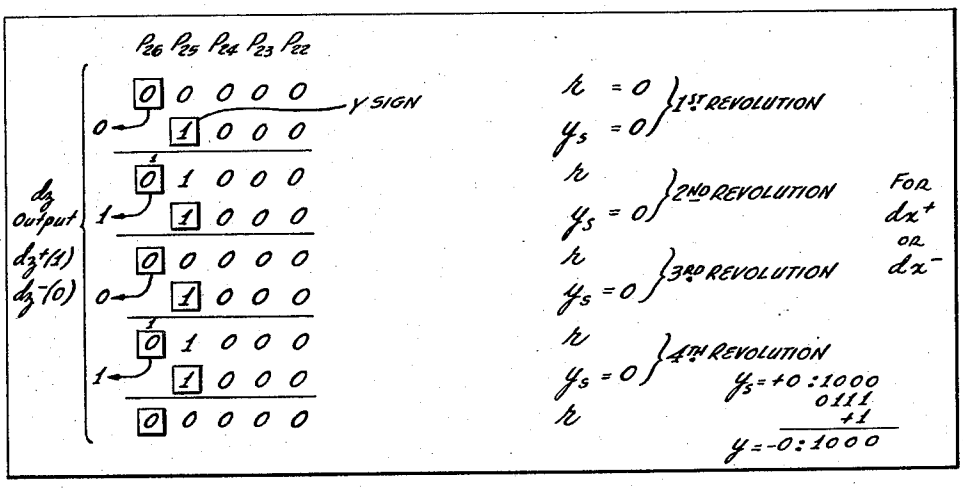
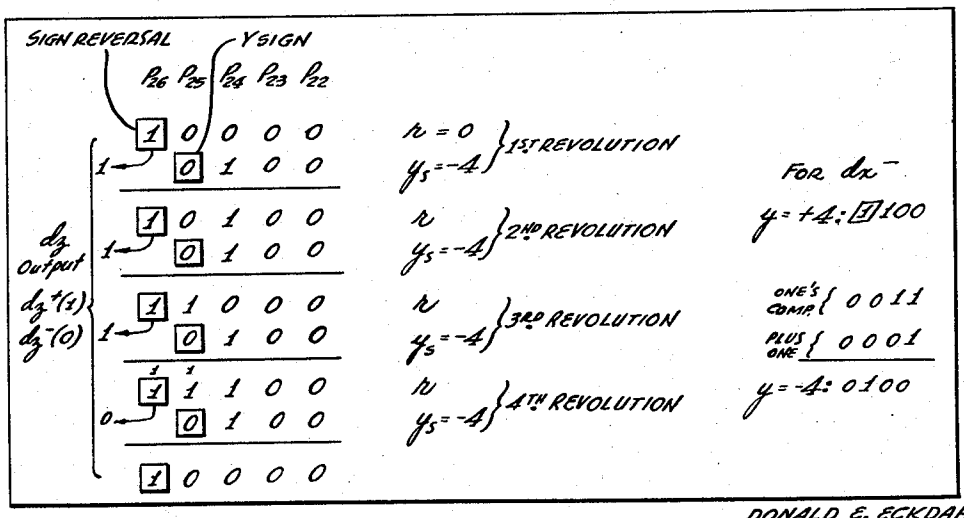


Fig. 62



INVENTORS:
DONALD E. ECKDAHL
JOHN F. DONAN
ALBERT E. WOLFE, JR.
RICHARD E. SPRAGUE
HRANT H. SARKISSIAN
CARL L. ISBORN
BERNARD T. WILSON
By Mason, Mason & Sheridan
PATENT ATTORNEYS

Feb. 9, 1960 D. E. ECKDAHL ET AL 2,924,381

DIGITAL DIFFERENTIAL ANALYZER

Filed April 22, 1952 48 Sheets-Sheet 33

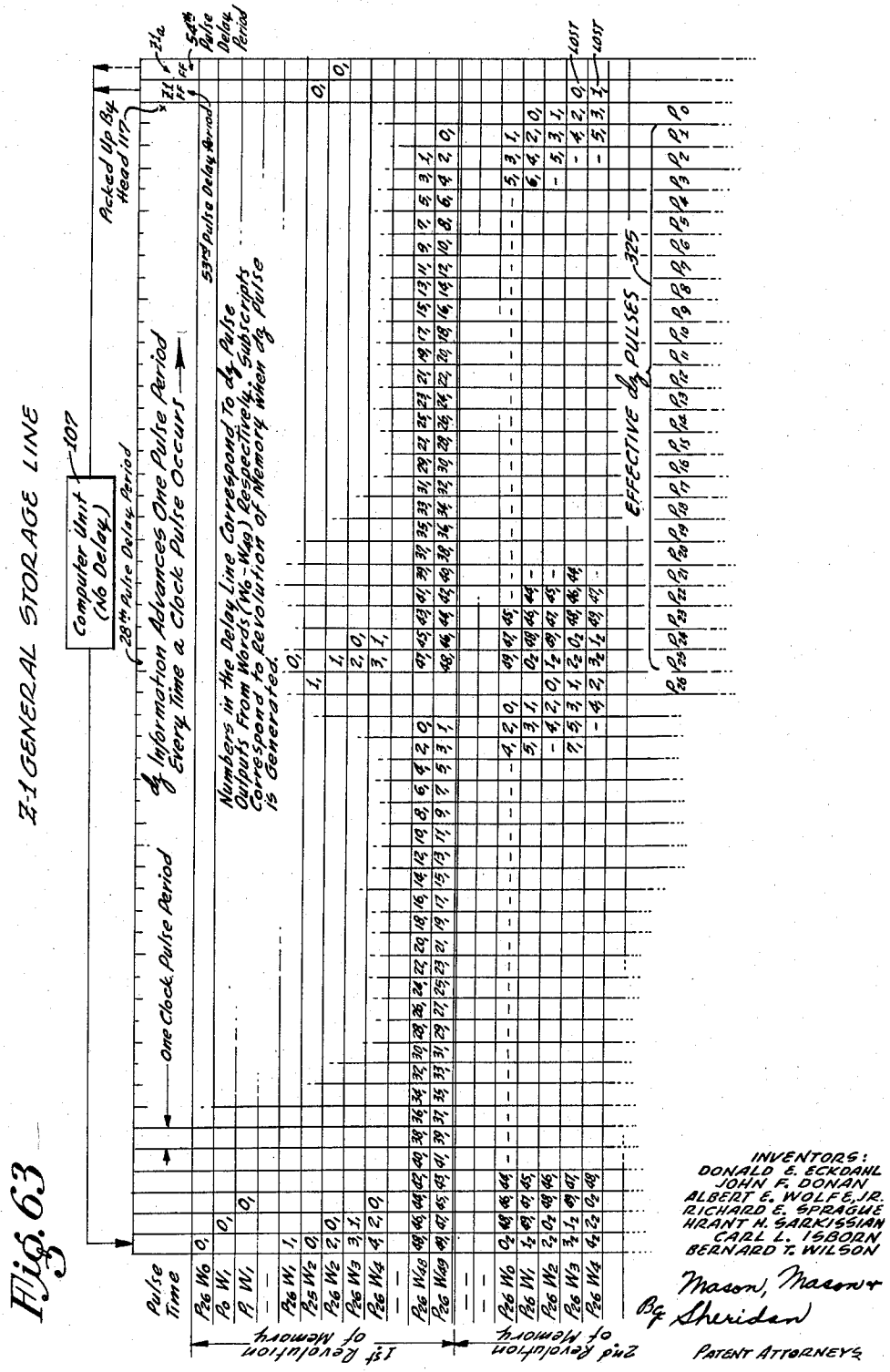


Fig. 6.3

INVENTORS:
DONALD E. ECKDAHL
JOHN F. DONAN
ALBERT E. WOLFE, JR.
RICHARD E. SPRAGUE
HRANT H. SARKISSIAN
CARL L. ISBORN
BERNARD T. WILSON

By Mason, Mason & Sheridan
PATENT ATTORNEYS

DIGITAL DIFFERENTIAL ANALYZER

Filed April 22, 1952 48 Sheets-Sheet 34

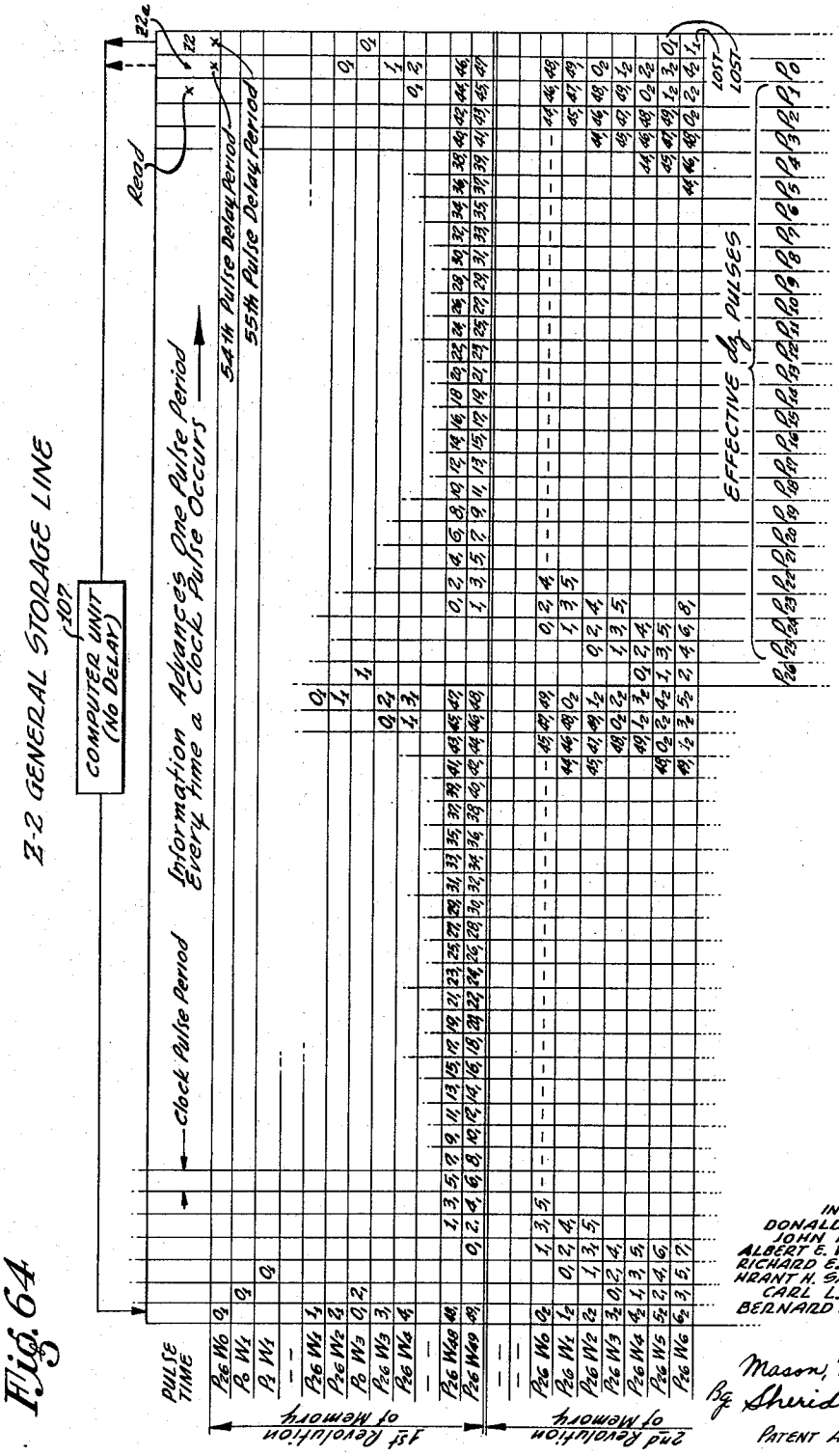


Fig. 64

INVENTORS:
DONALD E. ECKDAHL
JOHN F. DONAN
ALBERT E. WOLFE, JR.
RICHARD E. SPRAGUE
HRANT H. SARKISSIAN
CARL L. ISBORN
BERNARD T. WILSON

Mason, Mason & Sheridan
By
PATENT ATTORNEYS

Fig. 65

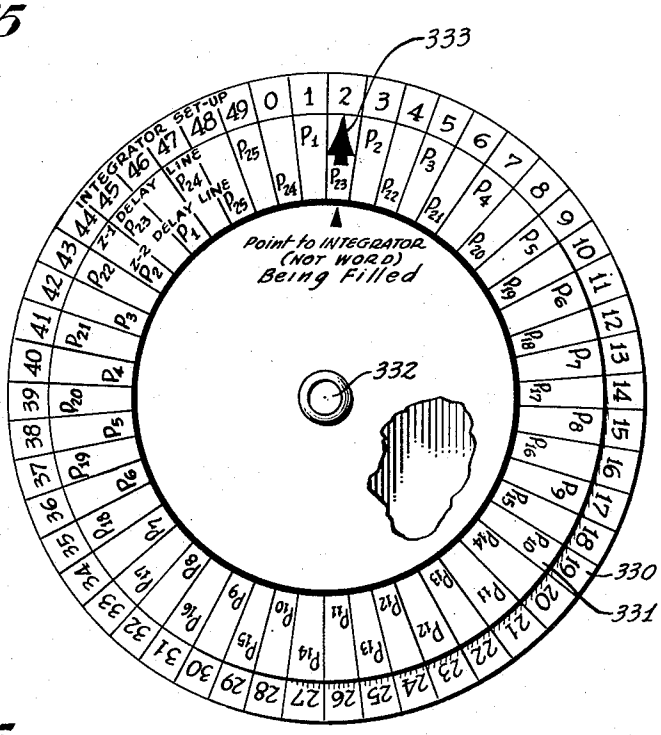
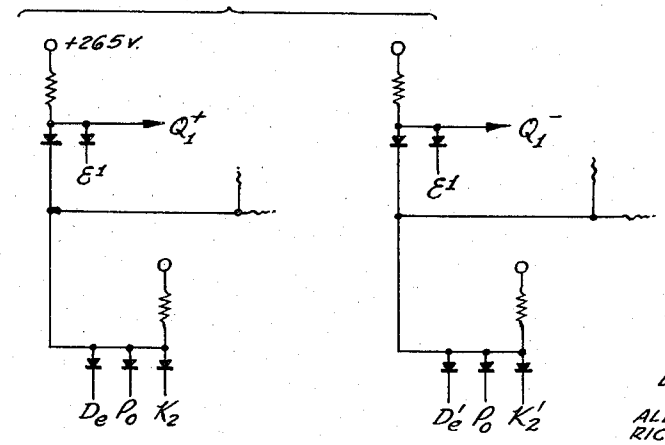


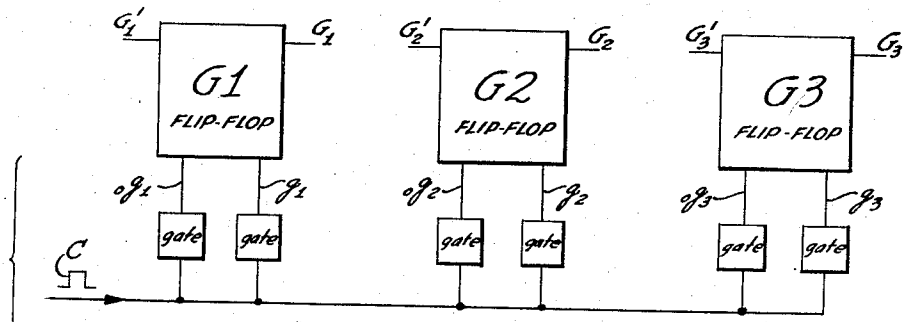
Fig. 65a



INVENTORS:
DONALD E. ECKDAHL
JOHN F. DONAN
ALBERT E. WOLFE, JR.
RICHARD E. SPRAGUE
HRANT H. SARKISSIAN
CARL L. ISBORN
BERNARD T. WILSON

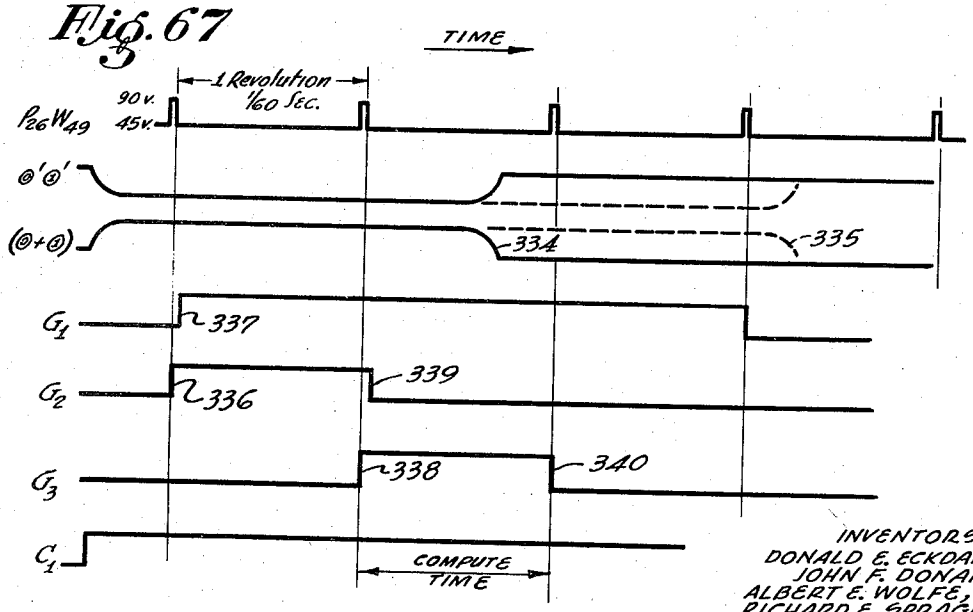
By Mason, Mason & Sheridan
PATENT ATTORNEYS

Fig. 66



COMPUTATION: $g_1 = (\textcircled{o}+\textcircled{i})G_2\,C$ $_0g_1 = \textcircled{o}'\textcircled{i}'G_3'P_{26}W_{49}\,C$	COMPUTATION: $g_2 = (\textcircled{o}x_1' + \textcircled{i}x_1)G_1'G_3'P_{26}W_{49}C$ $_0g_2 = G_3\,C$	COMPUTATION: $g_3 = P_{26}W_{49}G_1G_2G_3'C$ $_0g_3 = P_{26}W_{49}(C_1+H_1)G_3\,C$
	FILL: REFERENCE $g_2 = \phi A_x'B_x'P_{26}W_{49}\,C$ $_0g_2 = B_x\,C$	OVERFLOW: $_0g_3 = P_{26}F_1G_3\,C$

Fig. 67



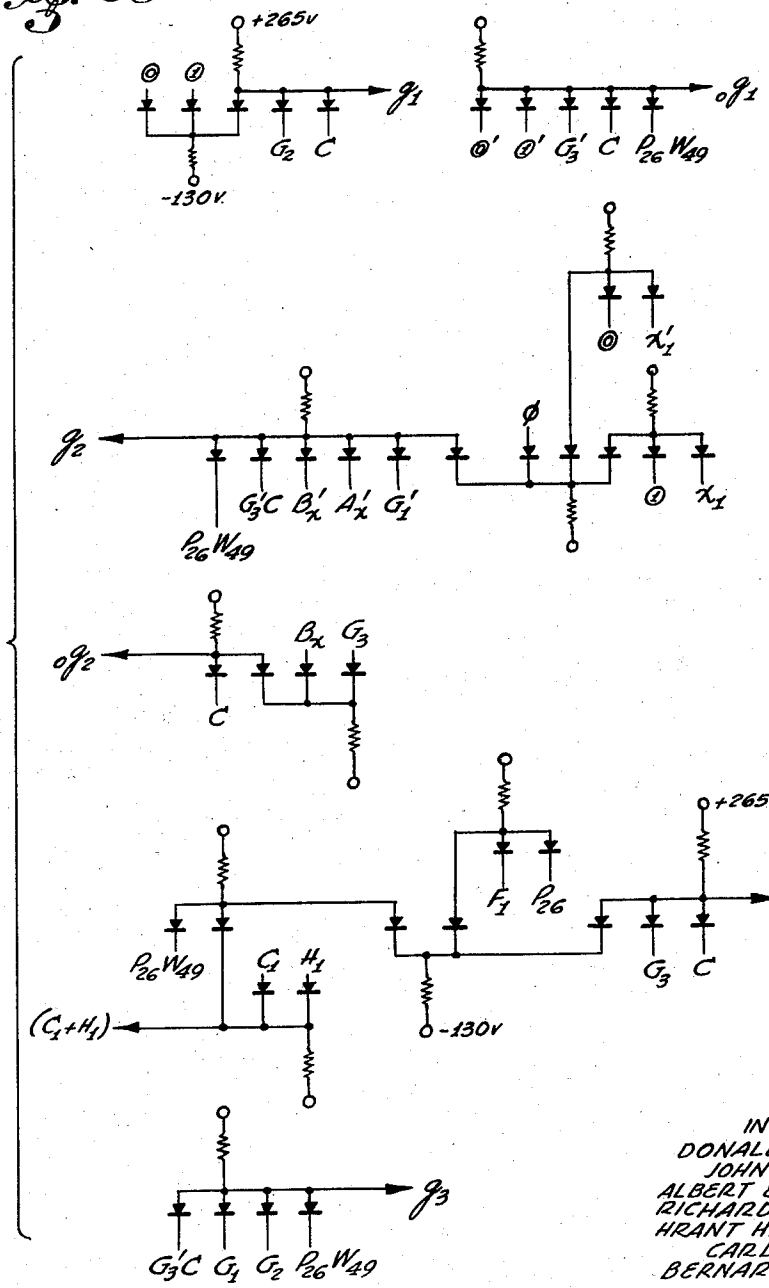
INVENTORS:
DONALD E. ECKDAHL
JOHN F. DONAN
ALBERT E. WOLFE, JR.
RICHARD E. SPRAGUE
HIRANT H. SARKISSIAN
CARL L. ISBORN
BERNARD T. WILSON

By Mason, Mason & Sheridan
PATENT ATTORNEYS

DIGITAL DIFFERENTIAL ANALYZER

Filed April 22, 1952 48 Sheets-Sheet 37

Fig. 68



INVENTORS:
DONALD E. ECKDAHL
JOHN F. DONAN
ALBERT E. WOLFE, JR
RICHARD E. SPRAGUE
HRANT H. SARKISSIAN
CARL L. ISBORN
BERNARD T. WILSON

By Mason, Mason & Sheridan
PATENT ATTORNEYS

Fig. 69

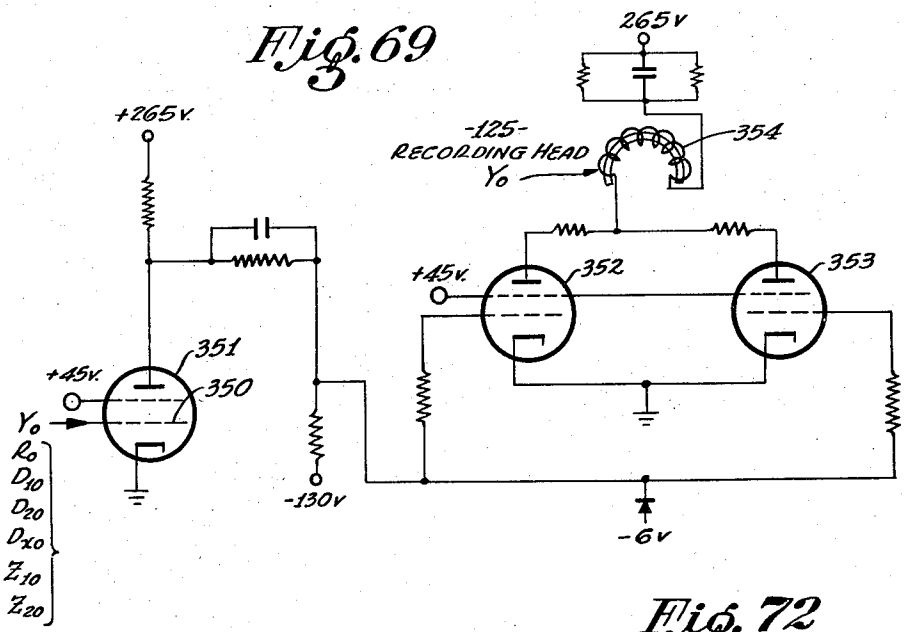
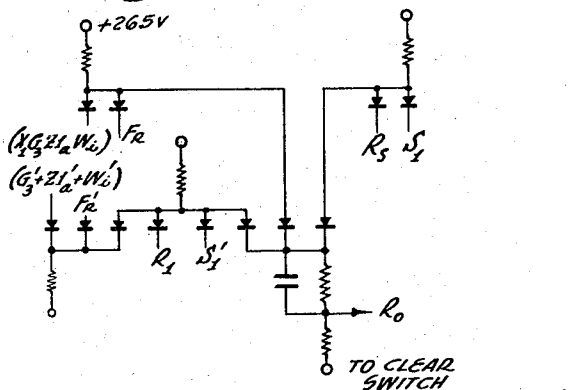
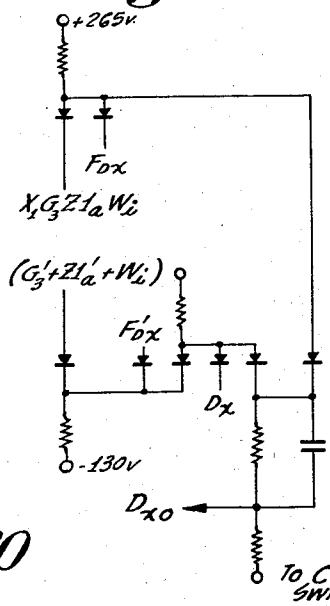


Fig. 71



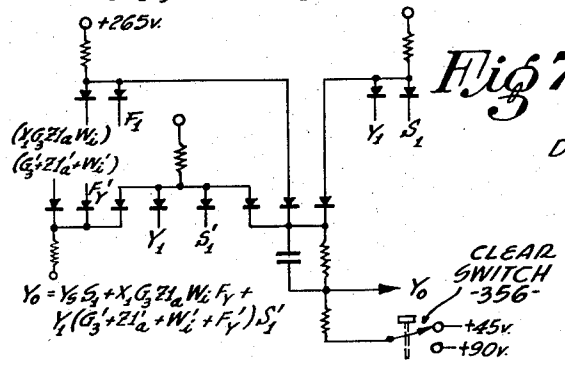
$$R_o = R_S S_1 + X_1 G_3 Z1_a W_i F_R + R_1 (G_3' + Z1_a' + W_i' + F_R') S_1'$$

Fig. 72



$$D_{X0} = X_1 G_3 Z1_a W_i F_{DX} + D_X (G_3' + Z1_a' + W_i' + F_{DX}')$$

Fig. 70



$$Y_o = Y_S S_1 + X_1 G_3 Z1_a W_i F_Y + Y_1 (G_3' + Z1_a' + W_i' + F_Y') S_1'$$

INVENTORS: DONALD E. ECKDAHL
JOHN F. DONAN
ALBERT E. WOLFE, JR.
RICHARD E. SPRAGUE
HRANT H. SARKISSIAN
CARL L. ISBORN
BERNARD T. WILSON

Mason, Mason & Sheridan
PATENT ATTORNEYS

DIGITAL DIFFERENTIAL ANALYZER

Filed April 22, 1952 48 Sheets-Sheet 39

Fig. 73

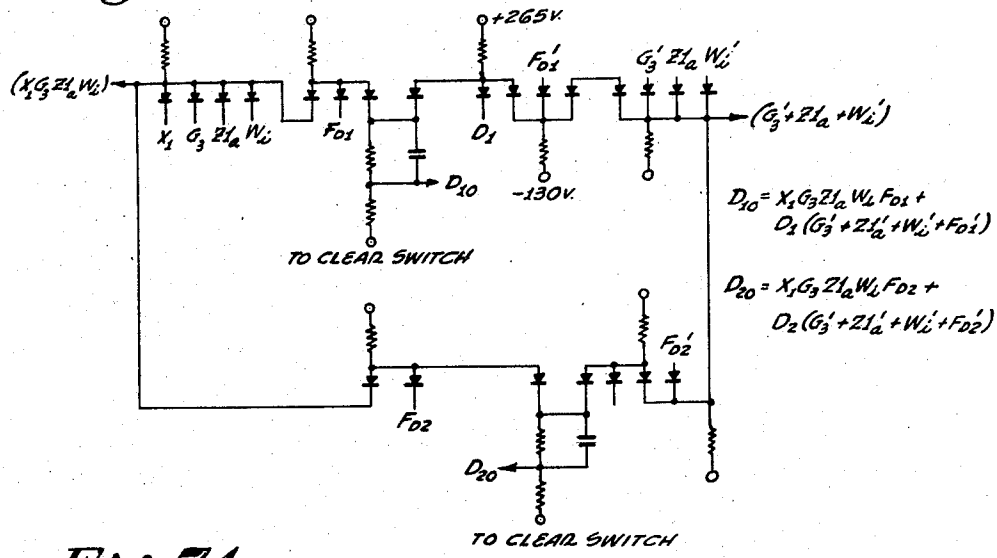
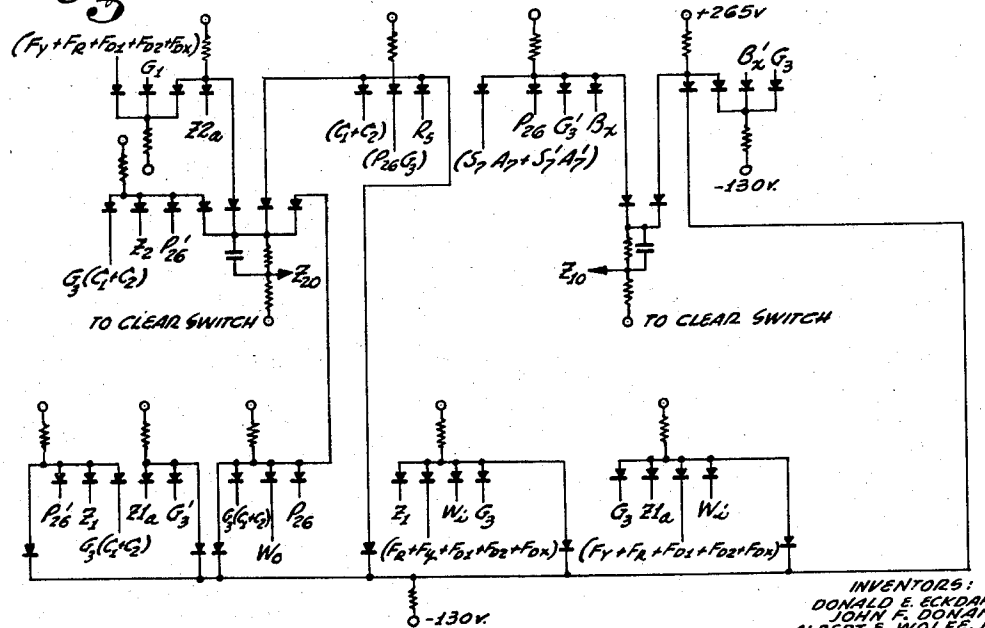


Fig. 74



$$Z_{10} = [P_{26}' Z_1 G_3 (C_1 + C_2) + W_0 P_{26} G_3 (C_1 + C_2) + (F_R + F_Y + F_{01} + F_{02} + F_{0X}) Z_1 W_L G_3 + R_5 P_{26} G_3 (C_1 + C_2) + Z1_a G_3' + (F_R + F_Y + F_{01} + F_{02} + F_{0X}) Z1_a W_L G_3] (B_X' + G_3) + P_{26} (S_7 A_7 + S_7' A_7') B_X G_3'$$

$$Z_{20} = R_5 P_{26} G_3 (C_1 + C_2) + Z_2 P_{26}' G_3 (C_1 + C_2) + Z Z_a G_3' + P_{26} W_0 G_3 (C_1 + C_2)$$

INVENTORS:
DONALD E. ECKDAHL
JOHN F. DONAN
ALBERT E. WOLFE, JR.
RICHARD E. SPRAGUE
HRANT H. SARKISSIAN
CARL L. ISBORN
BERNARD T. WILSON

By Mason, Mason & Sheridan
PATENT ATTORNEYS

DIGITAL DIFFERENTIAL ANALYZER

Filed April 22, 1952 48 Sheets-Sheet 40

Fig. 76

CODE REGISTER: P_0 CODE	$e_1 = X_1 P_0 G_3 C$ $_0e_1 = X_1' P_0 G_3 C$	$e_2 = Y_1 P_0 G_3 C$ $_0e_2 = Y_1' P_0 G_3 C$	$e_3 = D_1 P_0 G_3 C$ $_0e_3 = D_1' P_0 G_3 C$	$e_4 = D_2 P_0 G_3 C$ $_0e_4 = D_2' P_0 G_3 C$
P_{26} CODE	$e_1 = D_X P_{26} G_3 C$ $_0e_1 = D_X' P_{26} G_3 C$	$e_2 = Y_1 P_{26} G_3 C$ $_0e_2 = Y_1' P_{26} G_3 C$	$e_3 = D_1 P_{26} G_3 C$ $_0e_3 = D_1' P_{26} G_3 C$	$e_4 = D_2 P_{26} G_3 C$ $_0e_4 = D_2' P_{26} G_3 C$
COUNTER: Counting Net	$e_1 = E_1' G_3' T C$ $_0e_1 = E_1 G_3' T C$	$e_2 = E_1 E_2' G_3' T C$ $_0e_2 = E_1 E_2 G_3' T C$	$e_3 = E_1 E_2 E_3' G_3' T C$ $_0e_3 = E_1 E_2 E_3 G_3' T C$	$e_4 = E_1 E_2 E_3 E_4' G_3' T C$ $_0e_4 = 0$
COUNTER RESET	$_0e_1 = P_{26} E^9 G_3' C$			$_0e_4 = P_{26} E^9 G_3' C$

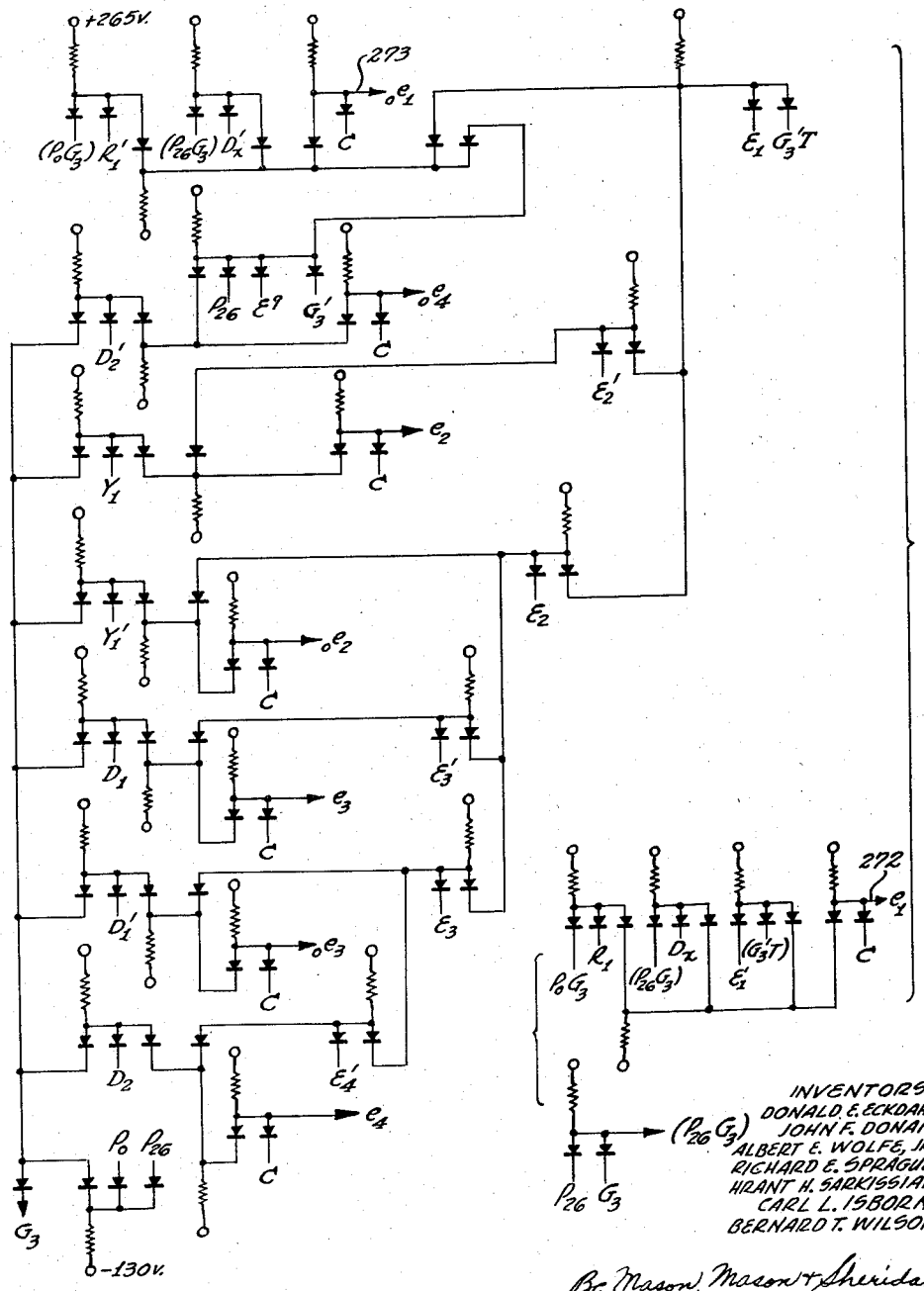
Fig. 75

	FLIP-FLOPS				
	E1	E2	E3	E4	
E^0	0	0	0	0	NO SELECTION
E^1	1	0	0	0	
E^2	0	1	0	0	
E^3	1	1	0	0	
E^4	0	0	1	0	
E^5	1	0	1	0	
E^6	0	1	1	0	
E^7	1	1	1	0	
E^8	0	0	0	1	ARBITRARY INPUT & OUTPUT CODE
E^9	1	0	0	1	
E^{10}	0	1	0	1	
E^{11}	1	1	0	1	
E^{12}	0	0	1	1	
E^{13}	1	0	1	1	
E^{14}	0	1	1	1	
E^{15}	1	1	1	1	NUMERICAL OUTPUT CODE

INVENTORS:
DONALD E. ECKDAHL
JOHN F. DONAN
ALBERT E. WOLFE, JR.
RICHARD E. SPRAGUE
HRANT H. SARKISSIAN
CARL L. ISBORN
BERNARD T. WILSON

By Mason, Mason & Sheridan
PATENT ATTORNEYS

Fig. 77


+265V.
273
272
-130V.
INVENTORS:
DONALD E. ECKDAHL
JOHN F. DONAN
ALBERT E. WOLFE, JR.
RICHARD E. SPRAGUE
HRANT H. SARKISSIAN
CARL L. ISBORN
BERNARD T. WILSON
By Mason, Mason & Sheridan
PATENT ATTORNEYS

DIGITAL DIFFERENTIAL ANALYZER

Filed April 22, 1952 48 Sheets-Sheet 42

Fig. 78

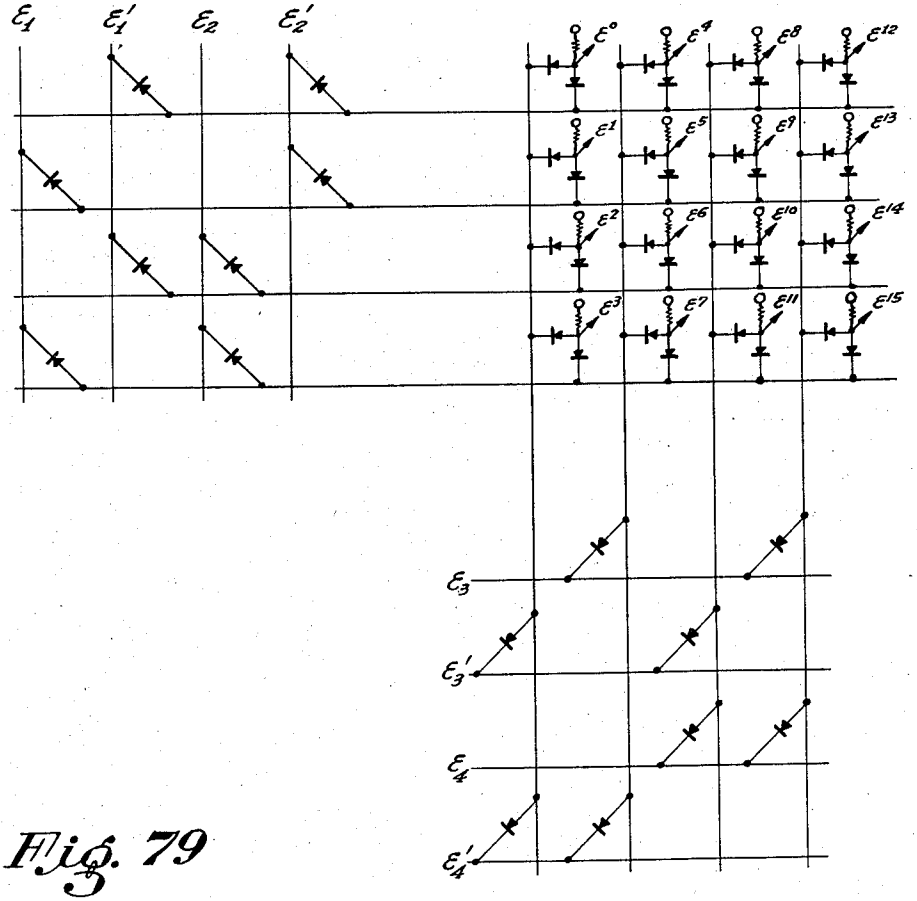
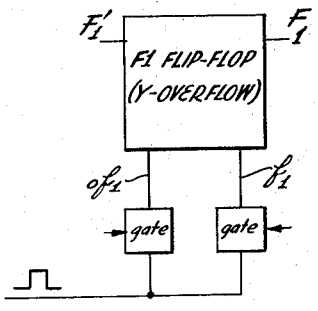


Fig. 79



$f_1 = (C_1 + C_2)\, P_1 R_1 G_3 C$

$of_1 = [P_{25}(Y_1' B_6' + Y_1' K_1' + K_1 B_6) G_3 + P_0 W_0 G_3] C$

INVENTORS:
DONALD E. ECKDAHL
JOHN F. DONAN
ALBERT E. WOLFE, JR.
RICHARD E. SPRAGUE
HRANT H. SARKISSIAN
CARL L. ISBORN
BERNARD T. WILSON

Bge Mason, Mason & Sheridan
PATENT ATTORNEYS

DIGITAL DIFFERENTIAL ANALYZER

Filed April 22, 1952 48 Sheets-Sheet 43

Fig. 80

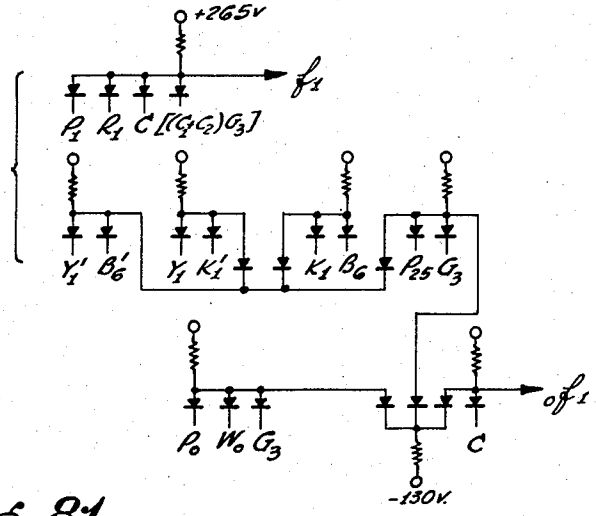
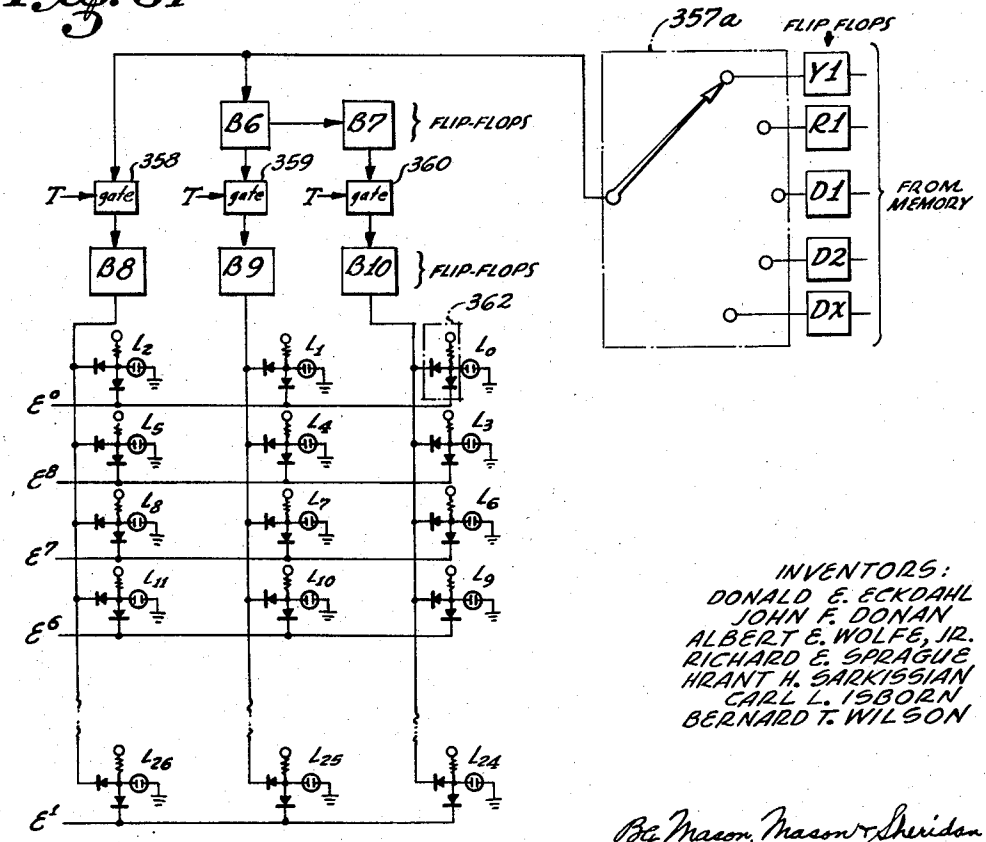


Fig. 81



INVENTORS:
DONALD E. ECKDAHL
JOHN F. DONAN
ALBERT E. WOLFE, JR.
RICHARD E. SPRAGUE
HRANT H. SARKISSIAN
CARL L. ISBORN
BERNARD T. WILSON

By Mason, Mason & Sheridan
PATENT ATTORNEYS

Fig. 84

ORDER OF READING OUT DIGITS			
1st Rev.	$L_{24} = B_{10}E^1$	$L_{25} = B_9E^1$	$L_{26} = B_8E^1$
2nd Rev.	$L_{21} = B_{10}E^2$	$L_{22} = B_9E^2$	$L_{23} = B_8E^2$
3rd Rev.	$L_{18} = B_{10}E^3$	$L_{19} = B_9E^3$	$L_{20} = B_8E^3$
4th Rev.	$L_{15} = B_{10}E^4$	$L_{16} = B_9E^4$	$L_{17} = B_8E^4$
5th Rev.	$L_{12} = B_{10}E^5$	$L_{13} = B_9E^5$	$L_{14} = B_8E^5$
6th Rev.	$L_9 = B_{10}E^6$	$L_{10} = B_9E^6$	$L_{11} = B_8E^6$
7th Rev.	$L_6 = B_{10}E^7$	$L_7 = B_9E^7$	$L_8 = B_8E^7$
8th Rev.	$L_3 = B_{10}E^8$	$L_4 = B_9E^8$	$L_5 = B_8E^8$
9th Rev.	$L_0 = B_{10}E^0$	$L_1 = B_9E^0$	$L_2 = B_8E^0$

Fig. 83

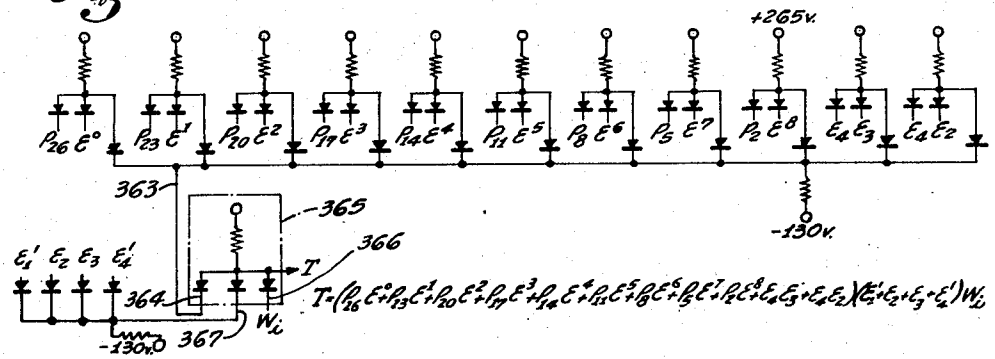
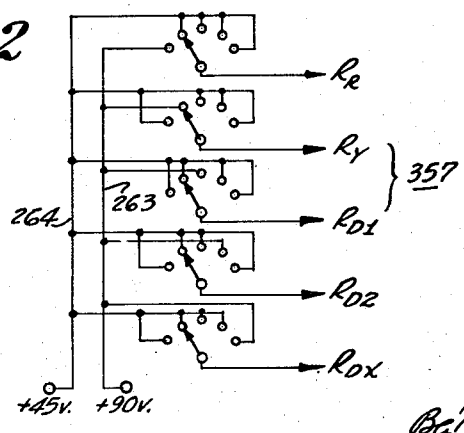


Fig. 82



INVENTORS:
DONALD E. ECKDAHL
JOHN F. DONAN
ALBERT E. WOLFE, JR.
RICHARD E. SPRAGUE
HRANT H. SARKISSIAN
CARL L. ISBORN
BERNARD T. WILSON

By Mason, Mason & Sheridan
PATENT ATTORNEYS

Fig. 88

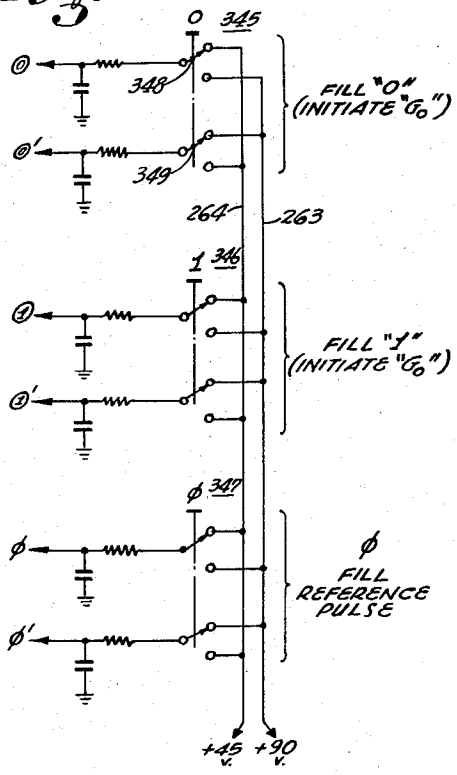


Fig. 85

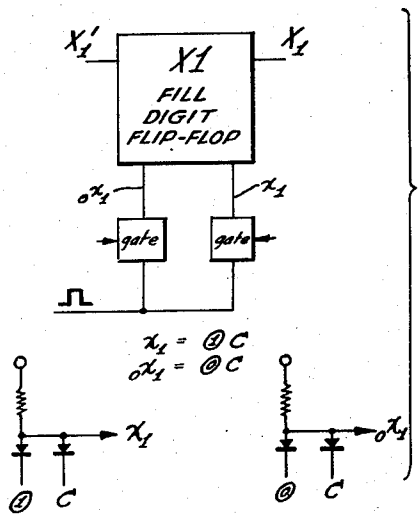


Fig. 87

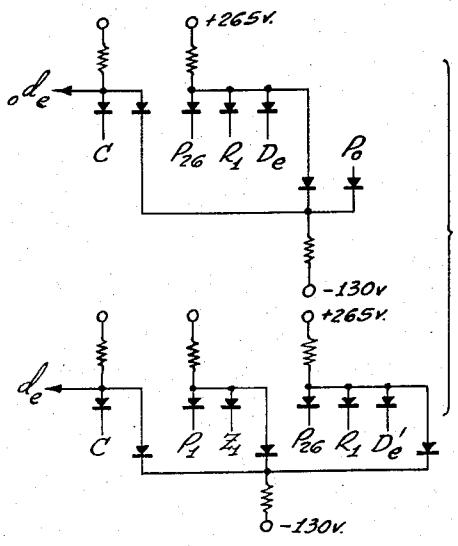
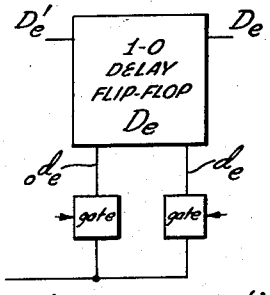


Fig. 86



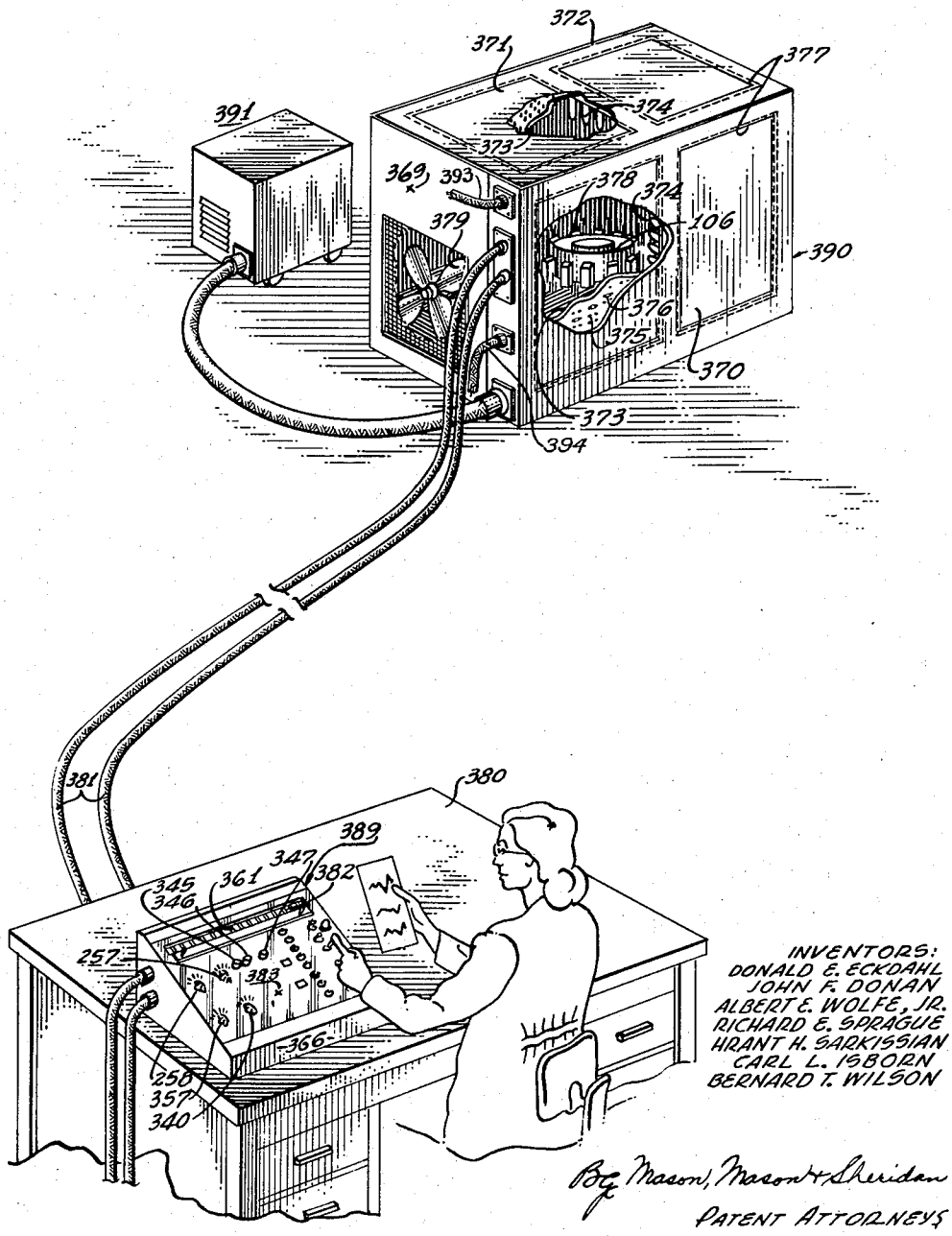
$d_e = (P_1Z_1 + P_{26}R_1D_e')C$

$_0d_e = (P_0 + P_{26}R_1D_e)C$

INVENTORS:
DONALD E. ECKDAHL
JOHN F. DONAN
ALBERT E. WOLFE, JR.
RICHARD E. SPRAGUE
HRANT H. SARKISSIAN
CARL L. ISBORN
BERNARD T. WILSON

By Mason, Mason & Sheridan
PATENT ATTORNEYS

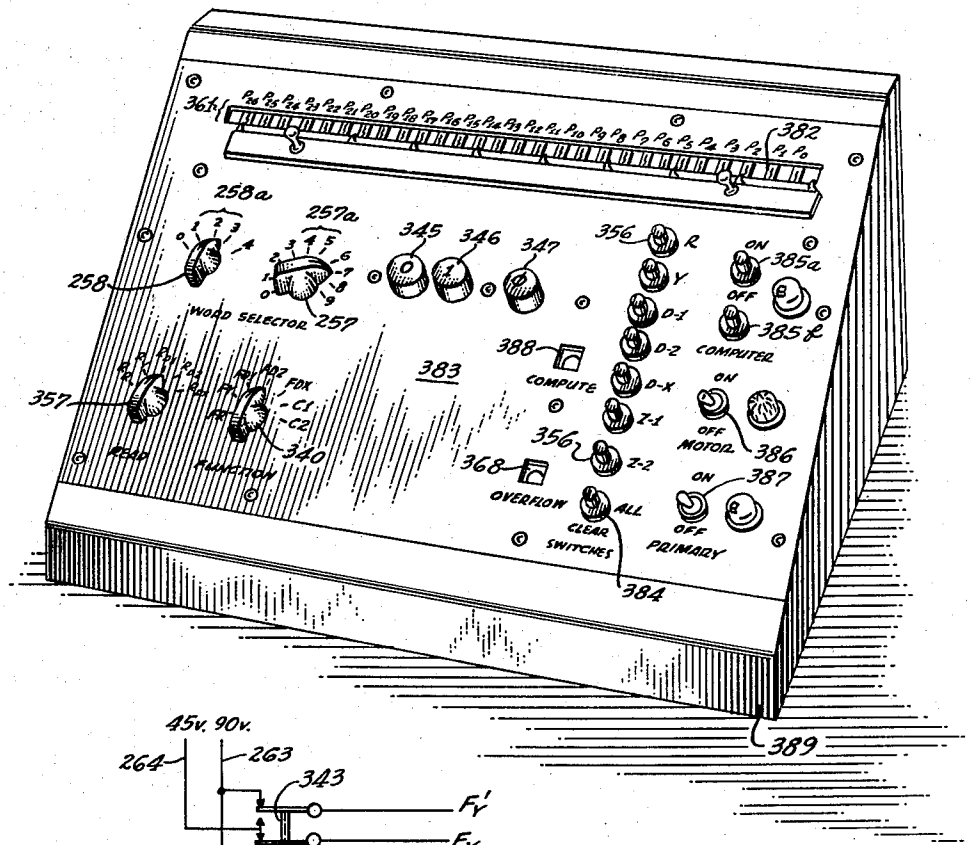
Fig. 89



INVENTORS:
DONALD E. ECKDAHL
JOHN F. DONAN
ALBERT E. WOLFE, JR.
RICHARD E. SPRAGUE
HIRANT H. SARKISSIAN
CARL L. ISBORN
BERNARD T. WILSON

By Mason, Mason & Sheridan
PATENT ATTORNEYS

Fig. 91



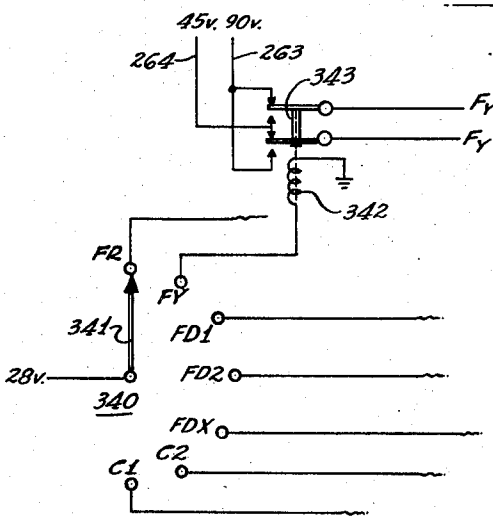


Fig. 90

INVENTORS:
DONALD E. ECKDAHL
JOHN F. DONAN
ALBERT E. WOLFE, JR.
RICHARD E. SPRAGUE
HIRANT H. SARKISSIAN
CARL L. ISBORN
BERNARD T. WILSON

By Mason, Mason & Sheridan
PATENT ATTORNEYS

Feb. 9, 1960 D. E. ECKDAHL ET AL 2,924,381

DIGITAL DIFFERENTIAL ANALYZER

Filed April 22, 1952 48 Sheets-Sheet 48

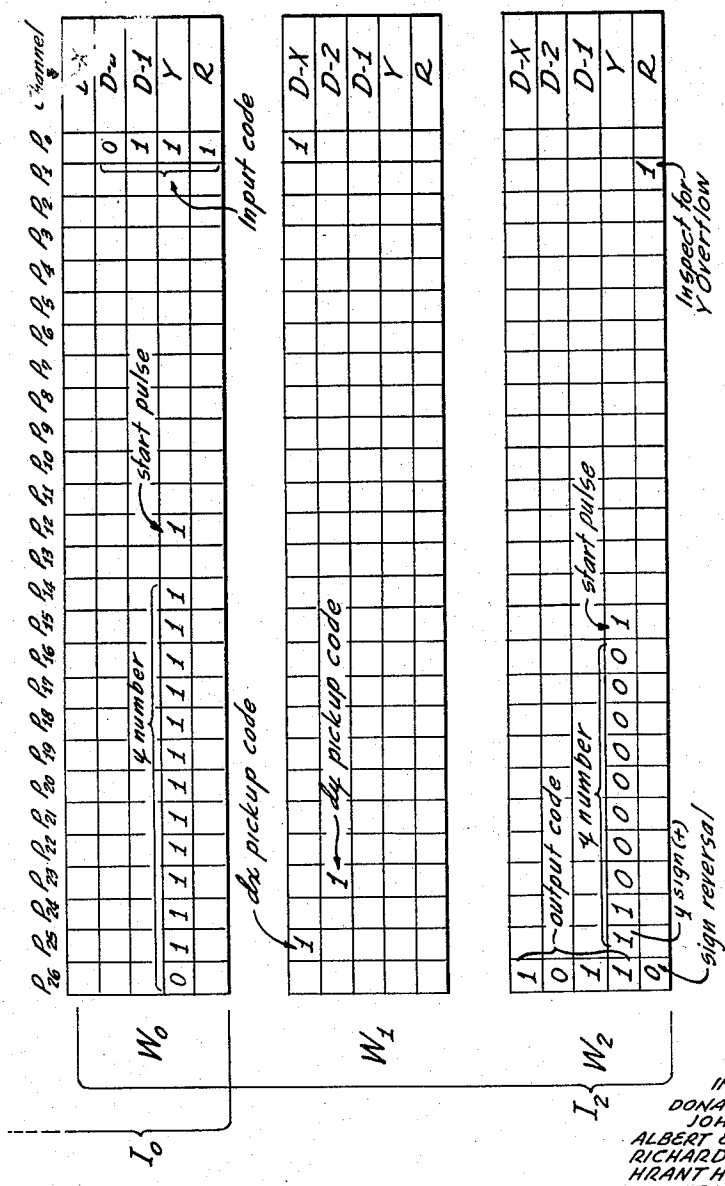


Fig. 92

United States Patent Office 2,924,381 Patented Feb. 9, 1960

2,924,381

DIGITAL DIFFERENTIAL ANALYZER

Donald E. Eckdahl, Torrance, John F. Donan, Reseda, Albert E. Wolfe, Jr., Compton, Richard E. Sprague, Torrance, Hrant H. Sarkissian, Pacific Palisades, Carl L. Isborn, Hawthorne, and Bernard T. Wilson, Los Angeles, Calif., assignors, by mesne assignments, to The National Cash Register Company, a corporation of Maryland

Application April 22, 1952, Serial No. 283,622

16 Claims. (Cl. 235—152)

This invention pertains to electronic computers and more particularly to a highly compact digital differential analyzer especially suitable for functioning as a control device.

A method for performing digital differential analysis has heretofore been proposed which integrates differential equations directly as do the analogue differential analyzers. Using this method, integration is accomplished by operating on pairs of numbers by an additive transfer process which can be explained as follows: A first register means "*n*" digits in length, containing a number "*y*," has its contents added into a second register means of equal length upon each occurrence of an input unit "*dx*." All carries from the second register means are denoted as incremental output units "*dz*." The number "*y*" in the first register means can be increased or decreased by an input unit "*dy*." The designations "*dx*" and "*dy*" for the inputs, and "*dz*" for the output, are differential notations for their respective variables. The result of the above arrangement is that the variables are related by the equation

$$dz=\frac{ydx}{B^n}$$

where B is the base of the number system used.

In the present invention information pertaining to the status of an integration process as above described is simulated by recording a pair of numbers representing the contents of the first and second register means, together with appropriate coded instructions, onto a portion of the periphery of a revolving magnetic drum functioning as a memory. Such recording shall be hereinafter designated as an "integrator setup." A plurality of such integrator setups, as required to solve a particular problem, are recorded in a serial fashion on this drum which is adapted to cooperate with a single computer unit. The information from the drum is serially passed through the computer unit which operates thereon to advance each integrator setup a given step. The information representing the altered integrator setup, obtained as a result of each computation, is redeposited back onto the revolving drum where it is stored until it is again serially passed through the computer unit during the next revolution.

By this arrangement the single computer unit is used over and over again to compute in turn on each of several integrator setups provided on the magnetic drum. As a result of each integrator computation, a function output, the "*dz*" output, is fed into a general register provided on the drum. By inspecting the information in this general register in accordance with a code associated with each integrator setup, the "*dz*" outputs generated as a result of the previous computations are selectively fed at the proper time as inputs into the computer unit. These inputs are the ones used to alter the integrator setup information as it is being operated upon.

One feature of the present invention is that it provides a highly flexible magnetic drum digital differential analyzer suitable for use on a wide variety of computation, test, and control problems. Not only will it solve linear and non-linear ordinary differential equations, but also means have been provided so that it is particularly adapted for receiving inputs, for example, from sensory instruments; performing computations on these inputs; and emitting outputs useful for controlling actuating devices such as motors, switches, or visual output indicators.

The essence of a computer of this nature lies in the arrangement of the information for the integrator setups which constitute the words recorded on the memory drum. This arrangement determines the means required in the computer unit and its mode of operation. In addition to storing numbers in each word section of the drum, coded information is stored therein which sets up the computer unit so as to properly receive the inputs, operate on the numbers, and emit the outputs.

Thus another feature of this invention provides a novel arrangement of the integrator setup pattern on the memory drum which enables it to contain the code needed for communication with external devices and which enables the computer unit associated therewith to perform the process of digital integration in a highly efficient manner.

Still another feature of the computer of the present invention is that it enables the computer unit, while the machine is operating, to receive external inputs from a wide variety of instruments or other computers without the necessity of special additional synchronization equipment for each such external device. In addition, the inputs may be coded to supply certain arbitrary information to any of the integrators in the machine. It should be further appreciated in connection herewith that a digital differential analyzer is basically a "system" for performing a plurality of integration processes and accordingly the term "system," as used in the present specification, refers to a set of circuits which are all operable with reference to the same timing source. Thus this feature of the present invention provides for introducing into the system, during the course of a computation, signals which are generated by outside sources in a random or arbitrary timed fashion relative to the timing of the system.

There is provided as still another feature of the device a control unit which enables the operator to perform all operations such as starting, filling, reading out, and stopping without going near the computer unit proper. In other words, the present invention provides for all controls and answer indicating equipment to be located remotely from the computer itself and connected thereto by cables. Such an arrangement is highly desirable, particularly for military and mobile applications. For example, the control unit may be used to initially fill the computer in a mobile installation and then be disconnected. Also, when used for pure computational work and engineering, this feature provides that the bulk of the machine need not be on the desk or even in the engineering office. Furthermore, this system may be adapted to be used with more than one remote control and readout unit.

Operation on each integrator setup as herein described can be broadly divided into two steps. During the first step, the computer unit picks up the "*dz*" outputs from the general register, and during the second step it operates on the numbers representing the status of the integrator setup in accordance with the inputs. Since these two steps are performed in order, the means employed in the computer unit for picking up the "*dz*" outputs operates only during the first half of an integrator setup time, and the means employed for operating on

the numbers operates only during the second half of an integrator setup time.

In order to provide an efficient computer unit and to speed up the number of integration processes which can be performed in a given time, the present invention presents a unique method and arrangement for performing integration whereby the means employed for picking up the "*dz*" outputs and the means for altering the numbers can operate simultaneously. This is essentially arrived at by always picking up the "*dz*" outputs required for altering the numbers in the following integrator setup during the same time that the numbers of the present integrator setup are being processed. What this amounts to is an overlapping of information, or a paralleling of operations, which results in a highly efficient use of the computer unit means, and the added feature that less time is required to perform a series of integrations. In the present invention, the paralleling of the pickup code information to be used during later integration processes with the number information descriptive of the present integration process is referred to as a "word." Each "word" is comprised of a block of information representing a fixed number of channels crosswise and a fixed number of pulse positions lengthwise. Information in the same pulse position of each of the channels is simultaneously observed by the computer unit as the "word" is scanned lengthwise. Thus a "word" represents the basic pattern of information to which the computer unit is designed to respond. Stated another way, the serial cyclical operation of the computer unit is coextensive with the information in a "word."

In the present invention, in order to provide a highly flexible computer, it is desirable to be able to feed into any integrator the outputs of any other integrator. To do this it is necessary to scan the outputs of all the other integration processes during a single cycle of the computer unit, i.e. one word period.

On the other hand, in order to provide a machine of large capacity and one which utilizes its equipment in a highly efficient manner, it is desirable that the number of integration processes provided for exceed the number of pulse periods in the basic cycle of the computer unit. Specifically, in the present embodiment, the number of integrators provided for is equal to twice the length of the basic cycle.

Thus another feature of the invention provides a novel arrangement of the general register for the integrator outputs comprising two relatively short parallel channels associated with the magnetic drum. These channels are arranged so that they operate over a delay of approximately two word periods. As each word is scanned the information in the last half of the two short parallel channels is also scanned. On each successive word, the information in each of the short channels precesses, i.e. steps with respect to its previous corresponding position in the word. These two short channels are arranged such that each precesses the information stored therein in a direction opposite to the other. By this means, while each integrator setup is being operated upon, the circuits are arranged so as to scan all the other integrator setup outputs in a single cycle of the computer unit, i.e. in a number of pulse periods equal to one-half the number of integration processes provided.

Still another feature of this invention provides novel means for visually reading out information in a given word channel of the memory drum. Since the signal representing a unit bit of information is only available while it is being scanned (approximately 10 microseconds), it is necessary to provide means for storing this information for a long enough period so that visual means, such as neon bulbs, energized thereby, can be observed by the operator.

In order to overcome the need of providing a storage for every digit to be read in a word channel, the present invention provides novel circuitry which effectively stores and feeds out only three binary digits at a time to the readout unit. Due to the speed of this operation, however, and the persistence of vision of the operator, he effectively sees all the digits in a word channel at one time.

Briefly the present invention is comprised of a rotating magnetic drum on whose periphery information is stored in a novel binary fashion. This information represents coded numerical and instructional data required for a plurality of digital integrator setups. A single highly flexible computer unit is provided for cooperating with this dynamic storage. The information for each integrator setup is serially picked up from the drum by the computer unit and operated thereon to accomplish the computations required to advance the integration one step.

The information required for a particular integrator setup is provided in a plurality of successive word times. That is, each integration process is divided into a plurality of steps or stages. The word structure is so arranged, however, and the computer unit is so devised that during any single word period information corresponding to all stages of an integration may be processed, but the information associated with each stage is for a different integrator setup.

In general, the computer unit is composed of a plurality of bistable state circuits, such as flip-flop circuits, representing logical propositions required to accomplish manipulations on binary information, such as: counting, shifting, arithmetic processes such as addition and subtraction, and various commands.

The flip-flop circuits are interconnected by sets of diode networks which operate according to logical equations. Each of the equations defines when and how a computer flip-flop will change its state. The outputs of the flip-flops, together with information picked up from the magnetic drum storage, represent the terms of the equations which are combined by "logical and" and "logical or" operations which subsequently are referred to as "logical multiplication" and "logical addition" operations, respectively. (As will be noted later, these latter designations for logical operations are preferable since they are descriptive of the notation employed in the equations.) These operations are physically performed by the diode networks which interconnect the lines carrying the two valued voltages representative of the terms.

Clock pulses, continuously generated by a recording placed on a separate channel of the magnetic drum, provide the timing of the computer. These clock pulses which define each pulse position synchronize the operation of the computer unit with the memory drum and also provide the signals for effectively beating the inputs to the flip-flop circuits so as to advance the state of the computer unit through its basic cycle.

More specifically, the correspondence between the information on the memory drum and the operation of the computer unit is obtained by counting circuits, triggered by the clock pulses, which continually recycle to generate timing signals defining word periods and pulse positions within the word periods. These timing signals help set up the conditions in the computer unit so that it properly responds to the information being picked up from the word channels of the memory drum.

Associated with the computer unit are circuit means for feeding arbitrary input rate information into certain integration processes, and circuit means for feeding output rate information from chosen integration processes. These input and output rates are tied in with the proper integrator setups by coded information recorded as initial conditions in particular word times of the memory drum.

The present invention will be made more fully apparent by reference to the following detailed description of a preferred arrangement of the invention wherein

reference will be made to the accompanying drawings in which:

Figure 1 is a block diagram of a circuit introducing the additive transfer principle for performing the mathematical process of digital integration.

Figure 2 is a graph illustrating generally the method by which the circuit of Figure 1 performs the integration of the area under a curve.

Figure 3 is a perspective view of the magnetic drum memory showing broadly how the information is arranged thereon and how this information is routed through the computer unit.

Figure 4 is a detailed illustration showing how the binary information is arranged in each word time of the memory.

Figure 5 is an illustration showing the time-wise relation of information in a word to the Z general memory channels.

Figure 6 is a block diagram which broadly shows the external input and output circuits associated with the computer.

Figure 7 shows the electrical circuits provided for serially picking up information from the drum and transferring it to the memory flip-flops.

Figure 8 is a circuit diagram showing how the clock pulses are generated by signals picked from the clock channel on the memory drum.

Figure 9 is a circuit diagram showing how the information picked up from the memory drum is synchronized with the clock pulses before being utilized in the computer unit.

Figure 10 is a block diagram showing how the E counter controls the neon readout circuitry.

Figure 11 is a time chart showing how the T proposition is used for controlling the neon readout circuitry.

Figure 12 is a diagram showing how the pickup and record heads are arranged on the Y channel of the memory drum.

Figures 13 and 14 are diagrams showing how the pick-up and record heads are arranged on the Z–1 and Z–2 general memory channels, respectively.

Figure 15 is an overall block functional diagram of the computer showing, generally, how the information is handled therein.

Figure 16 is a timing chart showing how and when the information required for a particular integration process is made available to the computer unit.

Figure 17 is a table showing the states of the A1–A5 flip-flops for indicating the P counts which divide up the word period.

Figure 18 is a block diagram of the A1–A5 flip-flops together with the symbolic logic equations which define how the outputs of the flip-flops are connected to the inputs in order to count in accordance with the table shown in Figure 17.

Figure 19 is a schematic circuit diagram showing how the A1 flip-flop is connected so as to operate as the first stage of the P counter.

Figure 20 is a set of waveforms presented for explaining the operation of the circuit in Figure 19.

Figure 21 is a logical network for solving the triggering equations associated with the A1–A5 flip-flops.

Figure 22 is a diode matrix for solving the P timing propositions shown by the logical equations in Figure 17.

Figure 23 is a table showing the states of the A7–A10 flip-flops for defining the radix 10 counting stage of the W counter.

Figure 24 is a table showing the states of the A11–A13 flip-flops for defining a radix 5 counting stage of the W counter.

Figure 25 is a block diagram of the A7–A10 flip-flops together with the symbolic logic equations defining how they change states.

Figure 26 is a block diagram of the A11–A13 flip-flops together with the symbolic logic equations defining how they change states.

Figure 27 is a logical network for solving the equations associated with the A7–A13 flip-flops.

Figure 28 is a circuit diagram of the W_1 selector switch used for manually selecting a word period in the computer.

Figure 29 is a circuit diagram showing the interconnections between the W_1 selector switch and the W counter flip-flops.

Figure 30 is a simplified block diagram showing how the random input information is picked up by the Σdy counter.

Figure 31 is a block diagram of the U1 and V1 flip-flops used for initially storing the random inputs, together with the symbolic logic equations defining when the flip-flops are to be reset.

Figure 32 is a logical network for solving the reset equations associated with the U*i* and V*i* flip-flops.

Figure 32*a* is a logical network for feeding out a *y* number from an integration setup onto an external line, together with the logical equation defining the same.

Figure 33 is a block diagram of the V*a* flip-flop used for picking up the negative random inputs, together with the symbolic logic equation for its grids.

Figure 34 is a block diagram of the U*a* flip-flop used for picking up the positive random inputs, together with the symbolic logic equations for its grids.

Figure 35 is a logical network for solving the grid triggering equations associated with the positive and negative random input flip-flops.

Figure 36 is a table showing the binary number system in which the Σdy counter operates.

Figure 37 is a table showing the binary number system used for representing the *y* number system.

Figure 38 is a block diagram of the Σdy counter flip-flops together with the symbolic logic equations defining how these flip-flops are interconnected to perform counting.

Figure 39 is a diagram of the logical network for generating the equations defining how the Σdy counter flip-flops operate.

Figure 40 is a diagram of the logical network used for decoding the arbitrary input information so as to determine how to reset the Σdy counter.

Figure 41 is a diagram of the logical network used for decoding the Z–1 and Z–2 general memory channels to determine the *dy* inputs.

Figure 42 is a block diagram of the B6–B10 flip-flops constituting the Σdy register, together with the symbolic logic equations defining how these flip-flops change state.

Figure 43 is a logical network showing how the logical equations presented in Figure 42 are physically solved.

Figure 44 is a block diagram of the start flip-flop, together with the logical equations defining its grid inputs.

Figure 45 is a logical network showing how the input equations for the start flip-flop are physically solved.

Figure 46 is a block diagram of the halt flip-flop, together with the logical equations defining its grid inputs.

Figure 47 is a logical network showing how the input equations for the halt flip-flop are physically solved.

Figure 48 is a truth table for revealing how the logical equations are obtained for summing the old *y* number obtained from the memory drum with the Σdy count obtained from the Σdy register.

Figure 49 is a block diagram of the K1 carry flip-flop, together with the logical equations defining its grid inputs.

Figure 50 is a truth table for revealing how the logical equations are obtained for summing the new *y* number with the *r* number obtained from the drum.

Figure 51 is a block diagram of the K2 carry flip-flop together with the logical equations defining its grid inputs.

Figure 52 is a logical network for solving the equations for the K1 carry flip-flop and the new y number.

Figure 53 is a logical network for solving the equations associated with the K2 carry flip-flop.

Figure 54 is a logical network for solving the equations representing the new r number.

Figure 55 is a block diagram of the Bx flip-flop which picks up the dx pulse from the Z–1 or Z–2 line, together with the logical equations for its grids.

Figure 56 is a logical network for solving the equations associated with the Bx flip-flop.

Figure 57 is a block diagram of the Ax flip-flop for storing the dx pulse after it has been picked up, together with the logical equations defining its grid inputs.

Figure 58 is a logical network for solving the equations associated with the Ax flip-flop.

Figure 59 shows, by way of example, the method by which the $(y+\Sigma dy)$ adder operates.

Figure 60 shows the method of operation of the $(r+y_s)$ adder when the dx input is positive in sign.

Figure 61 shows the method of operation of the $(r+y_s)$ adder when the effective dz output is zero.

Figure 62 shows the method of operation of the $(r+y_s)$ adder when the dx input is negative in sign.

Figure 63 is a chart illustrating the operation of the Z–1 general memory channel.

Figure 64 is a chart illustrating the operation of the Z–2 general memory channel.

Figure 65 is a circular slide rule used for determining the positions of the dy and dx pickup codes in an integrator setup.

Figure 65a is a logical network for feeding out the dz function from an integrator setup while the computer is operating, together with the symbolic logical equations defining the same.

Figure 66 is a block diagram of the G1, G2 and G3 flip-flops which are used for remotely initiating the computing operation, together with the logical equations defining their grid inputs.

Figure 67 is a set of graphs for explaining the operation of the G1, G2 and G3 flip-flops.

Figure 68 is a logical network for generating the equations associated with the G flip-flops.

Figure 69 is a diagram of the circuit associated with the Y channel recording head.

Figure 70 is a logical network for generating the Y channel record proposition together with the logical equations defining this network.

Figure 71 is a logical network for generating the R channel record proposition together with the logical equations defining this network.

Figure 72 is a logical network for generating the D–X channel record proposition together with the logical equations defining the network.

Figure 73 is a logical network for generating the D–1 and D–2 channel record propositions together with the logical equations defining the network.

Figure 74 is a logical network for generating the Z–1 and Z–2 channel record propositions together with the logical equations defining this network.

Figure 75 is a binary table showing the states of the E flip-flops for representing the P_0 input code, the P_{26} output code, and straight binary counting.

Figure 76 shows a block diagram of the E flip-flops together with the symbolic logic equations of its grid inputs.

Figure 77 is a logical network for generating the symbolic logic equations associated with the E flip-flops.

Figure 78 is a diode matrix for generating the E counts on separate lines.

Figure 79 is a block diagram of the F1 flip-flop for indicating the Y overflow, together with the equations defining the grid inputs.

Figure 80 is a logical network for solving the F1 flip-flop triggering equations.

Figure 81 is a partially block diagram showing, generally, how the B6–B10 flip-flops are used for reading out coded octal digits associated with a word time.

Figure 82 is a schematic diagram of the read switch used for selecting a channel on the memory drum.

Figure 83 is a logical net for generating the T proposition which triggers the E counter so as to select the proper neon lights in which to transfer the coded octal digit currently picked up.

Figure 84 is a chart showing the logical equations defining the coded octal digit inputs and the order in which the digits are read out.

Figure 85 is a block diagram of the fill digit flip-flop X1, together with the equations and diode nets for controlling its state.

Figure 86 is a block diagram of the De flip-flop used for preventing an oscillation of the dz output when it is effectively zero.

Figure 87 is a logical network for solving the equations associated with the De flip-flop.

Figure 88 is a wiring diagram of the push button switches used for filling information into the computer.

Figure 89 is an overall view of a preferred embodiment of the present computer.

Figure 90 is a schematic wiring diagram of the computer main function switch.

Figure 91 is a perspective view of the control panel.

Figure 92 is a work sheet showing how information for a particular integrator is set up by the coder prior to placing this information into the computer.

General theory

In order to fully understand the improvements on the digital differential analyzer herein disclosed, the additive transfer principle of performing integration will first be introduced by describing the basic digital integrating circuit shown in Figure 1. This circuit is composed of a first counter functioning as an integrand storage **100**; a second counter functioning as a remainder accumulator **101**; and, a transfer network **102** interconnecting the two counters. A first and second input lead **103** and **104** are connected to the integrand storage **100** and transfer network **102**, respectively, and a single output lead **105** feeds out from the reminder accumulator **101**.

Each pulse, designated dy, received on the input lead **103** to the integrand storage **100** represents a small unit increase of the integrand y and causes the content of integrand storage **100** to increase by "one." The nature of the transfer network **102** is such that as a result of each input pulse, representing the variable of integration dx, received on input lead **104**, a value, equivalent to the number y existing at that particular time in the integrand storage **100**, is transferred in an additive manner to the number in the remainder accumulator **101**. It should be noted that the y number in storage **100** is not altered as a result of this transfer.

The remainder accumulator **101** has a given capacity which is equal to that of the integrand storage **100**. As the y number is successively added into the remainder accumulator **101**, in accordance with the dx inputs, the accumulator **101** soon reaches its capacity. The output lead **105** then emits an output pulse dz, and accumulator **101** starts to count from a remainder again.

It is to be noted that the integrand storage **100** does not provide for an overflow output. Rather, its capacity limits the maximum value which y may attain during a computation, and extra precautions must be taken to ensure that this value is not exceeded.

In order to further understand how the digital integrating circuit performs the integrating process, digital integration will next be described as a limiting process of finding the area under a curve $y=f(x)$ which is identified only as a set of numerical values of the ordinate corresponding to a set of successive values of the abscissa. As shown in Figure 2, this is accomplished

by subdividing the area under the curve into a succession of adjacent incremental sub-areas such as the cross hatched area A_n. The height of each rectangular sub-area A_n varies in steps equivalent to one or more fixed increments dy so that, for instance, its left ordinate y_n at all times tends to intersect the curve $y=f(x)$. The width of each of the sub-areas A_n is the same and is determined by the value fixed for the increment dx. The total area, which is the sum of the sub-areas, corresponds to the approximated integral. It is seen that this method of defining integration can be approximated to any degree of accuracy desired by decreasing the value of the increment assigned to each dx and, of course, each dy which is of the same order.

In Figure 2 it is noted that since the incremental dx, in determining each incremental sub-area A_n, defined as ydx, is fixed throughout an integration, dx can be assumed to be dropped out of the notation by being assigned the value "one." This in no way affects the proportionality of an ordinate y to its corresponding sub-area. Thus, each time a dx pulse is received on input lead **104** to the integrator circuit of Figure 1, a digit "one" may be considered to be received.

Again referring to Figure 2, it is noted that the change in height of a sub-area A_n, i.e. the change in magnitude of the successive ordinates y, is to be made in very small fixed steps denoted as dy. Each of these dy increments is represented by a pulse received on the input lead **103** which algebraically alters the number content of the integrand storage **100** by one unit. Thus each dy pulse, similarly to the dx input pulse, can be considered as a digit "one" input to the integrator circuit.

The digit integrator circuit of Figure 1 can thus be considered to have two "one" inputs. One of these inputs, dy, is proportional to the incremental change in height of a sub-area A_n, and the other input dx, is the instruction to additively transfer.

As the sub-areas of Figure 2 corresponding to the y ordinates are successively accumulated in the remainder accumulator **101** of Figure 1, every time the capacity of the accumulator **101** is reached, a carry out pulse dz is emitted on output lead **105**. This carry out pulse dz, which is likewise to be considered as a digit "one" in the logic of the digital integrator, will be due to a pulse accumulation corresponding to a number N where N is at least one greater than the largest number which either the accumulator **101** or the storage **100** will hold.. Assuming a binary number counter is used, then $N=2^n$, and (2^n-1) is the number in the binary accumulator **101** when all of the stages contain the digit one. Any time a carry out pulse dz is emitted as a "one" digit on output lead **105**, a remainder term designated as r/N, which is always less than such a "one" carry out pulse dz, is left in the accumulator **101**. Note the number r is the contents of the accumulator **101** at this time. The next transfer of a y number from the storage **100** thus adds to the remainder r/N in the accumulator **101**, and may or may not, depending on the combined value, cause a single carry out pulse dz on output lead **105**.

Looking at this principle of digital integration from another angle, the basic circuit just described may be considered to be a means for combining two input rates

$$\frac{dy}{dt}$$

and

$$\frac{dx}{dt}$$

to obtain a desired output rate

$$\frac{dz}{dt}$$

As one instance of its versatility, the

$$\frac{dz}{dt}$$

output rate of such a device can be coupled back to form its own

$$\frac{dy}{dt}$$

input rate to produce an exponential function.

Thus, these digital integrators may be considered broadly as a set of devices which communicates between themselves by means of unitary pulses representing "ones" and which operate upon the inter-communications by means of counters storing numbers expressed in any numbering system convenient. Numbers may be looked upon here as a means for establishing and accurately generating streams of "ones." These "ones," in turn, are counted, merging to form other numbers, or are used for instructions.

Broadly, the stored numbers act upon a received rate of "ones" by deleting units from it. In other words, the effect of an integrator is to transmit an effective unitary rate dz which is equal to or less than the dx input rate, never greater. If the value of a function, Z, is to be found by counting the dz output rate, then, for example, between 10^4 and 10^5 units or pulses must be counted in order to express Z as a five place decimal number. If the rate reduction of an integrator is nominal, this may, ordinarily, be accomplished by the admission of no more than 10^5 dx inputs. If the rate reduction of the integrator is excessive, the number of dx inputs required to achieve the same accuracy of expression for Z must be greater. Since the time required to perform an addition cycle in the integrator will usually be fixed, the accuracy achievable in a limited time and with a limited amount of equipment will, therefore, depend, among other things, upon the extent of rate reduction in the integrators.

The explanation of the basic method for performing digital integration, so far as given, is necessarily simplified and general in nature. The description of the preferred embodiment of the present invention will further reveal how the signs of the variables are taken into account and how a plurality of such basic integrating circuits can be interconnected to generate particular solutions to relatively complex differential equations.

General description

Referring next to Figure 3, a schematic drawing shows a magnetic memory drum **106** and a computer unit **107**. The magnetic drum **106** which may be, for example, eight inches in diameter, is shown vertically mounted on bearings **109** in a housing **108**, and rotatably driven in a clockwise direction by a motor **110** at substantially, but not necessarily, a constant speed. On the periphery of drum **106** is a coating **111** of ferric oxide which provides the medium for magnetically storing bits of information. Around the periphery of magnetic drum **106**, eight circumferential channels are provided as designated by the notations R, Y, D–1, D–2, D–X, Z–1, Z–2 and C. The width of the magnetic track for each of these channels is about $\frac{1}{10}$ of an inch. Of these magnetically recorded channels C, is a permanently recorded "clock" channel; and R, Y, D–1, D–2 and D–X may be considered to function as relatively long re-circulating channels; while Z–1 and Z–2 function as relatively short recirculating channels.

In the present embodiment of the invention, the permanent clock channel C on the memory contains 1440 clock pulses serially arranged circumferentially about the periphery of drum **106** in a closed loop. A clock period defines the smallest time division of the computer and spacially represents an incremental arc on the drum periphery in which a bit of magnetic information, rep-

resenting a "zero" or a "one," may be magnetically recorded.

As will be pointed out more clearly later, the basic cycle of computer unit **107** is 27 clock pulses in length, i.e., every time 27 consecutive clock pulses are generated, the computer unit **107** repeats its operation. Furthermore, the machine is set up to pass through 50 such basic cycles for approximately each revolution of the drum **106**. The passing of the machine through 50 basic cycles is known as a major cycle. The basic cycle of 27 clock pulses corresponds to the arcuate length of the drum on which a "word" is recorded. Thus a "word" is a fixed pattern of information in the R, Y, D–1, D–2 and D–X channels around which the operation of the computer unit **107** is designed. That is to say, the computer unit **107** interprets the information sensed in each pulse position of each of the channels according to a predetermined arrangement build into the computer unit circuits.

Referring to the perspective view of the magnetic drum in Figure 3, the arcuate space occupied by a word is denoted by the capital letter W with the numerical subscripts 0–49. These designations are for convenience of reference, placed on sectors of the drum as shown. These words are located serially around the periphery of drum **106** in a counterclockwise order. It should be noted that the space required for these words does not extend over the entire length of the circumference of the channels provided on the magnetic drum **106**. Rather there is a "dead'" zone on the circumference whose sector **120** is shown shaded in Figure 3.

The manner in which the information is picked up from the memory drum **106**, fed into the computer unit **107**, and redeposited back onto the memory will now be generally described.

Associated with each memory channel R, Y, D–1, D–2, D–X, Z–1, Z–2 and C is a pickup head **112**, **113**, **114**, **115**, **116**, **117**, **118** and **119** respectively. These heads are schematically shown as being stationarily positioned in the drum housing **108** in close relationship with their respective channels on the drum **106**.

As the drum **106** rotates in a clockwise direction, the information recorded thereon is picked up by these channel pickup heads and fed into the computer unit **107** via input lines **121**.

The clock pulses are generated at approximately a 100 K.C. rate. This represents the speed at which successive bits of information are fed into the computer unit **107**. In the computer unit **107**, these clock pulses are counted to provide signals for letting the computer unit know the "word," and "pulse position" in the word, which is at the moment being scanned. Each bit of information in the R, Y, D–1, D–2 and D–X channels, on being sensed by the computer unit **107**, is utilized therein to logically generate output information which is deposited back onto the magnetic drum **106** via output lines **122** and record heads **124**, **125**, **126**, **127** and **128**, respectively. It should be noted that this information is recorded ahead of the point from which information was picked up, equivalent to the arcuate length of dead zone **120**. What this amounts to is that each bit of information is redeposited onto the revolving magnetic drum at such a point with respect to the pickup heads that it is delayed approximately 50 word times, thus permitting the rest of the information on the drum to be operated upon before the information in question is again picked up and fed into the computer unit **107**.

The information picked up by heads **117** and **118**, associated with the Z–1 and Z–2 channels, is of a different nature than the information represented by the "words." The Z–1 and Z–2 channels operate as a general storage for the dz information representing the outputs of the integration processes.

After being sensed by the computer unit **107**, each bit of Z–1 and Z–2 channel information is routed via return lines **129** and record heads **130** and **121**, respectively, to a point on the drum periphery approximately two word lengths behind the point from which it was picked up. Note that this may be considered to be the opposite direction from which the word channel information, picked up at the same time, is deposited.

The portion of the housing **108** between, for example, pickup head **112** and the record head **124**, is used for holding the memory erasing heads **132**, one of which is associated with each of the channels except the clock channel C. Note that the erasing heads **132** are always positioned within the arc associated with the dead zone of the drum **106**. These erasing heads effectively remove all the old information from the drum **106** by magnetizing each of the channels, in the direction corresponding to the "zero" of the binary information code.

It should be noted that the record of the information on the recirculating channels of the drum **106** is actually of a dynamic nature in that each bit of information changes its position, with respect to the drum, once each revolution, on passing through the computer unit **107**.

Word structure

The arrangement of the information in each of the words of the drum **106** will next be illustrated in Figure 4. Each word is 27 clock pulses long, the positions being indicated by notations P_0 through P_{26}. The Y and R channels contain in binary position P_2 through P_{24} a representation of a binary number indicating the contents of the integrand storage **100** and remainder accumulator **101**, respectively. These are the numbers associated with each digital integrating process, as explained in connection with Figure 1. Because of the scaling problem in the computer, the first binary stage, 2^0, of the y number may be located in pulse position P_2 or any pulse position thereafter. A code pulse, indicated as a start pulse, signals the computer unit to let it know when this fisrt digit of the y number is to be picked up. Hence, this start pulse is located in the pulse position just prior to the one used by the 2^0 stage of the y number.

In the particular illustration shown, the start pulse is located in position P_1, which is its first possible location, and the 2^0 stage of the y number is shown to be located in the P_2 position. The most significant y number digit is always located in the P_{24} position.

The P_{25} position of the Y word channel is reserved for the sign of the y number. A pulse in this position, i.e. a binary one notation therein, indicates the sign of the y number associated therewith is positive, while a lack of a pulse, or a binary zero notation indicates the y number sign is negative.

Referring to the R channel, corresponding stages of the r number are located in the same positions of the word as those reserved for the y number stages. The P_{25} position of the R channel is considered to hold the carry out digit of the r number corresponding to the dz output. As will be shown later, by example, the sign of the y number in the P_{25} position of the y number can be treated as an additional y stage and combined with the carry out of the r number like an ordinary binary digit.

The P_1 position of the R channel is reserved for a signal which, if present, indicates to the computer unit that it should inspect the output of the adder circuits generating the new y number to note if the y magnitude is exceeding its maximum value. This feature is advantageous during certain computations where the coder may not be able to precisely know how to scale a problem so as to keep the dependent variables within the ranges required by the machine. When a pulse is placed in this P_1 position of the R channel, a checking circuit is introduced into the computer which will stop the machine if the y number overflows beyond the capacity provided for.

The P_{26} position of the R channel of a word is reserved for a sign reversal pulse. When feeding the dz outputs of an integrator into the inputs of other integrators, it is

often necessary, as dictated by the differential equation to be solved, to reverse the sign of the *dz* output. A binary "one" signal in this position causes the computer unit **107** to accomplish this function.

The D–1 and D–2 channels of a word contain in predetermined pulse positions, which may be any of the positions P_1 through P_{25}, the *dy* pickup code information. The computer is designed to permit an integrator to receive as many as 15 *dy* pulses. Stated another way, as many as 15 *dz* outputs found in the general storage may be fed as *dy* inputs into the computer unit **107** during a single word time to modify the *y* number contained therein. The reason for providing two *dy* pickup channels, D–1 and D–2, is because two *dz* storage channels, Z–1 and Z–2, comprise the general storage. As will be more clearly shown in the ensuing description, it is because of this arrangement that it is possible for an integrator to pick up these 15 *dy* outputs from any of the 50 integrators in the computer during a lapse of only 25 clock pulse periods, namely P_1 through P_{25}.

The D–X channel is provided for picking up the *dx* code pulse from the Z–1 or Z–2 general storage channels. Only one *dx* pulse is picked up from the general storage during a word time. This *dx* code pulse may be located in any of the pulse positions P_1 through P_{25} of the D–X channel. As with the *dy* pickup code, in order to be able to pick up a *dz* output pulse from any of the 50 integrators, it is necessary to inspect both the Z–1 and Z–2 general storage channels. The decision as to which general storage channel will be inspected is determined, as far as the computer unit is concerned, by the content of the P_0 position of the D–X channel. A binary "one" signal in this position causes the *dx* code pulse in the following P_1 through P_{25} position to inspect the Z–1 general storage channel, while a "zero" in this position results in the Z–2 general storage channel being inspected.

As previously mentioned, one feature of the present computer is that information external to the machine may be fed into certain integrators according to coded instructions associated with the word. Another feature of the present computer provides for feeding information out of certain integrators according to code associated with the word.

Before describing the arrangement of this code in a word, the external input and output circuits associated with the computer, for accomplishing this transfer of information, will be broadly described and illustrated in Figure 6. Here it is shown that ten sets of input lines **216** are provided for feeding information into the computer. The positive input line of each set is designated by the letter "*u*" and the negative input line by the letter "*v*." Numerical subscripts 1 through 10 distinguish the various sets. These lines carry into the computer, electrical signals corresponding to *dy* unit increments. These inputs are fed in from instruments, for example, while the computer is operating.

Ten sets of output lines **217** are shown for feeding *dz* outputs from the computer. These outputs are designated by the capital letter Q with numerical subscripts 1 through 10 distinguishing the various sets. The *dz* outputs are, likewise, fed out while the machine is computing.

In addition to feeding out *dz* outputs, another set of outputs **218** is provided in the computer for feeding out information while the computer is operating. These output lines **218** are employed for serially feeding out a different *y* number, stage by stage, on each of ten lines designated S_6 through S_{15}. This readout circuit makes it possible to serially transfer a *y* number from a particular word time into an external register, for example.

The neon readout **361**, shown in Figure 6, provides a means for visually indicating one of the word channels. This readout circuit is operative while the machine is idling and will be discussed later since it is not controlled by coded instruction.

Referring back to Figure 4, it will now be shown how the coded instructions located in a "word" of the memory provide the signals for operatively connecting particular integration processes in the computer with these external input and output lines, while the computer is generating a problem.

The P_0 positions of the R, Y, D–1 and D–2 channels contain the P_0 code which in effect "connects" a particular integrator to one of the ten sets, u_1–v_1, u_2–v_2, etc., of external rate input lines **216**. This is accomplished by assigning a binary number ranging from 0001 to 1010 to each set of the external input lines **216**. These binary numbers are assigned according to the decimal equivalent of the subscripts used for designating the lines. Referring to the table in Figure 75, these binary code numbers are shown. They are symbolically noted as E^1, E^2 . . . E^{10}.

Any time a particular word time contains one of these binary numbers in its P_0 code positions, circuit means come into play such that the information on the corresponding set of input lines **216** is transferred into the computer unit **107**. This P_0 code is located in a particular word with the least significant of its four binary digits appearing in the R channel, and the succeeding digits in the Y, D–1 and D–2 channels respectively. Any time the code is in binary form 0000, which is actually a lack of code, nothing is picked up from the external input by the computer.

As will be explained more thoroughly in the ensuing description, because of delays in transferring the external inputs into a computer, this arbitrary input information is not used for computations until two word times later. Because of this, the P_0 code is also available to be used for selecting one of the ten output lines S_6 through S_{15} to pass an electrical indication of the *y* number immediately following, i.e., of the same word time, out of the computer.

As shown in the table of Figure 75, the binary P_0 code numbers for this latter operation ranges from 0110 to 1111, designated E^6 through E^{15}. The *y* output lines **218** are each assigned one of these codes in accordance with the decimal equivalent of its numerical subscript as shown in Figure 6.

It will be noticed in the table of Figure 75 that five code numbers, 0110 to 1010 (E^6 through E^{10}), are used for both of the coding operations so far described, but since the *y* number is fed out during the word time in question, and the external input is used two word times later, it is possible for the coder to utilize both operations by arranging the integrator setups in the required order on the drum. Thus all twenty operations can be performed with the fifteen code numbers.

An output four place code number, known as the P_{26} code, is provided in the P_{26} positions of the Y, D–1, D–2 and D–X channels of each word time. The least significant digit of this binary code number appears in the D–X channel and the following significant digits appear in the Y, D–1 and D–2 channels respectively.

This P_{26} code provides a signal for feeding the *dz* outputs currently being generated, while the machine is computing, on to one of the ten sets, Q_1 through Q_{10}, of external output lines **217**, shown in Figure 6. As with the P_0 code for the external inputs **216**, the binary numbers 0001 and 1010 are assigned to each of the sets of external output lines **217** and a similar number in the P_{26} code of one of the words effectively "connects" the *dz* output currently being generated to the corresponding output line.

Pickup circuits

Referring to Figure 7, the pickup circuits for sensing the information on each of the channels of the magnetic drum will next be described.

The information recorded on each channel of the drum periphery is in the nature of magnetic flux generated in either one direction or the other. When the flux is in one direction in a given pulse position, it represents a binary digit "one," and when it is in the other direction

it represents a binary digit "zero." Since the non-return-to-zero method of storing information on the drum is employed, the recorded flux pattern only changes on successive pulse positions when the binary digit in a number changes from a 0 to a 1, or vice versa.

The magnetic pickup head, such as the Y pickup head **113**, essentially differentiates the recorded flux pattern on the drum **106**. Assume the flux pattern M, shown in Figure 7, is recorded on the magnetic drum. This pattern when scanned by the Y pickup head **113** produces a positive and a negative pulse y^+ and y^-, as shown by waveform **134**. These pulses are fed through a two stage amplifier, comprised of tubes **135** and **136**, which emits pulses of the same polarity as the input. These latter pulses are then coupled into a phase inverter circuit comprised of tubes **137** and **138** whose cathodes are joined at point **139** to a common resistance **140** connected to a negative source of 130 volts. Each of the plates of phase inverter tubes **137** and **138** are held by clamping diodes **141** and **142** respectively, between 45 volts and 90 volts.

Referring to the operation of these circuits, the tubes **137** and **138** of the phase inverter are normally both conducting such that their plates are being held at 45 volts by the clamping diodes. The output from the plate of amplifier tube **136** is applied onto the control grid of phase inverter tube **137**. The positive pulse y^+ when impressed on the grid of tube **137** increases the current flow therethrough. This would ordinarily lower the plate voltage of tube **137**, but the plate voltage does not change in the present circuit because of the clamping diode **141** which holds it at +45 volts. Thus, the positive pulse y^+ has no effect on the output of tube **137**. But the second tube **138** in the phase inverter generates a positive pulse pattern whenever the y^+ pulse is applied onto tube **137**. This is due to the momentarily increased current flow of tube **137**, which, upon flowing through common resistor **140**, raises the potential of the cathode of second tube **138** causing this tube to momentarily reduce its current flow.

When negative pulse y^- is impressed on the grid of tube **137**, the current through this tube is decreased, and the clamped plate output line **143** of tube **137** gives a pulse output inverted with respect to this input. This cutting off of tube **137** lowers the potential of point **139** but does not change the output of tube **138** since it is clamped at 45 volts.

Thus, referring to the output lines **143** and **144** connected to inverter tubes **137** and **138** respectively, the positive pulse on output line **144** represents the timing of the leading edge, and the positive pulse on output line **143** represents the timing of the trailing edge of the magnetic flux pattern M.

These clamped outputs, **143** and **144**, are D.C. coupled through a resistor **145** and a capacitor **146**, in parallel, such as shown for output line **143**, to the control grids of tubes **147** and **148** which serve to invert the pulses applied thereto. The outputs of tubes **147** and **148** are clamped between 45 volts and 90 volts, similarly to the outputs of tubes **137** and **138**. As a result of these clamping operations, the pulses have an abrupt rise and fall, i.e. the pulses are squared. These inverted pulses are then applied onto a *Ym* flip-flop circuit comprised of tubes V_1 and V_2. The manner in which these tubes are coupled together for providing a two stable state circuit of the Eccles-Jordan type will be fully described later, along with its associated circuits, in Figure 19. It should be understood, however, that these flip-flop circuits should not be limited to this type since other flip-flop circuits can be readily adapted to operate in the manner hereafter to be described.

In the present description, it is assumed that the *Ym* flip-flop is functioning with tube V_2 in an operative state and tube V_1 in an inoperative state prior to their response to the signals generated by the magnetic pattern M.

Thus, the negative square waveform representing the leading edge of the flux pattern M as obtained from the plate of tube **147** is differentiated in differentiating circuit **150** and the positive portion of the resulting wave-form is attenuated in the back resistance of diode **151**, while the negative portion is applied onto the control grid of the right tube V_2 of the flip-flop *Ym*, resulting in this tube becoming inoperative and tube V_1 flipping to an operative condition.

As a result of triggering the right tube V_2 into an inoperative condition, the right output line **154** connected to its plate is swung to a relatively high potential and the left output line **155**, connected to the plate of tube V_1, is swung to a relatively low potential. These high and low potentials correspond to the clamping limits, 90 volts and 45 colts, maintained on the output lines by clamping diodes **156** as shown on the right output **154**.

After a time duration corresponding to the number of clock periods represented by the width of the flux pattern M, the square waveform representing the trailing edge of this flux pattern M, as obtained from tube **148**, is applied onto the input circuit connected to the grid of left tube V_1 of the flip-flop *Ym* where it is differentiated and limited by differentiating circuit **150***a* and diode **151***a*. The steep negative pulse obtained, triggers the *Ym* flip-flop back into its original state. As a result of this latter triggering of flip-flop *Ym*, the right output **154** is now swung to the relatively low potential of 45 volts, and the left output **155**, which corresponds to the inverse of the flux pattern M, is swung to a relatively high potential, 90 volts.

Thus the potential of *Ym*, the right output **154** of flip-flop *Ym*, is seen at all times to follow the magnetic pattern M, as sensed on the Y channel of the drum.

The circuits for transferring information from the magnetic drum memory to the memory flip-flops are similar for the R, D–1, D–2, D–X, Z–1 and Z–2 channels. Thus, the respective memory flip-flops R*m*, D1*m*, D2*m*, DX*m*, Z1*m* and Z2*m* are made to follow the magnetic pattern on their respective channels.

The pickup circuitry for the clock channel C, as shown in Figure 8, will next be described. The magnetic wave pattern **157**, as picked up by the clock pickup head **119** sensing the clock channel, is a sine wave. As with the previously described pickup circuits, the head **119** feeds its output into a two-stage amplifier **158**. The output of amplifier **158**, after being clipped by diode **159***a*, is coupled into a familiar Schmidt triggering type circuit **159** on whose output a square wave **160** is generated. This square wave **160** is applied onto both grids of a pair of driver tubes **161***a*, **161***b* connected in parallel. The common plate output **164** from the driver tubes **161***a*, **161***b* is clamped by upper and lower diodes **162** and **163** respectively, to 90 volts and 45 volts. The clock pulses C used for timing and synchronizing the computer are thus obtained from the common plate output **164**.

The reason for having two driver tubes is because a considerable amount of current is required for driving the gate circuits controlling the inputs to the flip-flop circuits. Furthermore the clock pulse source must be of relatively low impedance in order to generate the desired waveforms in these gate circuits.

Having described the manner in which the information is picked up from the channel of memory drum **106** and fed into the channel memory flip-flops, the method by which this information is synchronized or put in step with the clock pulses will next be described. Because this information is being transferred from an electronic medium to a mechanical medium and then, after a given time delay, back into the electronic medium, the information is likely to get out of step with the electronic circuits which are all designed to operate in synchronism with the clock pulses. This nonconformity may come about due to vibration or "jitter" in the operation of the magnetic drum, for example.

Referring to Figure 9, the Y*m* flip-flop is schematically shown. The right output line **154** is marked Y_m and the left output line **155** is marked Y_m'. The right and left grid inputs to the Y*m* flip-flop are simply denoted by lines marked y_m and $_0y_m$, respectively.

The Y*m* flip-flop operates, as previously shown and described in Figure 7, to follow the magnetic pattern M on the drum.

The Y1 flip-flop is the circuit by which the computer unit **107** senses the information in the Y channel. This Y1 flip-flop, which is identical to the Y*m* flip-flop, is comprised of right and left outputs designated Y_1 and Y_1', and right and left grid inputs designated y_1 and $_0y_1$, respectively.

Each of the grid inputs to the Y1 flip-flop has a circuit associated therewith, such as left gate **165**, which is coupled to input $_0y_1$ by differentiating circuit **166** and diode **167**.

Each of the gates is comprised of a pair of input crystal diodes such as diodes **168** and **169**, as shown for the left gate **165**. The anode-ends of these diodes are joined to a common junction **170** which is connected to a relatively high potential source, +265 volts. The operation of left gate **165** is such that any time the input to the cathode-end of either diode **168** or **169**, or both of them, is relatively low in potential the common junction **170** is relatively low. However, when inputs to diodes **168** and **169** are both relatively high in potential, the output of gate **165** is relatively high.

In the circuit shown in Figure 9, the right output Y_m is connected to one input of the right gate **165***a* and the left output Y_m' of the Y*m* flip-flop is connected to one input of the left gate **165**. The clock pulse source C is connected to each of the other inputs of these gates.

Assume, for example, that a unit pulse **171**, as shown, is impressed on the left output Y_m' of the Y*m* flip-flop. As this pulse is applied on the left gate **165**, the clock pulse C generated by clock pickup head **119** is so timed, by the positioning of the C pickup head **119** on the drum **106**, that the clock pulse C is straddled by the unit pulse **171** as shown by the waveforms in Figure 9. Corresponding to the time when both inputs to the left gate **165** are high in potential, which represents the duration of the clock pulse C, the output of gate **165** is high. Thus, a gate output pulse **171***a* similar to the clock pulse C is applied onto differentiating circuit **166** and diode **167** which permits only the negative differential pulse, corresponding to the trailing edge of the clock pulse, to pass to the left grid of the Y1 flip-flop, thus swinging the left output Y_1' to a high potential.

When the Y*m* flip-flop changes state such that the Y_m' potential drops and the Y_m output rises, this latter rise, when applied onto right gate **165***a*, along with the next clock pulse, results in the Y1 flip-flop changing state so that the Y_1' output abruptly swings to a low potential.

It should be understood that the Y1 flip-flop outputs are clamped by diodes (not shown) to swing between +45 and +90 volts, similar to that shown for the Y*m* flip-flop in Figure 7.

It is thus seen that the Y*m* flip-flop output is delayed slightly in order to become synchronized with the clock pulse periods. Except for this delay, it can be said that the Y1 flip-flop follows the Y*m* flip-flop in its operation.

Similar circuitry, as shown for synchronizing the Y*m* flip-flop information, is used for aligning all the other information picked up from the drum, with the operation of computer unit circuits.

In Figure 9, it is noted that a YD flip-flop is shown with its inputs connected to corresponding grid inputs of the Y1 flip-flop. This YD flip-flop changes exactly in accordance with the Y1 flip-flop so that outputs Y_D and Y_D' represent additional sources of current which can be substituted for the Y_1 and Y_1' outputs. A driver circuit such as the YD flip-flop, is required in the computer circuits whenever a particular flip-flop, such as Y1, cannot supply all the current needed to drive the logical nets associated with its outputs.

Referring next to Figures 12, 13, and 14, the spacing of the pickup and record heads for each of the channels of the memory drum **106**, broadly referred to in Figure 3, will now be described in detail.

Figure 12 reveals the operation of the Y channel recirculating loop. As previously shown, the Y channel pickup head **113** senses the information on the drum **106** and feeds it into the Y*m* memory flip-flop whose outputs vary directly in accordance with this information. The information in the Y*m* flip-flop is then slightly delayed approximately ½ clock period, before it is transferred to the Y1 flip-flop. The computer unit **107** then senses the outputs of the Y1 flip-flop, and utilizes this information to logically generate an output bit of information which is recorded, without delay, onto the drum **106** by means of Y record head **125**. This information is recorded ahead of the point on the drum by an arcuate distance equal to the "dead" zone **120**. In the present embodiment, having 1440 clock pulses located around its periphery, this "dead" zone corresponds to 90½ clock periods. However, the more significant arcuate distance is the complement of the dead zone arc which figures out to be 1349½ clock periods. Since 50 words (integrator setups) are provided on the memory drum and each word is comprised of 27 pulse positions, a total of 1350 pulse periods is required in order to serially pass all the required information through the computer unit **107**. Thus, the output information from the computer unit **107** is recorded 1349½ clock pulse periods ahead of the point from which it is picked up. The added ½ clock period delay between memory flip-flop Y*m* and the Y1 flip-flop makes the total delay of the memory equal to the 1350 clock periods.

The spacing for the heads on the R, D–1, D–2 and D–X channels is identical to that for the Y channel and will not be further presented.

In Figures 13 and 14, the spacing for the heads on the Z–1 and Z–2 channels are presented. As previously described, these channels represent the general storage for the *dz* output pulses generated as a result of advancing each of the integrator setups on the drum an incremental step.

As shown, the information in the Z–1 and Z–2 channels of the drum **106**, after being sensed by pickup heads **117** and **118**, is recorded in memory flip-flops Z1*m* and Z2*m*, respectively, in a manner similar to that described in detail for the Y channel circuitry.

Referring first to the Z–1 channel circuitry in Figure 13, the Z1*m* information is delayed approximately ½ clock period on being transferred to the Z1 flip-flop. The Z1 flip-flop information is then sensed by the computer unit **107** and deposited without delay by means of Z–1 record head **130** onto a point of the drum 52½ clock periods behind the point from which it was picked up. Note that the total delay thus obtained in the Z–1 circuitry is 53 clock periods. This is one clock period less than the number of pulse positions in two word lengths of the drum. As will be more clearly shown later, by making the Z–1 storage line 53 clock periods long, the information therein precesses with respect to the information in successive words of the memory drum. Since the Z–1 line is one clock pulse short of two word lengths, the information advances one pulse position earlier each time it is picked up.

As shown in Figure 13, the Z1 flip-flop information is also transferred, with a one clock period delay, into the Z1*a* flip-flop.

It is thus possible, as for instance when it is desired to idle the computer, to sense the Z1*a* flip-flop information rather than the Z1 flip-flop information by the computer unit **107**. When operated in this manner, the Z–1 information does not precess with respect to the word information, since the Z–1 storage line for this case is

54 clock periods in length, which is equal to two word times.

The Z–2 channel circuitry is presented in Figure 14. This circuitry is similar to that in the Z–1 channel. Here the Z2*a* flip-flop is the one which is shown to follow the Z2*m* flip-flop with approximately a ½ clock period delay. The Z2 flip-flop then follows the Z2*a* flip-flop with a one clock period delay. The computer unit **107** observes the contents of the Z2 flip-flop during the computing operation when it is desired to precess the information, and it observes the content of the Z2*a* flip-flop when it is desired to recirculate the information during idling. As previously mentioned, the Z–2 channel information precesses oppositely to that of the Z–1 channel. Thus, note that the delay in the Z–2 channel circuitry adds up to 55 clock periods when the computer unit senses the Z2 flip-flop. This delay is one clock period longer than two word periods. Thus each time a bit of information is recorded in Z–2 record head **131**, it precesses one pulse position later with respect to corresponding pulse positions in the words.

The operation of the Z–1 and Z–2 channel circuits, known also as the general storage lines, will be described in further detail in the ensuing discussion.

Referring to Figure 15, a functional diagram will next be shown and described which represents broadly the "system of thought" generated by the present computer unit **107** (Figure 3).

Functional description of computer unit

The operation of the computer unit **107** is actually carried on in a parallel manner in that during each operation time, as represented by a clock period, information from a given pulse position of each of the channels is observed by circuits in the computer unit **107**. This information, along with that already stored in the computer unit flip-flops, determines output information which is recorded back onto the drum during this same clock period. In addition, certain flip-flops in the computer unit, interconnected by logical nets, are caused to change their states at the end of this clock period. The logical conditions for controlling the triggering of these flip-flops vary in accordance with the memory inputs and the previous states of the flip-flops, representing stored information, as observed during the clock period.

In Figure 15, the channels of the memory are denoted in the block representing the magnetic drum **106**. The information obtained from these channels is considered to be fed out from the right side of the memory, and the information utilized and altered by the computer unit is considered to be recorded back into the left side of the memory.

The computer circuits, functionally shown, represent digital logical processes which are performed in a serial manner by flip-flop circuits and logical nets. It should be understood that the "system of thought" here presented represents the processes which can be performed by a computer unit **107** during a single word time. As will be more clearly explained in connection with Figure 13, however, in order to perform all the operations associated with a particular integration process, information from as many as three successive word times of the memory drum must be received.

Referring first to the clock channel C, the clock pulses are picked up and fed via clock line **175***a* into the P counter **175**, in which 27 clock pulse positions, corresponding to the basic word cycle, are counted off. Certain of these P counts and ranges of P counts, are manifested as high voltages on output lines, such as **176**, to provide timing signals for the computer unit.

The overflow output of the P counter **175** feeds into a W counter **177** which counts through the major cycle of 50 words and manifests word counts, so obtained, as high voltages on individual output lines, such as **178**.

The process by which the *dy* inputs for an integrator are obtained will next be broadly described.

During P_0 of a word time, an arbitrary input may be fed into a Σdy counter **180**. This input is obtained from the U*a* and V*a* flip-flops which have been previously filled by one of the external set of *u–v* inputs **216** shown in Figure 6. The U*a* and V*a* flip-flops are able to store a plus and minus *dy* unit, respectively.

During P_1 through P_{25} pulse times the *dy* inputs are obtained from the information located in the Z–1 and Z–2 channels of the memory. In order to obtain these latter *dy* inputs, the information from the D–1, D–2, Z–1 and Z–2 channels of the drum are fed into a *dy* decoder circuit **179**. The exact logical circuitry provided for performing this decoding will be described later. Generally, a pulse received during a given pulse time, on either channel D–1 or D–2, dictates that the *dz* output in the corresponding pulse time in either the Z–1 or Z–2 channels, respectively, be picked up and fed into the Σdy counter **180**. The coincident output of the Z–1 or Z–2 channel may be either a pulse, representing a $+dy$, or a blank, representing a $-dy$ unit increment. Since a unit input from either one or both, or neither, of the two Z channels may be picked up during a given pulse time, and since the signs of the inputs may be either plus or minus, the algebraic sum of *dy* unit inputs fed into the Σdy counter **180** during any single pulse time may be equal to 0, +1, −1, +2 or −2. The function of the *dy* decoder **179** is to thus analyze the *dy* inputs and manifest the resulting algebraic sum by a high potential on one of the appropriate *dy* input lines **179***a* feeding the Σdy counter **180**

Thus during P_0 of a word time, the external *dy* input circuitry is made operable to determine if any *dy* inputs are available, and during P_1/P_{25} pulse times of the following word period the Z channels are decoded to determine the remaining *dy* inputs.

During P_0 time of any word period, while the Σdy counter **180** is observing the inputs from the arbitrary storage flip-flops U*a* and V*a*, the contents of the Σdy counter **180**, as obtained during the previous word period, is observed by a Σdy register **181**. This observation is schematically presented by showing gate **188** to be open only at P_0 time. At the end of the P_0 pulse time this content is transferred bodily into the Σdy register **181**.

The binary number so obtained is the Σdy register **181** is fed out, digit by digit, into a $y+\Sigma dy$ adder **183**. This operation is obtained by bodily stepping the contents of the Σdy register **181**, each clock pulse time, by means of stepping circuit **182**, toward the least significant stage. Thus the $y+\Sigma dy$ adder **183**, which operates in a serial manner, need see only the content of the least significant stage of the Σdy register **181**. This stepping operation is initiated by the start pulse located in the Y channel of each word time. Referring to the Y channel in Figure 4, the start pulse is the first pulse therein after the P_0 position.

This start pulse is used for triggering a start flip-flop **184** into its "on" position, i.e. true state, as represented by a high potential on the right output S_1 of this flip-flop. The binary information following the start pulse on the Y channel represents the *y* number of an integrator setup. This *y* number is fed from the magnetic drum memory, digit by digit, by means of connection **183***a*, into the $y+\Sigma dy$ adder **183**, along with the information in corresponding stages of the Σdy register **181**. The $y+\Sigma dy$ adder **183** may be considered to add binary digits similar to the manner in which a human adder adds corresponding digits. Any carries obtained as the result of a stage addition are stored in a carry flip-flop K1 which then supplies this carry into the $y+\Sigma dy$ adder **183** along with the digits for the following stage. The digit outputs of the $y+\Sigma dy$ adder during the time the start flip-flop is "on" corresponds to the new y number, denoted y_g. This information is recorded back onto the

Y channel of the magnetic drum memory **106** by way of y_s record line **186** containing a y_s gate **186***a* and a Y mixer **187**. The y_s gate **186***a* is controlled in accordance with the S_1 output of the start flip-flop.

It should be noted that the output on the Y channel from the magnetic drum memory **106** takes two routes. During the time the start flip-flop is "on," as noted by its output S_1 being of a relatively high potential, the information may be considered to be passing along route **183***a* to the $y+\Sigma dy$ adder **183**. During the time the start flip-flop is "off," as noted by its output S_1' being of a relatively high potential, the Y channel information is routed through Y recirculating gate **185** and Y mixer **187** back onto the Y channel of the drum **106**. This latter route makes it possible for the information in the Y channel, such as the instructions and code, to be recirculated without change.

In addition to feeding the new y number, y_s, back onto the magnetic drum, this information is also fed into the $r+y_s$ adder **189** by way of two parallel routes **190** and **191** controlled by respective adder gates **192** and **193**, only one of which may be opened at a time. Since it is the sign of the dx pulse, fed into the computer unit **107** during each word period, which controls the adder gates **192** and **193**, the manner of obtaining this information will next be described.

The D–X channel contains the coded information which enables the dx input to be obtained by scanning either the Z–1 or Z–2 channels during a word time. The P_0 position of the D–X channel decides whether the Z–1 or Z–2 channels should be scanned to obtain the dx input. If a pulse, i.e. a "one," is coded in this position, the information from the Z–1 channel is scanned. If the P_0 position is blank, i.e. a "zero" is coded therein, information from the Z–2 channel is scanned. This channel selecting is denoted by block **194** in the diagram. During the following P_1 through P_{25} pulse times, the dx information is obtained from the selected channel. Only one dx code pulse is located in the pulse range P_1/P_{25} of the D–X channel so that only the corresponding position of the selected channel is observed. This position of the selected Z channel is decoded by dx decoder **195**. If a pulse appears in the position of the selected Z channel, corresponding to the position of the dx pickup pulse in the D–X channel, a "plus" dx is recorded in a dx counter **196**. If a blank appears in this corresponding position, in the selected Z channel, a "minus" dx is recorded in the dx counter. This dx input is picked up during the P_1–P_{25} pulse times of a word, and then transferred during the following P_{26} pulse time into a dx register **197**. It is thus during the word period following the one in which it is picked up that the dx input controls the adder gates **192** and **193**.

For reasons which will be explained more thoroughly in the ensuing discussion, a positive dx stored in the dx register **197** opens adder gate **192** permitting the output binary digits of the $y+\Sigma dy$ adder **183** to be fed via adder mixer **198** into the $r+y_s$ adder **189**. When the dx input stored in the dx register **197** is negative, the adder gate **193** is opened. In this case, before the new y binary number digits are fed through gate **193** they are reversed in block **201**, i.e. ones are changed to zeros and zeros to ones. What this accomplishes is a means for subtracting the new y number from the r number in the $r+y_s$ adder **189** by adding the complement of the new y number. This method of subtraction requires that a unit be added to the least significant stage of the number. Hence, whenever dx_1 is minus in sign, a carry gate **202** is permitted, in effect, to pass a P_0 pulse through adder mixer **199** to trigger the K2 flip-flop into an "on" or true state, thus feeding an initial carry into the $r+y_s$ adder **189**.

As the new r number, denoted r_s, is generated by the $r+y_s$ adder **189**, it is deposited digit by digit back onto the R channel of the memory drum by way of r_s record line **203**. An r_s gate **204**, in this line, permits information to pass therethrough only when the start flip-flop is in an "on" state, thus insuring that information pertaining only to the r_s number is deposited through R mixer **205** onto the drum.

The R channel, of course, has other information therein during the periods the start flip-flop is false, as denoted by its output S_1' being of a high potential. Return gate **206**, controlled by the S_1' term, feeds this information back onto the drum by way of R mixer **205**. This information includes the code in the P_0 position, a possible y overflow indicating pulse in the P_1 position, and the sign reversing code pulse in the P_{26} position. All this information is recorded initially in a particular word time on a drum and does not change during the computation of a particular problem, consequently it is always recorded without changes. All the information initially recorded in the D–X, D–2 and D–1 channels does not change during the computation of a particular problem. Thus, this information is recirculated back onto the drum by way of line **207** for the D–X channel; by way of line **208** for the D–1 channel; and by way of line **209** for the D–2 channel.

The output from the $r+y_s$ adder **189** during the P_{26} pulse time represents the dz overflow pulse obtained from the additive transfer process. This output is fed through function gate **211** into both the Z–1 and Z–2 general storage or precession channels by way of Z–1 and Z–2 mixers **219** and **218***a*, respectively, connected in parallel. Since during a word period both the Z channels are recirculating all their information, except that which is altered during the P_{26} pulse time, a Z–1 gate **212** is provided in the Z–1 recirculating line **213** and a Z–2 gate **214** is provided in the Z–2 recirculating line **215**. These gates are open during a word period, except the P_{26} time, as denoted by P_{26}' to redeposit the information back onto their respective channels.

In Figure 16 a time chart shows further how the system of computation of the present invention is accomplished.

It was previously noted that although all forms of the information required in an integration process are included in a word section of the memory, and the computer unit can process all this information during one word period, all the information so processed is not for the same integration process.

Thus, when referring to a given integrator setup in the memory, in order to pick up the dx and dy inputs and then to operate on the numbers, two successive word periods are required. If the process of picking up arbitrary rate inputs is included in a given integrator setup, then three successive word periods are required to perform an integration process.

Referring to the chart in Figure 16, it is there shown that an integrator setup I_i in the present invention is defined by three successive word periods. Assuming the integrator setup I_i is characterized by the subscript associated with the word period when the dz output is generated, the first process in integrator setup I_i occurs in the word period defined by W_{i-2}.

During P_0 of the W_{i-2} period the P_0 code is set up in the E register (see Figure 76) so as to virtually connect the computer unit to the external arbitrary input lines. During P_1 of the W_{i-2} period any arbitrary inputs so received are sensed and stored in the U*a* and V*a* flip-flops. These arbitrary inputs are so stored for the remaining portion of the W_{i-2} word period.

During P_0 of the succeeding word period, W_{i-1}, these arbitrary inputs are transferred into the Σdy counter **180** where they are totaled, along with the dy inputs as picked up from the Z–1 and Z–2 storage lines during pulse times P_1 through P_{25} of the same word period. By transferring the contents of the U*a* and V*a* storage flip-flop, and having previously reset the E register, this equipment is free to operate again during the W_{i-1} word period.

Simultaneously with the totaling of the *dy* inputs as above described, the *dx* input is also obtained during the W_{i-2} period. The proper channel, either the Z–1 or Z–2 storage line, is first selected during the P_0 time and then the *dx* input is picked up and stored during the succeeding P_1 through P_{25} time.

Having thus obtained the *dy* and *dx* inputs during the W_{i-2} and W_{i-1} periods, the main integration routine for integrator setup I_i is now ready to be performed in the W_i word period.

As previously mentioned, it is desired that this main integration routine be accomplished by separate equipment so as to free the pickup circuits, namely, the Σdy counter **180** and the *dx* counter **196**, so that they can immediately start to function again. Thus, the contents of the Σdy counter and the *dx* counter are transferred bodily into the Σdy register **181** and the *dx* register **197**, respectively.

This transfer could take place any time after the P_{25} time of the W_{i-1} word period since all the inputs *dx* and *dy* have been received by then. Thus, in the present embodiment, the contents of the *dx* counter **196** is transferred during the P_{26} time of the W_{i-1} word period, while the contents of the Σdy counter **180** is transferred, one pulse time later, which is the P_0 time of the W_i word.

As previously described in connection with Figure 15, when the start pulse is received during a word period, the adders effectively start functioning. Consequently, during the W_i period, when the start pulse is received, the Σdy register **181** commences to step its content into the $y+\Sigma dy$ adder **183**, where it is algebraically added digit by digit to the old *y* number being picked up from the memory.

Simultaneously with starting the $y+\Sigma dy$ adder **183**, the $r+y_s$ adder **189** becomes activated. The contents of the *dx* register **197**, depending on its sign, sets up the proper circuitry in the $r+y_s$ adder **189** so that the new *y* number being generated is either added or subtracted from the existing *r* number being picked up from the memory.

It should be noted that the operation of both the $y+\Sigma dy$ adder **183** and the $r+y_s$ adder **189** is completed by the end of the P_{25} pulse time of the W_i period.

During the P_{26} pulse time of the W_i word period, two final functions are performed. The P_{26} code is set up in the E register to effectively "connect" the *dz* output being generated as the result of the $r+y_s$ addition, to the external output lines Q_1, Q_2, etc. (Figure 6) and the sign reversal pulse, if any, is sensed by the computer unit **107** to reverse the sign of the *dz* output before it is fed out.

It should be understood, as previously described in connection with Figure 6, that the P_0 code which is set into the E register during the W_i period, can be used also for causing the *y* number appearing in the following pulse positions, P_1 through P_{25} of the same word, to be fed out digit by digit on an output line **218** into an external register.

The processes above described for the W_{i-2}, W_{i-1} and W_i word periods are the ones required to accomplish a single integration process in the computer. These contents of the three successive word periods, when fed into the computer unit **107** (Figure 3) result in advancing the integration process in question, one digital step.

The chart of Figure 16 now reveals the exact nature of how the computer unit operates. Note that the descripton of the operations in the W_{i-1} and W_{i-2} time blocks **221** and **222** respectively are duplicated below the W_i time block **220**, as denoted by time blocks **221***a* and **222***a*.

Actually, during a given word time, the processes of all three blocks may be accomplished by the computer. That is to say, enough apparatus is made available in the computer unit so that all three stages of an integration process are going on simultaneously. However, as is now evident, the process in block **221***a* of the W_i period is operating to advance the I_{i-1} integrator setup, and the process in block **222***a* is operating to advance the I_{i+2} integrator setup. Of course, if there are no such latter integration processes set up in the memory, as in some instances is the case, this equipment remains idle for these particular word periods.

Having described the computer unit **107** broadly, the detailed description of these circuits will next be presented.

Counter Circuits

The P and W counter circuits and their operation will first be described. As seen in Figures 3 and 15, the P counter **175** is fed by clock pulses generated in head **119** associated with the C channel on the drum **106**. The P counter counts repeatedly through a cycle of 27 pulses corresponding to the pulse positions P_0 through P_{26} which divides the length of each word recorded on the drum. Certain pulse counts are manifested as high potentials on count lines **176** for a clock period duration corresponding to the pulse count. These high potential count signals are used in the diode networks, as required, for controlling the operation of the computer unit as it operates on the information picked up from the drum **106** during a word time.

Each time the P counter **175** counts through its cycle, a pulse is fed into the W counter **177** which has a capacity corresponding to the number of word times provided on the drum periphery. Each word count is manifested as a high potential on a word line, such as **178**, for a duration of 27 clock pulse periods. The high potential word signals are used, for example, for selecting a particular word space on the memory in order to fill initial conditions.

Before going into a description of the details of the counter circuitry, the convention employed for presenting the building blocks for physically generating the "system of thought" of the present invention will be explained.

Logical propositions may be considered to be represented in the circuits by flip-flops which are electronic devices having two and only two possible steady state conditions. One of these conditions is referred to as "true" (in tables sometimes represented as "1") and the other condition is referred to as "false" (in tables as 0).

The true and false conditions of a proposition are preferably referred to as terms which are represented throughout the circuits by a D.C. voltage at a point. This voltage can exist at either of two D.C. levels. When a term is effective, the voltage is high, in the present embodiment this voltage is 90 volts; and when the term is ineffective, the voltage is low, 45 volts.

Thus by connecting output lines, for example, to each of the plates of the tubes of a flip-flop circuit, the output line having a high potential determines the effective state (or term) of the flip-flop. The other output line, having necessarily a low potential, then represents the ineffective state.

In accordance with the present scheme, in order to trigger a proposition flip-flop to either its true or false state by signals applied thereto on separate inputs, an input line is coupled to the grid of each of the tubes of the flip-flops.

The nomenclature employed for the present invention uses combinations of capital letters and numbers, or lower case letters, for designating the proposition flip-flops themselves. The outputs of the flip-flops are characterized by corresponding capital letters with the number, or lower case letter, as a subscript. In order to characterize the true state output of a flip-flop from the false, the latter is distinguished from the former by an affixed prime.

As for the inputs to a flip-flop, they are designated by corresponding lower case letters with the associated number or letter as a subscript. The input which, when energized, renders a flip-flop into a false state is further characterized by a subscript zero prefixing the lower case letter.

The flip-flops A1 through A5, used for generating the P counts, are shown in Figure 18. These flip-flops correspond essentially to stages of a binary counter, where flip-flop A1 is the least significant stage and flip-flop A5 is the most significant stage.

The counter arrangement is a parallel one in that a clock or time pulse, otheriwse known as a C pulse, generated by the revolving drum, is applied on all flip-flop inputs simultaneously. The inter-connections of the flip-flop outputs, however, only allow certain flip-flops to be triggered by the successive C pulses so as to change their states in an orderly fashion to indicate the cycle counts.

In Figure 17 the combinations of the states of the flip-flops characteristic of each digit content of the counter is shown by a table. This table is a binary representation of the pulse counts P_0 through P_{26}.

It is noted in the table that in order to count through a cycle of 27 consecutive counts, an initial binary count corresponding to decimal 5 is arranged to be set into the flip-flops, i.e., the counter flip-flops are so inter-connected that whenever they count to capacity, which is the binary equivalent of decimal 32, they reset to the binary equivalent of decimal 5.

On examining the changes in states of the A flip-flops, moving down the table in Figure 17, it is noted that when desiring to make the A1 flip-flop true, i.e., have a 1 therein on occurrence of the next C pulse, the A1 flip-flop must have a 0 therein, i.e., be in a false state. Similarly the A1 flip-flop must be in a true state in order to be switched to a false state on the next C pulse.

It is thus noted, in general, that the A1 flip-flop changes state every time a clock pulse is fed therein.

There is one exception, as can be seen by the binary table in Figure 17. When the P counter resets from P_{26} to P_0 it is not desired to trigger the A1 flip-flop to the "zero" state, but rather it is necessary to maintain the A1 flip-flop in its "one" state. This is because the binary reset number, equivalent to decimal 5, is represented by a "one" in the first stage, A1, of the counter.

Referring next to Figure 19, a detailed schematic circuit diagram is shown, indicating how the A1 flip-flop is connected so as to operate as the first stage of the P counter.

The flip-flop circuit as used in the present invention is a well known Eccles-Jordan type of trigger circuit comprised of two tubes, V_1 and V_2. As shown, each of the tubes has its plate cross coupled to the control grid of the opposite tube by a resistor R_p in parallel with a capacitor C_p. The plate of each of the tubes is connected through a separate load resistor, like resistor R_q, to a positive D.C. source of 265 volts, and the cathode of each triode is grounded. Each of the grids of the tubes is joined through a separate grid resistor R_t to a negative bias —E. The flip-flop circuit is provided with triggering circuits associated with each of its grids and output circuits connected to each of its plates.

Whenever the flip-flop is considered to be in a "one" state, neon light L, connected in series with a limiting resistor R_w across the left load resistor R_q, lights up; and when the flip-flop is in a "zero" state, neon light L is out. Thus the "one" state occurs when tube V_1 is conducting. At this time a low potential is on the left plate of the flip-flop, the plate of tube V_1; and a high potential is on the right plate of tube V_2.

The output lines, A_1 and A_1' from the A1 flip-flop are taken from the right and left plates respectively. In order to maintain the swing of the plate voltage between voltage levels 90 volts and 45 volts, clamping diodes, such as diodes *223a* and *223b* associated with the right output A_1, are provided on each of the output lines.

The inputs to the flip-flop are controlled by gate circuits **224** and **225** associated with the grids of the V_1 and V_2 tubes, respectively. Each of the gates is coupled through a differentiating circuit **226** and blocking diode **227**, as shown in particular for the left grid, the grid of tube V_1.

For this particular counting stage, the right plate output A_1 is connected to one input of the left gate **224**, and the left plate output A_1' is connected to one input of the right gate circuit **225**. In order to inhibit the change in state of the A1 flip-flop, when the counter resets from P_{26} to P_0 (see Figure 17), a potential corresponding to the P_{26}' term is applied to left gate **224**. The clock pulse C is applied simultaneously to the final inputs of each of the left and right gate circuits **224** and **225**.

These gate circuits **224** and **225** are herein defined as typical logical product diode nets. In such a circuit, as noted in particular for left gate **224**, the inputs therein are applied on the cathode-end of crystal diodes **228**, **229** and **230** whose anode-ends are joined to a common junction **231** which is connected to the positive source 265 volts through a load resistor **232**.

Each of the inputs to a gate circuit can be either at the high potential of 90 volts or the low potential of 45 volts, depending on the state of the flip-flop from which it is derived. Anytime all the inputs to a gate circuit are high in potential, the clock pulse C is, in effect, passed to the gate output. Referring to the left gate **224**, the pulse passed therethrough is differentiated in differentiating circuit **226** and the positive portion thereof is blocked by diode **227**, while the negative portion is applied to the grid of the left tube V_1 and thus triggers the tube into a non-conductive state.

It should be noted that the diode **227** is returned through the resistor of the differentiating circuit **226** to a positive voltage of 6 volts. This serves to place a bias voltage of 6 volts across diode **227**. Thus in order to pass current through diode **227**, it is necessary for the signals derived from the differentiating circuit **226** to be more than 6 volts negative. This bias circuit arrangement functions to clip spurious signals resulting from slight variations of the gate output voltage level caused by cutting off of current flow through one of the input diodes of gate **224**. The increased current flow through the other diodes results in the gate output level being slightly lower. This results from the small forward resistance inherent in each of the diodes. The differentiation of these negative drops in voltage levels creates the spurious signals which the bias circuit arrangement effectively prevents from passing to the grid.

In Figure 20 the graphs of the waveforms appearing at different points of the A1 counting stage circuit above described are shown. In line I the regularly recurring clock pulses C are shown; in line II the high potential corresponding to the P_{26}' term is represented; in line III the A_1 plate output is shown to be initially of a high voltage, 90 volts; and in line IV, the A_1' plate output is shown to be initially of a low voltage, 45 volts. As shown in line V, whenever both the waveforms A_1 and C, and of course P_{26}', are relatively high in potential, the term ${}_0a_1$ is considered to pass through the gating circuit **224** as a rectangular pulse similar in waveform to the clock pulse C. The clock pulse source is of very low impedance so as to insure that the trailing edge of the wave is not rounded but relatively square. On line VI the pulse form impressed on the input to the left grid is shown to be the differentiated trailing edge **233** of the rectangular pulse ${}_0a_1$. It is thus noted that the A1 flip-flop changes state on the trailing edge of the ${}_0a_1$ pulse (clock pulse C). It is also noted that as a result of triggering the left tube V_1 off, the left plate output A_1' rises in potential according to the time constant of the flip-flop circuit. The output A_1' is now at a high potential so that on occurrence of the next clock pulse C, the right gate **225**, in effect, allows the clock pulse C to pass therethrough, and hence the differentiated trailing edge **234** of this latter pulse, shown in line VIII of Figure 20, triggers the A1 flip-flop back to its original state.

It should be noted that the circuit arrangement shown

enables the flip-flop circuits in the present embodiment to be reliably operated without the use of any delay circuits. Referring to the circuit shown in Figure 19, the change in the output voltage level of flip-flop A1, induced by a particular clock pulse, controls the opposite gate so as to permit it to pass the next clock pulse. Thus it is very important to control the point at which each of the clock signals triggers a flip-flop so as to eliminate spurious pulses from passing through the gates.

For example, if the change in the A1 flip-flop, induced by a particular clock pulse, were to be sensed by the opposite gate before the clock pulse referred to is dissipated, it is possible that this gate will pass an unwanted signal. The present arrangement of circuit elements overcomes the possibility of this. First it should be pointed out that each clock pulse appears simultaneously on all gates. Now then, triggering at the trailing edge of a clock pulse ensures that the resulting change in flip-flop output voltage levels will necessarily always follow the clock pulse which creates the change. These flip-flop voltage level changes are communicated to the inputs of the gates connected thereto, after the clock pulse in question, thus setting up these gates to allow the next clock pulse to pass.

Stated in other words, during the first part of a clock pulse period, when the voltage from the clock source is low, the transients of the circuitry are occurring. For reliability, these should be completed before the leading edge of the clock pulse arrives. During the time of the clock pulse period when voltage from the clock source is high, the logical net circuits can be thought of as observing the flip-flops and the other sources of inputs so as to know if a pulse should pass onto the grid of any of the flip-flops. The clock pulse must be broad enough so that, taking into account its rise time, it reaches its maximum voltage level before the end of the clock period. The clock pulse must also have a low impedance source, so that a square edge can be created on the trailing edge of the pulse passing through the grid gates. These conditions make it possible to create, by differentiating the abrupt trailing edge, a sharp negative pulse, coincident with the end of the clock period, which can be used for triggering the flip-flops.

Referring next to Figures 18 and 21 the simplified manner in which the remaining circuitry of the present invention is to be presented will now be described.

Instead of showing the wiring diagrams of the remaining flip-flops of the P counter, together with logical circuits, as in Figure 19, the remaining circuits present simplified block diagrams of the flip-flops. It should be understood, however, that all the flip-flop circuits are identical. As shown in Figure 18, only the input and output lines for the flip-flop are shown and these are marked in accordance with the convention previously described. Furthermore, the grid input differentiating and blocking circuits are omitted in the block diagrams for simplicity. Only the gates, indicating the logical product of the control input and clock input, are shown at each of the inputs so as to emphasize the fact that the clock pulses are applied simultaneously to all the flip-flop inputs.

In accordance with the scheme of the present invention, after the "system of thought" to be accomplished is explained by means of binary tables or similar means for systematizing the thought, the terms of the system are represented by the conditions of the flip-flop circuits, or other sources of potential having two possible levels.

Logical equations are then written which define when and how the flip-flop circuits are to change in accordance with the effective terms of the system. The logical equations, so devised, are then presented below the block diagram of the flip-flops.

Writing the logical equations for the grid triggering of a flip-flop circuit is no more than stating the terms which have to be simultaneously of a high potential during a clock period in order that the particular flip-flop should trigger into the particular state at the end of a clock period.

Two distinct notations are used for the equations. The first, "logical multiplication," means that all the terms in the particular product have to be of a high potential in order to make the product effective in a particular equation.

The second, "logical addition," means that at least one term of the sum has to be of high potential in order to make that sum effective in a particular equation.

Thus for example, the equation

$$_0b_1=(U_aV_a+U_a'V_a')P_0C$$

can be interpreted as meaning that the B1 flip-flop, later to be introduced in connection with Figure 38, will change to a "false" state after the following three terms are at a high potential: complex term $(U_aV_a+U_a'V_a')$, and single terms P_0 and C. The complex term $(U_aV_a+U_a'V_a')$, itself, will be of high potential if either (U_aV_a) or $(U_a'V_a')$ is of a high potential, or if both are.

The particular representation of these logical equations has been chosen because these equations can be operated upon according to certain well known rules in a manner similar to that by which algebraic equations are operated upon.

Having once described the means for physically generating a typical logical product and logical sum, the present techniques enable the logical circuitry for solving the entire logical equations to be set up directly by reference to the equations alone. The logical circuitry generally ends up as one large interconnected net made up of the two fundamental circuits. In reducing the equations to physical circuitry, recognition is made of the fact that certain common complex terms and partial products can be generated separately and used repeatedly where needed.

It should be noted that circuits to solve "logical multiplication" are also called "gates" and circuits to solve "logical addition" are also called "mixers."

Returning to Figure 18 the conditions required for flipping the A1 flip-flop are now noted to be represented by symbolic logic equations: $a_1=A_1'$ and $_0a_1=A_1P_{26}C$. Note that these equations are simply a shorthand for stating the mode of operation of the A1 flip-flop as presented in detail in Figures 19 and 20.

By examining the states of the A flip-flops, as presented in table of Figure 17, the symbolic logic equations for the A2 flip-flop can be similarly determined. On moving down the table it is noted that the conditions necessary to make flip-flop A2 trigger to a true state, i.e., change from a 0 to a 1 state, are that flip-flop A1 be in a true state and flip-flop A2, itself, be in a false state. This can be symbolically noted by $a_2=A_2'A_1C$. In a similar manner, the conditions required to make the flip-flop A2 false are that flip-flop A2 be true and flip-flop A1 be true, i.e., $_0a_2=A_2A_1C$.

Thus in general, it is noted by the binary table in Figure 17 that all of the A flip-flops change from a "zero" to a "one" state when, and only when, it is in a "zero" state and all the respective lower order flip-flops are in a "one" state.

In a similar manner, the condition required, in general, for enabling a flip-flop to change from a "one" to a "zero" state on the next clock pulse is for it and all the previous flip-flops to be in a "one" state.

Thus the logical equations for the triggering inputs to the A3, A4, and A5 flip-flops are readily obtained and presented below their respective flip-flops in Figure 18. It should be noted that, as for the A1 flip-flop, when resetting from P_{26} to P_0, it is desired to maintain the A3 flip-flop in its "one" state. This exception is only true for the change from a P_{26} to a P_0 pulse time. Thus by including a P_{26}' term in the $_0a_3$ equation, the A3 flip-flop will operate as above designed except when changing from

the P_{26} count, since P_{26}' will then be of a low potential.

The logical diode networks for physically solving all the triggering equations for the P counter will next be presented by reference to Figure 21.

The network for the A1 flip-flop equation $a_1=A_1'C$ is the gate circuit **225** as shown in Figure 19. Here it is simply shown by designating the inputs to the gate **225**, known as a typical two input product gate, by terms of the a_1 equation. The net for the equation ${}_0a_1=A_1P_{26}'C$ is made up of circuits **235** and **236**. It is noted here that the circuit for the ${}_0a_1$ equation is slightly different than that shown in Figure 19, in that, first the A_1 and C terms are fed through diodes to a common line **237** in circuit **235** whose output, in turn, is then fed, along with the P_{26}' term into the two input product circuit **236**.

The outputs of gates **225** and **236** are marked respectively a_1 and ${}_0a_1$. Each of these product circuits is such that whenever any of the inputs are of relatively low potential, the output is also of relatively low potential; however when all the inputs are of relatively high potential, the output is also of relatively high potential. In other words, the output potential equals the lowest input potential.

The remaining circuitry in Figure 21 represents a family tree network for generating the remaining grid equations of the P counter flip-flops.

Referring to these logical equations below the flip-flops in Figure 18, it is noted that certain products, generating solutions to the grid input equations for the lower order stages, represent partial products in the solutions generated for the higher order stages. Thus in Figure 21, the partial product A_1C generated in block **235** is not only cascaded into the block **236** to generate the ${}_0a_1$ equation, but it is also fed into all the other equation circuits by being impressed on vertical line **238** which represents the main trunk of the tree network.

At points such as **239** and **240**, proceeding up the trunk **238**, terms, are multiplied to the partial product made up of all the terms introduced below such points by the circuits to the left of the main trunk **238**. Thus point **239** represents the partial product A_1A_2C, and point **240** represents the partial product $A_1A_2A_3C$. The terms forming the various equations which are not common to other solutions are then combined with points of the trunk in separate product circuits on the right side of the main trunk **238**, such as solution a_3, which is formed in a two input product circuit **241**. Here the unique term A_3' is applied on one of these inputs **242** and the partial product A_1A_2C, obtained from the point **239** of the trunk **238**, is applied on the other point **243**.

It should be understood that all the logical product circuits in the family tree, such as block **236**, have their load resistors connected to the 265 volt positive voltage source. This is true for the product circuits throughout the logical network of the invention. Similarly, as will be subsequently described in connection with Figure 22, all of the sum circuits of the logical network are returned to a negative 130 volts.

Having provided logical circuitry for interconnecting the P counter flip-flops to count through the basic cycle, the circuitry required for manifesting certain of these counts as high potentials on separate outputs will next be considered.

A P count is characterized by a unique combination of states of the A1 through A5 flip-flops. Referring to the binary table in Figure 17, the binary digits in each row correspond to the states of the flip-flops associated with each P count.

As previously explained, when a flip-flop contains the digit "one," its right output is high in potential; and when a flip-flop contains the digit "zero," its left output is high. Thus in order to obtain a logical product whose solution represents a given count, the states of the A1 through A5 flip-flops, which when high define a count, are symbolically multiplied together. Thus

$$P_0=A_1A_2'A_3A_4'A_5,\ P_1=A_1'A_2A_3'A_4A_5'$$

and the equations for other P counts are similarly written and presented to the right of the binary table shown in Figure 17.

In Figure 22 the logical net for physically generating the P count equations will next be introduced.

Here the matrix type of diode net is employed. This form of circuitry is useful when the equations generated are made up of different combinations of a plurality of terms. The outputs of flip-flops A1 and A2 are fed in on one set of matrix inputs **244**, and the outputs of flip-flops A3, A4 and A5 are fed in on another set of matrix inputs **245**. Interconnecting diodes, such as diodes **247** and **248**, connect the appropriate outputs of each flip-flop to vertical and horizontal partial product lines, respectively, such as **249** and **250**. These partial product lines are then combined in product circuits, such as circuit **251** which generates the P_0 count.

It should be noted that instead of generating the P_2 count directly, it is desired to generate first its logical negative P_2' which can be applied onto a grid of a driver tube. The plate output of the driver tube then provides a P_2 term which can be used to drive a number of other circuits.

It can be shown by symbolic logic that equation

$$P_2=A_1A_2A_3A_4'A_5'$$

can be operated upon to obtain its logical negative such that $P_2=A_1'+A_2'+A_3'+A_4+A_5$. Thus, it is this latter equation which is generated in network **253** of Figure 22. This circuit is defined herein as a typical logical sum network, also known as an "inclusive or" network. Each of the input terms to this sum network is fed through a diode, such as diode **254** to a common junction **255** connected to −130 volts through a resistor **256**. The operation of the circuit is such that when any one, or all, of the input terms are relatively high (90 volts) in potential, the output P_2' is relatively high (90 volts) in potential. Slated in another way, the output potential equals the highest input potential. It should be understood that all the logical sum networks in the computer unit have their resistors returned to a −130 volts.

Referring next to Figures 25 and 26, the W counter flip-flops, together with their grid input logical equations, will next be presented.

The W counter requires a range of 50 counts from W_0 to W_{49}. Instead of providing a single set of flip-flops, as in the P counter, to count through this cycle, two sets of flip-flops are provided. One of these sets corresponds to the unit stage and the other set corresponds to the tens stage of a decade counter.

The unit stage of the W counter, illustrated in Figure 25, will first be described. This unit stage part of the counter is comprised of four flip-flops A7 through A10. These flip-flops are interconnected by a logical counting net to indicate the decimal equivalent digits 0 through 9. The table in Figure 23 shows the states of the flip-flops which, in combination, uniquely define each of these decimal equivalents.

Note that these flip-flops operate similar to the stages of a binary counter with flip-flop A7 representing the first stage, 2^0; flip-flop A8 as the next stage, 2^1; etc.

As was described for the P counter flip-flops, the logical grid equations for the flip-flops are determined by noting the states the other flip-flops in the counter have when the flip-flop in question is to change state on the next clock period.

A logical equation is then symbolically written, defining this change which holds for all such changes of the flip-flop in question during the counting operation. Thus it is noted, by moving down the table in Figure 23, that the first flip-flop A7 changes state each time an input is

received. This input is generated each time the P counter reaches its capacity as denoted by the P count, P_{26}, being of a high potential.

The input equation for rendering the A7 flip-flop true is thus shown, in Figure 25, to be $a_7=A_7'P_{26}C$, and the equation for rendering the A7 flip-flop false is shown as ${}_0a_7=A_7P_{26}C$.

The logical equations for the triggering of the remaining flip-flops A8, A9 and A10 are obtained by inspecting their binary table shown in Figure 23 and are presented below their respective flip-flops in Figure 25.

As previously stated in connection with the P counter, the condition required to control the triggering of a flip-flop ordinarily represents a fixed pattern as obtained by inspecting its own state and all the lower order flip-flop states. However, when it is necesary to reset a counter after reaching its capacity, this triggering pattern of the flip-flops is interrupted and other conditions must be introduced, such as those represented by the higher order flip-flops. Thus when resetting from decimal equivalent 9 to decimal equivalent 0 in the present counting circuit, it is necessary to inhibit the A8 flip-flop from changing to the "one" state which its binary triggering pattern dictates. It is noted, by referring to the binary table in Figure 23, that any time the A8 flip-flop must change to a "one" state, the A10 flip-flop is in its "zero" state. Thus the a_8 triggering equation is shown to be equal to $A_8'A_{10}'(A_7P_{26}C)$ in order to provide for the anomaly in the triggering pattern caused by the counter reset requirements.

The tens stage of the W counter is shown in Figure 26. This stage is comprised of three flip-flops, A11, A12 and A13. These flip-flops are interconnected by a logical counting net to indicate the decimal equivalent digits 0 through 4. That is, this counter has a radix of five which is all that is required since the W counter resets on its W_{49} count.

The overflow of the unit stage of the W counter is used for triggering the tens stage. This input is defined by the logical product $A_{10}A_7P_{26}C$. The combination of terms $A_{10}A_7$, as noted in Figure 23, characterizes the decimal 9 equivalent of the units stage, and the P_{26} term defines the change of a word time.

The logical input triggering equations for the W counter are obtained, as for the other counting flip-flops, by inspecting the binary table in Figure 24, and will not be further described.

Likewise the description of the physical circuitry for generating the W counter flip-flop grid input equations, shown in Figure 27, follows from the previous discussion, and will not be further described.

The physical circuitry provided in the present embodiment for manually selecting a given word time on the memory drum, while it is revolving, will next be shown by references to Figures 28 and 29.

This feature is required to record or read information in or out of a given word time of the drum. On the control box of the computer, see Figure 91, a set of manual W_i selector switches 257 and 258 is provided. One of these switches 257 operates in conjunction with the units stage and the other switch 258 operates in conjunction with the tens stage of the W counter.

When it is desired to manually obtain a signal in the computer indicative of one of the word times W_0 through W_{49}, the switches 257 and 258 are set to the respective digits of the desired word count, as indicated by indicia 257*a* and 258*a* (Figure 91).

The circuitry of the W_i selector switches in Figure 28 shows that the tens stage switch 258 is comprised of three sets of wafer switches designated S11, S12 and S13. There are two wafers such as wafer 260 in each set, each wafer having an arm 261 associated therewith. The arms of all the wafers are mechanically interconnected to operate in unison. Each set of wafer switches S11, S12 and S13 correspond to the states of the A11, A12 and A13 flip-flops, respectively, as shown in Figure 24. Thus wafer S_{13}' of set S13 corresponds to the "false" state and wafer S_{13} of set S13 corresponds to the "true" state of the A13 flip-flop. Each of these wafers, such as S_{13}', has five terminals 0, 1, 2, 3 and 4 which correspond to the digits associated with the tens stage of the W counter.

Referring to the table in Figure 24, the states of the flip-flops and the setting of the wafers which correspond to each of the four decimal digits in the tens stage of the W_1 number is shown to be identical. The 0 digit of the tens stage is characterized by terms S_{11}', S_{12}' and S_{13}' being simultaneously of a high potential. Thus in Figure 28, wafers S_{11}', S_{12}' and S_{13}' have their 0 terminals connected to the 90 volt line 263, while the 0 terminals of the remaining wafers are connected to the 45 volt line 264. In a similar manner the 1 digit is seen in Figure 24 to be comprised of terms S_{11}, S_{12}' and S_{13}'. Thus the 1 digit terminals of the corresponding wafers are connected to the 90 volt line 263. The remaining terminals of the wafers are similarly connected for the 3 and 4 digits of the tens stage of the W number.

The block 262 in Figure 28 corresponds to a similar set of wafer switches which comprise the unit stage 257 of the W number. These wafers (not shown) have their terminals connected to either the 90 volt line 263 or the 45 volt line 264, according to the scheme in the table of Figure 23.

This method of interconnecting the wafer switches provides two sets of output lines 265 and 266 corresponding to terms which in combination characterize the digits of the W number.

The circuit means whereby the output selected by the W_1 selector switches 257 and 258 are compared with the outputs generated by the W counter flip-flops so as to give a signal W_1 is shown in Figure 29.

The coincidence of corresponding states of all the W counter flip-flops and W_1 selector switch wafers, is needed to give a W_1 output indication; hence, symbolically, this is noted as $(S_7A_7+S_7'A_7')$ $(S_8A_8+S_8'A_8')$ etc. By means of symbolic logic, it can be shown that these logical sums and products can be operated upon to give (A_7+S_7') $(A_7'+S_7)$ (A_8+S_8') $(A_8'+S_8)$ etc., which can be generated by a logical network having fewer diodes. In Figure 29 the logical network for generating the W_1 signal in accordance with this scheme is presented.

Thus the circuitry has been presented which shows how the operator can manually select a given word period of the computer by setting the W_1 selector switches 257 and 258 on the control panel (Figure 91). This word period is represented by a high voltage on output line W_1 of Figure 29.

The equipment required and the logical nature of the circuitry involved in performing the processes in the W_1, W_{1-1}, and W_{1-2} word periods of an integrator setup, as shown in Figure 16, will next be described in detail. The word periods W_{1-1} and W_{1-2} will first be described together because they combine to give the inputs used for altering the numbers in an integrator setup.

Arbitrary input circuits

A block diagram illustrating how the arbitrary rate inputs are introduced into the Σdy counter is shown in Figure 30.

In the present embodiment of the computer, as mentioned in connection with Figure 6, inputs from as many as ten sensory instruments (not shown) may be received. The voltage lines from the input sensory instruments are at a high voltage (as contrasted to a low voltage), whenever it is desired to introduce a dy unit input into the computer unit. The positive input voltage lines are designated as u_1 through u_{10} inclusive and the negative input voltage lines as v_1 through v_{10} inclusive.

Whenever one of these input lines is at a high potential, an input rate flip-flop U1 through U10, associated with the positive input lines; or, an input rate flip-flop V1 through

V10, associated with the negative input lines, is energized into a true state.

Each of the sets of inputs u_1–v_1, u_2–v_2, etc. is assigned one of the ten count indications E^1 through E^{10} available in the E counter.

As previously noted in connection with Figure 4, during a P_0 pulse time of a word, the P_0 code is provided for selecting one of the ten sets of arbitrary rate input lines. This is accomplished, as here shown, by transferring the P_0 code number from the memory into the E register **267**.

The E register circuitry, upon receiving this code causes its corresponding count indicating line, E^1 through E^{10}, to be energized to a high potential.

Associated with E register **267**, Figure 30, is shown a pair of coincidence circuits **268** and **269**. Right coincidence circuit **268** interconnects the E register output lines with the U*i* input rate flip-flops, and left coincidence circuit **269** interconnects the E count output lines with the V*i* input rate flip-flops.

The operation of the left coincidence circuits **269** is such that if, during a P_1 pulse time, corresponding outputs of the V*i* flip-flops and E register are of a relatively high potential, then a negative dy unit input is stored in the V*a* flip-flop. The right coincidence circuit **268** operates in the same manner to store a positive dy unit input into the U*a* flip-flop.

The U*a* and V*a* storage flip-flops then transfer their content into the Σdy counter **180** during the P_0 pulse time of the next word period W_{i-1} (see Figure 16).

Having thus broadly presented the method by which the arbitrary *dy* inputs are fed into the Σdy counter, the detail circuitry for accomplishing this operation in the present embodiment will be described.

The flip-flops E1 through E4 shown in Figure 76 function as the code register during the computing phase of the operation of the computer.

As will be described later, these flip-flops also function as a counter when the operator sets the controls to visually readout information in one of the channels of a word.

These two distinct functions of the E counter are distinguished by the G_3 term. As noted in the equations below the E flip-flops, the G_3 term is included in the equation for the code register and its inverse term, G_3', is included in the equations for the counter.

Whether the E flip-flops function as a register or a counter, the combination of states of these flip-flops define the E count, E^0, E^1, E^2, etc. shown in the table of Figure 75. These E counts are each manifested as a potential on a separate line thus making them available to be introduced as terms in the logical operation of the computer. Thus the diode matrix shown in Figure 78 generates each of these E counts by logically combining the proper E flip-flop outputs according to the pattern in Figure 75.

The schematic block diagram and logical grid equations for the E register flip-flops, E1 through E4, will first be presented in Figure 76. As previously noted in connection with Figure 4, the P_0 code of each word determines the particular external input (u–v) which is to be picked up. This four place binary code, comprised of digits in the P_0 position of the R, Y, D–1 and D–2 channels, is transferred at the end of the P_0 time into the E1, E2, E3 and E4 flip-flops, respectively. This transfer occurs whenever the machine is in its computing operation, as characterized by the G_3 term being true. Thus, referring to Figure 76, the triggering equations for transferring the P_0 code to the E register flip-flops all contain the combination of terms P_0G_3C. In addition, each of the true and false triggering inputs of the E1, E2, E3 and E4 flip-flops may be said to reflect the corresponding true and false output of the flip-flops R1, Y1, D1 and D2, respectively. Thus the transfer input equations for flip-flop E1, for example, can be written as $e_1=R_1P_0G_3C$ and ${}_0e_1=R_1'P_0G_3C$. These equations are generated as one of the solutions on lines **272** and **273**, respectively, in Figure 77.

The schematic circuits of the positive and negative random input flip-flops, generally referred to as U*i* and V*i* flip-flops, will next be presented in Figure 31.

Only the U1–V1 set of input flip-flops together with their logical input grid equations will be described since the remaining sets of flip-flops are all similar.

The receipt of a random unit input into a computer can be made at any time so long as the input rate to a particular integrator setup does not exceed the basic rate of computation of the machine. As previously indicated, each integrator setup is computed upon once each revolution of the memory. Thus, not more than one random input indication can be fed into an integrator setup during a memory revolution. As shown in Figure 31, the positive random input voltage u_1 is applied directly onto the right tube of flip-flop U1. As soon as the voltage U_1, representing this incremental unit, has been transferred into the U*a* flip-flop as shown in Figure 30, it is desirable to trigger the U1 flip-flop into a false state at the end of the next pulse time P_2, so as to enable the U1 flip-flop to sense the next incremental change which might be fed in on its u_1 input any time during the next revolution. On examining the conditions of the circuits involved for rendering the U1 flip-flops false, the ${}_0u_1$ triggering equation is found to be equal to $P_2U_aE^1C$. Note that count E^1 of the E register is still true at this time. In a similar manner, each of the remaining U*i* and V*i* flip-flops have a similar logical equation for defining when they trigger to a false state. These equations are presented in the lower part of Figure 31, and the logical networks for solving these equations are presented in Figure 32. Thus the output ${}_0u_1$ is represented as the output of a two input logical product circuit **270** having the term E^1 fed in one input **271***a*, and the partial product P_2U_aC, generated in product circuit **273** fed in on the other input **271***b*.

Referring next to Figures 33 and 34, a schematic circuit is shown of the U*a* and V*a* storage flip-flops together with their logical triggering equations. As previously explained, at P_1 time of a word the contents of the U*i* and V*i* input flip-flops are transferred to the U*a* and V*a* storage flip-flops. Hence a combination of the proper conditions, for example, $P_1U_1E^1C$, will trigger the U*a* flip-flop to a true state. As shown in Figure 30, the terms for setting up the transfer of the contents of the U*a* flip-flop into the Σdy counter **180** are observed during the P_0 pulse time, consequently, the triggering equation for resetting this flip-flop is noted to be ${}_0u_a=P_0C$, which results in triggering the U*a* flip-flop to a false state at the end of the P_0 time. The schematic block diagram of the V*a* flip-flop together with its triggering equations is shown in Figure 34. The logical triggering equations for the V*a* flip-flop are identical with those for the V*a* flip-flop with V terms substituted for U terms. The physical circuitry for solving the U*a* and V*a* flip-flop equations is shown on Figure 35. Since the understanding of this circuitry follows directly from the equations, together with the previous discussion, it will not be further described.

During pulse time P_0 of the W_{i-1} word period, as shown in Figure 16, the contents of the U*a* and V*a* storage flip-flops are observed to enable a transfer to be made to the Σdy counter **180** at the end of this pulse time. In order to explain how this transfer is effected the operation of the Σdy counter **180** will first be broadly explained.

The Σdy counter **180**, shown in Fig. 38, is comprised of flip-flops B1 through B5, which represent binary stages. The B1 counter represents the 2^0 stage, the B2 flip-flop the 2^1 stage, the B3 flip-flop the 2^2 stage, and the B4 flip-flop the 2^3 stage. The B5 flip-flop corresponds to the sign of the binary count whose magnitude is represented by the B1 through B4 flip-flops.

The scheme employed for indicating the binary content of the Σdy counter **180** is revealed by the table in Figure 36. Note that the coded decimal zero content of the Σdy counter is represented by a 0 in all the stages

including the sign stage B5. As unit *dy* increments are added into the Σdy counter, the flip-flop trigger in a conventional binary manner to indicate the binary count. As previously noted, the computer is designed so that as many as 15 *dy* input units can be received during a word period. Since these *dy* input units may be all positive or all negative in sign, the Σdy counter content has a range from plus 15 to minus 15.

The plus sign of the *dy* count is indicated by a zero content of the B5 flip-flop, and the minus sign is indicated by a one in the B5 flip-flop.

Thus, as shown in Fig. 36, the Σdy counter represents a decimal content (−1) by a one in all of the flip-flop counting stages, B1 through B5. The content (−2) is then represented by triggering the first stage B1 to a zero state, and the content (−3) is represented by triggering the first stage B1 back to its one state and triggering the second stage B2 to a zero state. It can thus be seen that the remaining negative contents of the Σdy counter 180 are obtained by triggering the flip-flops in a similar binary fashion.

The Σdy counter must be quite versatile in its operation since, in addition to counting either up or down through a range of +15 to −15, it must be able to reset to a value indicative of the random input; count up one or two units at a time from any given count; and count down one or two units at a time from any given count.

The resetting of the Σdy counter 180 to the value indicated by the U*a* and V*a* flip-flops will first be described.

Referring to the logical reset equations shown near the bottom of Figure 38, it is noted that the Σdy counter 180 must be capable of being reset during the P_0 time to indicate a (+1), a (0), or a (−1).

As seen in the table of Figure 36, a decimal count of +1 is represented by: a "one" in the B1 flip-flop, and a "zero" in the remaining flip-flops of the Σdy counter. Thus the equation $b_1=U_aV_a'P_0C$ sets the B1 flip-flop true whenever the U_a term is high and the V_a term is low in potential. Note that it has previously been shown that when the U_a term is high in potential a +1 is stored in the U*a* flip-flop.

As for the B2, B3, B4 and B5 flip-flops, whenever the U_a term is high in potential during P_0 time, the false inputs of these flip-flops are triggered to record zeros therein at the end of the P_0 time, as shown, for example, by equation ${}_0b_2=U_aP_0C$ for the B2 flip-flop. Note that the V_a term has been left out of these latter equations since, irrespective of whether term V_a is true or false, if term U_a is true, the B2, as well as the B3, B4 and B5 flip-flops, are rendered false.

Whenever both the U*a* and V*a* flip-flops are in a true state, as, for example, when both a positive and negative indication is simultaneously received by the computer unit from a sensory instrument, it is desired to reset the Σdy counter 180 to a zero state. It is, of course, also desired to reset the Σdy counter 180 to a zero state whenever no input is received, i.e., when both the U*a* and V*a* flip-flops are in a false state.

As shown in the table of Figure 36, the zero state of the Σdy counter 180 is represented by a zero in all the flip-flop stages. Thus whenever terms U_a and V_a are both true, symbolically noted as (V_aU_a), or both false ($V_a'U_a'$), the first flip-flop B1 is triggered to a false state as indicated by equation ${}_0b_1=(U_aV_a+U_a'V_a')P_0C$. As for the remaining flip-flops B2, B3, B4 and B5, it was stated above that whenever term U_a is true, irrespective of term V_a, these flip-flops are rendered false. Hence the equation presented for rendering these flip-flops false when resetting the Σdy counter to +1, are also suitable for setting the Σdy counter into the 0 count when condition U_aV_a is true.

Taking the condition $U_a'V_a'$ of the arbitrary inputs, which requires resetting the B2, B3, B4 and B5 flip-flops to zero, it can be shown that whenever term V_a' is true (i.e. high in potential) these flip-flops are rendered false. Note that this is the case irrespective of the condition of term U_a. Hence equation, such as ${}_0b_2=V_a'P_0C$, as used for the B2 flip-flop, is similarly used for the remaining flip-flops of the Σdy counter to reset it to zero.

In order to reset the Σdy counter to a minus 1, as seen in the table of Figure 36, all the flip-flops must be recording a one, i.e., the positive input of each of the B1 through B5 flip-flops must be triggered. Thus for the B1 flip-flop, as well as all the remaining flip-flops, this is accomplished by an equation such as, for example, $b_1=U_a'V_aP_0C$.

Referring next to Figure 40, a diode net for generating the above described reset equations for transferring the arbitrary inputs into the Σdy counter will be presented.

Since various combinations of the U*a* and V*a* flip-flop terms are required along with the P_0 time pulse, a family tree network is presented for building up these various combinations. The P_0 term is fed into the lower end of the family tree network and applied simultaneously into one input of each of the partial product circuits 274, 275, 276 and 277 which have the terms V_a', U_a, V_a and U_a', respectively, applied to their other input.

Partial products P_0V_a' and P_0U_a, as obtained from product circuits 274 and 275, respectively, are then fed into a two input sum circuit 278 which generates $P_0(U_a+V_a')$. As noted in Figure 38 this combination of terms corresponds to the false triggering conditions required for the B2, B3, B4 and B5 flip-flops in order to set the Σdy counter to (+1) and (0).

The partial products P_0V_a' and P_0U_a' as obtained from product circuits 274 and 277 are then fed into a product circuit 279 whose output $P_0V_a'U_a'$ when logically added to $P_0U_aV_a$, as obtained from product network 280, generates $P_0(U_aV_a+U_a'V_a')$ on output line 281. Note that this combination of terms is required for equation ${}_0b_1$ in Figure 38.

The output P_0V_a from product circuit 276, and output P_0U_a' from product circuit 277, are then logically multiplied in circuit 282 having an output 283. This output generates $P_0(U_a'V_a)$ which corresponds to the true triggering conditions required for the B2 through B5 flip-flops in order to reset the Σdy counter to (−1). This output is further combined in logical sum circuit 284 along with a product $P_0U_aV_a'$, obtained by combining the outputs from partial product circuits 274 and 275. Thus the complex logical term $P_0(U_aV_a'+U_a'V_a)$ is generated as the output of sum circuit 284. This latter combination of terms represents the solutions of the b_1 triggering equation required for resetting the Σdy counter to (+1) or (−1).

The combining of these solutions with other counting equation solutions to control the overall operation of the B1–B5 flip-flops will be explained in the ensuing description.

The manner in which the *dy* inputs are received from the Z1 and Z2 storage lines by the Σdy counter, and the logical equation for sensing this information will next be described.

The dy decoder

The decoding circuitry for the *dy* inputs can best be explained by referring to Figure 5. Here the pulse pattern in a word time is shown together with the information simultaneously picked up in the Z–1 and Z–2 channels.

The following logical equations are the ones which define the operation of the *dy* decoder shown in Figure 15.

$$\alpha=(Z_1D_1D_2'+D_1'Z_2D_2)P_0'P_{26}'$$
$$\beta=(Z_1'D_1D_2'+D_1'Z_2'D_2)P_0'P_{26}'$$
$$\gamma=Z_1D_1Z_2D_2P_0'P_{26}'$$
$$\delta=Z_1'D_1Z_2'D_2P_0'P_{26}'$$

The α equation states that if during a word time, except for the P_0 and P_{26} pulse times, a coincidence of pulses (ones) occurs on either the Z–1 and D–1 or Z–2 and

D–2 channels, but not on a Z channel and both the D–1 and D–2 channels, a signal indication of a positive one input will be fed into the Σdy counter.

The β equation states that if during a word time, except for the P_0 and P_{26} pulse times, a pulse is located on either the D–1 or D–2 channels, but not both, and there is no pulse (a zero) in the corresponding pulse position of the Z–1 and Z–2 channels, then a signal indication of a negative one input will be fed into the Σdy counter.

The γ equation causes a positive two unit input to be fed into the Σdy counter. This results when, during the period stipulated, a pulse is picked up from the D–1 and D–2 channels simultaneously with a pulse on the Z–1 and Z–2 channels.

The final dy decoder output δ feeds a negative two unit input into the Σdy counter. This occurs when, for the duration indicated, a pulse is present on both the D–1 and D–2 channel inputs together with a lack of pulse on both the Z–1 and Z–2 channels.

The equation for the α, β, γ and δ inputs are not generated by logical nets as above presented, but rather their inverses are generated as shown in Figure 41. These inverse equations are obtained by applying well known rules of symbolic logic. The outputs α', β', γ', δ', so obtained and generated are then fed through vacuum tubes (not shown) to obtain the α, β, γ and δ terms required

An example of how each of the α, β, γ, and δ inputs are coded is shown by pulse positions P_7, P_9, P_{12} and P_{17}, respectively, in Figure 5.

Returning back to Figure 38, the counting logical equations associated with the B1 through B5 flip-flops, enabling them to respond to the dy inputs on the α, β, γ and δ lines, will next be described. Note that the triggering input equations to these flip-flops are separated according to the nature of the inputs.

The Σdy counter

Referring to the table in Figure 36, during positive unit counting of the Σdy counter, the B1 flip-flop changes state upon receipt of each α input. This is symbolically noted by equation $b_1=\alpha B_1'C$ and ${}_0b_1=\alpha B_1C$. The remaining flip-flops, B2, B3, B4 and B5 change states on receiving positive unit inputs, according to the binary counting triggering pattern previously explained.

In order to perform negative unit counting, the β input must be of relatively high potential, i.e., true. For negative unit counting, instead of observing the change in states of the B1 through B5 flip-flops by moving up the binary table in Figure 36, the changes in states of these flip-flops are noted on moving down the table. The equations for the B1 flip-flop are similar to those for positive unit counting with β substituted for α. The equations for the remaining flip-flops are written by observing the state of the flip-flop in question and all lower order flip-flops.

For positive two unit counting, the γ term must be true. The B1 flip-flop is not affected by changes of two units. Thus $b_1=0$ and ${}_0b_1=0$. However, the B2 flip-flop changes state each time a γ input is received as indicated by equations $b_2=\gamma B_2'C$ and ${}_0b_2=\gamma B_2C$.

Noting when the B3 flip-flop changes from a 0 to a 1 state on moving up the table in Figure 36 it will be noted that the B2 flip-flop is always true and the B3 flip-flop, of course, is always false. Hence $b_3=\gamma B_2B_3'$. Likewise when the B3 flip-flop changes from a 1 to a 0 state, the logical pattern of conditions is seen to be ${}_0b_3=\gamma B_2B_3$. The B4 and B5 logical triggering equations are similarly obtained.

Negative two unit counting, as characterized by input δ being of a high potential, is expressed by logical equations obtained by noting the triggering pattern of the flip-flops upon moving down the table in Figure 36. These equations are analyzed similarly to those for positive unit counting and will not be further described.

Referring next to Figure 39 the overall logical counting network is shown for performing all the functions of Σdy counter **180**. As an example of how the triggering input circuits are arranged, the B3 flip-flop equations will be described. The positive unit counting equation $b_3=\alpha B_1B_2B_3'C$ is obtained by first generating partial product αB_1 in a left multiplying circuit **286**. This output, after being applied on left sum circuit **287**, is fed into a left partial product circuit **288** where it is multiplied along with term B_2 to generate partial product αB_1B_2 on left line **289**. Left line **289** is then fed into a common sum circuit **290** whose output feeds into a right product network **291** which multiplies in the term B_3'. (It is to be noted that the terms of a partial product are not changed in passing through a sum circuit.) The partial product $\alpha B_1B_2B_3'$ is then fed into a final right sum circuit **292**, and finally to a right clock product circuit **293** which generates the solutions to equation b_3 on its output.

The positive unit counting equation ${}_0b_3=\alpha B_1B_2B_3C$ is obtained by feeding αB_1B_2 as obtained from common sum circuit **290** into a left product network **294** which multiplies in the B_3 term. This result, after passing through left sum circuit **295**, is then multiplied by C in left clock product circuit **296** to obtain the output corresponding to solution ${}_0b_3$.

The negative unit counting equation $b_3=\beta B_1'B_2'B_3'C$ is obtained by cascading the output of right multiplying circuit **297** through right sum circuit **298**, right partial product circuit **299**, common sum circuit **290**, right product circuit **291**, right final sum circuit **292** and, finally, right clock product circuit **293**. Similarly the negative unit counting equation ${}_0b_3=\beta B_1'B_2'B_3'C$ is obtained by multiplying the partial product $\beta B_1'B_2'$ as obtained from right line **300** with terms in the circuits **293**, **295** and **296**.

The remaining equations for the B3 flip-flop are generated by feeding proper alternate terms into the sum circuits already introduced. Thus solutions $b_3=\gamma B_2B_3'C$, and ${}_0b_3=\gamma B_2B_3C$ are obtained by feeding γ into left sum circuit **287**. The resulting partial product γB_2, now generated on left line **289**, is then fed through common sum circuit **290** where it is either cascaded to the left and multiplied by B_3 to generate ${}_0b_3=\gamma B_2B_3C$, or it is cascaded to the right and multiplied by B_3', to generate

$$b_3=\gamma B_2B_3'C$$

Likewise, by feeding the δ term into the right sum circuit **298**, the partial product $\delta B_2'$ is obtained on right line **300** which is fed through common sum circuit **290**, and then cascaded either to the left to obtain

$${}_0b_3=\delta B_2'B_3'C$$

or to the right to obtain $b_3=\delta B_2'B_3C$.

The reset equations for the arbitrary input indications are generated by feeding alternate term $P_0(U_a'V_a)$, as obtained on lead **301** into right final sum circuit **292**. The output of right clock product circuit **293** then generates $b_3=P_0U_a'V_aC$. Similarly the complex term $P_0(U_a+V_a')$ generated on lead **302** is fed into left final sum circuit **295** whose output feeds into left clock product circuit **296** to generate ${}_0b_3=P_0(U_a+V_a')C$.

The circuitry in Figure 39 is an example of a multilevel network which is defined as one that involves logical products driving logical sums which in turn drive logical products, etc. In order to provide a multilevel network such as this, to solve complex logical propositions, it is necessary that forward current be always flowing in the product diodes when a product circuit drives a sum circuit and forward current be always flowing in the sum diodes when a sum circuit drives a product circuit. As will be explained herein, in a network having this characteristic, each change from one type of logical circuit to another is represented by a different level of operation of the diode network.

The arrangement and operation of a multilevel logical circuit such as this will be described by noting the circuitry for generating the equation $b_3=\alpha B_1B_2B_3'C$. As shown in Figure 39, this circuitry represents a seven level network. Each succeeding logical circuit for generating this equation, starting from the right clock product circuit **293**, for example, and working backward, has a smaller resistor value. Thus note how product circuit **293** has a product resistor of 220K ohms; sum circuit **292** has a sum resistor of 130K ohms; product circuit **291** has a product resistor 82K ohms, etc.

In these circuits, the resistor (equal to 220K ohms) in the clock product circuit **293** is first determined by noting the input capacity to the tube, for example, which is driven by this resistor so that the time constant of the rise or fall of the b_3 proposition is consistent with the operating frequencies of the circuit. Having this resistor, it is possible to calculate the current flow from the product circuit **293** when the sum circuit output is at 45 volts. In order to properly operate the sum circuit **292**, which drives the product circuit **293**, it is necessary that the sum resistor of circuit **292** be small enough so that the current flowing through it from the product circuit **293** will not raise the voltage at the output of sum circuit **292** higher than 45 volts. If the current through the sum resistor, for this condition, raises the output voltage to some level below 45 volts, forward current through the other diodes of the sum circuit will always flow, and this current will maintain the output voltage at the 45 volts level. Thus note that the current flow through the product resistor and the sum resistor is the same (disregarding back currents), but the voltage drop to be developed across the sum resistor is less than 45—(—130)=175 volts, while the voltage drop across the product resistor is 265—45=220 volts. As a result, the value of the sum resistor must be less than the value of the product resistor by a ratio of at least 175/220. Taking resistor tolerances into account as well as certain safety factors, this ratio becomes even smaller and circuit **292** is provided with a sum resistor of 130K ohms.

The ratio of the resistors when a sum circuit is driven by a product circuit will next be described.

The condition of interest here is when the product circuit **291** output is high (90 volts) which means current is flowing from circuit **291** into the resistor of sum circuit **292**. The current drawn by the resistor of sum circuit **292** is determined by the voltage 90—(—130)=220 volts. In order to have proper operation of the product circuit **291**, its resistor must be small enough so that current flowing through it will not drop the voltage at its output lower than 90 volts. In this case it is the current, drawn in the sum circuit **292** by a voltage differential of 220 volts, which must not drop the voltage at the output of the product circuit **291** to a point lower than 90 volts. That is, the voltage drop to be created by the resistor of the product circuit **291** must be less than 265—90 volts= 175 volts. Holding this voltage at some level above 90 volts, ensures that forward current will always be flowing in the other input diodes of product circuit **291**, thus maintaining the output voltage at 90 volts. Thus the value of the product resistor must be less than the value of the sum resistor by the ratio of at least 175/220. By taking into account tolerances and safety factors as noted before, this ratio becomes even smaller, resulting in product circuit **291** being provided with a resistor having a value of 82K ohms.

In a similar manner the resistor values of each of the other logical circuits, **290**, **288**, etc., is determined.

It is now evident that the resistor values of succeeding circuit levels must be reduced in value for proper operation of the circuits. As a result the current flow in the circuits tend to increase in a multilevel network and the power requirements rise. It can be shown that the power requirements can be kept down by making the swing voltage of the propositions small and by having the positive and negative return voltages of the diode networks relatively far apart. In other words, the swing of the propositions between 90 volts and 45 volts should be no greater than necessary, and the voltage return of 265 volts for the product resistors should be as high above 90 volts as possible, while the voltage return of —130 volts for the sum resistors should be as low below 45 volts as possible, consistent, of course, with other operating considerations.

It should be noted that making the positive and negative returns of the diode network equally away from the high and low swings of the propositions, respectively, results in a simplified design, because of its symmetry.

One other consideration is given in support of having the positive and negative returns of the diode networks as far apart as possible. Because of the clamping of each of the input voltage sources applied to the diodes, the rise time or fall time of the output of a logical circuit becomes only a part of the rise time inferred by the time constant of the load resistor and shunt capacity. For example, in an unloaded product circuit the charge of the equivalent condenser is toward 265 volts, but its rise is halted at 90 volts because of the clamping effect which limits this rise. This effect, likewise, makes it possible to use higher resistor values, thus reducing the power requirements of the circuits.

The Σdy register

As previously explained in connection with Figures 15 and 16, it is desired to bodily transfer the contents of the Σdy counter **180** into a Σdy register **181** during the P_0 time following the word period during which the *dy* counter received its inputs. Flip-flops B6 through B10 comprise the Σdy register **181** as shown in Figure 42. This transfer operation is accomplished by stepping, in parallel, the contents of the B1, B2, B3, B4 and B5 flip-flops into the B6, B7, B8, B9 and B10 flip-flops, respectively. This operation is characterized by a G_3 term being true. This G_3 term (see Figure 66) corresponds to a signal which is required to render the computing circuits in the machine active. The logical triggering equations for this transfer operaton are shown below their respective flip-flops in Figure 42. The transfer equations for the B6 flip-flop are $b=B_1P_0G_3C$ and ${}_0b_6=B_1'P_0G_3C$. These equations state that if G_3 is true during a P_0 time: the true input equation b_6 triggers the B6 flip-flop to a true state if the B1 flip-flop is in a true state; and the false input equation ${}_0b_6$ triggers the B6 flip-flop to a false state if the B1 flip-flop is false. The transfer equations for the B7, B8, B9 and B10 flip-flops are similarly written.

As shown in Figure 42, in addition to the transfer operation, the B6 through B10 flip-flops also function to step out the Σdy content into the $(y+\Sigma dy)$ adder. In addition, these flip-flops are used for reading out the *y* content of an integrator setup. The stepping function of the B6–B10 flip-flops will be described shortly, after the start flip-flop is introduced. The readout function of the B6–B10 flip-flops occurs when the machine is not computing and will be described later in connection with Figure 81.

Referring to the transfer equations above described, the b_6 equation is generated as shown in Figure 43 by partial product circuit **319** connected into the clock product circuit **320** by means of sum circuit having a common line **319***a*. The output of circuit **320** in this instance represents the b_6 transfer solution. Similarly the ${}_0b_6$ transfer equation is generated by circuit **321** connected to circuit **322** by means of a sum circuit having a common line **321***a*.

Before describing the $(y+\Sigma dy)$ adder which generates the new *y* and number y_s, the scheme used for designating the *y* and the Σdy numbers will be compared by reference to Figures 36 and 37. These numbers have their magnitudes both represented by the same binary notation. The positive numbers use the conventional binary nota-

tion. The negative number notation is based on a system whereby decimal equivalent −1 is denoted by a "one" content in all of the stages, and the stages are changed in a binary fashion to obtain the succeeding negative numbers. However, the sign convention for the y number is just opposite that employed for the Σdy number. A one is recorded in the P_{25} pulse position of a y number to indicate a positive sign, and a zero is recorded therein to indicate a negative sign. On the other hand, a one in the B5 flip-flop of the Σdy number indicates the sign of the Σdy magnitude is negative while a zero in this flip-flop indicates the sign of the Σdy magnitude is positive. As will be brought out in the ensuing description, this sign convention makes it possible to serially add together the y and Σdy numbers, treating the sign as an extra digit of the numbers. This method results in the sign of the sum $(y+\Sigma dy)$ being automatically and correctly provided for.

The block diagram of the S1 flip-flop together with the logical equations controlling its grids will next be shown in Figure 44. The true triggering equation

$$s_1=Y_1S_1'P_0'P_{26}'G_3(C_1+C_2)C$$

states that, as the binary information is picked up from the Y channel of the drum, the first digit "one" picked up, after the P_0 but before the P_{26} pulse position, is used for triggering the start flip-flop S1 into a true state. The computing combination $G_3(C_1+C_2)$ must also be true for this to happen. The significance of this latter complex term will be described later. The false triggering equation for the S1 flip-flop, ${}_0s_1=P_{25}C$, states that at the end of the P_{25} pulse time, the start proposition is always rendered false. These start triggering equations are generated by the network in Figure 45.

The S_1 term thus obtained is used to signal the circuits associated with the Σdy register to step out the Σdy content, stage by stage. This is accomplished by bodily stepping the contents of each of the flip-flop stages B6–B10 (Figure 42) to a lower order stage on each successive clock pulse. Thus the stepping equations for the B6 flip-flop are noted to be $b_6=B_7S_1C$ and ${}_0b_6=B_7'S_1C$. It follows that the stepping equations for each of the remaining Σdy register flip-flops are obtained by substituting the terms of the next higher order flip-flop as shown in Figure 42. Referring to Figure 43, this b_6 stepping equation is generated by partial product circuit 323 connected to the sum circuit having the common line 319*a*. This line is then connected to clock product circuit 320 whose output will in this instance represent the b_6 stepping solution. Similarly stepping equation ${}_0b_6$ is generated by circuit 324, common line 321*a* and circuit 322.

The $y+\Sigma dy$ adder

The circuitry for the $y+\Sigma dy$ adder 103 will next be described in detail by reference to Figures 48, 49 and 52. The binary contents observed during a clock period in the Y1 and B6 flip-flops are added in this $y+\Sigma dy$ adder. In order to keep track of the carries resulting from the addition of a given stage of the numbers, a carry flip-flop K1 is provided as shown in Figure 15. Thus, it can be said that the inputs to to the $y+\Sigma dy$ adder are provided by the Y1, B6, and K1 flip-flops.

The logical equations which define the operation of the $y+\Sigma dy$ adder 183 (Figure 15) are determined by reference to the table in Figure 48. In this table, a column is provided for each of the Y1, B6 and K1 flip-flops. It is noted that all combinations of inputs possible by these three flip-flops can be represented by the binary counts corresponding to decimal 0 through 7 which is recorded in the table. Assuming the inputs during a given clock period can be any one of the horizontal sets shown, the binary sum and carry resulting from each set is denoted in columns 304 and 305 respectively. In column 306, a record is made of the comparison of the contents in each row of the K1 column with the "carry" column 305. When the binary contents of these columns are the same, a 0 is recorded in column 306. A change from 0 to 1 on matching these columns is noted by a 1 in column 306. A similar comparison is made of these columns for a change from 1 to 0, which change is noted in column 307 by a 1.

The table of Figure 48 contains the information required to write the logical equations for the $y+\Sigma dy$ adder circuits and the triggering equations for the K1 flip-flop.

As shown below this table, the sum proposition Y_s is equal to a 1, i.e., is true for four different sets of states of the inputs from flip-flops Y1, B6 and K1. These sets are obtained from the table and logically added to give the sum equation: $Y_s=Y_1B_6K_1+Y_1B_6'K_1'+Y_1'B_6'K_1$.

The contents of columns 306 and 307 indicate the triggering equations for the K1 flip-flop. A single 1 is present in each of these columns. Note in column 306 that when the K1 flip-flop changes from a 0 to a 1; that is, rendered true, the Y1 and B6 flip-flops are true. Likewise as shown in column 307 when the K1 flip-flop changes from a 1 to a 0, that is, made false, the Y1 and B6 flip-flops are false. The logical equations for expressing these K1 flip-flop changes are denoted below the table in Figure 48 as:

$$0\xrightarrow{\Delta}1=Y_1B_6, \text{ and } 1\xrightarrow{\Delta}0=Y_1'B_6'$$

Referring next to Figure 49, the K1 flip-flop is schematically depicted along with its logical triggering equations, $k_1=Y_1B_6S_1C$ and ${}_0k_1=(Y_1'B_6'K_1S_1+P_0)C$. These equations include the terms obtained from the table in Figure 48, together with other terms to be now described. The S_1 term is introduced here in each of the equations as one of the required conditions so as to insure that the $y+\Sigma dy$ counter does not set up the carry in the K1 flip-flop before the y and Σdy digits have been received. Note the K_1 term in the ${}_0k_1$ equation is redundant but has been included because the combination of terms $Y_1'B_6'K_1$ is required for the sum equation Y_s. Thus this combination can be generated once and used for both equations. The logical addition of P_0C to the ${}_0k_1$ triggering equation automatically resets the carry flip-flop K1 to a zero state at the beginning of each word.

In Figure 52, the logical nets are shown for physically generating the logical equations for the $y+\Sigma dy$ adder. This includes the triggering equations for the K1 flip-flop and the equation for the sum proposition Y_s. The k_1 triggering equation is generated in product circuit 307*a*. The triggering equation ${}_0k_1$ and the sum proposition Y_s both include the product $K_1B_6'Y_1'$. Hence this product is generated in circuit 308 and cascaded to the right into sum circuit 309, along with the three other products of the sum equation to generate Y_s. The output of circuit 308 is multiplied by S_1 and the result is impressed on point 309*a* of the network. This point is then multiplied by the clock pulse C before being fed out as the ${}_0k_1$ solution. Point 309*a* also corresponds to alternate term P_0 of the ${}_0k_1$ equation. Thus the reset term of this latter equation is provided for.

An example will now be presented in Figure 59 of how the binary code numbers corresponding to $y=+20$ and $\Sigma dy=-1$ are added in the $y+\Sigma dy$ adder to obtain a binary code number corresponding to $y_s=+19$. Here it is assumed that start pulse is located in the P_{18} position of the Y channel. Thus the first digit of the y number, corresponding to stage 2^0, is in pulse position P_{19}. The following pulse positions through P_{25} contain the remaining y digits. These y digits represent the binary code of +20 in accordance with the scheme employed for the y number shown in Figure 37. It should be noted that since the sign of y is positive here, a 1 is placed in the P_{25} pulse position.

The Σdy count is the binary code corresponding to a −1. As shown in Figure 36, this count is represented by 1's in all the stages including the sign stage. The

first digit of the Σdy count is sensed in the P_{19} pulse position and the following pulse positions through P_{23} contain the remaining Σdy digits. The P_{23} digit is the sign digit which is, prior to the addition, contained in the B10 flip-flop of the Σdy register.

The corresponding digits of the numbers are added in a conventional manner through pulse position P_{23} which contains the Σdy sign digit. This latter digit is then, in effect, carried along and added to all the subsequent stages of the y number including the y sign digit in pulse position P_{25}. This explains the reason for showing a digit one in positions P_{24} and P_{25} of the Σdy number in the example of Figure 59.

As a result of adding the numbers in this manner, the new y number, $y_s=+19$, follows the binary code for the y numbers as set up in Figure 37. Note that the P_{25} position of the y_s number contains a one, indicating this number is positive as desired.

In order to further clarify the manner in which the $y+\Sigma dy$ adder operates, Figure 59 also shows by way of Example 2 how a $y=-40$ when added to a $\Sigma dy=+8$ results in a $y_s=-32$. In this case the binary word, representing -40, follows the convention for negative y numbers. A zero in the P_{25} position shows that the y number is minus. The Σdy number is positive as shown by a zero in its sign stage B10. On summing these binary numbers, the binary number -32 is obtained. In this case, as before, the B10 content of the Σdy number is carried along and added to the remaining digits of the y number, including the y sign digit in the P_{25} position. The result gives a zero digit in the P_{25} position of y_s indicating the binary number representing the sum is negative in sign.

As shown by the system of thought of the present computer presented in Figure 15, the new y number digits generated by the $y+\Sigma dy$ adder are next combined, in the $r+y_s$ adder, with the r number digits picked up from the drum.

Before describing the logical equation and required circuits for performing this addition, the means for picking up the dx input to an integrator setup must be described.

The dx pickup and storage circuit

As previously explained the dx input pulse is obtained either from the Z–1 or the Z–2 channels of the memory drum 106. The P_0 pulse position of the D–X channel determines whether the information in the Z–1 or Z–2 channel will be selected. A pulse in this P_0 position indicates the Z–1 channel should be scanned, while a lack of a pulse in this P_0 position indicates the Z–2 channel should be scanned.

The decoding and storing of the dx input is logically determined by the triggering equations for the Bx flip-flop. As shown below, Figure 55, the Bx flip-flop ends up with either a plus dx unit or a minus dx unit stored therein. The plus dx unit can be obtained from either the Z–1 or Z–2 channel, and the negative dx unit can be likewise obtained from either channel. The logical triggering equations are separated according to the sign of the dx input and the channel from which it is selected.

It should be noted that initially the Bx flip-flop is always in a false state.

The conditions which result in picking up a $+dx$ from the Z–1 channel are that a pulse be in the P_0 position of the D–X channel, and the G_3 term be in a true state, thus $b_x=P_0G_3D_xC$.

Note that this indication of a $+dx$ input from the Z–1 channel is made prior to the period P_1 through P_{25} when the dx pulse is actually picked up. The scheme is such that if there is a pulse in the Z–1 channel corresponding to the position held by the single dx code pickup pulse in the D–X channel, then the state of the Bx flip-flop does not change but remains in its previously acquired state. However, if the Bx flip-flop has been triggered into a true state as indicated above, because of a pulse in the P_0 position of the D–X channel, and if there is no pulse in the Z–1 channel corresponding to the position of the dx code pulse, then the Bx flip-flop is retriggered to a false state as indicated by equation ${}_0b_x=D_xZ_1'B_xG_3C$.

Assuming there is no pulse in the P_0 position of the D–X channel, the Bx flip-flop is then triggered to a true state if, during P_1 through P_{25} pulse times, there is a pulse on the Z–2 channel corresponding to the position in which the dx pickup pulse is located in the D–X channel. Another condition which must be satisfied is that the Bx flip-flop not be already set into a true state. Thus the logical equation for picking up a $+dz$ from the Z–2 channel is $b_x=Z_2B_x'P_0'P_{26}'G_3D_xC$.

Assuming there is no pulse in the Z–2 channel corresponding to the pickup pulse located in the D–X channel, then the Bx flip-flop is not rendered true by the above equation, but remains in its false state. At P_{26} pulse time the Bx flip-flop is always rendered false, that is, the Bx flip-flop is reset to a false state at P_{26} pulse time so that at the beginning of each word the Bx flip-flop can be considered to be initially in the false state.

It should be understood that whenever a statement is made that a flip-flop changes state at a given pulse time, what is meant is that the flip-flop changes state at the end of that pulse time. That is, it is the conditions which are observed during a pulse time which determines if a triggering input equation is satisfied, but it is the differential pulse derived from the trailing edge of a clock pulse (coincident with the end of this pulse period) which actually triggers the circuits.

As shown in Figure 56, the b_x equations for the Bx flip-flop are physically generated so that their solutions appear on line 310 feeding into sum circuit 311. As will be described later, the Bx flip-flop also serves another function in the machine. The other input 312 to sum circuit 311 represents the b_x solution of this alternate function, which is in the fill reference circuit. In the lower network in Figure 56 is shown the solutions to the ${}_0b_x$ equations. Here sum circuit 313 receives not only the dx input equations for the function of the Bx flip-flop above described, but also the equations for the fill reference circuit, which function is to be described later.

Referring to Figure 57, the Ax flip-flop is shown. This flip-flop is used as a dx register for the dx input previously recorded in the Bx flip-flop. As shown in connection with Figures 15 and 16 this transferring of the dx input out of the Bx flip-flop into the Ax flip-flop is necessary in order to free the Bx flip-flop so that it can, during the following word time, pick up the next dx input, while the previous dx input is used for controlling the operation of the $r+y_s$ adder. The triggering equations for the Ax flip-flop state that the Ax flip-flop follows the condition of the Bx flip-flop as observed during the P_{26} pulse time, providing the G_3 term is true. It should be noted the Ax flip-flop also functions in the fill reference circuits, as shown by the other set of triggering equations in Figure 57.

In Figure 58 the triggering equations are shown for the Ax flip-flop when functioning both as the dx register and a fill reference flip-flop. These a_x and ${}_0a_x$ equations for the dx register function are generated on inputs 314a and 315a to sum circuits 314 and 315 respectively. When the Ax flip-flop functions as part of the fill reference circuit, the alternate inputs to the sum circuits shown are effective.

The r+y$_s$ adder

The $r+y_s$ adder circuits will next be described by reference to Figures 50, 51, 52, and 54. As with the $y+\Sigma dy$ adder the binary table in Figure 50 aids in determining the logical equations for summing the states of Ya, R1 and K2 inputs feeding into the $r+y_s$ adder. The Y_a term is equal to $Y_sA_x+Y_s'A_x'$. That is to say,

anytime the Y_s term is true and the *dx* input is positive, as represented by term A_x being true; or, anytime the logical negative of these terms, namely, Y_s' and A_x' is true, then Y_a is true. This is nothing more than a statement that if the signs of the y_s number and the *dx* input are the same then the sign of their algebraic product is positive. On the other hand, anytime terms Y_s and A_x are not both true, then the Y_a' term is true, i.e., the sign of $(y_s dx)$ is negative. All the possible combinations of the *Ya*, R1 and K2 inputs are represented by a binary table equivalent to decimal 0 through decimal 7. Assuming the set in each row is received, the sum and carry resulting therefrom is shown in columns **316** and **317**, respectively. The inverse of the "sum" proposition in column 316 is then presented in column **318**. The digits in column K2 are then matched with the digits in carry column **317** in order to determine when the K2 flip-flop must change states for the various sets of input condition. As shown in columns **328** and **329**, a single change from 0 to 1 is noted in column **328** and a single change from 1 to 0 is noted in column **329**.

Below Table 50, the triggering of the K2 flip-flop from a 0 to a 1 state is defined by the logical product Y_aR_1 and, the triggering of this flip-flop from a 1 to a 0 state is defined by logical product $Y_a'R_1'$. The logical equations for the sum logical proposition R_s is also shown below Table 50. This sum proposition is true, as shown in column 316, for the combination of inputs represented as: $R_s=(Y_a'R_1'K_2+Y_a'R_1K_2'+Y_aR_1'K_2'+Y_aR_1K_2)$. to obtain its logical inverse:

$$R_s=(R_1K_2+R_1'K_2')Y_a'+(R_1'K_2+R_1K_2')Y_a$$

Note that this latter equation can also be obtained directly from the table, by reference to column **318**.

Referring to Figure 51, the K2 flip-flop is presented along with the triggering equations for its grids. Note that the term S_1 is added to the terms in the equations as obtained from the table in Figure 50. This S_1 term ensures that the adder does not commence to operate until the digits of the *r* and y_s numbers are received. When the *dx* input is negative as represented by term A_x' being true, it is desired to subtract the y_s number from the *r* number. This is performed in the computer by adding the complement of the y_s number digits to the *r* number digits. This is accomplished by reversing the 1's and 0's of the y_s number and adding a "one" to the first stage of the *y* number as shown in Figure 15. Thus, whenever the A_x' term is true, the K2 flip-flop is triggered into a true state at P_0 time. This "one" is then fed into the computer along with the y_s and *r* digits when the $r+y_s$ adder starts operating at some later pulse time of the word.

The ${}_0k_2$ and k_2 triggering equations are generated by the circuit in Figure 53; and, the circuit for physically generating the sum equation R_s' associated with the $r+y_s$ adder is shown in Figure 54.

It should be noted in the equation below the circuit in Figure 54 that additional terms have been added to the R_s' equation as obtained from the table in Figure 50. The P_{26} term has been added to Y_a' term in the first main solution of the equation, and the P_{26}' term has been multiplied into the last main solution of the equation. Referring to the word structure in Figure 4, it is noted that the P_{26} position of the Y channel is reserved for a digit of the P_{26} code. Since this is not part of the *y* number, it is desired to inhibit the information in this pulse position from entering the $r+y_s$ adder. The P_{26} term introduced in the latter solution of the R_s' equation ensures that it is not effective at P_{26} pulse time or a word period. The R_s' output will be false unless the first solution is true.

Examining the first solution of the R_s' equation, at P_{26} time, if K_1 and R_1 are both true, or both false, the R_s' output is true and accordingly the R_s term will be false. Note that here the Y_a' term is redundant during the P_{26} pulse time since, irrespective of the Y_a' term, the sum $(Ya'+P_{26})$ is always true during this time.

Examples of the operation of the $r+y_s$ adder and the associated circuits will next be shown in Figures 60, 61, and 62. In Figure 60, it is assumed that the *r* number as obtained from the drum is equivalent to decimal 0, and the y_s number, as generated in the $y+\Sigma dy$ adder, is equal to a decimal +7. The *dx* input for this operation is assumed to be equal to $+dx$. Thus the y_s number digits are fed into the $r+y_s$ adder without being complemetend. The binary digits representing these numbers are shown in the example. The start pulse is assumed to be in pulse position P_{21}. The following binary digits, received during the P_{22} through P_{26} time, are added stage by stage in an ordinary manner. The sum represents the new *r* number r_s, which is recorded back onto the drum. Any carry resulting from the addition of the P_{25} pulse time represents a *dz* output. Note that the *y* sign digit **326** in pulse position P_{25} is considered as an extra digit of the *y* number and added to the corresponding binary digit in the P_{25} pulse position of the *r* number. In the particular example shown, for the addition during the first revolution of the drum, no carry results from the P_{25} pulse time addition. The P_{26} position of the *r* number represents a reversing pulse, and since this is zero, the output is not changed. Thus a binary digit "zero" (notated as **327** on the drawing) is fed out at P_{26} pulse time as a result of advancing this particular digital integrator setup one step. As will be more clearly explained later, a "zero" *dz* output as a result of an incremental stepping of an integrator setup is considered as a $-dz$ output. What this amounts to, is that as a result of processing each integrator setup, either a $+dz$ or a $-dz$ output is generated. In order to find the value of a function (Z) generated as a result of a particular integration process, these plus and minus *dz* outputs must be added up.

During the next revolution of the memory, for the example shown in Figure 60, the *r* number obtained as a result of the previous calculation is fed into the $r+y_s$ adder along the new *y* number y_s, as obtained from the $y+\Sigma dy$ adder. Assuming this y_s number has not changed, the result of the addition during the second revolution of the memory is shown in Figure 60. Note that as a result of this computation a carry "one" is generated as a result of adding the P_{25} pulse time digits. This carry is, in effect added to the P_{26} pulse time digit in the *r* number. Actually, the operation during the P_{26} pulse time is not part of the addition process but rather a means for reversing the *dz* output. In this case, the P_{26} position of the *r* number contains a zero and so it has no effect when added to the carry "one" in the $r+y_s$ adder. Thus a $+dz$ output designated **328** is generated as a result of this incremental advance of the integrator setup.

During the third revolution of the memory, the *r* number is again picked up and added to the *y* number which is assumed to be the same. Here again, a carry "one" is generated as a result of adding the binary digits during the P_{25} pulse time. Consequently a $+dz$ output **329** is again fed out at P_{26} time.

It should be noted that the binary code information in the P_{26} pulse position of the *r* number is fixed throughout a problem. This binary code results in a means for reversing the *dz* outputs, if so desired. It is recorded into an integrator setup, by the coder, along with other initial conditions of a problem.

In order to further show the nature of the *dz* outputs generated as an overflow from the $r+y_s$ adder, an example will next be shown of how the function Z, generated as a result of an integration process, can be effectively made to remain constant.

Assuming the *y* number in an integrator setup is initially equal to zero and no *dy* inputs are received, it is conceivable that the function Z generated should remain

constant. In Figure 61, during the first revolution of the memory, the y_s number and r number are both assumed to be equal to decimal 0. The dz carry output generated during this revolution is equal to binary zero and thus corresponds to a $-dz$ increment. During the second revolution of the memory, as shown, the result of the addition generates a carry "one" which is fed out as a $+dz$ increment. During the third revolution of the memory, the binary digit carry out is a "zero," which represents, again, a $-dz$ increment. The fourth revolution generates a $+dz$ increment. On adding up the dz increments shown, the effective change in the dz outputs is zero. Thus it is now made evident that the function Z, which is the algebraic sum of the dz units generated, actually oscillates by one dz unit when its change is zero.

Note that the example shown in Figure 61 represents the operation of the $r+y_s$ adder whether the dx input during the computation is positive or negative. Referring back to Figure 16, when dx is negative the y number binary digits are reversed, and a digit "one" is added to the first stage. As shown in the lower right corner of Figure 61, when the binary equivalent to a decimal number 0 is complemented, the resulting binary number does not change.

In Figure 62, an illustration is presented which clearly brings out the operating characteristics of the $r+y_s$ adder when the dx input is minus in sign and the P_{26} position of the r number has a reversing pulse located therein.

In this case the binary y_s number is equivalent to a decimal $+4$. However, as dictated by the dx input being negative, the binary digit representation of the dy number is reversed and a binary one is added resulting in actually feeding a $y_s=-4$ into the adder.

Referring to the addition during the first revolution of the memory, a carry "zero" is generated as a result of the P_{25} pulse time addition. However, when this carry is added to the reversing "one" digit in the P_{26} position of the r number, a one, or $+dz$ output is fed out of the computer. As seen in the illustration, a $+dz$ output is fed out again during the second and third revolutions, and a $-dz$ output is fed out during the fourth revolution resulting in a total effective change of $+2$ dz units for the four incremental advances of the digital integrator setup shown.

The dz outputs generated during the P_{26} pulse time of each word (integrator setup) are fed into the Z–1 and Z–2 general storage lines where they are made available to be picked up as inputs into other words (integrator setups).

Z general storage lines

The mode of operation of the Z–1 and Z–2 general storage lines or delay lines will next be broadly described. As previously noted, it is necessary for this operation to be such that all 50 dz outputs can be scanned by the dx and dy pickup codes located in the 25 pulse positions constituting a word time. Broadly, this is accomplished by a novel precession arrangement for the information in the Z–1 and Z–2 lines. First it is to be understood that all the dz outputs are fed into both the Z–1 and Z–2 delay lines. Each of these lines then functions to automatically divide the dz information therein into two groups. Defining each output by the numeric subscript of the word period W_0–W_{49} when it is generated one of these groups includes the outputs 0, 2, 4, 6 . . ., i.e. the even numbered dz outputs; and, the other group includes the outputs 1, 3, 5, 7 . . ., i.e., the odd numbered dz outputs. Thus, delay line Z–1 has an even and odd grouping of the dz outputs, and the delay line Z–2 has an even and odd grouping of these same dz outputs. Because of the precession arrangement of the information in the Z–1 and Z–2 channels, however, at any given time, the even grouping of dz outputs in the Z–1 delay line is temporally positioned opposite the odd grouping of dz outputs in the Z–2 delay line, and the odd grouping of the dz outputs in the Z–1 line are temporally positioned opposite the even grouping in the Z–2 line. It is now evident that because of this arrangement of the dz information, all the dz outputs can be picked up during the $P_{1/25}$ pulse times of a word since the dz outputs not in the Z–1 line during the pickup code time are necessarily in the Z–2 line.

In Figures 63 and 64, a detailed chart is presented for illustrating the operation of the Z–1 and Z–2 general storage lines.

Assuming the computer unit is just beginning to compute, the position of the dz outputs in the Z–1 and Z–2 lines during the first revolution of the drum memory will be described, then enough of the precession operation of each of the delay lines during the second revolution of the memory will be presented to show that the operation is cyclical.

Referring first to the Z–1 delay line in Figure 63, each row of the chart shows the positions of the dz outputs in the delay line during the pulse time of the computer defined in the left column. As previously shown in Figure 13, the Z–1 delay line is comprised of the Z–1 channel of the drum and the Z–1 flip-flop. The Z–1 delay line is known as the short precession line in that its length, during computation, is one pulse time short of two word periods.

Referring to the $P_{26}W_0$ pulse time of the computer, during this time the dz output of the word W_0, or integrator setup I_0, is generated, i.e., set up by the logical circuits and recorded onto the Z–1 channel of the magnetic drum. As shown and described in connection with Figure 13, this recording is made approximately 52½ clock pulse periods behind the position of the Z–1 pickup head 117. The position of head 117 is indicated near the right end of the chart, the output end of the Z–1 delay line.

It should be noted that the computer unit 107 observes the Z1 flip-flop during computation, which is the operation presently being described. The computer unit 107 may observe the contents of the Z1*a* flip-flop, at which time the information in the Z–1 delay line recirculates without being shifted or precessed. The latter operation of the Z–1 delay line is provided for when the computer is idling.

The dz output at $P_{26}W_0$ time of the computer is denoted in the first pulse position of the first row of Figure 63 by the designation 0_1. The subscript associated with output, 0_1 corresponds to the revolution of the memory. One clock pulse time later, as defined by P_0W_1 in the left hand column, this dz output 0_1 is shown to have advanced one additional clock pulse period toward the position of the pickup head 117. The next pulse time as defined by P_1W_1, advances the dz output 0_1 still another clock pulse period.

At $P_{26}W_1$ pulse time, the dz output 1_1 is generated and recorded into the Z–1 channel of the drum. Since this occurs 28 clocks pulse periods after the 0_1 output was recorded in the Z–1 delay line, the 0_1 output has by now advanced to the 28th pulse delay period. The positions of these 0_1 and 1_1 dz outputs are next shown for the $P_{25}W_2$ pulse time. Note that the 0_1 output by now has been picked up by head 117 and is recorded in the Z1 flip-flop. The operation of the Z–1 delay line during computation is such that the Z1 flip-flop contents is sensed by the computer unit 107. During the same clock period that this Z1 flip-flop contents is sensed by the computer unit, an output is recorded into the input end of the Z–1 channel. This recorded output is identical to the Z1 flip-flop content during pulse times P_0 through P_{25}, changing only for the P_{26} pulse time of each word when a new dz output is generated.

Thus during $P_{25}W_2$ time, the contents of the Z1 flip-flop

which is the *dz* output 0_1, is also, as shown, recorded in the first pulse delay position of the Z–1 line. The *dz* output 1_1 is located in the 27th pulse delay position during this time.

One pulse time later, $P_{26}W_2$, the *dz* output 2_1 is generated and recorded into the Z–1 line following the *dz* output 0_1 which has advanced to the second pulse delay position .

Note that one word time later, namely $P_{26}W_3$, the *dz* outputs 2_1 and 0_1 have moved to the 28th and 29th pulse delay positions respectively, while the *dz* output 3_1 is located in the 1st pulse delay position following the *dz* output 1_1 which is in the 2nd pulse delay position.

By word time $P_{26}W_4$, the *dz* outputs 0_1 and 2_1 have advanced to the end of the Z–1 delay line, through the computer unit, and back onto the front end of the Z–1 line so that *dz* output 4_1, generated during this time, now follows directly behind output 2_1. The *dz* output 3_1 and 1_1 are now in the 28th and 29th pulse delay positions.

It is now evident how the even and odd *dz* outputs are caused to be arranged into separate groups in the Z–1 line. As shown at $P_{26}W_{48}$ and $P_{26}W_{49}$ pulse time, each of these groups follows a definite pattern in that, in each group, successive *dz* outputs are for alternately spaced words on the drum memory. Note that the successive *dz* outputs are arranged in an orderly manner, such that each *dz* output is followed by the *dz* output generated two word periods later.

Another operating characteristic of the Z–1 delay line brought out by observing the $P_{26}W_{48}$ and $P_{26}W_{49}$ pulse times, is that at time $P_{26}W_{48}$ the even group of *dz* outputs is located in the first half of the Z–1 delay line and the odd group is located in the last half of the Z–1 line. On the other hand, at $P_{26}W_{49}$ pulse time, one word period later, the even group of *dz* outputs is now located in the last half of the Z–1 delay line with the odd group in the first half.

Whenever the P_{26} pulse time of a word is presented in Figure 63, the information shown in the chart, in the 27 pulse positions of the Z–1 delay line following the Z1 flip-flop, corresponds to the information to be sensed by the computer unit during the following word period. Thus the data in the 52nd pulse delay position of the delay line corresponds to that sensed by the computer unit at P_0 time of the following word, and the information in each pulse delay position thereafter corresponds to succeeding pulse times of this word. The positions in the Z–1 delay line corresponding to the P_0 through P_{26} pulse times of the word following the P_{26} pulse times shown are noted by the row **325** of P designations placed on the lower right half of the chart. As previously pointed out in describing the structure of a word in Figure 5, the *dy* and *dx* codes for picking up the *dz* information from the Z lines can only be located in the P_1 through P_{25} pulse positions, thus these pulse positions **325** in the Z–1 delay line contain the effective *dz* outputs.

In the lower half of the chart in Figure 63, the *dz* outputs generated during the second revolution of the memory are presented.

At $P_{26}W_0$ time, the *dz* output 0_2 (the subscript 2 corresponds to the second revolution) is recorded so as to follow output 48_1 which was the last even *dz* output generated during the first revolution. At $P_{26}W_1$, the *dz* output 1_2 is shown to follow the *dz* output 49_1. It should be noted that at $P_{26}W_1$ time of the second revolution, the *dz* output 0_2 is in the same position in which the *dz* output 0_1 was located at $P_{26}W_1$ time during the first revolution. Similarly, it is to be noted that for a given pulse time, corresponding *dz* outputs are located in corresponding positions of the delay line for each revolution. The operation of the Z–1 delay line is thus shown to repeat in a cyclical pattern.

Since at every P_{26} pulse time, a new *dz* output is recorded into the Z–1 channel from the computer unit, the delay line soon fills up and some of the information therein gets replaced. As shown at pulse time $P_{26}W_3$ of the 2nd revolution of the memory, the *dz* output 0_1 is in the Z1 flip-flop. However, as shown in Figure 63, during the P_{26} pulse time of each word, the information in the Z1 flip-flop is, so to say, ignored, by the computer unit. Thus, this *dz* output, 0_1, having already served its useful purpose, is now discarded, so as to make room for the *dz* output 3_2. Similarly at $P_{26}W_4$ time, the *dz* output 1_1 is discarded and replaced by *dz* output 4_2.

The operation of the Z–2 general storage line will next be described. As shown in Figure 14, this line is known as the long precession line because it is 55 pulse periods in length, i.e., one clock pulse period longer than two word periods. The computer unit observes the contents of the Z2 flip-flop for this operation.

Referring to the chart in Figure 64, at $P_{26}W_0$, the same *dz* output 0_1 referred to in connection with the Z-1 line is also recorded into the Z–2 channel of the drum. This 0_1 output advances one pulse delay position each clock period, as described in connection with the Z–1 delay line, but notice here that since this delay line is 55 clock pulse periods in length, two clock pulse periods longer than the Z–1 line, the 0_1 output is not routed through the computer unit back onto the drum until P_0W_3 time. Note, by referring back to Figure 63, that this is two clock pulse periods after $P_{25}W_2$ which was when this same *dz* output 0_1 was recorded back onto the Z–1 line.

What this amounts to is that the *dz* output 0_1 is now made to follow the 2_1 output in the Z–2 delay line, as shown at P_0W_3; whereas in the Z–1 delay line, the 0_1 output preceded the 2_1 output.

In a similar manner, as noted at $P_{26}W_4$ time, the *dz* output 4_1 is recorded onto the Z–2 channel, just before the 2_1 and 0_1 outputs, shown at the end of the delay line, are to be routed via the computer unit **107** back onto the Z–2 channel.

In the Z–2 delay line (as shown also for the Z–1 delay line) the *dz* outputs 0_2, 1_2, 3_2 generated during the second revolution of the memory appear in identical positions of the delay line in which the 0_1, 1_1, 3_1 *dz* outputs were located in corresponding pulse times of the first revolution. Thus, the operation of the Z–2 line is shown to be of a cyclical nature.

It should be noted, by comparing the charts of Figures 63 and 64, that at $P_{26}W_{48}$ time the even group of *dz* outputs are effective in the Z–2 delay line, while the odd group of *dz* outputs are effective in the Z–1 delay line. During $P_{26}W_{49}$ the following word time, the reverse groups are effective in the two delay lines.

As noted with the Z–1 delay line, the information in the Z–2 line is recirculated through the computer unit **107**, except at P_{26} time when the information in the Z1 flip-flop is lost since the computer unit replaces this information by the new *dz* output generated at this time.

Having described the Z–1 and Z2 delay lines, it will next be shown how to determine the positions in which the *dx* and *dy* pickup code are to be recorded in the D–X, D–1 and D–2 channels of a word so as to pick up the desired *dz* outputs from the general storage lines.

Channels D–1 and D–2 hold the *dy* pickup codes of the Z–1 and Z–2 precession lines, respectively. As noted all integrator setups feed their outputs to both the Z–1 and Z–2 lines; however, the *dy* code pulses for a particular integrator setup will only pick up, from the Z–1 line, the outputs of the integrator setups which are an odd number of integrator setups behind the one under consideration. On the other hand, the *dy* code pulses for a particular integrator setup will pick up from the Z–2 line, the outputs of integrator setups which are an even number of integrator setups behind the one under consideration. Stated another way, *dy* code pulses for picking up integrator setup outputs oddly spaced from the integrator setup being considered should be placed in the D–1 channel, while *dy* code pulses for picking up integrator setup

outputs evenly spaced from the integrator setup being considered should be placed in the D–2 channel.

The general formulas for the pulse position of the D–1 and D–2 channels that will pick up a given integrator setup output from the Z–1 and Z–2 delay lines, respectively, are given below.

For channel D–1, a *dy* pulse in word W_n (which will actually be used for integrator setup I_{n+1}) will pick up at P_1 pulse time from the Z–1 delay line, the output of integrator setup I as determined by equation:

(1) $$I=[n-2(26-i)+1]$$

For channel D–2, a *dy* pulse in word W_n will pick up at P_1 pulse time from the Z–2 delay line, the output of integrator setup I as determined by equation:

(2) $$I=[n-(2i+4)]$$

Since an integrator setup *dx* pickup is limited to one input, only one address line, the D–X channel, is used to pick up the *dx* information from the two lines Z–1 and Z–2, by means of a selection code in its P_0 position. A "one" in the P_0 of channel D–X will cause the following single pickup code pulse to inspect the Z–1 delay line while a zero therein will cause inspection of the Z–2 delay line. The *dx* code pulse position in the D–X channel is thus determined by either the equation for the Z–1 delay line, or the Z–2 delay line given above.

As an example of the use of these equations, consider coding integrator setup I_2 such that it picks up its own *dz* output as a *dy* input, and picks up the *dz* output of integrator setup I_{49} as a *dx* input.

The code position in channel D–2 by which integrator setup I_2 picks up its own *dz* output as a *dy* input, is obtained from the pickup equation for the Z–2 delay line. Thus by substitution in Equation 2 above: $2=2-(2i+4)$.

A solution here for *i* gives a negative answer; but since there are 50 integrators operating in a continuous ring, it is possible to use *n* modulo 50 in the equation, i.e., $n+50=52$ is substituted for *n*. Now then:

$$2=52-(2i+4).$$

Solving this equation gives a solution $i=23$. Thus channel D–2 has a *dy* pickup code placed in its P_{23} pulse position. Note that if a solution had been sought for picking up I_2 from delay line Z–1, an integral solution for *i* would not have been obtained.

To pick up the output of integrator setup I_{49}, as the *dx* input for integrator setup I_2, the pickup equation for the Z–1 delay line is employed. Thus by substitution in Equation 1 above: $49=52-2(26-i)+1$. As before, *n* modulo 50 is used, i.e., $n+50$ is substituted for *n*.

Solving this equation gives a solution $i=25$. It should be remembered here that if the D–X channel is to pick up information from the D–1 channel, an additional pulse should be placed in the P_0 position of channel D–X. Thus the code employed for picking up the output of integrator setup I_{49}, as the *dx* input for integrator setup I_2, consists of a digit "one" in positions P_0 and P_{25} of the D–X channel.

In order to facilitate determination of the positioning of the *dx* and *dy* code pulses, a slide rule has been included with the machine which eliminates computation of the code positions by the above equations. As shown in Figure 65, this slide rule consists of an outer disc **330** and an inner disc **331**, each having its circumference divided into 50 equal divisions. Each of the divisions of the outer disc **330** is marked from 0 to 49, respectively, in a clockwise direction. Each of these divisions corresponds to an integrator setup of the computer. The inner disc **331** which may be rotated relative to the outer disc **330** about a common center **332**, has an arrow reference **333** which is used to point to the integrator setup being filled as indicated by the appropriate division on the outer disc **330**.

The inner disc **331** has its circumference divided into an outer and inner set of 50 equal arcuate divisions. Alternate divisions of the outer set are each indicated as pulse positions P_1 through P_{25} arranged in a clockwise direction. These positions correspond to the effective pulse positions in the Z–1 delay line. The inside set of divisions of the inner disc are likewise indicated as pulse positions P_1 through P_{25}, but in this case, the P_1 position in the inside set is located five divisions in a counterclockwise direction from the P_1 position in the outer set. The successive P positions P_2, P_3, etc. are then marked on alternate divisions of this inside set in a counterclockwise order. These pulse positions correspond to the effective pulse positions of the Z–2 delay line.

By setting the arrow reference **333** on the integrator setup (not the word), to be filled, the inner disc **331** shows the pulse positions in which the integrator setup outputs will be found in the Z–1 and Z–2 delay lines. The coder thus determines the position in which to put the code pulses in the D–1, D–2, and D–X code channels.

Consider the example of coding, discussed above. The integrator setup I_2 to be filled is represented by division **2** on the outer disc **330**. This division is set opposite the arrow reference **333** of the inner disc **331**, as shown. To pick up integrator setup I_2 as a *dy* input, that is, in order for I_2 to pick up its own *dz* output the coder looks opposite **2** on the outer disc **330** and finds P_{23} of delay line Z–2. In order for I_2 to pick up the output of integrator setup I_{49}, division **49** on outer disc **330** is noted to be opposite position P_{25} of the Z–1 delay line. The determination of the code positions, as obtained from the slide rule, thus checks with the code positions previously calculated by Equations 1 and 2.

Manual control circuits

Before describing the logical equations for the record propositions which deposit information during the computing operation from computer unit **107** onto each of the channels of the memory, the filling operation of the computer, and the circuits required therefor, will be presented. This approach is used because the record propositions necessarily include the logical equations for both the computing and filling operations.

Referring first to Figure 90, a schematic diagram of the main function switch **340** of the computer is shown. This switch has a switch arm **341** which can be manually positioned by the operator onto any one of the seven switch contacts shown. Five of these contacts: FR, FY, FD1, FD2, and FDX set up the computer for performing filling operations; and the two remaining contacts C1 and C2 set up the computer for performing computing operations.

Each of the contacts of the main function switch **340** has a circuit associated therewith, such as the circuit shown connected to the contact FY (Figure 90). This circuit is comprised of a solenoid **342** which is adapted to control the setting of a movable switch bar **343**. The upper terminal of switch bar **343** is designated F_Y' and the lower terminal, which is insulated from the upper, is designated F_Y.

Whenever contact FY of the main function switch is open, as in the example shown, switch bar **343** connects its upper terminal F_Y' to the 90 volt line **263** and its lower terminal F_Y to the 45 volt line **264**. When switch arm **340** is positioned on contact FY, the solenoid **342** is energized and repositions switch bar **343** to connect upper terminal F_Y' to the 45 volt line **264** and lower terminal F_Y to the 90 volt line **263**. Thus when the FY switch is open, its output term F_Y' is of a high potential, and when FY switch is closed, its output term F_Y is of a high potential. Each of the other contacts of the main function switch has a similar circuitry for providing outputs characteristic of the state of the switch. It is now evident that only one of the function switches can be in a true state at a time.

In order to solve a problem on the computer, the first operation which must be performed is to place the initial information in the Y, R, D–1, D–2 and D–X channels of each of the integrator setups to be used on the memory. The filling of each of these word channels is done independently of the other and is characterized by the operator moving the main function switch **340** to the appropriate contact.

After filling the computer memory, to select the computation process the operator moves the main function switch either to contact C1 or contact C2. Contact C1 limits the computation to one turn of the memory, while contact C2 causes the computer to continue operating for a period equivalent to a preselected fixed increment of the independent variable X. This latter computing routine is usually characterized by a large number of turns of the memory.

Before describing the filling operation of the machine, reference will be made to three push button switches **345**, **346** and **347** schematically shown in Figure 88. Switches **345** and **346** are known as the "0" and "1" fill digit switches inasmuch as they are used for selecting the digit 0 or 1 to be recorded into the machine. Switch **347** is known as the ϕ switch and is used for placing a reference pulse into the machine just prior to filling a word channel. This fill reference routine, which is to be distinguished from the filling operation routine, will be explained shortly.

Referring to Figure 88, each of switches there shown has two output lines, such as ⓪ and ⓪′ associated with the "0" button switch **345**. These button switches are spring loaded so that they are normally in the up position, in which, as shown for the "0" button switch, the output is connected by means of first movable arm **348** to the 45 volt line **264**, and the output ⓪′ is connected by means of second movable arm **349** to the 90 volt line **263**. Whenever the "0" button is depressed, the arms **348** and **348** reverse the voltage on outputs ⓪ and ⓪′, thus connecting output ⓪ to the 90 volt line **263** and output ⓪′ to the 45 volt line **264**.

Thus it is noted that the output ⓪′ is normally of a relatively high voltage, but that when the "0" button is down the relatively high voltage is evidenced on the true state output ⓪.

The "1" button and the ϕ button circuits are identical to the "0" button circuit in that they generate on their outputs, terms, indicative of the state of the respective switch.

The 0 and 1 buttons have their positive state outputs ⓪ and ①, respectively, connected to the grid inputs of the fill digit flip-flop X1 schematically shown in Figure 85. The logical equations for the grids, and the diode networks for generating them, shown below the X1 flip-flop, indicate that when either the "1" or "0" button is tapped, the X1 flip-flop stores the selected binary digit. For example, the output X_1 is of a high potential when the digit **1** is stored, and output X_1' is high when the digit 0 is stored.

In addition to using the "0" and "1" buttons for storing a desired digit in the X1 flip-flop, these buttons are used to initiate the "go" flip-flop series.

This sharing of functions of the 0 and 1 buttons is possible because the X1 flip-flop output terms are only effective when the main "go" flip-flop G3 is in a true state. Hence initiation of the "go" flip-flops is required whenever it is desired to fill a digit into the computer.

Referring next to Figure 66, the "go" flip-flops, together with the logical triggering equations for their grids, are schematically shown. These "go" flip-flops are used for controlling the precise time when the computer unit circuits are to be made effective during the filling and computing operations. The operator may initiate the "go" flip-flops by depressing the 0 or 1 buttons at any time, but the G3 term which controls the starting and stopping of these operations, can only be made true or false at the beginning of the P_0W_0 pulse period.

Three flip-flops designated G1, G2 and G3 are required to perform this timing function. The G3 flip-flop is the main "go" flip-flop since its output G_3 is the controlling term desired.

In order to describe the operation of these "go" flip-flops, in addition to the logical equations in Figure 66, the waveforms in Figure 67 will be referred to.

Initially all the G flip-flops are in a false state as noted by waveforms G_1, G_2 and G_3 being of a low voltage. Whenever the 0 or 1 button (Figure 88) is manually closed by the operator, the waveform (⓪+①) changes from a low to a high voltage. On the next $P_{26}W_{49}$ pulse, the G2 flip-flop is triggered to a true state; that is, the G2 term is now represented by a high voltage as shown at point **336** of Figure 67. This triggering of the G2 flip-flop to a true state is symbolically expressed by equation,

$$g_2=(⓪X_1'+①X_1)G_1'G_3'P_{26}W_{49}C$$

Note that the inclusion of the X_1' and X_1 terms (Figure 85) as products with the ⓪ and ① terms, respectively, are provided to ensure greater reliability in the circuit.

Immediately after the G2 flip-flop changes to a true state, on the next clock pulse, the G1 flip-flop is made true as expressed by the equation $g_1=(⓪+①)G_2C$. This change in the G1 flip-flop is shown at point **337** of the waveforms.

The G3 flip-flop is made true on the next $P_{26}W_{49}$ pulse after the G1 flip-flop is made true, as shown at point **338** of the graph. This triggering of the G3 flip-flop to a true state thus occurs one revolution of the memory after the G2 flip-flop is made true, as shown by the equation $g_3=P_{26}W_{49}G_1G_2G_3'C$. Note that the G_3 term is always made true at the beginning of a major cycle of the computer, namely at the end of the $P_{26}W_{49}$ pulse time, or, stated another way, at the beginning of the W_0P_0 pulse time.

One clock pulse time after the G3 flip-flop is rendered true, the G2 flip-flop is triggered into a false state as shown at point **339** of the waveforms. This is expressed by equation ${}_0g_2=G_3C$.

It is now evident that the function of the G2 flip-flop is to ensure that the G3 flip-flop, the main "go" flip-flop, is not rendered true until after the 0 or 1 buttons, which are manually closed, have reached a steady state in the closed position, i.e., that the transients caused by the closing of the switches have died out.

It should be noticed that this method of initiating the G3 flip-flop into a true state is obtained by a D.C. potential. A D.C. potential is desired, instead of, for example, a differentiated pulse, because of the remote control feature of this machine. For example, if a differentiated pulse were initiated directly by the 0 or 1 buttons this pulse might be attenuated in the circuit before it could be effective since, in the preferred embodiment, the control panel (see Figure 89) is placed a considerable distance from the computer.

When using the D.C. potential (⓪+①) to initiate the triggering of the G flip-flops, there must be some way of insuring that the manual closing of the 0 or 1 button does not initiate the G2 flip-flop more than once. For example, it is often desired to have the G3 flip-flop, once it is triggered into a true state, to remain there for only a single revolution of the memory, at which time it automatically is rendered false. Under these conditions the G3 flip-flop is only true for $\frac{1}{60}$ of a second. It is thus possible that the operator may inadvertently hold the 0 or 1 buttons down long enough to reinitiate the go circuits after they have completed their cycle. Thus some means must be provided for making the closing of the 0 or 1 button cause the flip-flop G3 to be true for only one revolution of the memory drum, and for this

to be a positive action, independent of when the D.C. potential representing the term (⓪+①) is removed by the operator lifting his finger from the 0 or 1 button. The G1 flip-flop serves this purpose. Thus this flip-flop functions to take over the control of the "go" circuits by cutting off the effect of the D.C. proposition represented by the 0 and 1 buttons. The manner of doing this can be made clear by reference to the waveforms in Figure 67. First it should be noted that whenever the C_1 proposition is true, it is a command to advance all of the integrator setups one step. This mode of operation is used in the computer when it is desired to study a problem, for example. This is accomplished by triggering the G3 flip-flop to a false state after a single revolution of the memory which corresponds to approximately 1/60 of a second. Thus, at point 340 of the waveforms, the G3 flip-flop is triggered to a false state as described also by the equation ${}_0g_3=P_{26}W_{49}(C_1+H_1)G_3C$.

Note that when the C_1 proposition is not true, the H_1 term governs when the G3 flip-flop is triggered into a false state as indicated by the logical sum (C_1+H_1) in the ${}_0g_3$ equation. Ordinarily the H_1 term is controlled so as to remain in a false state until after an appreciable number of turns of the memory, in which event it does not enter into the present problem.

As shown in Figure 67, the G3 flip-flop is triggered to a false state at point 340, irrespective of the state of the (⓪+①) proposition. Thus the operator may release the 0 or 1 buttons at time point 334 or he may wait until time point 335 without affecting the operation of the circuits.

The use of the G1 flip-flop is now made clear. The G1 flip-flop is triggered into a true state after the G2 flip-flop is triggered into a true state. The G1 flip-flop then remains true until both the 0 and 1 buttons are opened and the G3 flip-flop is re-triggered to a false state. This is expressed by equation

$${}_0g_1=⓪'①'G_3'P_{26}W_{49}C$$

Having once triggered the G1 flip-flop into a true state, the G2 flip-flop cannot, itself, be triggered into a true state again to enable the G3 flip-flop to be made true.

In Figure 68, a diode network is shown for physically generating the equations associated with the G1, G2 and G3 flip-flops. In addition to the triggering equations discussed, the G2 flip-flop also functions in the fill reference circuit, and the G3 flip-flop functions in the overflow circuits as shown by the triggering equations in Figure 66. These latter triggering equations are also generated by the network in Figure 68.

As has been indicated, the G_3 term must be true to render the filling and computing circuits operable. The previous discussion of the go flip-flops describes how the G_3 term is controlled to be true for a single revolution of the memory. It is also desirable to have the G_3 term remain true for a plurality of revolutions of the memory so that the computation will be carried on for a predetermined interval of the independent variable x. To stop computation under these conditions, integrator setup I_0 is coded so that the r number has a carry out, a $+dz$ output, when this stopping point has been reached. This carry out is used to trigger the H1 flip-flop (Figure 46), which in turn stops the computation. This coding of the I_0 intergrator setup is obtained by filling the Y channel of word W_0 with a y number equal to:

$$(3) \qquad 2^{24}-\left(\frac{2^{24}}{\Delta X S_x}\right)$$

where ΔX is the interval desired, and S_x is the scale of the independent variable. By not coding any dx pickup for integrator setup I_0, the desired overflow will occur, as the y number will be continuously subtracted from the r number.

This overflow from the I_0 integrator setup (word W_0) is used for triggering the halt flip-flop H1 into a true state. The H_1 term in the ${}_0g_3$ equation (Figure 66) is thus indicative of when the computation process of the machine should stop after a plurality of revolutions.

The connnections controlling the halt flip-flop H1 will next be described by reference to Figure 46. The P_{26} position of the R channel is always 0 in the I_0 integrator setup, hence for an output to occur, the $(r+y_s)$ adder proposition R_s must be true at this time. This is possible only when a carry results from the P_{25} pulse time addition. In addition, the G3 flip-flop and the compute operation C_2 must both be true, i.e., of a high potential at the same time, to permit the next clock C to trigger the H1 flip-flop into a true state. It is thus seen that the halt flip-flop H1 is made true at the $P_{26}W_0$ time of the last revolution to be made by the drum. The computation process is then stopped, as shown by the triggering equation for the ${}_0g_3$ input to G3 flip-flop, at the $P_{26}W_{49}$ pulse time of that revolution. To ensure that the H1 flip-flop will be in a false position until triggered by the W_0 output during the last revolution of a computation, the false input ${}_0h_1$ is defined as $P_{26}W_{49}C$, thus the last clock pulse of the revolution is used to render the H1 flip-flop false.

The h_1 and ${}_0h_1$ triggering equations are generated by the logical nets shown in Figure 47.

Fill operation

The filling operation of the computer will now be described in detail. This operation is accomplished by feeding binary digits, one at a time, into the 27 pulse positions of a word selected by setting the function switch 340 to the appropriate channel, and the word switches 257 and 258 to the number of the word.

Before this filling operation can proceed, however, a reference pulse must be placed in the P_{26} position of the Z–1 general storage line.

The function of this reference pulse in the Z–1 line is to determine the position in the selected word in which to deposit a binary digit. The first binary digit, fed into the machine by the operator tapping the 0 or 1 buttons, is deposited into the P_{26} position of the selected word corresponding to the P_{26} position in which a reference pulse is located in the Z–1 line. The operation of the Z–1 line during the fill routine is such that, after the digit is deposited in the selected word, the reference pulse is automatically stepped within the Z–1 line to its next lower position. Thus, after the first digit is deposited, the reference pulse is shifted into the P_{25} position. After the second digit is filled into the P_{25} position of the selected word (by the operator again tapping the 0 or 1 buttons), the reference pulse is shifted in the Z–1 delay line to the P_{24} pulse position. The digits are thus filled one at a time into the selected word. When the last digit is filled in position P_0, the reference pulse is lost and the operator must place a new reference pulse in the P_{26} of the Z–1 storage line in order to fill another word of the memory.

It should be noted that the routine for filling the reference pulse into the Z–1 delay line is done independently of the setting of the main function switch 340 for the computing and filling operation, as shown in Figure 90.

The computing and filling operations are both characterized by the G_3 term, previously described, being true. In order to distinguish the fill reference operation therefrom, the G_3 term is false in this routine, i.e., the G_3' term is true.

Thus, whereas a closing of the 0 or 1 buttons, initiated the go flip-flops to perform the computing and filling operations, the closing of the ϕ button (Figure 88) initiates the fill reference pulse operation. As a result of this initiation a series of flip-flops Ax, G2 and Bx are activated so as to generate a D.C. term B_x which, when true, characterizes the fill reference operation. The sequence of operation of the Ax, G2 and Bx flip-flops, to generate the B_x term, is similar to that of the G1, G2 and G3 flip-flops (see Figures 66 and 67) which is used

to generate the G_3 term. The B_x term, similarly to the G_3 term, remains on for a single revolution of the memory and its timing is independent of the length of time the operator holds down the ϕ button.

The fill reference flip-flops A*x*, G2 and B*x* are schematically shown in Figures 57, 66 and 55 respectively. The logical equations below these flip-flops show how they are interconnected to function in the fill reference routine.

In this latter routine, all three flip-flops are initially in a false state. The G2 flip-flop is then triggered into a true state at the beginning of a computing cycle by the tapping of the ϕ push button. As soon as the G2 flip-flop is true, it sets up the condition for allowing the A*x* flip-flop to be triggered into a true state. The combination of the G2 and A*x* flip-flops being true when permits the B*x* flip-flop to be made true at the beginning of the next computing cycle, whereupon the G2 flip-flop is then rendered false.

The G2 flip-flop then remains insensitive to the state of the ϕ button until this B*x* flip-flop is turned off and the ϕ button released.

This system enables the fill reference circuit to be initiated by the D.C. state of the ϕ push buttons and yet be independent of the timing of the D.C. state of this switch. A D.C. state of the ϕ push button is required for initiating this routine because of the remote control operation of the machine.

Note that the function of the G2 flip-flops is the same in both the go series and the fill reference series of flip-flops. As previously described, the A*x* and B*x* flip flops function as the *dx* counter and *dx* register, respectively, during the computing operation. The fill reference routine is always performed when the computer unit is idle. Thus the A*x* and B*x* flip-flops are always inactive during the fill reference operation and have been, so to say, borrowed from the computer circuits to serve in the fill reference series of flip-flops. The false term G_3' noted in these fill reference flip-flop trigger circuits show that one of the conditions required to make these circuits active is that the computer is not in the computing operation.

The logical equations for the record propositions, i.e., for the terms that are recorded onto the channels of the drum memory from the computer unit, will next be described.

Computer unit recording circuits

As previously noted, the information recorded in the memory is obtained as a result of two main operations performed on the machine. One of these is the filling operation wherein the operator manually places initial information, according to the problem to be solved, onto the drum memory. The other is the computing operation wherein the computer automatically operates on the information picked up from the memory and records it back onto the drum.

Each of the filling and computing operations can themselves be separated by the laws of logic into two groups. During one such logical group of operations the information on the drum is altered, that is, the information recorded onto the memory from the computer logical nets is subject to be changed from that picked up from the corresponding position of the memory. During the other logical group of operations, the information in the computer does not change, that is the information recorded onto the memory from the logical nets is identical to that currently being observed on the drum by the pickup heads. This latter logical group includes the idle operation of the computer which is distinguished by the G3 flip-flops being in a false state.

With these main operations and logical groupings in mind, the logical equations for the Y_0 and R_0 record propositions will now be described together since they are similar in their nature.

As shown in Figure 70, the Y_0 logical equation equals:

$$Y_0=X_1G_3Z1_aW_iF_y+Y_sS_1+Y_1(G_3'+Z1_a'+W_i'+F_y')S_1'$$

and, as shown in Figure 71, the R_0 logical equation equals:

$$R_0=X_1G_3Z1_aW_iF_R+R_sS_1+R_1(G_3'+Z1_a'+W_i'+F_R')S_1'$$

Each of these equations is seen to have three main solutions which are logically summed together. The first of these solutions functions to change information on the memory during the filling operation, the second solution functions to change information on the memory during the computing operation, and the third solution functions to recirculate information without change during both the filling and computing operations.

It should be understood that during the filling operation, the operator records only one binary digit at a time in the selected word channels. Each time a digit button is tapped, the G_3 term is made true for one revolution of the memory. The only time information changes during this one revolution of the memory is when this digit is deposited in the single pulse position of the selected word channel, corresponding to the position of the reference pulse in the Z–1 delay line.

The first solution of the Y_0 and R_0 equations

$$X_1G_3Z1_aW_iF_y \text{ and } X_1G_3Z1_aW_iF_R$$

respectively, generates the fill digit so that it can be recorded in the proper pulse position of the drum. Analyzing this first Y_0 solution: the term W_1 selects the proper word; term F_y selects the channel; the G_3 term characterizes a condition which precisely defines a period equal to the major cycle of the memory; the $Z1_a$ term defines the position of the reference pulse in the Z–1 delay line; and, finally, the X_1 term indicates the digit to be recorded into the memory is the digit 1. At the so defined clock pulse time of the major cycle, when all these terms are simultaneously true, a pulse is deposited onto the memory.

As is now evident, in view of the previous detailed description of the computation process of the machine, the only changes made in the Y and R channels, during computation, is an altering of the *y* and *r* numbers therein.

As previously noted in Figure 15, the start flip-flop S1 is triggered into a true state by a start pulse in the Y channel which is located in a pulse position just prior to the position of the first digit of the *y* and *r* numbers. The start flip-flop then is triggered to a false state at the end of the P_{24} pulse time which corresponds to the last position of the *y* and *r* numbers. Thus the state of the start flip-flop logically separates the computing process into the period when the information in the Y and R channels on the memory is subject to change, and when the information in these channels is recirculated without change.

Thus, referring to the second solution of the Y_0 and R_0 equations shown, namely, Y_sS_1 and R_sS_1, these solutions define the period of each word, during computation, when the *y* and *r* number digits are subject to change.

The third solution in each of the Y_0 and R_0 record logical equation defines when the information is recirculated for both the filling and computing processes. These solutions are:

$$Y_1(G_3'+Z1_a'+W_i'+F_y')S_1'$$

and

$$R_1(G_3'+Z1_a'+W_i'+F_R')S_1'$$

The logical sums:

$$(G_3'+Z1_a'+W_i'+F_R')$$

and

$$(G_3'+Z1_a'+W_i'+F_R')$$

in these solutions are the logical negatives of the products defining the period during the filling operation when a

digit is recorded onto the memory. Thus these logical sums represent the "not fill" period of the filling operation, or the period when information is recirculated without change.

The S_1' term in the above solutions correspond to recirculation, without change, during the computing operation. Referring to the Y_0 logical equation, the F_y' term is also true during the computation process. Hence the particular product $Y_1F_yS_1'$, inherent in the third solution of the Y_0 equations, ensures that the third solution is true during the computation period defined by S_1', so as to recirculate the Y channel information. Now then, the S_1' term is always true during the entire filling operation. Thus, logically multiplying the S_1' term by the "not fill" term does not prevent the "not fill" term from defining the period during the filling operation when the third solution, which governs recirculation, is true.

In Figure 70 the logical network for physically generating the Y_0 record proposition is shown; and in Figure 71, the logical network is shown for generating the R_0 record propositions.

The logical equations for the D_{X0}, D_{10}, and D_{20} record propositions presented in Figures 72 and 73 will next be described. The information in these channels is never altered during computation. It is only during the filling operation that information is changed in these channels. The logical equations for these propositions are as follows:

$$D_{X0}=X_1G_3Z1_aW_iF_{DX}+D_X(D_3'+Z1_a'+W_i'+F_{DX}')$$
$$D_{10}=X_1G_3Z1_aW_iF_{D1}+D_1(G_3'+Z1_a'+W_i'+F_{D1}')$$
$$D_{20}=X_1G_3Z1_aW_iF_{D2}+D_2(G_3'+Z1_a'+W_i'+F_{D2}')$$

The first solution of each of these equations defines the filling of a digit in a particular pulse position of a word, as determined by the location of the reference pulse in the Z–1 delay line.

The second solution of each of these equations is comprised of a logical sum which represents the "not fill" period of the filling operation. Thus when the second solution is true, the information in the remaining digit positions of the word being filled, as well as all other word information, is recirculated in the memory without change.

The product D_xF_{DX}', D_1F_{D1}', and D_2F_{D2}', inherent in each of the second solutions of the above equations, defines the recirculation of these channels during the computing operation of the machine. This results from the fact that the fill switches are necessarily false during a computation routine and the terms such as F_{DX}', F_{D1}' and F_{D2}' are thus true.

The logical network for the D_{X0} proposition is shown in Figure 72, and the net for the D_{10} and D_{20} propositions is shown in Figure 73.

Referring next to Figure 74, the logical equations for the Z_{10} and Z_{20} record propositions will next be shown and described. Seven major solutions are required for the Z_{10} proposition, and four major solutions for the Z_{20} proposition.

During the computation process three Z_{10} record solutions are required. The first of these, $W_0P_{26}G_3(C_1+C_2)$, generates the independent variable dx input, representing the time constant of the machine. This dx input is generated once each revolution of the memory and recorded into the Z–1 delay line by one of the integrator setups, in this case I_0, whose output is obtained at W_0P_{26} time. The term G_3 indicates either a computing or fill-digit operation is being carried on. The sum (C_1+C_2) distinguishes the operation as a computation for either a single revolution of the memory, or for a fixed number of revolutions of the memory. This latter of the two operations is carried on for a predetermined fixed increment of the independent variable x.

The second computing solution of the Z_{10} record proposition, $R_6P_{26}G_3(C_1+C_2)$, generates the dz function output obtained as a result of stepping each integrator setup through the computer unit. More particularly, this output results from the $r+y_3$ addition during the P_{26} pulse time. These dz outputs in the Z–1 general storage line are made available as inputs to other integrator setups.

The third computing solution of the Z_{10} record proposition, $P_{26}'Z_1G_3(C_1+C_2)$, is used to precess the information already in the Z–1 storage line, as shown and described in Figure 63. At P_{26} time, the new dz output is recorded in the Z–1 general delay line, hence this precession operation is true, except for the P_{26} pulse time.

For recirculating the information in the Z–1 channel when the computer is idling, the solution $Z1_aG_3'$ is made available.

The Z–1 and Z–2 delay lines are not initially recorded with information, as are the word channels, and hence no solutions are required for filling these lines. However, the Z–1 delay line operates during the filling operation as an aid in filling the word channels. The three remaining solutions of the Z–1 delay line are required for this purpose. The first of those latter Z_{10} record solutions is the fill reference solution:

$$P_{26}(S_7A_7+S_7'A_7')B_xG_3'$$

This equation states that when the B_x term is true and the G_3' term is false, which combination characterizes the fill reference operation, a reference pulse is placed in the P_{26} pulse position of the Z–1 general storage line. Introducing this reference pulse into the Z–1 storage line persents some problems. The Z–1 line is two word periods in length, so if a reference pulse is placed in the Z–1 line every P_{26} pulse time, then two reference pulses exist in the Z–1 line. Since the Z–1 line does not precess during the fill reference operation, the reference pulses generated every other time are positioned on top of each other.

In order to prevent two reference pulses from being deposited in the Z–1 delay line, use is made of the fact that the W_1 selector switch is always set so as to be indicating either an even or odd number word of the memory. Thus note that the state of the S7 set of wafer switches (see Figure 23) is such that either the S_7 term is true, or the S_7' term is true. Both of these terms cannot be true simultaneously. Now the word counter for the machine, which is in operation while the drum is rotating, has a flip-flop A7 which has its output A_7 true for odd numbered word periods and its output A_7' true for even numbered word periods. It is now evident that the complex term $(S_7A_7+S_7'A_7')$ is only true for every other word of the memory, since only term S_7 or S_7' is true during the fill reference routine, while terms A_7 and A_7' are true for alternate words of this routine. Thus the complex term $(S_7A_7+S_7'A_7')$ ensures that only one reference pulse will be positioned in the Z–1 delay line, since the product S_7A_7, or $S_7'A_7'$, will only be generated every other word time.

After a digit is deposited in the selected word channel, it is desired to automatically shift the reference pulse to the next lower pulse position in the Z–1 line. For example, after depositing a pulse in the P_{26} position of a word channel, the reference pulse automatically shifts to the P_{25} pulse position of the Z–1 line, thus providing the reference for the next digit to be filled. During the word periods not being filled, it is desired to recirculate the reference pulse in the Z–1 delay line, without shift.

The additional solutions of the Z_{10} record proposition, which aid in filling the other channels, are the ones which control the shifting and recirculating of the reference pulse as above described.

Solution $(F_R+F_y+F_{D1}+F_{D2}+F_{DX})Z_1W_iG_3$ is used for precessing or shifting the reference pulse to the next lower pulse position during the particular word time being filled. Note that the Z1 flip-flop output is observed by the com-

puter unit which results in the shifting taking place to the next lower position.

Solution $(F_R+F_y+F_{D1}+F_{D2}+F_{DX})Z1_aW_i'G_3$ enables the single reference pulse to be recirculated without shift, during all the word times, except the word being filled. In this case the Z1*a* flip-flop output (see Figure 13) is observed in the line which results in mere recirculation of the information without precession.

Note that all the computing and filling solutions are logically multiplied by $(B_x'+G_3)$. This term serves to distinguish these solutions from the fill reference solution which is characterized by term B_x being true.

Referring next to the Z_{20} record proposition also shown in Figure 74, the only solutions required are the ones associated with the computing process and for recirculating information during idling.

The Z–1 and Z–2 general storage lines both have the same information fed therein during computing and differ only in that they shift or precess their information in opposite directions. Thus the computing solutions previously described for Z–1 general storage line are similar to the Z–2 solutions which follow. The first Z_{20} solution is $P_{26}W_0G_3(C_1+C_2)$ which feeds the *dx* independent variable into the Z–2 delay line. The second solution is $R_sP_{26}G_3(C_1+C_2)$ which represents the *dz* output generated during the P_{26} pulse time of each word. The third solution is $P_{26}'Z_2G_3(C_1+C_2)$ which is used for precessing the *dz* outputs in the Z–2 delay line during all but the P_{26} pulse time.

The final solution of the Z_{20} record proposition, $Z2_aG_3'$, is used to recirculate the information in the Z–2 delay line when the computer is idling.

The solutions for the Z_{10} and Z_{20} record propositions are generated by the logical network shown in Figure 74.

Referring to Figure 69, a description will next be presented of the circuitry used for recording the information generated by the computer logical nets back onto the drum memory. The circuitry shown is for the Y_0 record proposition, but a similar circuit is provided for each of the other record propositions R_0, D_{10}, D_{20}, D_{X0}, Z_{10} and Z_{20}.

The signal generated on output Y_0 shown in Figure 70 is applied onto the grid **350** (Figure 69) of a first record tube **351** whose plate output is D.C. coupled to the grids of a pair of record tubes **352** and **353**. The plates of these latter tubes are both connected through magnetizing coil **354** to a positive source of potential, 265 volts.

Any time tubes **352** and **353** conduct, the current therethrough energizes the coil **354** of the Y_0 record head **125** to produce the magnetizing pattern M on the desired pulse positions of the memory drum **106**.

As shown in Figure 70, a clear switch **356** is connected to the record proposition output line whose output controls the grid **350** of first record tube **351** shown in Figure 69. This clear switch **356** is normally in its upper position where it places such a bias on the grid **350** so as to render tube **351** sensitive to both the high and low states of the Y_0 output. Whenever, clear switch **356** is depressed, however, as when the operator desires to clear the Y channel on the memory of all information, a negative bias is placed on grid **351** such that record tube **351** is prevented from conducting even when the Y_0 output is relatively high in potential.

A clear switch, such as **356**, is provided for each of the remaining record proposition circuits.

The circuit, broadly shown in Figure 6, for obtaining a visual recording of the contents of one channel of a word in the memory will next be described.

Visual readout

The means provided for accomplishing this operation includes setting the W_i switch, shown in Figure 28, to select the desired word period on the memory, and setting the readout switch **357**, shown in Figure 82, to select the desired channel.

It is noted that this readout operation takes place when the computer is not computing but rather is in an idling operation, with the memory rotating of course. This operation of the computer is characterized by the main go flip-flop G3 being in a false state, as evidenced by G_3' being of a high voltage.

Because of this, the flip-flop circuits required to perform this reading out operation are borrowed from those previously used for performing functions during computation.

One set of these flip-flops is the E1, E2, E3, and E4 flip-flops, previously described in Figure 76, which are used as the E register for storing the P_0 and P_{26} codes during computation. These flip-flops are here used to function as an E counter which programs the reading out (in sets of three) the 27 binary digits comprising a channel of a word.

Another set of flip-flops used in the recording operation is the B6, B7, B8, B9 and B10 flip-flops which comprise the Σdy register during computation. These latter flip-flops are here used for picking up three consecutive binary digits from a word channel of the memory, and storing them for approximately one revolution of the memory so that they can provide the signals required to energize the neon lights in the readout unit.

In Figure 81, a schematic diagram broadly illustrates the arrangement of the readout circuits. Each of the flip-flops Y1, R1, D1, D2 and DX shown contain on successive clock pulses one of the binary digits picked up from its associated channel on the drum. Read switch **357***a* which schematically corresponds to the readout switch **357** in Figure 82 is manually positioned to select one of the channel flip-flops, in this case flip-flop Y1. Thus in the diagram, the Y1 flip-flop is shown connected to the B6 flip-flop which in turn is connected to the B7 flip-flop. The Y1, B6 and B7 flip-flops are each further shown to be connected by means of read gates **358**, **359** and **360** to flip-flops B8, B9 and B10, respectively. Each of these connecting gates is controlled by a T proposition. The outputs of flip-flops B8, B9 and B10 are used for forming a diode matrix along with the count lines E^0, E^1, etc. of the E counter (Figure 78).

Referring to the table in Figure 84, logical equations are presented for controlling each of the **27** neon lights L_0, L_1, etc. which comprise the readout unit **361** (Figure **91**. The diode matrix in Figure 81 physically generates these equations. For example, the equation for neon light L_0 is $B_{10}E^0$. This signifies that the logical product of the B_{10} output of the B10 flip-flop with the E^0 count of the E counter will, when true, cause the neon light L_0 to be lit. Thus in Figure 81, the neon bulb L_0 has one of its terminals connected to the output of product circuit **362** and its other terminal connected to ground, and whenever the output of the product circuit **362** is high in potential, the neon bulb L_0 is fired.

The operation of the readout circuit in Figure 81 will now be more clearly described by reference to the time chart in Figure 11. This chart illustrates generally how the information in a word channel is fed out of the memory.

Note first that during successive revolutions of the memory, as shown in the chart, each time the word W_1 is scanned a T proposition is generated. As schematically shown in Figure 10, this T proposition is generated three clock pulse times earlier, each successive revolution of the memory, at the coincidence of pulse times such as $P_{26}E^0$, $P_{23}E^1$, etc. As shown, this T proposition serves two functions: it opens the gates **358**, **359** and **360** (Figure 81) so that the existing information in the Y1, B6 and B7 flip-flops can be stepped into the static registers flip-flops B8, B9 and B10, respectively; and it also triggers the E flip-flops so as to advance the E count one unit.

Assume that at the beginning of the readout operation the E counter flip-flops are recording count output E^0,

As the binary digits in the P_0 through P_{26} positions of the Y channel are serially stepped through the flip-flops Y1, B6 and B7, nothing happens until P_{26} pulse time when the first T pulse is generated. This T pulse causes the existing contents of the Y1, B6 and B7 flip-flops, which correspond to the P_{26}, P_{25}, and P_{24} pulse positions of the W_1 word time, to be transfererd to the B8, B9 and B10 flip-flops, respectively, as shown in Figure 81. This information remains in these latter flip-flops for approximately one revolution of the memory, that is until the next T pulse is generated.

Referring to Figure 10, the same T pulse used for transferring the information to the static registers B8, B9, and B10 is also used for advancing the E counter one unit. Thus it is the E^1 count which is high in potential for this present case. Thus until the second T pulse is generated during the next revolution of the memory, the L_{24}, L_{25}, and L_{26} neon bulbs are lit in accordance with the condition of the E^1 count line of the E counter and the information in the B10, B9 and B8 flip-flops, respectively.

During the second revolution of the memory, as shown in the chart of Figure 11, the T pulse is generated three pulse positions earlier than it was for the first revolution, i.e., at the coincidence of terms $P_{23}E^1$.

Thus the second T pulse again results in the contents of the Y1, B6 and B7 flip-flops (which in this case contains the information in pulse positions P_{21}, P_{22} and P_{23}, respectively) being transferred to the static registers B8, B9 and B10. As before, this T proposition immediately triggers the E counter to indicate the E^2 count therein.

Any of the neon lights L_{24}, L_{25} and L_{26} which were lit are now extinguished, since count E^1 is no longer true, and the neon lights L_{21}, L_{22} and L_{23} instead are subject to being ignited in accordance with the E^2 count line and the outputs of the B8, B9 and B10 flip-flops.

In a similar manner, on each successive memory reolution, the T pulse is generated three clock pulse times ahead of when it was generated in the previous turn of the memory, to obtain each of the remaining set of binary digit outputs in word W_1. As previously noted, such a set of three binary digits is referred to as a binary coded octal digit, since each set can readily be translated to denote its octal equivalent.

In this manner the entire word channel is visually indicated on the neon readout unit 361 shown on the control panel in Figure 91.

It should be noted that by this means for obtaining a visual indication of the channel output, three of the neon bulbs, representing an octal digit, are observed for approximately one revolution of the memory and that the following three binary digits are observed during the succeeding revolution. Only one set of binary digits (octal digit) is actually read at a time. Thus it takes approximately nine revolutions of the memory to read out the selected word channel and cause it to be visually manifested. Because of the serial manner of reading out the octal digits, a slight flicker is present on the neon readout unit 361, but the speed of the neon readout is sufficient so as to give to the observer the effect of a continuous reading.

The logical circuitry for accomplishing the readout operation will now be presented.

Referring first to the E1–E4 flip-flops in Figure 76, the logical equations for enabling them to function as a counter for the readout operation are shown below their respective flip-flops. These equations are obtained as for the other counting circuits, by noting the states of the E flip-flops when the flip-flop in question is to change states on the next T pulse, as revealed in Figure 75. Thus the input equations of the E1 flip-flop, when functioning as a counter, are $e_1=E_1'G_3'TC$ and ${}_0e_1=E_1G_3'TC$. Thus in order for the E1 flip-flop to be rendered true at the end of a clock period, it must be in a false state during the clock period as evidenced by the E_1' term being true. The T term is included in all of the counting equations since, of course, it is an indication of the unit which is being counted. Furthermore, the G_3' is included to distinguish this operation from the time when these E flip-flops are used as a code register which is characterized by the G_3 term being true.

Before describing the counter reset equations associated with the E counter, the logical equation defining the T proposition will be presented in Figure 83.

As noted in the chart of Figure 11, the T proposition is generated at the coincidence of pulse time and E counts shown such as $P_{26}E^0$, $P_{23}E^1$, $P_{20}E^2$, etc. (see Figure 10). These coincidences, written as logical products, are symbolically summed together as shown in the detailed circuitry in Figure 83. The line 363 carrying the solution of this logical sum is fed into one input 364 of a three input product circuit 365, a second input 366, of which, senses the output of the W_1 selector switch circuit. The third input 367 to product circuit 365 carries a solution, symbolically represented as

$$(E_1'+E_2+E_3+E_4')$$

Referring to the table of the E counts in Figure 75, it is noted that E^9 is symbolically defined as $E_1\ E_2'\ E_3'\ E_4$. Thus $(E_1'+E_2+E_3+E_4')$ represents the logical negative of E^9, or "not E^9." The reason for adding this solution to the T proposition is to limit the counting of the E counter to count E^9.

As seen in Figure 11, after count E^8, the last set of binary digits in a word channel have been read out, and the readout circuit is ready to start to read out the first set again, namely P_{26}, P_{25}, P_{24}.

If the E counter were immediately set to E^0 when the T pulse is generated during the 9th revolution, the P_2, P_1 and P_0 ouputs would be reflected in the L_2, L_1, L_0 neon bulbs for only the remainder of the W_1 word instead of for the longer time of approximately a revolution of the memory. This is because P_{26} pulse time of W_1 has as yet not come up during the current scanning of word W_1 and when it does, if E^0 count is true, the T pulse is generated again causing the neon readout to change too soon.

Thus the circuits are so arranged that under the above circumstance, at the coincidence of P_2E^8, the T proposition generated triggers the E counters to indicate count E^9. As a result the T proposition cannot change at the P_{26} time of the W_1 word. This ensures that the information in the B8, B9 and B10 static registers flip-flop does not change at this time. However, the T proposition not only functions to change the static register but also functions to advance the E counter. Since it is desired in the present case to reset the E counter after P_{26} time of the word W_1, to indicate the E^0 count, the counter reset equations, shown in Figure 76, serve this purpose.

Assume that, in the present instance, the E counter is indicating the count E^9. Referring to the table in Figure 75, in order to change count E^9 to count E^0, it is necessary to only change flip-flops E1 and E4 to false states since flip-flops E2 and E3 are already false. Thus the reset equations for the E counter which enables the readout cycle to repeat are shown to be ${}_0e_1=P_{26}E^9G_3'C$ and ${}_0e_4=P_{26}E^9G_3'C$.

The logical equations for the E flip-flop functioning as the E counter are shown in Figure 76. These equations are generated, in Figure 77, along with the networks for the other functions of these flip-flops, and will not be further described.

Referring to the T proposition in Figure 83, the function of the logical products E_4E_3 and E_4E_2 shown will now be explained by referring to the table in Figure 75. Note that it is possible that the E counter might be containing one of the code counts E^{10} through E^{15} when the readout operation is first started. Since the T proposition

is only rendered true for counts E^0 through E^8, while count E^9 causes the counter to reset itself, the E counter would be unable to get out of counts E^{10} through E^{15}. Note in the table of Figure 75, that counts E^{10} and E^{11} are distinguished by the logical product E_2E_4, and counts E^{12} through E^{15} are distinguished by the logical product E_3E_4. Thus, whenever these products are true, the T proposition is true and the E counter will count into the operative range E^0–E^8.

The logical nets for controlling the operation of the B6 through B10 flip-flops during readout are shown in Figure 43 and will next be described.

As previously shown in Figure 81, one of the channel flip-flops, depending on the setting of readout switch 357*a*, transfers its content, at the end of the current clock pulse, into the B6 flip-flop. Thus:

$$b_6=G_3'(Y_1R_y+R_1R_R+D_1R_{D1}+D_2R_{D2}+D_XR_{DX})C$$

and

$${}_0b_6=G_3'(Y_1'R_y+R_1'R_R+D_1'R_{D1}+D_2'R_{D2}+D_X'R_{DX})C$$

The B6 flip-flop transfers its content at the end of this clock period into the B7 flip-flop. Thus $b_7=G_3'B_6C$ and ${}_0b_7=G_3'B_6'C$.

Now then, depending on the T proposition being true, the contents of: the selected channel flip-flop; the B6 flip-flop; and the B7 flip-flop, simultaneously transfer their contents into the B8, B9 and B10 static register flip-flops, respectively. The logical equations for this readout transfer operation are shown below their respective flip-flops in Figure 42. These readout logical equations for the B6 through B10 flip-flops are generated in Figure 43 by diode nets as previously described.

In addition to reading out a word channel by use of the neon readout circuit above described, it is noted in Figure 6 that the *y* number content of ten integrator setups can be made externally available on output lines S_6–S_{15} by appropriate coding, while the computer is functioning. The P_0 code of a word, containing a binary equivalent of one of the decimal numbers 6 through 15, will "connect" the Y channel output in the following P_1 through P_{25} pulse times to the S output line having a corresponding subscript. Figure 32*a* shows the logical net which enables, for example, the S_6 output to be so energized. This line responds to the Y channel of a word during the period defined by $P_0'P_{26}'$ when the P_0 code of that word is equal to the binary equivalent of a decimal 6. Thus $S_6=E^6P_0'P_{26}'C$ is the equation generated by the net. The other S output lines are similar, being distinguished only by a different E count (P_0 code).

Referring next to Figure 79, a schematic diagram is presented of the F1 flip-flop together with the logical equations for its triggering inputs.

In scaling a problem for the differential analyzer, it is often difficult to foresee just what range a dependent variable may cover during a given computation. Thus for the convenience of the operator, an automatic check circuit comprised of the F1 flip-flop has been provided for guarding against a dependent variable exceeding its allowable range. This circuit permits the operator to code the machine so that an inspection can be made of any of the *y* numbers while it is being generated. Anytime an overflow is sensed beyond the most significant digit provided for, the computation automatically stops. A visual indicator signals the operator to show such an overflow has been the cause of the computer stopping.

As referred to in Figure 4, showing the arrangement of a word in the memory, the P_1 position of the R channel is reserved for a code pulse which instructs the F1 flip-flop to inspect the *y* number, generated in the following pulse positions, for a possible overflow digit.

Referring to the triggering input equations for the F1 flip-flops, the equation $f_1=(C_1+C_2)P_1R_1G_3C$ renders the F1 flip-flop true when the code pulse is present in the P_1 position of the R channel as evidenced by product P_1R_1. The $(C_1+C_2)G_3$ combination of terms in this equation corresponds to the computing operation as previously described. After once being triggered true, the F1 flip-flop is rendered false at the next P_{25} pulse time if the *y* number does not have an overflow digit. If an overflow digit is sensed, the F1 flip-flop does not get triggered into a false state at the P_{25} pulse time.

The grid equation for rendering the F1 flip-flop false is ${}_0f_1=[P_{25}(Y_1'B_6'+Y_1K_1'+K_1B_6)G_3+P_0W_0G_3]C$. The complex term $(Y_1'B_6'+Y_1K_1'+K_1B_6)$ in this latter equation is a logical sum indicative of when the inputs to the $y+\Sigma dy$ adder do not generate an overflow. The understanding of this complex term can be more readily understood by noting the logical expression representing when an overflow digit is generated.

When the result of adding the magnitude of the *y* digits to the Σdy digits results in a carry at P_{24} pulse time, the K1 flip-flop is rendered true for the P_{25} pulse time. Now then, if the sign of the *y* number as noted in the P_{25} pulse time is positive, the Y_1 term is true. If the Σdy number added therto is likewise positive in sign, the B_6' term is true during this P_{25} pulse time. The combination of adding a positive Σdy number to a positive *y* number with the carry term K_1 true, logically noted as $Y_1B_6'K_1$, generates an overflow digit indicative of the new *y* number exceeding its positive capacity.

When the sign of the *y* number is negative at the same time the sign of the Σdy number added thereto is negative, then the sum exceeds the negative capacity of the *y* number if no carry is derived as a result of the P_{24} pulse time addition. This last set of conditions is represented as $Y_1'B_6K_1'$.

It can be shown by symbolic logic that the logical negative of the logical sum of these two sets of overflow conditions, namely, $(Y_1B_6'K_1+Y_1'B_6K_1')'$ is equal to $(Y_1'B_6'+Y_1K_1'+K_1B_6)$. Thus the complex term in the ${}_0f_1$ equation is interpreted to mean a "no overflow" condition.

The networks for generating the f_1 and ${}_0f_1$ triggering equations is presented in Figure 80.

If the *y* number inspected does generate an overflow digit, the F1 flip-flop remains in a true state during P_{26} pulse time, as noted by term F_1 being of a high voltage. When this occurs, the computing operation terminates at the end of the P_{26} pulse time.

The computing operation requires that the G_3 term be true. Thus by rendering the G_3 flip-flop into a false state at P_{26} time, if the F1 flip-flop is in a true state, the G_3' term is now true and computation ceases.

Referring to the G_3 flip-flop shown in Figure 66, the equation for rendering this flip-flop false when an overflow occurs is shown to be ${}_0g_3=P_{26}F_1G_3C$.

In the perspective view of the control panel 383 shown in Figure 91, an indicating neon bulb 368 is lit when the F1 flip-flop is in a true state, to let the operator know that an overflow has occurred.

After an overflow has stopped the computation process, the F1 flip-flop remains in its true state; but the F1 flip-flop should be in its false state at the beginning of a computation in order to function properly. Thus when this process is again initiated, the F1 flip-flop is triggered into a false state at the beginning of the memory cycle as defined by P_0W_0. Thus the additional solution, $P_0W_0G_3$, for the ${}_0f_1$ equation is accounted for.

In Figure 86, the 1–0 delay flip-flop De is shown and its function will now be described.

External dz output circuits

As previously seen in connection with Figure 6, circuit means are provided for feeding on separate lines Q_1^+, Q_1^-; Q_2^+, Q_2^-; etc., the *dz* outputs of certain integrator setups in accordance with the P_{26} code located in the words. This feeding out of the *dz* outputs is accomplished while the machine is computing a problem.

In order to understand how the De flip-flop operates to

control the dz output increments fed out on the Q output lines, the nature of the dz outputs from an integrator setup should be reviewed. As shown in connection with the example in Figure 61, a zero dz output from an integrator setup is equivalent to a series of alternate $+dz$ and $-dz$ units. In other words, the change of a function in a given period is obtained by algebraically summing the positive and negative dz outputs generated during that period.

The De flip-flop is used to eliminate this $+dz$, $-dz$ equavilent of zero in the operation of this type of machine. It does this by virtually biasing out the 1–0 oscillation. The De flip-flop serves as a means for matching a dz output currently generated in an integrator setup with the dz output generated by the same integrator setup during the previous revolution of the memory. In order for a dz output to be fed out of the computer by this system, the current dz output must match the previous dz output.

Referring to the d_e and ${}_0d_e$ triggering equations below the block diagram of the 1–0 delay flip-flop De in Figure 86, to simplify the explanation, the second solutions shown for these equations will be omitted in the present part of the discussion and will be introduced later.

As seen by the first solution of the ${}_0d_e$ equation, the De flip-flop is triggered into a false state at the beginning of each word period, i.e., ${}_0d_e=P_0C$. Now then, by referring to the chart in Figure 63, it can be determined that the dz output generated by any word, during the previous revolution of the memory, can be picked up from the Z–1 delay line during the present revolution of the memory, at the P_1 pulse time of the word. This latter dz output, if equivalent to a $+dz$, is used to trigger the De flip-flop to a true state as evidenced by the first solution of the d_e equation, i.e., $d_e=P_1Z_1$. If the dz output in the Z–1 delay line is equivalent to a $-dz$, the De flip-flop remains in its false state. Thus the state of the De flip-flop at the end of the P_1 pulse time indicates whether the previous dz output was positive or negative.

Now then, referring to the output lines Q_1^+, Q_1^-; Q_2^+, Q_2^-; etc. used for feeding out the dz outputs, the logical equations for the first set of these lines will be presented in Figure 65*a*, along with the diode network for generating them. These equations are: $Q_1^+=G_3P_0K_2D_eE^1$, and $Q_1^-=G_3P_0K_2'D_e'E^1$.

The equations for the remaining sets of dz output lines are all similar and will not be further described. They are distinguished only by the fact that a different E count is used for selecting each of the 10 different sets. In the present case, it is assumed the integrator outputs being sensed are coded with the E^1 count.

These equations for the output lines essentially compare the current dz output with the previous dz output. The current dz output is sensed in these equations by noting the condition of the K2 carry flip-flop. If the K2 flip-flop is true, a $(+dz)$ output is generated and if false, a $(-dz)$ output is obtained. The previous dz output, having been stored in the De flip-flop, is sensed by the state of that flip-flop.

If the K_2 and D_e terms are both true at P_0 time, then the current and the previous dz outputs are both positive in sign. Thus the Q_1^+ equation is satisfied and a positive voltage is impressed on the Q_1^+ output line.

If the K_2' and D_e' terms are both true at P_0 time, then the current and previous dz outputs are both negative in sign, resulting in a positive voltage being impressed on the Q_1^- output line.

If the state of the K2 and the De flip-flops are dissimilar, as when the current dz output does not match the previously generated dz output, then neither of the Q output equations are satisfied and both output lines have a low voltage impressed thereon. This latter is equivalent to a zero output.

Returning now to the De flip-flop, the second solutions of the d_e and ${}_0d_e$ triggering equations will be introduced back into the circuits. As previously noted, the function of a pulse in the P_{26} position of the R channel is to reverse the sign of the dz output.

Assuming no pulse is in this position, then the operation of the dz output circuits of the computer is as above described. Now then, if a reversing pulse is located in the P_{26} position of the word in question, then whatever state the delay flip-flop De reposes in after P_1 time, is reversed at P_{26} pulse time. The outputs of the De flip-flop to be matched with the outputs of the K2 flip-flop in the Q equations are reversed when the reversing pulse is present, thus resulting in a different voltage output on the output lines Q_1^+ and Q_1^-.

In Figure 89, an overall illustration of the preferred embodiment of the present invention is shown. The machine is divided into three units: the control box **389**, the computer **390**, and the power supply **391**. The computer **390** is connected to each of the other units by means of cables.

The principal unit of the machine is the computer **390** which is enclosed in a reinforced sheet metal housing **369**. The rectangular right side **370** and top **371** of this housing are rigidly joined together at their common edge to form a single unit which swings about a hinge **372** extending lengthwise along one corner of the housing **369**. The inside surfaces of the swinging side **370** and top **371** of the housing each have two substantially square diode boards **373** mounted thereon. The tubes **374** required for the flip-flop circuits are arranged in rows along the sides of the diode boards **373**. The outside surfaces of the diode boards **373** have clips **375** arranged in rows and columns for holding the crystal diodes **376** needed for the logical nets. Panels **377**, on the housing side **370** and top **371**, permit easy access to these diodes. Inside the housing **369** is located the memory drum **106**, with the pickup heads **378** arranged around its periphery, and a fan **379** for cooling the unit. Cable **393**, connected to the side of housing **369**, feeds the arbitrary inputs rates for the computer as received from external instruments for example, and cable **394** feeds the dz output rate generated by the computer to external devices.

Mounted on a table **380** is the remote control box **389** which is connected to the computer **390** by means of a pair of cables **381**. All the controls needed for manually operating the computer **390** are contained on the one control panel **383**.

A larger view of the control box **389** and panel **383** is shown in Figure 91. On the left of the control panel **383**, four multi-position switches are located. Main function switch **340** is used for setting the computer into either the fill operation or the computer operation. Switches **257** and **258** are used for manually selecting one of the word periods (0–49) in the memory, as required, for example, during the filling operation. Of these, switch **257** selects the units digits, and switch **258** selects the tens digit of the word number. Read switch **357** is used for selecting the channel of a particular word which is to be visually read out on the readout unit **361** provided along the top of the control panel **383**. This readout unit **361** is comprised of 27 neon tubes, such as tube **382**, corresponding to the pulse positions P_0–P_{26} of a word channel.

In the center of the control panel **383**, three push buttons are located, the 0 button **345**, the 1 button **346**, and the ϕ button **347**. The 0 and 1 buttons are used for filling digits into the memory of the computer, and for initiating the "go" flip-flops. The ϕ button is used for placing a reference pulse into the Z–1 delay line as a prerequisite to filling digits into the memory. Along the right side of the control panel **383** are two columns of two position switches. The push button switches in the left column are used for clearing information from the memory. The upper seven switches, **356**, each operate to clear one of the channels, as marked. The lower switch **384** is a master clear switch which when closed

clears all the channels of information, simultaneously. In the outer column, the power switches for the computer are located. Push button switches **385***a* and **385***b* control the D.C. power for the computer; two-position switch **386** connects the power to the memory motor **110**; and, two-position switch **387** connects the power supply **391** to the outside power source. A neon bulb signal is provided for indicating when each of the power switches is closed.

In addition, two other signal indicators are provided on control panel **383**. Whenever the G3 flip-flop is in the true state, the computer indicator **388** will light up; and any time the computation stops, because of an overflow in a *y* number, the overflow indicator **368** will light up.

Operation

The basic method of solving differential equations by the use of integrating means such as these is to feed the outputs of one or a plurality of integrator setups to some of the inputs of others in such a manner as to create a feed back system, and then "drive" the entire integrator system by feeding a pulse generated once each revolution of the memory into the remaining inputs of the system. This driving pulse, the independent variable pulse *dx*, is generated by an integrator setup not in the system referred to above. Generally this is the I_0 integrator setup.

On actuating the above system, particular solutions of the equation are defined by noting the values of certain *y* numbers corresponding to incremental values of the independent variable *x*.

The integrator setups required and the interconnections between them are determined by the differential equations to be solved and are well known in the differential analyzer art. The scale factors for the variable must also be determined. The relation of the scale factors for the variables of a particular integrator is determined by equation

$$S_z=\frac{S_xS_y}{2^n}$$

where S is the scale factor for the respective variable. Thus by arbitrarily choosing a scale factor for the *x* and *y* variables consistent with the accuracy and the capacity of the memory, the scale factor for the *z* variable is determined.

As an example of how a particular integrator setup would be coded into the machine, reference will be made to the illustration in Figure 92 which represents part of work sheet used by the coder to set up a problem preparatory to placing it in the computer. This work sheet shows the actual position of the information in the required words of the memory.

Consider coding, for example, the I_2 integrator setup such that it operates as follows: It picks up a random *dy* increment on input line 7, designated more particularly as inputs u_7–v_7; it picks up the output of integrator setup I_{49} as a *dx* input; the initial *y* number is +100,000,000; it inspects for a *y* number overflow; there is no sign reversal required of its *dz* output; and its *dz* output is fed out on output line which is composed of outputs Q_7^+–Q_7^-.

As a further requirement of coding, the I_0 integrator setup is to be coded to stop computation, after an interval of the independent variable equal to $X=2$. The scale factor for the *x* variable, S_x, will be considered to be 2^{10} for this case.

The I_2 integrator setup is illustrated on the sample code sheet of Figure 92. As previously shown in Figure 16, it requires three successive word periods to completely code such a single integrator setup. In this instance, words W_0, W_1 and W_2.

In summary, to code integrator setup I_2: the random input code P_0 is located in word W_0; the *dx* and *dy* input codes are located in word W_1; and, the initial *y* number, the random output code P_{26}, the sign reversal pulse for the *dz* output, and the inspect pulse for the *y* number overflow are located in word W_2.

Referring first to word W_0, in order to pick up the proper random input, the decimal 7, in its binary form 0111 is coded with one binary digit in each of the P_0 positions of columns D–2, D–1, Y and R, respectively. (It should be noted that this coding can also be used to pass the *y* number of word W_0 to an external step register by means of output line S_7, if desired.)

Referring next to word W_1, the code position required for integrator setup I_2 picking up its own *dz* output from the Z general storage lines as a *dy* input is obtained from the setting shown for the circular slide rule in Figure 65. There it is shown that since the Z–2 line is to be observed, channel D–2 of word W_1 is provided with a code pulse 1 at position P_{23} to accomplish this.

The code position required to pick up the output of integrator setup I_{49}, from the Z general storage, as a *dx* input, is shown by the same setting of the slide rule to give a solution of position P_{25} in the D–X channel. It should be remembered here that if a code pulse in the D–X channel is to pick up information from the Z–1 storage line, an additional pulse is required in pulse position P_0 of the D–X channel. Thus the code for the *dx* input is a "one" in P_{25} and P_0 of the D–X channel of the W_1 word.

Referring next to word W_2, to fill the initial *y* number into the I_2 integrator setup, a one is needed in the P_{25} position to represent the plus sign. A binary one in position P_{24}, and binary zeros from P_{22} through P_{16} represent the *y* magnitude 100,000,000. A one is filled in P_{15} to serve as the start pulse which determines P_{16} as the least significant digit of the *y* number.

To inspect for *y* overflow, P_1 of the R channel is filled with a digit one. Since no sign reversal pulse is stipulated, the P_{26} position of the R channel is left zero.

Finally, the decimal 7 for selecting the output lines, Q_7^+–Q_7^-, on which to feed out the *dz* outputs, is filled as a binary number 0111 with one of the binary digits in the P_{26} pulse position of each of the D–2, D–1, Y and D–X channels, respectively.

The information thus arranged in the W_0, W_1, and W_2 words represents the I_2 integrator setup. The other integrator setups (not shown) required for obtaining a solution to the assumed problem are then similarly set up in the code sheet.

The I_0 integrator setup, used to control the stopping of the computation at a given interval of the independent variable *x* will next be coded.

Using Equation 3 previously presented, in order to stop computation at regular intervals of $\Delta X=2$, and with $S_x=2^{10}$, the *y* number to be filled in the word W_0 is equal to:

$$2^{24}-\left(\frac{2^{24}}{2\cdot 2^{10}}\right)=2^{24}-2^1$$

This number is set up in the *y* channel by placing a digit 1 in pulse positions P_{14} through P_{25} of word W_0.

After setting up the word sheet as shown in Figure 92 for all the integrator setups required in a particular problem, this information is next filled into corresponding positions of the memory.

Referring to the control box in Figure 91, the operator, after turning on the power switches, taps the ϕ button **347** to place a fill reference pulse in the Z–1 storage line. The word switches **257** and **258** are then set to the word desired, and the main function switch **340** is set to the appropriate channel to be filled. The operator then taps the 0 button **345** or the 1 button **346** in accordance with the digits shown in the channel of the word being filled, as shown in the code sheet in Figure 92. After filling one of the channels, the operator again taps the ϕ button **347**, selects the next channel, and then taps

the 0 and 1 buttons according to the digits on the code sheet.

After filling the memory in this manner, with all the integrator setups required, the generation of the particular solution to the problem is obtained by the operator first setting the main function switch to a computing setting, say C2. The 0 or 1 button is then tapped to initiate the computing routine.

When the r number of integrator I_0 reaches capacity, it emits a pulse which stops the computation. The operator can then read the value of the desired y numbers, in the read-out unit **361**, by setting the word switches **257** and **258** to the desired word and the read switch **357** to the R_y position.

While the circuits as shown and described herein are admirably adapted to fulfill the objects and features of advantage previously enumerated as desirable, it is to be understood that the invention is not to be limited to the specific features shown but that the means and construction herein disclosed are susceptible of modification in form, proportion, and arrangement of parts without departing from the principle involved or sacrificing any of its advantages, and the invention is therefore claimed in embodiments of various forms all coming within the scope of the claims which follow.

What is claimed is:

1. An electronic system for generating a digital process of logic comprising: a plurality of flip-flop circuits each having two stable conductivity states; a pair of output lines for and associated with each respective flip-flop circuit and having high and low potentials in alternation thereon representative of the respective state of the flip-flop; a pair of input circuits for and associated with each respective flip-flop circuit for controlling the states thereof; means for introducing into the system from an outside source information in the form of high and low potentials similar in nature to those on the flip-flop circuit output lines; circuit means for connecting according to a predetermined arrangement at least one output line from each of certain flip-flop circuits and said outside source to establish control lines, one such control line existing for and associated with each of respective one of said input circuits, each of said control lines capable of having a high potential thereon only when the connections to the flip-flop circuits and said output source simultaneously have a high potential thereon; an additional input means connected to a source of recurring clock pulses defining the timing of the system; and separate gating means each controlled by one of said control lines for connecting said clock pulses simultaneously to all of the flip-flop input circuits; whereby each clock pulse results in the status of process of logic being advanced one step in accordance with said predetermined arrangement.

2. An electronic system for generating a digital process of logic comprising: a plurality of flip-flop circuits each having two stable conductivity states; a pair of output lines for and associated with each respective flip-flop circuit and having high and low potentials in alternation thereon representative of a respective state of the flip-flop; a pair of input circuits for and associated with each respective flip-flop circuit, each capable of generating a trigger pulse for controlling one state thereof; circuit means for connecting together, according to a predetermined arrangement, one of the pair of output lines of each of certain flip-flop circuits to establish a first set of control lines; a plurality of input lines controlled by sources outside the system to have high and low potential thereon, representing a second set of control lines; one control line existing for each of said input circuits; each of said first set of control lines having a high potential thereon only when the connected flip-flop circuits are in the proper states; an additional input means connected to a source of recurring clock pulses defining the timing of the system; separate gating means each controlled by a respective one of said control lines for connecting said clock pulses to the flip-flop input circuits; and means for connecting together according to a predetermined arrangement one output line from certain flip-flops to form a line for feeding information out of the system, said latter line having a high potential thereon only during the time the connected flip-flop circuits are in the proper states; whereby each clock pulse results in the status of the process of logic being advanced one step in accordance with said predetermined arrangement.

3. An electrical system for generating a digital process of logic comprising: a separate line for supplying clock pulses, the spacing of said clock pulses defining a basic period of operation for the circuits in the system; a plurality of flip-flop circuits each having two stable conductivity states; a pair of output lines for and associated with each respective flip-flop circuit and having high and low potentials in alternation thereon representative of a respective state of the flip-flop; a pair of input circuits for and associated with each respective flip-flop circuit, each capable of generating, when energized, a trigger pulse for controlling one state thereof; circuit means for connecting together according to a predetermined arrangement one output line from each of certain flip-flop circuits to establish each respective one of a plurality of control lines, each control line having a high potential thereon only when the connected flip-flop circuits are in the proper states; a pair of input lines controlled by a source outside the system for directly energizing the pair of input circuits of one of said flip-flop circuits; and separate gating means, each controlled by one of said control lines for connecting said clock pulses to the input circuits of the remaining flip-flop circuits; whereby on the occurrence of each clock pulse certain flip-flop circuits are triggered in accordance with the high potentials on said control lines, thereby rendering other control lines operative to have high potentials thereon so that other flip-flop circuits are triggered on the occurrence of the next clock pulse.

4. A readout circuit for a computer of the class described comprising: a revolving drum provided with a channel of timing signals and a channel of digital information signals extending about the periphery thereof; a first digital storage means having a capacity equal to a group of digits to be simultaneously read out of said channel; means for sequentially stepping the information signals in said channel through said first storage means each revolution of said drum; means responsive to said timing signal channel for providing timing signals indicative of at least the last digit position of each group of signals sensed from said information channel; a counter for indicating counts assigned to each group of digits to be fed out; means for generating a transfer signal at the coincidence of a particular count with the last digit signal in a group; a second digital storage having a capacity equal to said first digital storage; gating means controlled by said transfer signal for transferring the digits in said first storage means to said second storage means; a readout unit having sets of visual indicators corresponding to the groups of digits in said channel; means for energizing one of said sets of visual indicators as selected by said counter in accordance with the contents in said second storage means; and means responsive to said transfer signal for triggering said counter to indicate the count assigned to the next group of digits to be read out; whereby successive groups of digits in said channel are visually indicated on said readout unit each revolution of said drum.

5. A circuit for feeding a function rate output from a digital integrating device comprising an electronic means having a cyclical operation for successively operating on a plurality of integrator setups recorded on the surface of a rotating drum memory, said circuit comprising: a plurality of pairs of output lines, one line in each pair for receiving positive output increments

and the other for receiving negative output increments generated by said electronic means; a recording of code number signals in certain integrator setups on said drum, one of said code number signals corresponding to each of said pairs of output lines; a code number register having output lines indicative of the number signal content therein; circuit means for transferring a code number signal from one of said integrator setups into said register; a flip-flop circuit; means for setting said flip-flop circuit at the beginning of each cycle operation of said electronic means in accordance with the function output increment generated for said integrator setup during the previous revolution of said drum; and means for feeding an output increment on one of the output lines of the pair corresponding to the code number signal contained in said register when the output increment generated at the end of a cycle corresponds to said previous output increment.

6. A device for solving differential equations by a digital process which comprises: a rotatable member having a magnetizable periphery; a timing signal channel extending circumferentially about the periphery of said member having signals thereon whose spacing defines the period of a unit of information signal; information signal channels extending about the periphery of said member having coded information signals thereon which have a timed relation with signals on said timing signal channel; means responsve to signals in said timing signal channel for subdividing said information signal channels into arcuate sections having a length equal to a fixed number of timing signal periods, the number of arcuate sections provided being larger than the number of timing signal periods in each section; a first and second general storage channel extending about the periphery of said member; pickoff means positioned to electrically sense information signals appearing in said information signal channels and said first and second general storage channels; recording means for each of said information signal channels positioned to record information signals thereon such that signals in all the arcuate sections provided are continually sensed in order by corresponding pickoff means; recording means for said general storage channels positioned to record information signals thereon substantially two arcuate sections ahead of respective pickoff means; electronic means having a cyclical operation coextensive with each of said arcuate sections for responding to information signals in all said pickoff means during each timing signal period to thereby generate the signals to be recorded by respective recording means back onto the information signal channels of said rotating member, said electronic means operating further to generate at the end of each of said cyclical operations an output signal to be recorded by respective recording means into both of said general storage channels; said general storage channels being so operated that all the output signals generated during the previous revolution of said member are sensed by the pickoff means associated with said general storage channels each cyclical operation of said electronic means according to a predetermined arrangement, the output signals generated at the end of even numbered cyclical operations of said electronic means being sensed by the pickoff means associated with one of said general storage channels, and the output signals generated at the end of the odd numbered cyclical operations of said electronic means being sensed by the pickoff means associated with the other of said general storage channels.

7. An electronic computer of the class described comprising: a rotating magnetic storage medium having a magnetizable periphery for storing information on successive portions thereof in the form of flux signals, each said portion capable of including a group of coded pickup signals; a reading device operatively associated with the periphery of said medium; an electronic means including a selector register responsive to said reading device; a plurality of signal sources, each corresponding to a respective one of said groups of coded pickup signals, and said signal sources each supplying respective information signals from external source means at a recurrence rate not in excess of once per revolution of said storage medium; means including timing means, and gating circuit means controlled by said timing means and said selector register and capable of connecting any of said sources to said electronic means, said electronic means being responsive in succession to successive ones of said groups of coded pickup signals, to set the selector register uniquely for each respective group of pickup signals to cause said gating circuit means to pass signals from respective ones of said sources in successive order into said electronic means.

8. An electronic computer of the class described comprising: a rotating magnetic storage medium having a magnetizable periphery for storing information on successive portions thereof in the form of flux signals, each said portion capable of including coded pickup signals; a reading device operatively associated with the periphery of said medium; and electronic means responsive to said reading device, said electronic means including a selector register, a pair of storage flip-flops and a counter; a plurality of pairs of input lines for supplying respective information signals from respective external sources, each at a rate not greater than one signal per revolution of said storage medium; a plurality of pairs of input flip-flops, each of said pair of input flip-flops directly responsive to the signals on a respective pair of input lines; gating means for connecting all of said pairs of input flip-flops to said pair of storage flip-flops, said electronic means operable to set said selector register in accordance with the coded pickup signals as sensed by said reading device in successive portions of the periphery of said storage medium to enable said gating means to introduce signals from a selected pair of input flip-flops into said pair of storage flip-flops; circuit means for successively resetting said counter in accordance with the contents of said storage flip-flops; and means responsive to the count output of said counter to alter the information signals successively recorded on said storage medium.

9. Means for introducing input increments from a source outside the system into a digital computing device having a rotating drum memory comprising: a plurality of pairs of input lines, one line in each pair for receiving positive increment signals from an outside source and the other for receiving negative increment signals from said outside source; means for recording information signals and code number signals on successive arcuate sections of the periphery of said drum; a code number register associated with said drum having output lines indicative of the content therein, one of said code numbers corresponding to each of said pair of input lines; circuit means for transferring said code number signals from an arcuate section of said drum into said register; a pair of storage flip-flop circuits; a plurality of coincident circuit means for connecting each pair of input lines to the inputs of said storage flip-flop circuits in accordance with the content of said code number register; an increment counter; means for resetting said increment counter in accordance with the signals in said pair of storage flip-flop circuits during each period defined by an arcuate section of the rotating drum; and means responsive to the count output of said increment counter to alter information signals on said drum.

10. A circuit for introducing signals representing increments of a dependent variable from one of a plurality of outside signal sources into a digital integrating system while a calculation is being performed, said integrating system characterized by having signals representing a plurality of integrator setups serially recorded on the surface of a rotating drum memory, comprising, in combination with said memory: time-period demarking means in-

cluding a clock channel recorded on the drum memory and counting means responsive to the clock channel recording for producing time signals for timing operations of the system, the timing means producing signals corresponding to the signal positions of each of the integrator setups recorded on the memory; a plurality of external signal lines each connected to a respective one of said outside sources; said drum memory having a code number signal corresponding to one of said outside sources capable of being included in each of the integrator setups recorded thereon; a code number register having output lines indicating the number content therein, said lines including one for each of said code number signals; means serially responsive to signals on the rotating drum memory and said timing means for transferring a code number signal sensed from an integrator setup into said register; a coincidence circuit connected to said register output lines and to said external signal lines to provide an output signal incident to coincidence of an external line signal and a register output line signal on correspondingly assigned lines, said signal representing an increment of the dependent variable being introduced into the system from the outside source; means for temporarily storing said signal; a dependent variable storage device; means for setting said storage device in accordance with said stored signal; and means responsive to the output of said storage device to alter the integrator setups on said drum memory.

11. An electronic system for generating a digital process of logic comprising: a memory for storing binary data signals and clock signals; a plurality of flip-flop circuits each having two stable conductivity states; a pair of output lines for and associated with each flip-flop circuit and having high and low potentials in alternation thereon representative of the respective state of the flip-flop; a pair of input circuits for and associated with each respective flip-flop circuit, each capable when energized to trigger the flip-flop to a respective one of its states; a plurality of input lines controlled by sources outside the system, each said input line having a high or low potential thereon representative of the state of the respective source; circuit means for connecting together, according to a predetermined logical arrangement, the output lines of each of certain flip-flop circuits and the input lines from said sources to establish each respective one of a plurality of control lines, each of said control lines having a high potential thereon only when the connected flip-flop circuits and sources are in the proper states; means responsive to clock signals stored by said memory for providing recurring clock signals defining the timing of the system; other means responsive to data signals stored by said memory for energizing the input circuits of at least one of said flip-flop circuits; and gating means each controlled by a respective one of said control lines having a high potential thereon to connect a clock signal to energize a respective input circuit of the remaining flip-flop circuits; whereby each clock signal provided by said memory results in the status of the process of logic being advanced one step in accordance with said predetermined logical arangement.

12. An electronic system for generating a cyclical digital process of logic comprising: a memory for storing binary information signals and clock signals; a plurality of flip-flop circuits each having two stable conductivity states; a pair of output lines for and associated with each flip-flop circuit having high and low potentials in alternation thereon representative of a respective state of the flip-flop; a pair of input circuits for and associated with each flip-flop circuit, each capable when energized to trigger the flip-flop to a respective one of its states; a plurality of input lines controlled by sources outside the system, each said input line having a high or low potential thereon representative of the state of the respective source; logical circuit means for connecting together, according to a predetermined arrangement, the output lines of each of certain flip-flop circuits and the input lines from said sources to establish each of a plurality of control lines, each of said control lines having a high potential thereon only when the connected flip-flop circuits and sources are in the proper states; means responsive to the clock signals stored by said memory for providing recurring clock signals defining the timing of the system; other means responsive to data signals stored by said memory for successively energizing the input circuits of at least one of said flip-flop circuits; and separate gating means each controlled by a respective one of said control lines having a high potential thereon for connecting said clock signals to energize the input circuits of selected ones of the remaining flip-flop circuits; whereby each clock signal provided by said memory results in the status of the process of logic being advanced one step in accordince with said predetermined logical arrangement, and wherein certain of said flip-flops are interconnected and arranged to operate as a cyclical counter responsive to said clock signals to provide count signals for indicating the steps of the cycle through which the digital process of logic advances.

13. An electronic system for generating a digital process of logic comprising: a memory for storing binary data signals and clock signals; a plurality of bistable state elements; a pair of output lines for and associated with each bistable state element having high and low potentials in alternation thereon representative of a respective state of the bistable state element; a pair of input circuits for and associated with each bistable state element, each capable when energized to trigger said element to a respective one of its states; a plurality of input lines controlled by sources outside the system, each said input line having a high or low potential thereon representative of the state of the respective source; circuit means for connecting together, according to a predetermined logical arrangement, the output lines of each of certain bistable state elements and the input lines from said sources to establish each respective one of a plurality of control lines, each of said control lines having a high potential thereon only when the connected bistable state elements and sources are in the proper states; reading means responsive to clock signals stored by said memory for supplying recurring electrical clock signals defining the timing of the system; other reading means responsive to data signals stored by said memory for supplying signals for energizing the input circuits of at least one of said bistable state elements; separate gating means each controlled by a respective one of said control lines for connecting said clock signals to energize the input circuits of the remaining bistable state elements; and other means for connecting together, according to a predetermined arrangement, the output lines of each of certain bistable state elements to form each respective one of a plurality of lines for feeding information out of the system, said latter lines each having a high potential thereon only during the time the connected bistable state elements are in the proper states; whereby in response to a clock signal provided by said memory certain bistable state circuits are triggered in accordance with high potentials on certain of said control lines thereby changing which of said control lines have high potentials thereon so that different bistable state circuits can be triggered in response to the next clock signal.

14. An electronic system for generating a digital process of logic comprising: timing means for providing recurring clock signals defining the timing of the system; a memory included in said system for storing binary data signals and operating in synchronism with said clock signals; a plurality of flip-flop circuits each having two stable conductivity states; a pair of output leads for and associated with each flip-flop circuit having high and low potentials in alternation thereon representative of a respective state of the flip-flop; a pair of input circuits for

and associated with each flip-flop circuit, each capable when energized to trigger the flip-flop to a respective one of its states; an input lead controlled by a source outside the system to have high and low potentials thereon; circuit means for connecting together, according to a predetermined arrangement, output leads of each of certain flip-flop circuits and the input lead from said source to establish each of a plurality of control leads, each said control lead having an effective potential thereon only when the connected flip-flop circuits and sources are in the proper states; reading means responsive to data signals stored on said memory for supplying signals for energizing the input circuits of at least one of said flip-flop circuits; separate gating means each controlled by a respective one of said control leads for connecting said clock signals to the input circuits of other flip-flop circuits; other means for connecting together according to a further predetermined arrangement the output leads of each of certain flip-flops to form a lead providing output data signals; and means for storing said output data signals in said memory; whereby each clock signal provided by said timing means results in the status of the process of logic being advanced in accordance with said predetermined arrangements.

15. A system for solving differential equations by a digital process which comprises: a rotating member having a magnetizable periphery; a timing signal channel extending circumferentially about the periphery of said member operable to provide a source of timing signals whose spacing defines the period of a unit of information; a first set of record heads positioned to record magnetic flux signal patterns representing digital integrator setups and code information in information channels extending circumferentially about the periphery of said member; means responsive to said timing signal channels for subdividing said information channels into successive arcuate sections each having a length equal to a fixed number of timing signal periods, the number of arcuate sections provided being larger than the number of timing signal periods in each section; a second set of record heads positioned to record flux signal patterns into a first and second general storage channel extending about the periphery of said member; pickoff heads spaced from said respective record heads and positioned to sense signals in each of said channels including said general storage channels; electronic means interconnecting the spaced heads on respective channels to thereby provide information delay devices, the delay device so provided for each said information channels being of sufficient length to contain the information in all of said arcuate sections, the delay device so provided for said first general storage channel being one timing signal period longer in length than two arcuate sections and the delay device so provided for said second general storage channel being one timing signal period shorter in length than two arcuate sections, said electronic means having a cyclical operation coextensive with an arcuate section during which each cyclical operation said electronic means responds to all said pickoff heads to generate signals representing an altered digital integrator setup for recording back on the information channels of said member by said first set of record heads, and said electronic means further operates to generate at the end of each said cyclical operation an integrator output signal for simultaneously recording on the general storage channels of said member by said second set of record heads; said electronic means including means operable during a cyclical operation of the computation for introducing signals representing increments of a dependent variable directly into said electronic means from one of a plurality of sources, outside the system, selected to be gated in accordance with code information provided in an arcuate section of said member; and said electronic means including further means operable during a cyclical operation of the computation for routing an integrator output signal generated by said electronic means, onto one of a plurality of lines, leading out of the system, selected to be gated in accordance with further code information provided in an arcuate section of said member.

16. Apparatus in accordance with claim 15 wherein said information channels for said arcuate sections include: a channel for storing signals in locations thereof corresponding to the locations of signals in one of the general storage channels representing increments of a dependent variable; another channel for storing signals in locations thereof corresponding to the locations of signals in the other general storage channel representing increments of a dependent variable; and still another channel for storing signals in locations thereof corresponding to the locations of signals in a selected one of said general storage channels representing increments of an independent variable, whereby the respective pickup heads for said channels feed into said electronic means during each cyclical operation of the computation signals representing increments of a dependent variable and signals representing increments of an independent variable as derived from said general storage channels.

References Cited in the file of this patent

UNITED STATES PATENTS

2,435,841	Morton	Feb. 10, 1948
2,476,066	Rochester	July 12, 1949
2,521,350	Dickinson	Sept. 5, 1950
2,539,623	Heising	Jan. 30, 1951
2,540,654	Cohen	Feb. 6, 1951
2,549,071	Dusek	Apr. 17, 1951
2,557,729	Eckert	June 19, 1951
2,564,403	May	Aug. 14, 1951
2,568,932	Rajchman	Sept. 25, 1951
2,575,342	Gridley	Nov. 20, 1951
2,584,811	Phelps	Feb. 5, 1952
2,585,630	Crosman	Feb. 12, 1952
2,587,532	Schmidt	Feb. 26, 1952
2,594,731	Connolly	Apr. 29, 1952
2,595,378	Den Hertog	May 6, 1952
2,603,123	Loukomsky	July 15, 1952
2,609,143	Stibitz	Sept. 1, 1952
2,609,439	Marshall	Sept. 2, 1952
2,611,813	Sharpless	Sept. 23, 1952
2,614,169	Cohen	Oct. 14, 1952
2,617,704	Mallina	Nov. 11, 1952
2,632,845	Goldberg	Mar. 24, 1953
2,636,133	Hussey	Apr. 21, 1953
2,637,812	Hagen	May 5, 1953
2,644,867	Wolfe	July 7, 1953
2,652,554	Williams	Sept. 15, 1953
2,655,598	Eckert	Oct. 13, 1953
2,675,427	Newby	Apr. 13, 1954
2,680,239	Daniels	June 1, 1954
2,700,504	Thomas	Jan. 25, 1955
2,700,755	Burkhart	Jan. 25, 1955
2,701,095	Stibitz	Feb. 1, 1955
2,703,202	Cartwright	Mar. 1, 1955
2,719,670	Jacobs	Oct. 4, 1955
2,737,342	Nelson	Mar. 6, 1956
2,739,299	Burkhart	Mar. 20, 1956
2,749,037	Stibitz	June 5, 1956
2,775,402	Weiss	Dec. 25, 1956
2,787,416	Hansen	Apr. 2, 1957
2,796,218	Tootill et al.	June 18, 1957
2,815,168	Zukin	Dec. 3, 1957
2,841,328	Steele et al.	July 1, 1958
2,850,232	Hagen et al.	Sept. 2, 1958

FOREIGN PATENTS

975,941	France	Oct. 17, 1950

(Other references on following page)

OTHER REFERENCES

Proc. of the IRE, "Electronic Computing Circuits of the ENIAC," by Burks (pages 756 to 767), August 1947.

Proc. of the National Electronics Conference, "Storage of Numbers on Magnetic Tape" by J. M. Coombs (pages 201 to 210), vol. III, Mar. 17, 1948.

Progress Report No. 2 Computation Lab., Harvard University (pages II–5 to II–9, total of five (5) pages), Nov. 10, 1958.

Fifth (5th) Interim Progress Report on the Physical Realization of an Electronic Computing Instrument, IAS Princeton (pages VIII–1 to VIII–6; block diagram opposite page VIII–3), January 1949.

"24 Digit Computer With Magnetic Drum Memory," by ERA dated Feb. 15, 1949, submitted by Engrg. Research Assoc., Inc. (pages 23 to 33, 65, 66, and 68; drawings Figures 3.2–2, 3.3–5, 3.3–11, 3.4–4, B–1, D–1).

Electronic Engineering, "A Magnetic Digital Storage System," by A. D. Booth (pages 234 through 238), July 1949.

Proc. of IRE, "Diode Coincidence and Mixing Circuits in Digital Computers," by Chen (pages 511 to 514, total of four (4) pages) May 1950.

"High Speed Computing Devices," by ERA, McGraw-Hill (pages 289 to 293). Copyright July 1950.

UNITED STATES PATENT OFFICE

Certificate of Correction

Patent No. 2,924,381 February 9, 1960

Donald E. Eckdahl et al.

It is hereby certified that error appears in the printed specification of the above numbered patent requiring correction and that the said Letters Patent should read as corrected below.

Column 11, line 75, for "121" read —131—; column 12, line 36, for "fisrt" read —first—; column 23, line 3, for "W_{i-2}" read —W_{i-1}—; column 25, line 7, for "otheriwse" read —otherwise—; column 28, line 75, for "designed" read —designated—; column 30, line 3, for "A_5," read —A_5',—; line 32, for "P_2" read —P_2'—; column 38, line 33, for "B_3'" read —B_3—; column 40, line 44, for "are b" read —are b_6—; column 42, line 38, after "Note" insert —that—; column 45, line 30, before "to obtain" insert —This equation can be revised by rules of symbolic logic—; same column 45, line 31, for "R_s" read —R_s'—; column 46, lines 10 and 11, for "complemetend" read —complemented—; line 41, after "along" insert —with—; column 53, line 29, after "of" insert —the—; line 38, for "348", second occurrence, read —349—; column 58, line 70, for "F_R'" read —F_y'—; column 59, line 9, for "$Y_1F_yS_1'$" read —$Y_1F_y'S_1'$—; line 30, for "D_3'" read —G_3'—; column 60, line 32, for "persents" read —presents—; column 63, line 39, for "reolution" read —revolution—; line 72, for "of" read —for—; column 64, line 37, for "ouputs" read —outputs—; column 66, line 21, for "therto" read —thereto—; line 50, for "G_3" read —G3—; column 67, line 11, for "equavilent" read —equivalent—; column 70, line 31, for "P_{22}" read —P_{23}—; line 55, for "—2^1" read ——2^{13}—; column 71, line 42, strike out "of", first occurrence; column 76, line 17, for "accordince" read —accordance—.

Signed and sealed this 23rd day of August 1960.

[SEAL]

Attest:

KARL H. AXLINE,
Attesting Officer.

ROBERT C. WATSON,
Commissioner of Patents.