5-1-1979

An electronic switching system: a computer model for traffic study

Richard McInnes

Follow this and additional works at: http://scholarworks.rit.edu/theses

Recommended Citation

This Thesis is brought to you for free and open access by the Thesis/Dissertation Collections at RIT Scholar Works. It has been accepted for inclusion in Theses by an authorized administrator of RIT Scholar Works. For more information, please contact ritscholarworks@rit.edu.
AN ELECTRONIC SWITCHING SYSTEM:
A COMPUTER MODEL FOR TRAFFIC STUDY

by

Richard D. McInnes

A Thesis Submitted
in
Partial Fulfillment
of the
Requirements for the Degree of
MASTER OF SCIENCE
in
Electrical Engineering

Approved by:

Prof. George A. Brown
(Thesis Advisor)

Prof. Name Illegible

Prof. Harvey E. Rhody
(Department Head)

DEPARTMENT OF ELECTRICAL ENGINEERING
COLLEGE OF ENGINEERING
ROCHESTER INSTITUTE OF TECHNOLOGY
ROCHESTER, NEW YORK
MAY, 1979
ACKNOWLEDGMENTS

I would like to thank Professor George A. Brown for his guidance and encouragement, Ms. Paula Victor for her long hours of typing, and my wife, Barbara, for her patience and understanding.
Abstract

When designing a telephone switching system, it is necessary to ensure that the system will be able to handle a prescribed quantity of traffic (telephone calls); both at any particular instant and during a given period of time. By modelling the switching system with a digital computer, it is possible to determine the traffic handling capability of the system prior to investing in a prototype. In this manner, the system architecture can be altered by merely changing software rather than by rearranging the hardware in a prototype. The savings in terms of time and money provide the stimulus for taking this type of approach.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Acknowledgments</td>
<td>i</td>
</tr>
<tr>
<td>Abstract</td>
<td>ii</td>
</tr>
<tr>
<td>List of Tables</td>
<td>iv</td>
</tr>
<tr>
<td>List of Figures</td>
<td>v</td>
</tr>
<tr>
<td>I. Introduction</td>
<td>2</td>
</tr>
<tr>
<td>II. Call Switching Matrix</td>
<td>6</td>
</tr>
<tr>
<td>III. Path Search Algorithm</td>
<td>10</td>
</tr>
<tr>
<td>IV. Program Operation</td>
<td>13</td>
</tr>
<tr>
<td>V. Program Utilization</td>
<td>33</td>
</tr>
<tr>
<td>VI. Summary</td>
<td>39</td>
</tr>
<tr>
<td>Bibliography</td>
<td>40</td>
</tr>
</tbody>
</table>
Table 1: Sample Statistics for 100 Call Attempts

35
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A Six-by-six Crosspoint Switch</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>A 4096 x 4096 Matrix</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>Schematic Representation of a Three Stage Matrix</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>Schematic Representation of a Folded Three Stage Matrix</td>
<td>5</td>
</tr>
<tr>
<td>5</td>
<td>One Grid Layout</td>
<td>7</td>
</tr>
<tr>
<td>6</td>
<td>Three Stage Matrix (Shown Schematically) Showing Redundant Paths</td>
<td>9</td>
</tr>
<tr>
<td>7</td>
<td>Matrix Simulation Program Flowchart</td>
<td>13</td>
</tr>
</tbody>
</table>
I. INTRODUCTION

The program described in this paper simulates the call switching matrix in a 4096 line telephone central office. Any of the 4096 lines can be connected to any other line with a maximum of 128 connections at any given time. The matrix consists of individual solid state switches called crosspoints which can be closed to complete an electrical circuit. These crosspoints are arranged in a grid as shown in Figure 1.

![Crosspoint Switch Diagram](image)

**Figure 1**
A Six-by-six Crosspoint Switch

- **Inlets**
  - A
  - B
  - C
  - D
  - E
  - F

- **Outlets**
  - 1
  - 2
  - 3
  - 4
  - 5
  - 6

- **Crosspoint**
  - Open Crosspoint
  - Closed Crosspoint
The six-by-six crosspoint in Fig. 1 has 36 individual crosspoints. Of these 36 crosspoints, 33 are open and three are closed. In this case, the closed crosspoints are designated (B,5),(D,4), and (E,1) and connect inlets B,D, and E to outlets 5,4, and 1 respectively. Any inlet can be connected to any outlet. The matrix is a bidirectional device. The inlets could have been 1 to 6 and the outlets A to F; it does not matter. It is the concept that is important.

It would seem that, if this system would work for a single six-by-six matrix, it would also work for a 4096 by 4096 matrix. Each line would have two appearances -- one as an inlet and one as an outlet as shown in Figure 2. As an added benefit, this system would permit a maximum of 4096/2 or 2048 simultaneous connections. Everyone could be using the phone at the same time!

![Figure 2](A 4096 x 4096 Matrix)
At first glance, this system seems ideal. However, this is not the case. To implement such a system would require a 4096-by-4096 matrix. This matrix would have 4096 x 4096 or 16,777,216 crosspoints. The size and cost would be prohibitive. The possibility of crosspoint failure leads to another drawback. If, for example, the (1,3) crosspoint were to fail, 1 could not call 3. This makes 100 per cent reliability a requirement of the system. Such reliability is not available at any price.

It is obviously not necessary for 100 per cent of the telephones to be in use at any one time. Normally, less than 20 per cent are in use at any one time. If 50 phones are in use, there are 50/2 or 25 connections. This indicates that there need be only half as many paths or ways to get through the switching matrix as there are telephones in use.

By using a multistage matrix, it is possible to decrease the cost of the connection matrix by using fewer crosspoints and at the same time improve its reliability. A three stage matrix is shown schematically in Figure 3 on the following page.
All of the inlets are at the left side of the diagram and all of the outlets are at the right. A path from an inlet to an outlet starts at the left and traces from left to right through the A, B, and C stages. It must be pointed out that a path does not actually exist in the matrix until the appropriate crosspoints are closed.

A telephone must be able to both make (originate) a call and receive (terminate) a call; therefore, each line must have two appearances -- one as an inlet and one as an outlet. This leads to the concept of the folded matrix shown schematically in Figure 4. The set of inlets is identical with the set of outlets. These appearances are called ports.
II. CALL SWITCHING MATRIX

The A stage of the three stage matrix for which the program in this paper was written consists of 128 A switches, each with 32 inlets and four (4) outlets. The B stage has 64 B switches, each with eight (8) inlets and four (4) outlets; and the C stage made up of 256 C switches, each with 16 inlets and 16 outlets. Each group of eight A switches and four B switches comprise a Grid. There are 16 Grids in a 4096 line switching system. One Grid and its connections to the C stage are shown in Figure 5 on the following page. The B and C stages are connected in the same fashion as the A and B stages. The ports are numbered from 0 to 31 to facilitate their representation in binary.

The inlets in each switch are referred to as columns and the outlets as rows. The outlets of an A switch are A rows and the inlets of a B switch are B columns.

Note in Figure 5 that row 3 from A stage, switch 0 goes to B stage, column 0 on B switch 3. (Recall that an A switch has four rows or outlets.) The matrix is designed so that the A row number is always the same as the B switch number and the A switch number is always the same as the B column number. This greatly simplifies the task of finding and setting up a path through the matrix.
Figure 5
One Grid Layout
The three stage matrix system makes it possible to construct the paths in steps rather than all at once as in the matrix pictured in Figure 2. Six steps are required in a three stage system - three to get from left to right and three more to get back from right to left. The six step path construction provides great flexibility because there are many possible variations of each step. This flexibility allows the small three stage matrix to do the work of a much larger one stage matrix. Reliability of the matrix is greatly increased because it is possible to connect any two ports with several different paths as shown in Figure 6 on the following page. Thus, if a crosspoint which forms part of one path fails, it is possible to use another path.
Figure 6
Three Stage Matrix
(Shown Partially)
Showing Redundant Paths
III. PATH SEARCH ALGORITHM

The path search begins with the loading of the originating and terminating equipment numbers into the path search register. The originating grid, originating A stage switch and A row zero are compared with the contents of the call store memory. If there is a match, it means that zero is busy and A row one is then tried in the same manner. If A row one is busy, A row two is tried and then, if necessary, A row three is tried. The call is blocked if A row three is busy.

When a free A row is found, the B stage switch is known. (A row equals B stage switch.) The B row is found by comparing the originating grid, originating A switch, and B row zero to the contents of the call store. If B row zero is busy, one is added to the B row and the new B row is tested for busy. If B row 3 is busy, one is added to the A row and the tests are repeated.

When a free B row is found, the C stage switch is known. (B row equals C stage switch.) The C row is found by comparing the originating grid, originating A switch, originating A row, originating B row, and C row zero to the contents of the call store. If the C row is busy, one is added to it and a new test is made. If C row 15 is busy, one is added to the B row and new tests are made on the B row and then on the C rows again until a free C row is found.

When the C row is found, the originating half of the
path is complete. It is not necessary to search for the terminating half of the path because the multistage matrix is designed so that there is only one way to get to any given terminating equipment number from any given originating C column.

The terminating C row equals the originating C switch and the terminating C switch equals the originating C row except when the originating C row equals the originating C switch. Then the terminating C switch equals the terminating C row. The terminating C switch equals the originating C switch plus one when the originating C switch is even or it equals the originating C switch minus one when the originating C switch is odd.

Path Search Sequence

1. Find originating A row (0 - 3)
   Then originating B switch = originating A row.

2. Find originating B row (0 - 3)
   Then originating C switch = originating B row.

3. Find originating C row (0 - 15)
   Then terminating C switch = originating C row
   And terminating C row = originating C switch
   (if originating C row ≠ originating C switch)

   OR terminating C row = terminating C switch

   And terminating C switch = originating C switch + 1
   (if originating C switch is even)

   OR terminating C switch = originating C switch - 1
   (if originating C switch is odd)

   In this manner, the entire terminating path is
known. Now all that remains is to check for busy on several parts of the terminating path. The path search checks for busy on the terminating B row. If the terminating B row is busy, one is added to the C row and the appropriate busy tests are then made. The terminating A row is checked for busy in an identical manner.

One last busy check is required: If the originating and terminating lines are on the same A switch, they might be connected to the same A row at this stage of the path search algorithm. The originating and terminating grids, A switches and A rows are compared. If they are equal, one is added to the originating C row and new busy tests are made. This concludes the path search.

Calls can be blocked in three ways. The first occurs when the terminating number already has a call, i.e. it is busy. The second occurs when the matrix contains calls whose paths block all possible paths between the new originating and terminating numbers. This call is referred to as being blocked. The third way a call can be blocked occurs when the new call is the 129th call in the matrix simultaneously. The call processing circuitry in the matrix is designed to hold a maximum of 128 calls simultaneously due to memory constraints. Thus, the 129th call is blocked because all timeslots are full.
IV. PROGRAM OPERATION

The operation of the Matrix Simulation Program is illustrated in Figure 7 below.

![Matrix Simulation Program Flowchart]

Figure 7
Matrix Simulation Program Flowchart
Introduction:

Lines 10 - 190:

The beginning of the program contains the necessary declaration statements and provides an explanation of its use. Descriptions of the two modes, manual and random, are printed.

Lines 200 - 220:

This is followed by initialization of the Call Store (M) memory matrix and setting the number of calls in the matrix (N) to zero.

Lines 230 - 330:

The operator is next asked to choose the mode of call programming. He responds by assigning a value to a mode select variable (Q0$) which is then tested and the program branches to the desired mode, manual or random simulation, or to the conclusion of the simulation.

Manual Mode:

Lines 340 - 390:

In the manual mode, the memory timeslot (T), number of calls attempted (Al), number of calls busy (Bl), and number of calls blocked (Kl) are initialized to zero.
Lines 400 - 470:

The operator is told how many calls are currently in the matrix (N) and is asked to assign a variable (Q1$) expressing his desire to add another call or finish the simulation. This variable (Q1$) is tested and the program branches to the appropriate location.

Lines 480 - 620:

The operator is asked to type in the Originating Equipment Number (N1) which is then broken down into Originating Grid (A), Originating A switch (B), and Originating A column (C).

Lines 630 - 690:

The Call Store memory matrix (M) is searched to determine whether the Originating Equipment Number (N1) is busy or idle.

Lines 700 - 770:

If it is busy, the operator is informed and asked to assign a variable (H0$) whose value depends on whether he wants to terminate the call or continue. The program then branches to the appropriate location.

Lines 780 - 810:

If the call is to be terminated, the idle-busy bit
(M(13, WD) is set to zero, the number of calls in the matrix (N) is decremented, and control jumps to line 480.

Lines 820 - 990:

The program asks for the Terminating Equipment Number (N2). It is compared to the Originating Equipment Number (N1) and, if they are equal, the busy call counter (Bl) and attempted call counter (A1) are incremented. The operator is informed that the originating equipment number (N1) is equal to the terminating equipment number (N2). Control then passes to line 480.

Lines 1000 - 1030:

If the Originating Equipment Number (N1) is not equal to the Terminating Equipment Number (N2), the Terminating Equipment Number (N2) is broken down into Terminating Grid (J), Terminating A switch (K), and Terminating A Column (L).

Lines 1040 - 1090:

The program searches the Call Store memory matrix (M) to determine whether the Terminating Equipment Number (N2) is idle or busy.

Lines 1100 - 1210:

If the Terminating Equipment Number (N2) is busy,
the busy call counter (Bl) and attempted call counter (Al) are incremented. The operator is told that the equipment number is busy and is asked to assign a variable (Hl$) telling whether he wants to terminate the call or continue. If it is idle, control passes to line 1260.

Lines 1220 - 1250:

If the call is to be terminated, the idle-busy git (M(13,XD) is set to zero and the number of calls in the matrix (N) is decremented. Control then jumps to line 430. If the call is not to be terminated, control jumps directly to line 480.

Lines 1260 - 1320:

The calls attempted counter (Al) and timeslot (T) are incremented. If the timeslot (T) is less than 129, control jumps to line 2130; otherwise, 128 is subtracted from the timeslot (T) and it is tested again until it is less than 129.

Random Mode:
Lines 1330 - 1400:

In the Random Mode, the operator is asked to enter the number of calls to be attempted (Al) and the average number of calls to be in the matrix at any time (E2).
Lines 1410 - 1470:

The calls attempted (Al), calls busy (Bl), calls blocked (Kl), and all timeslots full (Fl) counters are initialized at zero and the always add call flag (A3) is set to one. If the value of the calls attempted counter (Al) is greater than or equal to the number of calls to be attempted (El), control jumps to line 2960.

Lines 1480 - 1500:

If the number of calls in the matrix (N) is less than the average number of calls to be in the matrix (E2), control jumps around a statement which sets the always add flag (A3) to zero. Then, if the flag (A3) is equal to one, control jumps to line 1710 where a call is attempted.

Lines 1510 - 1570:

A variable (A4) takes on a random value between zero and one; exclusive of both. If that variable (A4) is less than 0.5, control jumps to line 1560 where the program tests to see if there are any calls in the matrix. If there are no calls, control jumps to line 1710 to add a call. If there is a call, it is deleted in the next block of the program. If the variable (A4) is greater than 0.5, control jumps to line 1710 where a call is attempted. Otherwise
(when $A_4 = 0.5$), control jumps back to line 1510 and a new random value is assigned.

Lines 1580 - 1700:

The calls in matrix counter $(N)$ is decremented and the delete timeslot counter $(V_l)$ is initialized to zero. The delete timeslot $(V)$ takes on a random integer value between one and 128, inclusive. Then the delete timeslot counter $(V_l)$ is incremented and tested. If it $(V_l)$ is greater than 128, control jumps to line 1470. Otherwise, the idle-busy flag $(M(13,V))$ is tested for busy condition - that is equal to one. If it $(M(13,V))$ is equal to one, control jumps to line 1690 where the flag $(M(13,V))$ is set to zero; then back to line 1510. If the flag $(M(13,V))$ is equal to zero - idle condition, the delete timeslot $(V)$ is incremented. If the timeslot $(V)$ is less than 129, control jumps back to line 1620; otherwise, 128 is subtracted from $(V)$ and then control jumps back to line 1620.

Lines 1710 - 1860:

The calls attempted counter $(A_l)$ is incremented and the add timeslot counter $(U_l)$ is initialized to zero. The add timeslot $(U)$ takes on a random integer value between one and 128, inclusive. Next, the add timeslot counter $(U_l)$
is incremented and tested. If it (Ul) is less than or equal to 128, control jumps to line 1810 where the idle-busy flag (M(13,U)) is tested for idle condition - that is equal to zero. If the flag (M(13,U)) is equal to zero, control jumps to line 1860 where the value of the add timeslot (U) is assigned to the path search timeslot (T). Otherwise, the add timeslot (U) is incremented and tested to determine if it is less than 129. If so, control jumps back to line 1760 where the add timeslot counter (Ul) is incremented. If the add timeslot (U) is not less than 129, then 129 is subtracted and it is tested again. If the add timeslot counter (Ul) is greater than 128, the number of times all timeslots are full counter (Fl) is incremented and control jumps back to line 1560.

Lines 1870 - 1960:

A random integer value between one and 4096, inclusive is assigned to the originating equipment number (Nl). The originating equipment number (Nl) is then broken down into originating grid (A), originating A switch (B), and originating A column (C). The originating equipment number (Nl) is compared to all equipment numbers in the matrix. If it (Nl) is busy, control jumps back to line 1870 and a new originating equipment number (Nl) is chosen. Otherwise, control passes to the next block of the program.
Lines 1970 - 2120:

The terminating equipment number (N2) takes on an integer value between one and 4096, inclusive. It (N2) is tested to determine if it is equal to the originating equipment number (N1). If they are equal, a new terminating equipment number (N2) is chosen and the test is repeated. Next the terminating equipment number (N2) is compared to all equipment numbers in the matrix. If the terminating equipment number (N2) is busy, the busy counter (Bl) and the number of calls attempted counter (Al) control jumps back to line 1470.

Path Search:

Lines 2130 - 2300:

The originating A row (D) is initialized at minus one and then incremented. It (D) is tested and if it is greater than three, control jumps to line 2190 where the calls blocked counter (K1) is incremented. Next, the value of the mode select variable (Q0$) is tested to determine whether the program is being run in the Manual or Random mode and control jumps back to the beginning of the appropriate block of code. If the call is not blocked, control jumps to line 2260. Then the originating grid (A) and the originating A row (D) are compared to the contents
of the matrix to determine if the (D) is busy - equal to one. If the originating A row (D) is busy, control jumps back to line 2150 where it (D) is incremented and tested again, first for being greater than three and then for being busy. If the originating A row (D) is not busy, control passes to the next section of the program.

Lines 2310 - 2400:

The originating B row (E) is initialized at minus one and then incremented and tested. If it (E) is greater than three, control jumps back to line 2150 where the originating grid (A), originating A switch (B), originating A row and the originating B row (E) are compared to the contents of the matrix to determine if the originating B row (E) is busy - equal to one. If the originating B row (E) is busy, control jumps back to line 2330 where it (E) is incremented and the tests are then repeated. If the originating B row (E) is not busy, control passes to the next section of the program.

Lines 2410 - 2500:

The originating C row (F) is initialized at minus one and then incremented and tested to determine if it (F) is greater than 15. If the originating C row (F) is greater than 15, control jumps back to line 2330 where the
originating B row (E) is incremented. Otherwise, the originating grid (A), originating A row (D) and originating C row (F) are compared to the contents of the matrix to determine if the originating C row (F) is busy - equal to one. If the originating C row (F) is busy, control jumps back to line 2430 where the originating C row (F) is incremented and the tests are repeated.

Lines 2510 – 2640:

The originating C switch (03) is computed. If the originating C row (F) equals the originating C switch (03), control jumps to line 2570 and a variable (0) takes on the integer value of half the value of the originating C switch (03). This is used to determine whether the originating C switch (03) is even or odd. If it (03) is even, the terminating C switch (T3) is equal to the originating C switch (03) plus one and if it (03) is odd, the terminating C switch (T3) is equal to the originating C switch minus one. In either case, the terminating C row (T) is equal to the terminating C switch (T3). If the originating C row (F) is not equal to the originating C switch (03), the terminating C row (F) is set equal to the originating C switch (03) and the terminating C switch (T3) is set equal to the originating C row (F).
Lines 2650 - 2720:

The terminating A row (G) and terminating B row (H) are computed and compared with the contents of the matrix to determine whether they are free or busy. If either the terminating A row (G) or terminating B row (H) is busy, control jumps back to line 2430 where the originating C row (F) is incremented.

Lines 2730 - 2970:

The originating grid (A), originating A switch (B), originating A column (C), originating A row (D), originating B row (E), originating C row (F), terminating A row (G), terminating B row (H), terminating C row (F), terminating grid (J), terminating A switch (K), and terminating A column (L) are all stored in the matrix and the idle-busy flag (M(13,T) is set equal to one. The value of the mode select variable (Q0$) is tested and if the program is running in the manual mode, the number of calls in the matrix counter (N) is tested and if it is less than 128, it is incremented. Then control jumps back to the beginning of the manual section of the program. If the program is in the random mode, the number of calls in the matrix counter (N) is incremented and control jumps back to the beginning of the random section of the program.
Lines 2980 - 3110:

The number of calls attempted (Al), the number of calls encountering busy (Bl), and the number of calls blocked (Kl) are printed out. Then if the program is in the manual mode, control jumps back to line 180. Otherwise, the number of calls blocked due to all 128 timeslots being full (F) is printed out and then control jumps to line 210. This is followed by the stop and end statements.
Variable Descriptions

M  Callstore memory matrix
N  Number of calls in the matrix
Q0$  Mode select
T  Timeslot
Al  Number of calls attempted
B1  Number of calls encountering busy
K1  Number of calls blocked
Ol$  Manual add question
N1  Originating equipment number
A  Originating Grid
B  Originating A Switch
C  Originating A Column
W  Iterative manual originating busy counter
Wl  Save iterative manual originating busy counter
H0$  Originating hang-up or continue
N2  Terminating equipment number
J  Terminating grid
K  Terminating A switch
L  Terminating A column
X  Iterative manual terminating busy counter
Xl  Save iterative manual terminating busy counter
H1$  Terminating hang-up or continue
E1  Enter number of calls to be attempted
E2  Average number of calls
A3  Always add a call
F1  Number of times all timeslots are full
A4  Add or delete
V1  Delete timeslot counter
V  Delete timeslot
U1  Add timeslot counter
U  Add timeslot
Variable Descriptions (Cont'd.)

Y  Iterative random originating busy counter
Z  Iterative random terminating busy counter
D  Originating A row
P  Timeslot counter 1
E  Originating B row
Q  Timeslot counter 2
F  Originating C row
R  Timeslot counter 3
03 Originating C switch
I  Terminating C row
T3 Terminating C switch
0  Originating C switch even
S  Timeslot counter 4
MATRIX

10 REM MATRIX SIMULATOR
20 REM
30 REM WRITTEN BY RICHARD D. MCINNES
40 REM
50 B9=BRK(0)
60 DIM M(13,128),Q0$(255),I1$(255),H1$(255)
70 REM INTRODUCTION
80 PRINT
90 PRINT "THIS PROGRAM SIMPLIFIES THE CALL SWITCHING MATRIX IN A 4096 LINE"
100 PRINT "CENTRAL OFFICE FOR TRAFFIC STUDY PURPOSES. THE LINES ARE NUMBERED"
110 PRINT "FROM 0 TO 4095. THE MATRIX CAN HOLD A MAXIMUM OF 128 CALLS AT ANY"
120 PRINT "GIVEN TIME. THESE CALLS CAN BE PROGRAMMED MANUALLY OR SELECTED AT"
130 PRINT "RANDOM BY THE COMPUTER. IF THE CALLS ARE PROGRAMMED MANUALLY ONLY"
140 PRINT "THE LAST 128 CALLS WILL BE IN THE MATRIX AT ANY TIME. ALL OTHERS"
150 PRINT "WILL BE REMOVED. THIS MODE IS USEFUL FOR STUDYING THE BLOCKING"
160 PRINT "CHARACTERISTICS OF THE MATRIX. IF THE COMPUTER IS ASKED TO SELECT"
170 PRINT "RANDOM CALLS, THOSE CALLS WILL BE SET UP AND TAKEN DOWN AT RANDOM."
180 PRINT "IF THE COMPUTER ATTEMPTS TO SET A CALL WHEN THERE ARE ALREADY"
190 PRINT "128 CALLS IN THE MATRIX, THAT CALL WILL BE BLOCKED."
200 REM INITIALIZE CALL STORE MATRIX
210 MAT M=ZER
220 N=0
230 REM MODE SELECT
240 PRINT
250 PRINT "IF YOU WISH TO ENTER THE ORIGINATING AND TERMINATING EQUIPMENT"
260 PRINT "NUMBERS MANUALLY TYPE 'MANUAL'. IF YOU WANT THE COMPUTER TO"
270 PRINT "SELECT RANDOM EQUIPMENT NUMBERS TYPE 'RANDOM'. IF YOU ARE"
280 PRINT "FINISHED TYPE 'FINISH'.":
290 INPUT Q0$
300 IF Q0$(1,1)="M" THEN 440
310 IF Q0$(1,1)="R" THEN 1330
320 IF Q0$(1,1)="F" THEN 3090
330 GOTO 230
340 REM MANUAL MODE
350 REM INITIALIZE TIMES, ATTEMPT, BUSY, AND BLOCKED COUNTERS
360 T=0
370 A1=0
380 B1=0
390 K1=0
400 REM ADD A CALL OR FINISH?
410 PRINT
420 PRINT "THERE ARE PRESENTLY "+N+" CALLS IN THE MATRIX. IF YOU WISH TO ADD"
430 PRINT "ANOTHER CALL TYPE 'ADD', OTHERWISE TYPE 'FINISH'.":
440 INPUT Q1$
450 IF Q1$(1,1)="A" THEN 450
460 IF Q1$(1,1)="F" THEN 2390
470 GOTO 400
480 REM ADD A CALL
490 REM ORIGINATING EQUIPMENT NUMBER (OEN)
500 PRINT
510 PRINT "ENTER ORIGINATING EQUIPMENT NUMBER ";
520 INPUT N1
530 IF N1=INT(N1)*0 THEN 570
540 IF N1>4095 THEN 570
550 IF N1<0 THEN 570
560 GOTO 690
570 PRINT
580 PRINT "EQUIPMENT NUMBERS MUST BE INTEGERS BETWEEN 0 AND 4095, INCLUSIVE."
590 GOTO 490
600 A=INT(N1/256)
610 B=INT((N1-256*A)/32)
C=INT(N1-256*A-32*B)
FOR w=1 TO 128
w=w
If w[1,w]=A AND w[2,w]=B AND w[3,w]=C AND w[13,w]=1 THEN 700
NEXT w
GOTO 820
REM DETERMINE WHETHER TEN IS IDLE OR BUSY
FOR w=1 TO 128
w=w
NEXT w
GOTO 820
REM TEN IS BUSY
HANG UP OR CONTINUE?
PRINT "THIS EQUIPMENT NUMBER IS BUSY. IF YOU WISH TO TERMINATE ITS CALL"
PRINT "TYPE 'HANG UP', OTHERWISE TYPE 'CONTINUE'."
input h0$
IF h0$="C" THEN 490
IF h0$="H" THEN 780
GOTO 700
REM HANG UP
M[13,w]=0
N=N-1
GOTO 480
REM TERMINATING EQUIPMENT NUMBER (TN)
PRINT "ENTER TERMINATING EQUIPMENT NUMBER ";
INPUT N2
IF N2<INT(N2)*0 THEN 900
IF N2>4095 THEN 900
IF N2<0 THEN 900
GOTO 930
PRINT "EQUIPMENT NUMBERS MUST BE INTEGERS BETWEEN 0 AND 4095, INCLUSIVE."
GOTO 920
IF N2=N1 THEN 950
GOTO 1000
B1=B1*1
A1=A1+1
PRINT "ORIGINATING EQUIPMENT NUMBER = TERMINATING EQUIPMENT NUMBER"
GOTO 490
REM BREAK DOWN TEN
J=INT(N2/256)
K=INT((N2-256*J)/32)
L=INT(N2-256*J-32*K)
PRINT "DETERMINE WHETHER TEN IS IDLE OR BUSY"
FOR x=1 TO 128
X1=X
NEXT x
REM TEN IS IDLE
GOTO 1260
REM TEN IS BUSY
M[13,x]=0
N=N-1
GOTO 490
REM INCREMENT ATTEMPT COUNTER
A1=A1+1
1280 REM INCREMENT TIMESLOT
1290 T=T+1
1300 IF T<129 THEN 2130
1310 T=T-128
1320 GOTO 1300
1330 REM RANDOM MODE
1340 REM ENTER PARAMETERS
1350 PRINT
1360 PRINT "ENTER NUMBER OF CALLS TO BE ATTEMPTED ",
1370 INPUT E1
1380 PRINT
1390 PRINT "ENTER AVERAGE NUMBER OF CALLS TO BE IN THE MATRIX ",
1400 INPUT E2
1410 REM INITIALIZE ATTEMPT, BUSY, BLOCKED, AND FULL COUNTERS
1420 A1=0
1430 B1=0
1440 K1=0
1450 F1=0
1460 A3=1
1470 IF A1 >= E1 THEN 2980
1480 IF N<E2 THEN 1500
1490 A3=0
1500 IF A3=1 THEN 1710
1510 REM ADD OR DELETE A CALL?
1520 A4=RND(1)
1530 IF A4<.5 THEN 1560
1540 IF A4>.5 THEN 1710
1550 GOTO 1510
1560 REM DELETE A CALL
1570 IF N <= 0 THEN 1710
1580 N=N-1
1590 V1=0
1600 REM SELECT A RANDOM TIMESLOT TO BE DELETED
1610 V=INT((128*RND(1))+1)
1620 V1=V1+1
1630 IF V1>128 THEN 1470
1640 IF M(13,V1)=1 THEN 1690
1650 V=V+1
1660 IF V<129 THEN 1620
1670 V=V-129
1680 GOTO 1620
1690 M(13,V1)=0
1700 GOTO 1510
1710 REM ADD A CALL
1720 A1=A1+1
1730 U1=0
1740 REM SELECT A RANDOM TIMESLOT
1750 U=INT((128*RND(1))+1)
1760 U1=U1+1
1770 IF U1 <= 128 THEN 1810
1780 REM ALL TIMESLOTS FULL!
1790 F1=F1+1
1800 GOTO 1560
1810 IF M(13,U1)=0 THEN 1860
1820 U=U+1
1830 IF U<129 THEN 1760
1840 U=U-128
1850 GOTO 1760
1860 T=U
1870 REM SELECT A RANDOM DEN
1880 N1=INT(4096*RND(1))
1890 A=INT(V1/256)
1900 M1=INT((N1-256*A)/32)
1910 C=INT(V1-256*A-32*B)
1920 REM TEST DEN FOR BUSY
1930 FOR Y=1 TO 128
1960 NEXT Y
1970 REM SELECT A RANDOM T3Y
1980 N2 = INT(4096 * RND(1))
1990 IF N2 = Y1 THEN 1970
2000 J = INT(Y2 / 256)
2010 K = INT((N2 - 256 * J) / 32)
2020 L = INT(N2 - 256 * J - 32 * K)
2030 REM TEST TEN FOR 9JSY
2040 FOR Z = 1 TO 128
2070 NEXT Z
2080 GOTO 2130
2090 REM T3Y IS BUSY
2100 R1 = R1 + 1
2110 A1 = A1 + 1
2120 GOTO 1470
2130 REM T3Y IS IDLE
2140 D = (-1)
2150 IF D = 3 THEN 2190
2160 GOTO 2260
2170 REM CALL IS BLOCKED
2180 K1 = K1 + 1
2190 IF Q0$[1, 1] = "R" THEN 1470
2200 PRINT "CALL IS BLOCKED."
2210 IF Q0$[1, 1] = "M" THEN 400
2220 PRINT "CALL IS IDLE."
2230 FOR P = 1 TO 128
2260 NEXT P
2270 REM CALL IS BUSY
2280 FOR P = 1 TO 128
2310 NEXT P
2320 REM CALL IS IDLE
2330 IF E = 3 THEN 2190
2340 E = E + 1
2350 REM DETERMINE IF OCR IS BUSY
2360 FOR R = 1 TO 128
2390 NEXT R
2400 REM OCR IS BUSY
2410 IF F = 15 THEN 2330
2420 F = F + 1
2430 REM OCR IS IDLE
2440 D = D + 1
2450 IF F = 15 THEN 2330
2460 REM DETERMINE IF OCR IS BUSY
2470 FOR R = 1 TO 128
2500 NEXT R
2510 REM OCR IS IDLE
2520 IF E = 3 THEN 2570
2530 E = E + 1
2540 IF F = 15 THEN 2330
2550 D = D + 1
2560 GOTO 2650
2570 D = INT(33 / R)
2580 IF D = 0 THEN 2570
2590 T3 = T3 + 1
I=T3
2610 GOTO 2650
2620 T3=T3+1
2630 I=T3
2640 GOTO 2650
2650 G=INT(T3/4)
2660 M=T3=4*G
2670 REM DETERMINE IF TAR AND TBR ARE BUSY
2680 FOR S=1 TO 128
2690 IF M(8,S)=J AND M(7,S)=2 AND M(8,S)=5 AND M(13,S)=1 THEN 2430
2700 IF M(10,S)=J AND M(11,S)=C AND M(7,S)=5 AND M(13,S)=1 THEN 2430
2710 NEXT S
2720 IF M=S AND B=S AND D=S THEN 2430
2730 REM TAR AND TBR ARE 'IDLE'
2740 REM STORE PATH IN MATRIX
2750 M[1,T]=A
2760 M[2,T]=B
2770 M[3,T]=C
2780 M[4,T]=D
2790 M[5,T]=E
2800 M[6,T]=F
2810 M[7,T]=G
2820 M[8,T]=H
2830 M[9,T]=I
2840 M[10,T]=J
2850 M[11,T]=K
2860 M[12,T]=L
2870 M[13,T]=M
2880 IF Q0$(1,1)="M" THEN 2910
2890 IF Q0$(1,1)="R" THEN 2950
2900 GOTO 230
2910 REM INCREMENT CALL COUNTER
2920 IF N>=128 THEN 410
2930 N=N+1
2940 GOTO 400
2950 REM INCREMENT CALL COUNTER
2960 N=N+1
2970 GOTO 1470
2980 REM FINISH MANUAL MODE
2990 PRINT
3000 PRINT "NUMBER OF CALLS ATTEMPTED "A1
3010 PRINT
3020 PRINT "NUMBER OF CALLS ENCOUNTERING BUSY "B1
3030 PRINT
3040 PRINT "NUMBER OF CALLS BLOCKED "C1
3050 IF Q0$(1,1)="M" THEN 210
3060 PRINT
3070 PRINT "NUMBER OF CALLS BLOCKED (ALL TIMESLOTS FULL) "F1
3080 GOTO 210
3090 REM FINISH
3100 STOP
3110 END
V. PROGRAM UTILIZATION

The program was designed to be self-explanatory in order to facilitate its use. Rather than explaining how the program is used, it is easier and more instructive to illustrate its use with several examples.

Random Mode:

The Random Mode enables the user to simulate the actual operation of a telephone exchange. The user first determines the number of calls to be attempted, thereby setting a limit on the amount of telephone traffic to be simulated. The traffic intensity is determined by the average number of calls to be in the matrix. The program selects random equipment numbers and attempts to set up and take down calls.

A simulation of 1,000 calls is shown on the following page.
IRUN MATRIX

This program simulates the call switching matrix in a 4096 line central office for traffic study purposes. The lines are numbered from 0 to 4095. The matrix can hold a maximum of 128 calls at any given time. These calls can be programmed manually or selected at random by the computer. If the calls are programmed manually only the last 128 calls will be in the matrix at any time. All others will be removed. This mode is useful for studying the blocking characteristics of the matrix. If the computer is asked to select random calls, those calls will be set up and taken down at random. If the computer attempts to set up a call when there are already 128 calls in the matrix, that call will be blocked.

If you wish to enter the originating and terminating equipment numbers manually type 'MANUAL'. If you want the computer to select random equipment numbers type 'RANDOM'. If you are finished type 'FINISH'. ?RANDOM

ENTER NUMBER OF CALLS TO BE ATTEMPTED ?1000

ENTER AVERAGE NUMBER OF CALLS TO BE IN THE MATRIX ?500

NUMBER OF CALLS ATTEMPTED 1000

NUMBER OF CALLS ENCOUNTERING BUSY 43

NUMBER OF CALLS BLOCKED 43

NUMBER OF CALLS BLOCKED (ALL TIMESLOTS FULL) 251

If you wish to enter the originating and terminating equipment numbers manually type 'MANUAL'. If you want the computer to select random equipment numbers type 'RANDOM'. If you are finished type 'FINISH'. ?FINISH

END OF PROGRAM
The table below illustrates the results of ten simulations of 100 random calls. In only one case was a call blocked. No calls were blocked due to all timeslots being full. This is expected because there are 128 timeslots.

Table 1. Sample Statistics for 100 Call Attempts

<table>
<thead>
<tr>
<th>Average Calls in Matrix Simultaneously</th>
<th>Calls Busy</th>
<th>Calls Blocked</th>
<th>Calls Blocked Due to All Timeslots Full</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>20</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>30</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>40</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>50</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>60</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>70</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>80</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>90</td>
<td>2</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>100</td>
<td>3</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Manual Mode:

The Manual Mode permits the user to set up various call patterns and to study their effect on additional attempted calls. In this manner, the blocking characteristics at the switching matrix can be studied. The simulation on the next page shows that the fifth call originating from an A stage will be blocked.
RUN
MATRIX

This program simulates the call switching matrix in a 4096 line central office for traffic study purposes. The lines are numbered from 0 to 4095. The matrix can hold a maximum of 128 calls at any given time. These calls can be programmed manually or selected at random by the computer. If the calls are programmed manually only the last 128 calls will be in the matrix at any time. All others will be removed. This mode is useful for studying the blocking characteristics of the matrix. If the computer is asked to select random calls, those calls will be set up and taken down at random. If the computer attempts to set up a call when there are already 128 calls in the matrix, that call will be blocked.

If you wish to enter the originating and terminating equipment numbers manually type 'MANUAL'. If you want the computer to select random equipment numbers type 'RANDOM'. If you are finished type 'FINISH'.

There are presently 0 calls in the matrix. If you wish to add another call type 'ADD'; otherwise type 'FINISH'.

ADD

Enter originating equipment number 70

Enter terminating equipment number 74095

There are presently 1 calls in the matrix. If you wish to add another call type 'ADD'; otherwise type 'FINISH'.

ADD

Enter originating equipment number 71

Enter terminating equipment number 74094

There are presently 2 calls in the matrix. If you wish to add another call type 'ADD'; otherwise type 'FINISH'.

ADD

Enter originating equipment number 72

Enter terminating equipment number 74093

There are presently 3 calls in the matrix. If you wish to add another call type 'ADD'; otherwise type 'FINISH'.

ADD

Enter originating equipment number 73

Enter terminating equipment number 74092

There are presently 4 calls in the matrix. If you wish to add another call type 'ADD'; otherwise type 'FINISH'.

ADD

Enter originating equipment number 74

Enter terminating equipment number 74091

Call is blocked.

There are presently 4 calls in the matrix. If you wish to add another call type 'ADD'; otherwise type 'FINISH'.

ADD

Enter originating equipment number 75

Enter terminating equipment number 74090

Call is blocked.

There are presently 4 calls in the matrix. If you wish to add another call type 'ADD'; otherwise type 'FINISH'.

ADD

Enter originating equipment number 76

Enter terminating equipment number 74089

Call is blocked.

There are presently 4 calls in the matrix. If you wish to add another call type 'ADD'; otherwise type 'FINISH'.

FINISH

Number of calls attempted 7
Number of calls encountering busy 0
Number of calls blocked 3

If you wish to enter the originating and terminating equipment numbers manually type 'MANUAL'. If you want the computer to select random equipment numbers type 'RANDOM'. If you are finished type 'FINISH'.

FINISH

Done
While not meant to be exhaustive or even to thoroughly treat the subject of telephone call switching with multistage matrices, these simulations illustrate the usefulness of computer simulated matrix testing.
VI. SUMMARY

It is desirable to create a computer model of a proposed telephone switching matrix prior to building a prototype. This approach saves time and money as well as facilitates analyzing the effects of design changes. The computer program described here was used to study the traffic-handling capability and blocking characteristics of a special three-stage matrix. Other matrix configurations can be modelled using a similar approach.
BIBLIOGRAPHY

Fundamental Principles of Switching Circuits and Systems


