PB 151409

NBS

Technical Note 50

MAGNETIC DRUM DIRECTORY AND PROGRAMMING SYSTEM FOR CODESORTING LETTER MAIL

U. S. DEPARTMENT OF COMMERCE
NATIONAL BUREAU OF STANDARDS

THE NATIONAL BUREAU OF STANDARDS

Functions and Activities

The functions of the National Bureau of Standards are set forth in the Act of Congress, March 3, 1901, as amended by Congress in Public Law 619, 1950. These include the development and maintenance of the national standards of measurement and the provision of means and methods for making measurements consistent with these standards; the determination of physical constants and properties of materials; the development of methods and instruments for testing materials, devices, and structures; advisory services to government agencies on scientific and technical problems; invention and development of devices to serve special needs of the Government; and the development of standard practices, codes, and specifications. The work includes basic and applied research, development, engineering, instrumentation, testing, evaluation, calibration services, and various consultation and information services. Research projects are also performed for other government agencies when the work relates to and supplements the basic program of the Bureau or when the Bureau's unique competence is required. The scope of activities is suggested by the listing of divisions and sections on the inside of the back cover.

Publications

The results of the Bureau's work take the form of either actual equipment and devices or published papers. These papers appear either in the Bureau's own series of publications or in the journals of professional and scientific societies. The Bureau itself publishes three periodicals available from the Government Printing Office: The Journal of Research, published in four separate sections, presents complete scientific and technical papers; the Technical News Bulletin presents summary and preliminary reports on work in progress; and Basic Radio Propagation Predictions provides data for determining the best frequencies to use for radio communications throughout the world. There are also five series of nonperiodical publications: Monographs, Applied Mathematics Series, Handbooks, Miscellaneous Publications, and Technical Notes.

Information on the Bureau's publications can be found in NBS Circular 460, Publications of the National Bureau of Standards ($1.25) and its Supplement ($1.50), available from the Superintendent of Documents, Government Printing Office, Washington 25, D.C.

NATIONAL BUREAU OF STANDARDS

Technical Note

50

JUNE 1960

MAGNETIC DRUM DIRECTORY AND PROGRAMMING SYSTEM FOR CODESORTING LETTER MAIL

Peter C. Tosini

DISTRIBUTED BY

UNITED STATES DEPARTMENT OF COMMERCE

OFFICE OF TECHNICAL SERVICES

WASHINGTON 25, D. C.

Price $1.75

TABLE OF CONTENTS

TABLE OF CONTENTS (CONT'D)

TABLE OF CONTENTS (CONT'D)

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGEMENTS

The author wishes to express his appreciation to Mr. Arthur Holt of the Rabinow Engineering Company whose report provided the basis for this study. Further thanks are due to Professor Harry Goode of the University of Michigan, Mr. Israel Rotkin of the Diamond Ordnance Fuse Labs, and Mr. Martin Stark and Dr. Bernard Levin of NBS, Section 12.5, for their valuable suggestions and comments.

ABSTRACT

This report is an analysis and extension of the Rabinow Engineering Company's proposal for a magnetic drum file-directory and special purpose computer to be used in a test coded-address ("codesorting") letter-mail sorting system scheduled for late 1960 installation at the Washington, D.C. Post Office. The report is composed of 6 major sections that successively examine the major considerations involved. A short introduction is presented first. Then the general problem of coding the fields composing addresses is discussed. This area is analyzed in terms of facilitating both the human coding process and the computer coded address-to-bin-number translation. Addresses are classified into two types: "standard" and "non-standard". When coded, their fields must be unique and yet be sufficiently flexible to specify a variety of addresses. They must furthermore adequately specify some more common forms of misaddressed mail that is presently correctly sorted manually. A slightly amended form of Coding Plan E-1 is then presented and analyzed in terms of the conditions specified for coded addresses in general.

After the presentation of the coding area, the local mail sort translation program of the Rabinow proposal is discussed in detail. It is then extended to include the outgoing mail translation program and non-standard and special address forms. The information and instruction characters necessitated by the programs are listed in detail and the effects of the file-directory source document preparation determined. A coding keyboard is then suggested.

The programming is then reexamined and it is determined that all addresses can be treated as special cases of the standard address forms at an increased cost in memory space and with a saving in the special instruction symbols required. Finally, alternative methods of file-directory information storage are explored and evaluated in terms of memory space requirements, table look-up and directory access times, and conceptual simplicity.

MAGNETIC DRUM DIRECTORY AND PROGRAMMING SYSTEM FOR CODESORTING LETTER MAIL

Peter C. Tosini

1. INTRODUCTION

1.0 Objectives and Scope

The purposes of this report are: (1) to examine the major concepts involved in designing a magnetic drum directory and programming system for coded-address sorting ("codesorting") letter mail in general; and (2) to develop these through modifications and extensions to accommodate local and outgoing sorting for the Washington, D.C. Post Office in particular. It is restricted to one distributor separating local mail to carriers and "directs" (destinations receiving large volumes of mail), and outgoing mail to cities and "state residues" (alphabetic ranges of cities within a state). The specialized problems encountered in multi-machine and multi-pass sorting, sorting to carrier walk, and sorting to time-varying destinations are not discussed.

Codesorting can be described as a series of five distinct operations:

(1) Address coding (Plan E-1, [1] the best presently available set of rules for abbreviating literal address items by a selective choice of letters, is assumed in this study),

(2) Imprinting coded address on envelope,

(3) Coded address reading and memory matching,

(4) Codewheel setting,

(5) Transporting letter to its associated bin.

Operations (2) and (3) are primarily concerned with coding, memory storage, and programming, and their analysis forms the basis of the report. The report first presents an expository discussion of the interrelated factors affecting memory storage and programming. It then relates these to the sequential steps comprising the memory matching program.

Essentially, the report is a detailed analysis and extension of the Rabinow serial programming proposal. [2] The analysis and extensions are developed through the following criteria:

---

[1] M. C. Stark, unpublished report.

[2] A. Holt, Rabinow Engineering Co., unpublished report partially presented in A. Holt, "An Electronic Directory for Sorting Mail," Proceedings of the Eastern Joint Computer Conference, Dec. 3-5, 1958, pp 79-90.

(1) Operational feasibility,

(2) Operational flexibility, and

(3) Reduction of coding operator effort.

2. CODED ADDRESS FIELDS

2.0 Introduction

The information specifying any address is generally composed of a set of variable-length fields. Local mail is usually described by building number, street name, street type, and, if necessary, section; while outgoing mail is designated by city and state. However, variations from these standard forms commonly occur. These are discussed in a later section of the report.

The abbreviated and coded field information is imprinted on the envelope and serves as the "source document" for the translation program which yields a bin number as output. The format of the imprinted fields was primarily designed to accommodate standard address forms. Four such fields are available for specifying local mail: [1]

(1) Building Number Field,

(2) Street Name Field,

(3) Street Type Field, and

(4) Section Designation Field.

Two fields describe outgoing mail:

(1) City Field, and

(2) State Field.

2.1 Evaluative Criteria

The fields provide the information for the establishment of a one-to-one correspondence between any coded address and its associated entry within the memory. In the most general case, this functional requirement imposes the following specifications:

---

[1] The final coding plan will make optimum use of "short codes" (one-or-two-character identifications for specific destinations receiving large volumes of mail). This will greatly diminish the average coding time per letter.

(1) The fields must be uniquely interpretable. Each field must be distinguishable from any other since each is independently necessary for the complete specification of an address. Several techniques are available for achieving this individuality: (a) a flag can be introduced before or after each field; (b) fields can be defined according to position, e.g., the n^{th} character of a coded local address would always signify the beginning of the Street Name Field if all n-1 characters of the Building Number Field were always filled; (c) exclusive use of certain characters in certain fields, e.g., direction symbols appearing only in the Section Designation Field; and (d) a combination of the above.

(2) The fields must be flexible in usage. Since a variety of address forms is used, the fields must be sufficiently adaptable to specify uniquely each form. This requires that the number of character positions per field and the symbols representable within each character position be adequate for specifying all the forms.

(3) The fields and the characters composing them must be in forms suitable for input to, and interpretation by, the logical circuitry which performs the memory matching.

(4) Coding effort should be minimized. As a general criterion, the average number of strokes per address should be minimized. Each key-stroke should add to the information content of a field until it is fully specified. Then, no more strokes should be expended in filling unused portions of fields.

2.2 Analysis of Field Usage

2.2.1 Uniqueness of Local Fields:

(1) An individual address may or may not require a section designation. No confusion arises from this if the characters for section designations are used in no other fields.

(2) The Street Type Field is used on every local address and its characters may be identical to any of those in the Street Name Field. But, the field remains uniquely identifiable if the address is referenced from right-to-left. Then, the first character that is not a section designation character is a street type character.

(3) Confusion can arise between the Building Number and Street Name fields since the same characters

can be used in both. This difficulty can be resolved however, by having the coder introduce a special symbol at tne end of the Building Number Field. An alternative, if the field lengths were fixed, would be to have the operator always fill completely either the Building Number or Street Name fields. This would require the entry of zeros in the unused portion of the field (an operation that might be performed much like tabulating on a typewriter). The fields could thereby be uniquely specified by reading the address from left to right if the Building Number Field were completely filled, and by reading from right to left if the Street Name Field were filled. The use of the special symbol denoting the end of the Building Number Field seems to be the more acceptable since it should simplify coding equipment without increasing coding time. Any character that is not used in either of the two fields would be acceptable as the special symbol.

2.2.2 Uniqueness of Outgoing Fields:

The City-State Field is generally composed of six consecutive characters. The last two always designate the state, and the first four the city. Therefore the fields are uniquely defined by their relative positions. One-or-two character short codes for certain cities will also be used. They will not require a state abbreviation and will not use the entire six character outgoing field. These will also be unique since any such address is known, by definition, to be a short-coded city. Confusion can occur, though, between local and outgoing destinations having identical short codes, if both are sorted simultaneously. In this case, a special identification symbol can be automatically added to the outgoing short code when the coding station letter-forwarding mechanism is actuated by the outgoing release key.

2.2.3 Field Flexibility and Ease of Coding Effort:

The local fields are quite flexible since they may be composed of a variable number of both alphabetic and numeric characters. No flexibility is required of the outgoing fields since they specify only the standard outgoing address form or short-coded cities. A detailed discussion of the requirements for local-field flexibility is presented in a later section of the report.

The analysis of field usage suggests that considerable simplification may result if the translation program can treat the Street Name, Type, and Section Designation Fields as a single field. As will be seen, this is indeed the case. Local standard addresses

are then specified by only two fields instead of four. These are separated by an End-of-Building-Number-Field symbol which simplifies their reading and transfer to registers. The suggested use of local and outgoing fields is presented in the following detailed description of Plan E-1.

2.3 Description of Plan E-1

Plan E-1 is a set of rules for reducing the length of addresses while retaining their information content. These rules determine the transcription, reduction, and abbreviation of address items into the following coding fields:

Local Mail

(1)	Building Number Field*	(up to 5 characters)
(2)	Street Name Field	(up to 4 characters)
(3)	Street Type Field	(1 character)
(4)	Section Designation Field	(1 character)

Outgoing Mail

(1)	City Field	(up to 4 characters)
(2)	State Field	(2 characters)

2.3.1 Local Mail -- Standard Case:

Local mail is generally specified by a building number, street name, street type, and section designation. These items are entered into their corresponding fields according to the following rules:

(1) Numeric items are transcribed directly.

(2) Alphabetic items are reduced according to the rules of Table I (P 8), with the exception of spelled-out numbers, which are treated as numerics.

(3) State names used as street names, are coded with the abbreviations of Table II (P 9).

---

* The Building Number Field is always followed by a one-character End-of-Building-Number-Field symbol which serves as a reference point ior address reading and differentiation between outgoing and local addresses.

(4) Fractions and letters following building numbers are ignored but letters preceding them are accepted. The letter x is used to designate the fraction 1/2 in half-streets; e.g., 6 1/2 Street, N.W.

(5) Street types are designated by the abbreviations of Table III (P11).

(6) Section designations are specified by the following four one-character symbols:

North or Northeast	(Ne)
East or Southeast	(sE)
South or Southwest	(Sw)
West or Northwest	(nW)

These symbols are distinct from the directional beginnings of Table IV.

Some examples of local coding are:

2502 Franklin Avenue	2502*frnia
10 Main Street	10*mains
146 Fifth Avenue	146*5a
1600 Commonwealth Avenue, N.E.	1600*cmnha(Ne)
257 44th Place, North	257*44p(Ne)

2.3.2 Outgoing Mail

Outgoing addresses are specified by city and state. These are entered into their corresponding coding fields according to the following rules:

(1) City names of four letters or less are transcribed directly.

(2) City names of more than four letters are reduced to four letters by the reduction rules of Table 1.

(3) City names consisting of more than one word are treated as a single word.

(4) Commonly used beginnings and endings of city names are abbreviated by the set of abbreviations in Table IV (P 12). (These are the only upper-case letters used in the coding process.)

(5) Numeric numbers are spelled out and treated as alphabetics.

(6) State names are abbreviated with the set of abbreviations in Table II.

Some examples of outgoing coding follow. As is seen, the state abbreviation immediately follows the city name.

Laustin, Tex.	lusiex
Stone, Ind.	stnein
Allen, Ga.	alengg
West Middlesex, Conn.	Zmdxct
Mount Carmel, N. Y.	Mcrlzq
Lexington, Ky.	lxnNky

2.4 Summary

The use of local and outgoing fields for specifying standardly addressed letters has been presented. Specification of non-standard address forms is discussed in a later section of the report.

TABLE I

Reduction Rules

Step		Operation
1	A	Take the first abbreviation.
	B	Take the first letter.
2	A	Take the second of two successive vowels.
	B	Take the second of a double consonant.
	C	Take the next consonant.
3	A	If only two letters remain, take them.
	B	Take the second of two successive vowels.
	C	Take the next consonant.
4	A	Take the ending abbreviation.
	B	If only one letter remains, take it.
	C	Take the less frequent of the last two letters (See Table V (P 13) for frequency alphabet.)

Note: Proceed through the steps, starting with Operation A of Step 1. If operation A applies, do it and go on to next step. If it does not apply, continue through the operations of the step until one does apply. Continue through steps 2, 3, and 4 in this manner until reduction is completed. Items of four or fewer characters are transcribed directly.

Source: Psychological Research Associates, Inc., unpublished report.

TABLE II

State Name Abbreviations

State Name	Abbreviation
Alabama	al
Alaska	aa
Arizona	az
Arkansas	ak
California	yf
Canal Zone	cz
Colorado	co
Connecticut	ct
Delaware	de
District of Columbia	dc
Florida	fl
Georgia	gg
Hawaii	hw
Idaho	id
Illinois	bp
Indiana	in
Iowa	iw
Kansas	ks
Kentucky	ky
Louisiana	lu
Maine	me
Maryland	md
Massachusetts	ma
Michigan	mh
Minnesota	mn
Mississippi	ms
Missouri	mo
Montana	mt
Nebraska	nb
Nevada	nv

Table II (Continued)

State Name	Abbreviation
New Hampshire	nh
New Jersey	js
New Mexico	hv
New York	zq
North Carolina	nc
North Dakota	nd
Ohio	oh
Oklahoma	ok
Oregon	or
Pennsylvania	xm
Puerto Rico	pr
Rhode Island	ri
South Carolina	sc
South Dakota	sd
Tennessee	tn
Texas	ex
Utah	ut
Vermont	vt
Virginia	vg
Washington	wg
West Virginia	wv
Wisconsin	wi
Wyoming	wy

Source: M. C. Stark and I. Rotkin, unpublished report.

TABLE III

Street Type Abbreviations [1]

Street Type	Abbreviation
street	s
avenue	a
road	r
place	p
drive	d
court	c
lane	l
terrace	t
boulevard	y
circle	y
crescent	y
heights	y
highway	y
manor	y
park	y
square	y
parkway	y
row	y
trail	y
turnpike	y
walk	y
way	y
alley	y

Source: M. C. Stark and I. Rotkin, unpublished report.

[1] These are the only street types explicitly coded. Other street types (quite rare) are incorporated with the street name and coded as an integral part of it. Then the letter s is entered as the resulting item's street type.

TABLE IV

Abbreviations for Commonly Used

Beginnings and Endings

Beginning	Abbreviation	Beginning	Abbreviation
North or Northeast	I	San, Santa, Saint	S
East or Southeast	O	Mont, Mount, Mountain	M
South or Southwest	X	Port	P
West or Northwest	Z	Glen	G
New	N	Lake	K

Ending*	Abbreviation	Ending*	Abbreviation
ville	V	lake	K
valley	Y	land	L
ton	N	son	S
town	T	spring	G
burg	B	fall	A
dale	D	field	F
creek	R	park	P
center	C	wood	W
city	T		

Source: M. C. Stark, unpublished report.

* Including the plurals of these.

TABLE V

Frequency Alphabet

(In descending order of frequency)

e
r
n
a
t
l
s
o
d
y
c
i
k
h
m
g
f
w
u
p
x
b
v
z
j
q

Source: Psychological Research Associates, Inc., unpublished report.

3. LOCAL AND OUTGOING TRANSLATION PROGRAMS

3.0 Introduction

It would appear that the general problem of translating coded destinations into their associated bin numbers can be accomplished in two ways:

(1) Coded destinations can be arithmetically manipulated to generate bin numbers (within acceptable limits of duplication).

(2) Coded destinations can be matched against a memory directory which includes all the coded destinations and their associated bin numbers.

Method (1) is the more direct of the two. But its use is precluded by a specific operational requirement of codesorting mail. Many different destinations must be sorted to the same bins e.g., all destinations comprising a carrier's route must be sorted to that carrier's bin. The translation procedure must therefore yield bin numbers which are arithmetically independent of the coded destination. Therefore, method (1) is not applicable (though it is useful in generating a beginning reading address for a random-access memory system using method (2)).

Method (2) does provide complete flexibility of bin number output and therefore satisfies the criterion. It is the basis of the Rabinow programming proposal. The imprinted address fields are read and sent to registers where comparisons are made with entries read in from the memory drum. When an equality match occurs, the appropriate bin number is read out and sent to the codewheel setter. The program is composed of instructions appearing before each memory item. These initiate and guide the arithmetic, comparative, and transfer operations the program comprises.

3.1 Memory Information Storage - Local Directory

The storage pattern of information corresponds to the sequence of operations necessary for bin determination. For local standardly addressed mail, a series of tests on different levels of inclusiveness is performed where each test successively narrows the search area for the corresponding memory entries. For instance, a standardly addressed local letter is described by street name, type, section, and building number. Each of these fields must be matched for bin determination. The memory reflects this by the organization of its

information. Each street is entered once, followed by its type and section. [1] Following these are listed entries which specify building numbers on the street. Here, use is made of the fact that carriers deliver mail to continuous groups of buildings. These groups are termed "breaks "and their lower and upper bounds are, respectively, the first and last buildings of the continuous set of carrier stops. Entered after the upper limit of each break is the bin number of the carrier serving that break. Consequently, the totality of breaks and carrier entries for any street completely specifies the carriers delivering letters to any building on that street. The break organization consequently obviates the need for individually specifying each building on every street.

Memory entries are stored on the channels of a magnetic drum. A variety of information storage patterns is therefore possible. These vary from full-parallel to full-serial. The full-parallel method stores coded fields of information in fixed-length sets of channels on a line along successive channels. Each set of channels specifies a different field, i.e., the first set, the street; the second, the type; the third, the section; the fourth, the break; and the fifth, the carrier. Successive vertical sets of these horizontal sets are required to describe the totality of breaks and carriers on any individual street. This results in a duplication of the information fixed for any street and is wasteful of memory space. Furthermore, since the number of channels per field is fixed and the information entered in each one varies in length, some of the sets will not be completely filled and will again waste memory space. The parallel mode also requires that all channels specifying an address be read simultaneously. This necessitates an amplifier for each channel being read.

The full serial method stores the coded information sequentially along a channel rather than linearly across the channels. It does not require fixed sets of memory space to the extent that the parallel mode does and thus permits more compression of the information with a resulting saving in memory space. And since only one channel is read at any time, only one amplifier is required. The serial method does require special flag symbols, though, to identify the stored items since the items' types are not determined by their location on the drum as was the case with the parallel method of storage. Information is entered as follows: The coded street name, type, and section are successively entered, followed by the upper limit of the first break on the street and the carrier serving it. Then

---

[1] This is directly analogous to the present scheme of distribution for manual sorting. An alternative organization would be according to building numbers instead of street names. That is, each hundred's block would appear once and be followed by all the street names having that specific block of building numbers. This method would not have the flexibility of the street name organization and would require more memory space.

the upper bounds of all the successive breaks followed by their bin numbers are entered. Preceding each of these entries is its flag symbol identifying the type of entry. These flag symbols also initiate the computer operations which perform the matching program.

Full-serial storage appears to be the more acceptable of the two because of its efficient memory space utilization and the flexibility afforded by extensions of its special order concept. But these apparent advantages necessitate increased complexity of the electronic circuitry for interpreting and acting upon the special symbols. The two storage methods require different programs for memory matching. Serial programming will now be presented. Parallel programming and a more detailed analysis of storage alternatives are discussed in Section 6.2.

3.2 Serial Programming - Local Mail Sort

The memory matching program is essentially a set of comparisons between the fields imprinted on the envelopes and stored in registers, and the information read from the drum directory and stored in temporary registers. A flow chart for standardly addressed local mail is presented in Figure 1 (P25). The successive steps are explained below. The chart is phrased in terms of the logical comparisons that must be made for memory matching. It is not implied that the operations fully represent the actual physical equipment requirements or information flow.

Step 1

The process begins when the coded fields on the envelope are sensed by reading heads, amplified, and transferred to registers. There are two registers: the Street Register (SR); and the Building Number Register (BR). The Street Register stores the Street Name, Type and Section fields as an integral unit, while the Building Number Register stores the Building Number Field.

This might be done by reading the coded envelope from right to left and transferring the characters into the Street Register until the end-of-building number symbol is sensed. Then, continuing reading from right to left, the building number is transferred into its register. The reading consequently proceeds from the least to the most significant characters of the fields. By transferring the characters to the registers as they are read (from right to left) the information in the registers will have its least significant character in the right-most register position. An alternative method might be to read the envelope fields simultaneously; the Building Number Field from left to right and the Street Fields from right to left until the end-of-building symbol is reached. The street fields could be transferred from right to left as before, but the building number would usually require shifting to the right for the least significant position criterion to be satisfied. This method seems faster but may be more expensive.

After the fields are sent to registers, channel selection is performed. This operation determines which of the drum's channels stores the information concerning the particular street in the street register. Here the concept of arithmetic interpretation of binary coded characters is used. By using a "collating sequence" binary-coded-decimal code (one in which the addition of unity to an alpha-numeric yields the succeeding alpha-numeric) the memory entries can be organized alphabetically. Consequently, all numeric streets and their breaks and carriers will be entered in numerical order followed by the alphabetically ordered alphabetic streets and their breaks and carriers. Therefore, the range of entries on any channel is specified by the magnitude of the first and last streets listed in the channel. This is the basis for the channel selection procedure proposed by Rabinow.[1/] The Rabinow method utilizes a separate pre-directory which is a consecutive listing of the last street entered in each channel, with each street followed by its channel number. These are compared against the street stored in the street register until one is found to be greater than or equal to that street. Since the streets are numerically ordered, this test specifies the desired channel. The channel number is then read and initiates the reading of the channel. If a street extends into another channel, the Building Number Field can be tested simultaneously with the street field. By "street" in the preceding and following discussions is meant the street name, type, and section. An alternative proposal performs channel selection by a similar series of tests between the stored street and the end-of-channel streets. The test path is in the form of a "tree" where each test (branch) successively halves the possible search area; i.e., the first test tells which half of the drum the street is in; the second, which quarter; the third, which eighth; etc. Therefore, n tests suffice for channel determination on any 2^n channel drum. e.g., six tests for a 64 channel drum. Again, this may be extended to include the case of a street extending into more than one channel by including a simultaneous test on the Building Number Field.

Step 2

The memory layout and programming are designed with the street entry as the key item. That is, street determination fixes the range where the carrier number is found on the channel. Since serial storage is used, a special symbol (s_1) is entered before each street. The next step after channel selection is than to find the s_1 symbol that passes the drum reading head. But, several problems arise in meaningfully interpreting the stored information since it is in binary form. The reading head can interpret only pulses and, therefore, without outside control cannot determine the number of "no pulses" it is "reading". This problem can apparently be resolved in at least three ways:

1/ A. Holt, Op. cit.

(1) The binary code for any character can be chosen so that a binary one is never followed by more than one binary zero and any character always begins with a binary one. This severely restricts the characters representable by any fixed number of bits.

(2) Binary ones could still be represented by positive pulses, but binary zeros by negative pulses instead of the absence of pulses.

(3) A drum channel can be reserved for reference pulses. That is, the channel is completely filled with binary ones. When a channel is read, it is continuously compared with this channel. When both have simultaneous pulses, a binary one is being read. When only the reference channel has a pulse, then a binary zero is being read.

(4) Since the drum speed is uniform to a high degree, a device rigidly coupled to the drum can generate reference pulses at the same rate as (3).

An allied problem is that without further control the reading head cannot distinguish whether it is reading the beginning, ending or middle of the s_1 symbol. This control can be provided in several alternative ways:

(a) A continuous comparison can be made between the stream of bits read from the channel and a stored s_1 symbol. An exact match signifies that the symbol has been reached and reading of the street name can begin immediately. This method is possible as long as any serial combination of the other character bit representations cannot produce the code for the symbol.

(b) A number of zeros greater than the greatest number of consecutive zeros in any character's code can be entered before the symbol. These can indicate that if follows and might even replace it.

(c) If all readings are begun at a fixed angular position of the drum, one timing pulse can determine the starting point for reading on all channels. This requires an average of one-half a drum revolution before reading can be initiated.

(d) An "address channel", marked with pulses at regular intervals could also be used. The s_1 symbols on all the other channels would be aligned with these pulses. This, as the first two alternatives, would permit low access time because of rapid entry, but would be wasteful of memory space because of the restrictions it imposes on the format of the entered items.

The interpretation problem is greatly simplified if all characters are specified by the same number of bits. It will be discussed further after completion of this discussion of serial programming.

After the s_1 symbol is sensed and interpreted, all the bits specifying the street are sent to the Temporary Street Register (TSR), But the imprinted coded streets will generally vary in length, i.e., some streets, with their type and section, can be specified by three characters while others require up to six characters. Furthermore, they appear in the SR with their least significant character in the right-most register position. The characters transferred to the TSR must therefore be aligned with those in the SR if the subsequent equality test is to be made. Some alternative methods of controlling the transfer of bits are:

(1) Streets can be entered in the memory so they take the maximum number of characters by inserting zeros in the unused character positions. Consequently, every street is specified by six characters and a constant number of bits is always transferred to the TSR after the s_1 symbol is interpreted. But this procedure is quite wasteful of memory space.

(2) Different s_1 symbols can be used depending upon the number of characters in the street item following them. This would necessitate six orders instead of only one. But the recognition circuitry could possibly be simplified if the orders were composed of two parts: (a) a "prefix" order (the usual s_1 symbol); and (b) a "suffix" group of bits specifying the number of characters to be sent to the register.

(3) The street entry is immediately followed by either of two instruction symbols (s_2 or s_3). Therefore, bits can be sent to the register until either of these is sensed.

If method (1) is used, transfer can proceed directly from left to right and completely fill the TSR, with the right-most character automatically located in the right-most position of the register. But the other methods will generally require some intermediate storage and counting devices for the shifting operation required to properly align the less-than-six character streets in the registers. 2/

2/ The alignment problem is easily resolved if the imprinted address fields are read from left-to-right and their corresponding registers are filled from left-to-right. Then the register fields are in direct alignment with the temporary register fields.

Method (3) also requires all alpha-numeric characters to be composed of the same number of bits. This condition does not appear strictly necessary for the first two methods. But to eliminate possible redundancy that could occur when serial combinations of character bit configurations are interpretable as a third character, the characters should each have the same number of bits in any of the methods.

Steps 3 and 4

The street item from the envelope, stored in the Street Register, is then compared for equality with the memory street stored in the Temporary Street Register. If equality occurs, it means that the correct street has been found and the program continues to Step 5. If equality does not occur, the program returns to Step 4 and continues searching for the appropriate street. If because of some error the desired street is not located in the particular channel being searched, the search will continue indefinitely in a loop. Consequently, some method must be provided to send error letters to an error bin when the drum has made more than one revolution after the beginning of reading.

Step 5

After the correct street is found, the building number must be matched with its associated entry within that street. Since sorting is done to the carrier, and since carriers deliver to continuous sets of buildings ("breaks"), it is necessary to specify only the break limits instead of all individual building numbers. However, certain buildings ("exceptions") are physically located within carriers' breaks but are not served by those carriers. This mail must therefore be sorted to its own bins. Exceptions are entered immediately preceding the break which numerically contains them. As was done with streets, exceptional building numbers and break entries are preceded by special instruction symbols (s_2 and s_3 respectively) which identify them and control the programming.

The program, after fixing on the street, reads the next memory character and tests to see if it is the s_2 symb ol. If it is, then the following item is known to be an exceptional building number and is sent to the TBR. Then, the TBR is compared for equality with BR which stores the building number read from the imprinted envelope. If equality occurs, the program transfers to S tep 7. If not, control is transferred back to Step 5. If the first or any other test against the s_2 symbol in Step 5 does not yield equality, control is transferred to Step 6.

Step 6

If the test against the s_2 symbol did not yield equality, the character is tested against the break symbol (s_3). If equality does not occur, control is transferred back to Step 5. If it does, the entry

Step 7

Each break or exception is followed by a bin number symbol (s_4) and the bin number of the carrier serving it. After finding the correct break or exception, the program reads the next character and determines whether it is an s_4[3/], [4/]. It them transfers the fixed length bin number to the codewheel setter. Finally all registers are cleared and the program is ready to process the next address.

3.3 Serial Programming - Outgoing Mail Sort

The programming for outgoing mail is similar to that of local mail. The imprinted information is read, sent to a register, and successively compared with the sequential entries on the drum. But it is much simpler since usually only one type of test is required after channel selection. A tentative program is presented in Figure 2 (P. 26), and the successive steps are explained below:

Step 1

The imprinted information consists of a four character City Field followed immediately by a two character S tate Field. The six characters are read and tested with key words that locate the correct channel. The channel entries consist of sequentially entered cities, each followed by its appropriate bin number. Each state's cities are grouped alphabetically and each group is ordered according to the alphabetical rank of the state. No two states'cities ever appear on a single channel. Channel selection can therefore be performed by placing the State Field before the City Field and testing on the resulting index word.

Step 2

After channel selection, the city field is transferred and stored in the City Register (CR). Reading of the channel can begin, as will be explained in Step 3, only at one position of the drum. When this position is reached, the first four character City Field is sent to the Temporary City Register (TCR).

[3/] If it is not, an error has occurred and the letter is directed to an error bin after the drum has completed one revolution.

[4/] It should be noted that comparison tests must be delayed until the registers are filled and not performed during the interval when the registers are being filled. This can be accomplished by inhibiting the output of the tests until the succeeding instruction is interpreted. Tests are thereby performed character-by-character but not passed or failed until the complete items are tested.

following the symbol is added to the TBR and stored there. This entry is the upper limit of the first break on the street. Before the addition, the TBR is empty, so the addition yields the upper limit of the first break on the street. The BR is then compared against the TBR. If it is less-than-or-equal-to the TBR, it means that the building number is included in that first break, and the program shifts to Step 7. If it is greater than the TBR, it means that the building number is not included in the first break. The program therefore shifts back to Step 5 and continues searching for the proper break or exceptional building number. Exceptions and breaks are specified by the arithmetic difference between the particular exception or break and the exception or break item preceding it. This difference is called the "Differential Building Number" (DBN) and for the case of the first entry on a street equals the upper limit of the first break, or the exceptional building number if one precedes the break.

Some memory space can be saved by incorporating a shift technique with the instruction symbols. The great majority of successive breaks differ by an even 100's number. Therefore, by entering only the 100's digit and shifting it before addition, the break limit is specified without being fully entered. However, certain successive break differentials do not follow this pattern and will have to be fully stored and transferred. This will require the use of a different s_3 symbol which will not perform shifting. Differences between successive exceptions, and exceptions and breaks, also generally do not follow the pattern and will require full differential storage and transfer. A similar proposed procedure is to store only the characters differing between successive entries rather than their arithmetic difference. This simplifies file-directory preparation and is also amenable to the shifting technique.

Bit transfer problems, identical to those of Step 4, also occur in Steps 5 and 6. Unless the special symbols specify the number of bits to be transferred, the bits must be transferred in fixed-length blocks, or else all characters and special symbols must be specified by the same number of bits and transfer continued until a special symbol is interpreted.

Step 3

The CR is then compared with the TCR for equality. If it occurs, the correct entry has been found and the fixed-length bin number is transferred to the codewheel setter and the registers are cleared. If equality does not occur, the next city is read in and the test re-performed. Letters to any city having an entry in the drum can be sorted to any chosen bin. However, many cities will not be entered in the drum. These must be sorted to alphabetical residue categories, as, for instance, a-l and m-z. This can be done by including alphabetically the residue "cities" lzzz and zzzz, and their residue bin numbers, in the drum directory. Then, by performing a less-than-or-equal-to test on these entries, the residue mail is grouped to its appropriate bin. This procedure requires that the residue entries be preceded by a special symbol instructing the computer to perform the less-than-or-equal-to test. Consequently, every character read during drum reading must be compared against the special symbol. The residue sorting concept further requires that drum reading begin at the lowest alphabetical city on the channel, thus precluding random entry into the drum and requiring an average of one-half a drum revolution before reading can begin.

The additive technique used in the local mail programming is extendable to outgoing mail. Any city can be entered as the numeric difference between itself and the city immediately preceding it. These differentials, when added to the cumulative total in the TCR, would then produce the desired codes. This requires all the differentials to either be four characters long, or to be preceded by special symbols specifying the number of characters composing them. It could conceivably require four characters for some differences so the fixed-length differentials would be four characters long and consequently would save no memory space. The variable-length alternative method would require three special symbols and their associated circuitry. Assuming the symbol would be one character, only two characters of memory space could be saved in the extreme case of a one character difference between successive city names. For the case of a four character difference, five characters of memory space would be required. A more reasonable proposed solution appears to be the entry of only the characters differing between successive cities.

The memory layout results in the repetition of many cities having the same name but located in different states. An alternative storage procedure might be to enter all cities alphabetically with no duplication and no organization according to states. Then any city name appearing in more than one state could be followed by the codes for those states. City name (4 characters) duplication is thereby eliminated at the expense of entering an equal number of previously unnecessary state abbreviations (2 characters). This procedure does not permit sorting to state residues and may require a special symbol for the interpretation of the state abbreviations.

3.4 Summary

Some characteristics of serial programming have been presented. Its major complexities arise from the variable length of address fields and the use of instruction symbols stored before every entry on the drum. Possible extension of the serial programming concept to accommodate nonstandard local address forms is discussed next.

4. EXTENSION OF SERIAL PROGRAMMING

4.0 Introduction

The proposed serial programs appear to be satisfactory for sorting standard address forms. However, they may be refined in the following areas:

(1) Short Coding

(2) Minimum redundancy coding

(3) Non-standard address coding, and

(4) Special situations.

These are discussed below and the suggested final use of E-1 summarized in Table VI (P33).

4.1 Short Coding

Much local mail is destined for relatively few large receivers of mail who will be specified by short codes. Under the present system, the operator would have to:

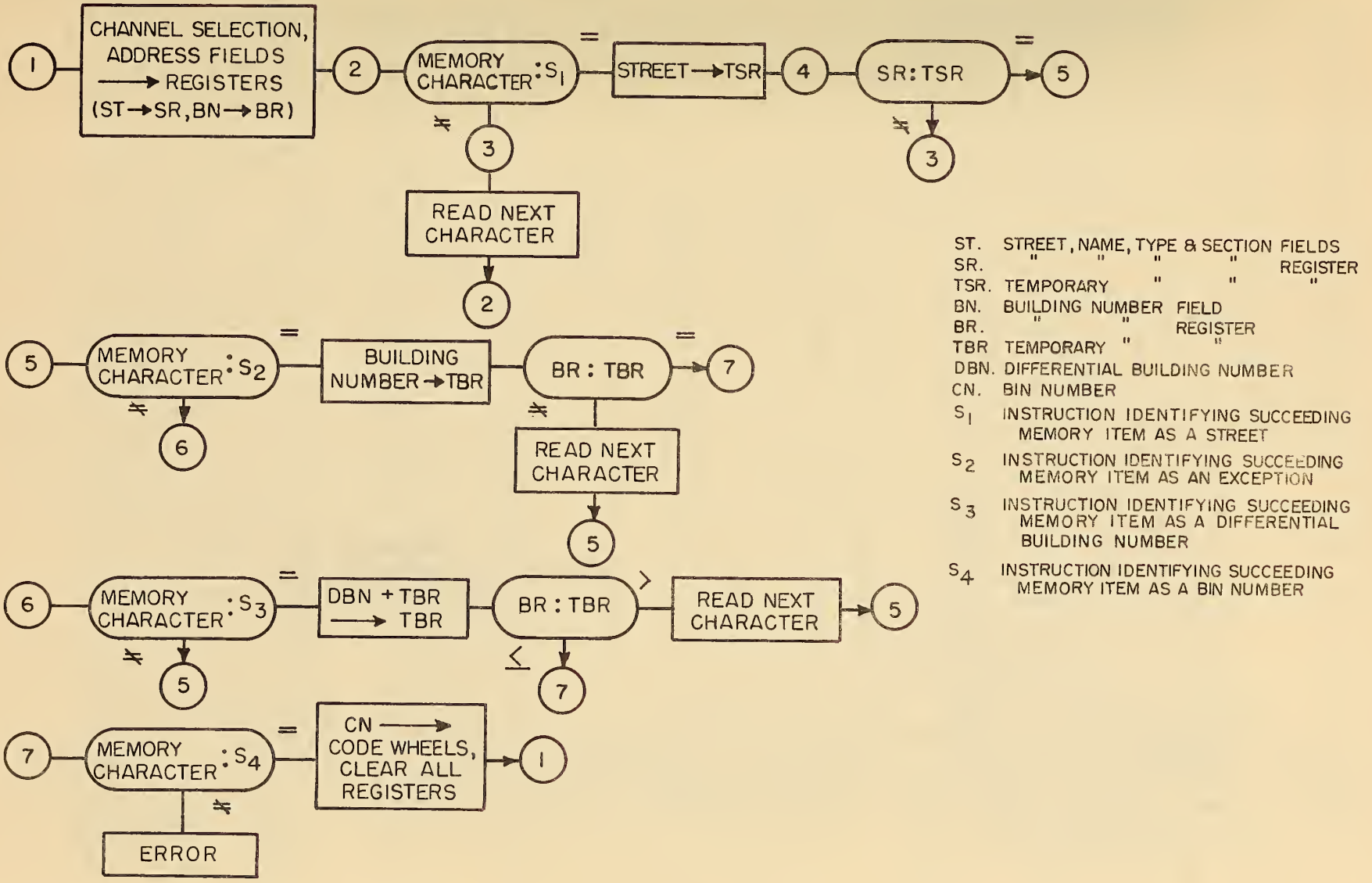
(1) Type the short code,

(2) Type an end-of-building number symbol,

(3) Type a letter which would serve as a street, since the street is the key item in the programming,

(4) Strike the Forward key.

This is clearly unsuitable since it greatly detracts from the anticipated time savings accruing from short codes. The programming must therefore be modified to accept short codes without special symbols. The operator should type only the code and strike the Forward key. This might be done by testing the number of characters on an envelope before Step 1 of the standard programming. If there are three or less characters, then it is a short code and can be immediately sent to the TSR and compared with the short codes entered in the channel or channels reserved for them.



ST. STREET, NAME, TYPE & SECTION FIELDS
SR. " " " " REGISTER
TSR. TEMPORARY " " "
BN. BUILDING NUMBER FIELD
BR. " " REGISTER
TBR TEMPORARY " "
DBN. DIFFERENTIAL BUILDING NUMBER
CN. BIN NUMBER

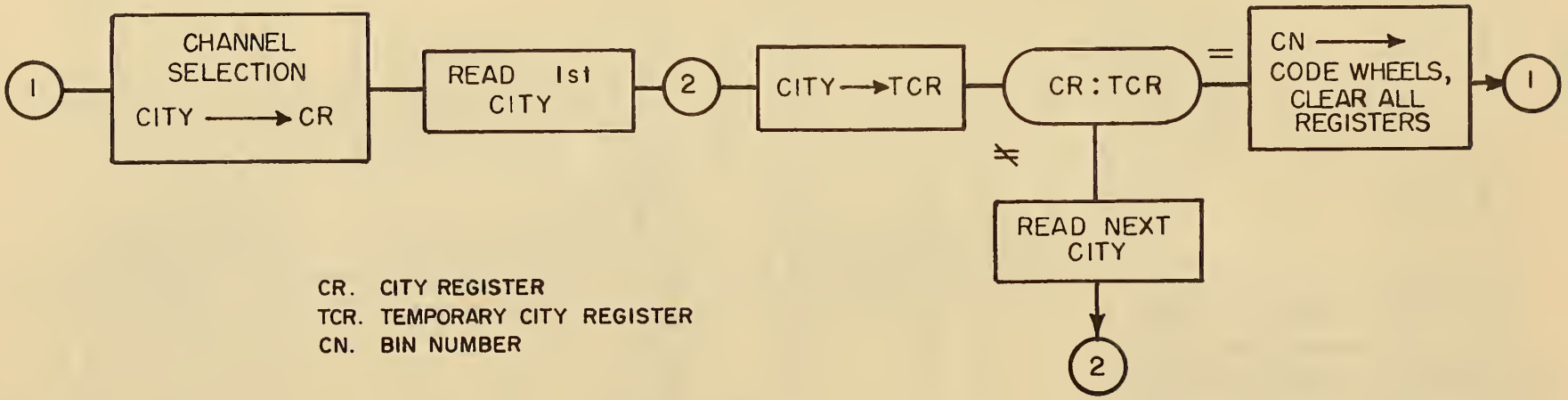
S_1 INSTRUCTION IDENTIFYING SUCCEEDING MEMORY ITEM AS A STREET

S_2 INSTRUCTION IDENTIFYING SUCCEEDING MEMORY ITEM AS AN EXCEPTION

S_3 INSTRUCTION IDENTIFYING SUCCEEDING MEMORY ITEM AS A DIFFERENTIAL BUILDING NUMBER

S_4 INSTRUCTION IDENTIFYING SUCCEEDING MEMORY ITEM AS A BIN NUMBER

FIGURE I. LOCAL MAIL SORT PROGRAM



CR. CITY REGISTER
TCR. TEMPORARY CITY REGISTER
CN. BIN NUMBER

FIGURE 2. OUTGOING MAIL SORT PROGRAM

The same problem occurs with outgoing mail since many cities will be similarly designated. A test can also be performed in this case on the number of characters.

4.2 Minimum Redundancy Coding

The break, street name, type, and section organization of Washington, D.C., provides a unique representation for every building. But it does this redundantly. That is, all fields are not required in every case to uniquely specify individual buildings. This arises from the exclusive nature of the break organization. All streets with the same street name do not include the same numeric range of breaks. Therefore, buildings on identically named streets in different sections often are still uniquely identifiable even without the Street Type and Direction fields. This characteristic implies a potential reduction in coding time if decision rules can be formulated for extracting from an address the minimum information required for its unique specification. The memory entries would of course have to correspond to these rules also.

Four interrelated rules seem to be applicable:

(1) Omit "street" as an entry in the Street Field for all streets.

(2) Omit the Street Type Field entirely for named streets.

(3) Omit NW as a section designation in the Direction Field for all streets.

(4) Omit Street Type and Direction fields for all named streets.

Study of the Postal Zone Directory, Washington, D.C., 1959, indicates that (1), (2), and (3), when applied, produce unique address specifications; i.e., letters are sorted to their proper carriers. Rule (4) results in some missorting, i.e., letters can occasionally be sorted to the wrong carriers, but these carriers generally seem to be in the correct zone. Consequently, only intra-station rather than inter-station transfer of this mail is required. However, even this level of redundancy is probably unacceptable.

The value apparently accruing from these rules may not be sufficient to justify their use. Omitting the Section Designation saves one key stroke. But the letters must still be released by the striking of a Forward key. If the Section Designation keys are linked to the "Forward" key so as to automatically actuate it when they are depressed, the one stroke saving is achieved without losing the section information. Furthermore, the operator's decision to type or omit the Street and/or Section Designation fields might take as long as the key striking time. In addition, the address organization redundancy was noted for the case of uncoded street names. Coded street names

are abbreviated and consequently have a lower information content.[1/] The previous redundancies may therefore be necessary for unique address specification. Another factor is that even though the omission rules may be practicable and economical they may have a high "opportunity cost". That is, the time and energy expenced by the operator in making these decisions as he codes might be more profitably applied in other activities, e.g., more careful examination as to whether or not the address has a short code.

4.3 Non-Standard Addresses

The public's addressing habits are such that mail is often addressed in a non-standard format. Some of the more prevalent variants will now be discus sed in relation to the serial program's flexibility for accepting them.

4.3.1 Section Designation

A common departure from the standard form is the omission of the section designation. This occurs more frequently for NW streets than for those in other sections and the present convention is to consider such streets to be in the NW with the exception of several major arteries in other sections. When confronted with an address whose section is omitted, the programming should ideally perform as follows:

(1) If the street and building number appear in only one section it should be sorted to that section.

(2) If the street and building number appear in more than one section, including the NW, it should be sorted to the NW section.

(3) If the street and building number appear in more than one section (excluding the NW) the letter should be sorted to an error bin.

The present serial programming is designed with the section as an integral part of the street field. Consequently, it can not now meet the above conditions. But modifications for achieving them can be made in the coded information fields and/or the memory entries and programming. A possible method would be to omit the section designation

---

[1/] Ideally, their relative information is the same. But, in general, two different street names might yield the same coded street name. Consequently, the street type and s ection fields would be required for their unique specification.

from the coded addresses and the memory for all NW streets and streets in other sections which: (1) are commonly specified without a section designation; and (2) do not have names and building numbers identical to any streets in the NW. This seems feasible but would require the coding operator to know and recognize the streets, other than in the NW, which are to be so treated. An objection is that some streets in the delivery area appear outside the District and have no section designations while having the same name as certain NW streets. They would apparently be sorted to the NW. But, fortunately, these streets' breaks and those of the NW section are exclusive. Thus, missorting is prevented.

An alternative method would be to code the section and retain its designation in the memory, but to extend the special order concept. The s_1 symbol preceding the NW street entries, would be modified to perform two tests: The first in the standard manner; and if the first did not yield equality, a second, performed on only the street name and type fields stored in the TSR. Then, a letter to a NW street would be sorted properly with or without its section designation. This procedure requires that the NW street entry be placed last in a group of identical street name entries. It further requires that if the second test was performed by shifting the TSR contents one character, they must be reshifted without losing the character after the test is performed. This special order extension can be applied to selected streets in other sections as long as these streets do not have any house numbers that are the same as those on the identically named NW streets. This method presumes no operator scheme knowledge but would probably require an excessive number of shifting operations and would unduly complicate the electronic circuitry.

A much more flexible solution would be to use a "Test Suppression Instruction" 2/ in the memory as follows:

(1) As the section designation for streets appearing in only one section.

(2) As the street type in streets having only one type.

The instruction character would have the property of always passing an equality test with any other character.

2/ Similar to the "ignore" character of general-purpose computers.

The coding operator would then specify the NW section for addresses with the section omitted, and "street" for addresses with their street type omitted. This would fulfill the conditions specified at the beginning of this discussion without requiring the coding operator to have any scheme knowledge. Furthermore, it would correctly sort letters with the "street" street type omitted.

In general then, the program can be modified so that NW mail with no section designation will be properly sorted and mail with no section designation, destined for streets in other sections, can also be correctly sorted. The restriction is that in any set of identically named streets, mail lacking a section destination and destined for building numbers common to that group of streets, is sorted to only the NW street. This restriction is not critical in view of the present convention of assuming such mail to be NW. By using the Test-Suppression character in the memory, the program can in addition sort letters having their street types omitted. The restriction in this case is that in any set of identically named streets, mail lacking a street type and destined for building numbers common to that group of streets, is sorted to only the "street" street.

4.3.2 Intersections

Buildings are often specified by an intersection of two streets rather than a building number and street. The program can handle this situation if the first street is entered in the Building Number Field, the second in the Street Name Field, and the Letter "i" in the Street Type Field. The corresponding directory entries would be entered as exceptions.

4.3.3 Box Mail

Lock box mail can be sorted by entering the box number in the building number field and a "b" in the street field. Corresponding memory entries would consist of a b "street" and the upper limit of the box numbers for successive stations with each followed by a bin number for that station. This procedure is analogous to the exception and break procedure used with the standard address form.

4.3.4 Named Destinations

Some mail for many destinations such as firms, embassies, hotels and apartments is addressed to the name of the destination and does not specify the building

number. Some of this mail is for short-coded destinations and consequently is easily sorted. But the remainder presents a problem. Some of the names could be entered in the memory, but many are difficult to reduce and code. Furthermore, for efficient utilization of coding time, the operator has to know whether the specific name being coded is in the memory. A reasonable method of handling these addresses would seem to be:

(1) Train operator to recognize and code names which are entered in the memory and do not have short codes. The recognition of names as non-short-coded should require less memory effort than the recall of short codes.

(2) Short-code the remainder to bins corresponding to the type of destinations; e.g., firms, schools, apartments, etc.

4.4 Special Situations

In addition to the above mentioned non-standard address forms, the following situations commonly occur:

4.4.1 Room Numbers

Mail for a few large multi-carrier buildings often includes the room or office number. By entering the room number in the Building Number Field and an abbreviation of the building's name in the Street Name Field, this mail can be sorted directly to the appropriate carriers rather than to a bin corresponding to the building. This of course presumes operator knowledge of the multi-carrier buildings stored in the memory.

4.4.2 Odd-Even Carriers

Some carriers deliver only to the odd or even side of certain breaks. The present programming does not differentiate between these two cases, but it could do so if the special order concept were extended. A new s3 symbol could be placed before the break. Following the break would be the bin numbers of the carriers serving the odd and even sides of the break. The symbol would initiate a test on the building number stored in the Building Register. If it were odd, the first bin number would be sent to the codewheel setter. If it were even, the second would be sent.

An alternative method would be to enter in the memory as exceptions the individual buildings composing the odd-even breaks.

4.5 Summary

Serial programming is sufficiently flexible to accommodate a variety of address forms if:

(1) A Test Suppression Instruction character is used in the memory entries for a specific set of streets.

(2) Psuedo "streets" and "building numbers" are used for the non-standard address forms. Table VI follows.

5. BINARY CODING AND DIRECTORY PREPARATION

5.0 Introduction

Three interdependent questions arise in the assignment of BCD bit combinations to the alpha-numeric and special order characters:

(1) How many characters are required?

(2) How many unique bit combinations are necessary for their specification?

(3) How many bits are required to specify each character, and should all characters be composed of the same number of bits?

5.1 Characters Required

The characters required differ for local and outgoing mail. Local mail utilizes:

(1) All digits and lower case alphabetics in the Building Number Field.

(2) All digits (including 1/2) and lower case alphabetics in the Street Name Field.

(3) The lower case alphabetics s, a, r, p, d, c, l, t, and y in the Street Type Field.

(4) Four Section Designation symbols in the Section Field.

Outgoing mail requires:

(1) All lower case alphabetics and all upper case alphabetics except E, H, J, Q, and U in the City Field.

(2) All lower case alphabetics in the State Field.

TABLE VI

Washington, D.C. Local Field Entries (Plan E-1)

	Address Type	Building Number Field		Street Name Field	Street Type Field	Section Designation Field
(1)	Standard Addresses:					
	(a) Complete	Bldg. No. [1]	*	Coded Street Name	s, a, r, p, d, l, c, t, or y	(Ne), (sE), (Sw), or (nW)
	(b) Street type omitted	Bldg. No.	*	Coded Street Name	See [2]	(Ne), (sE), (S w), or (nW)
	(c) Section Designation omitted	Bldg. No.	*	Coded Street Name	s, a, r, p, d, c, l, t, or y	See [3]
(2)	Intersection	Coded First Street Name	*	Coded Second Street Name	i	Blank
(3)	Boxes	Box Number	*	b	Blank	Blank
(4)	RFD Routes	Route Number	*	r	Blank	Blank
(5)	Named Destinations:					
	(a) Apartments			a		
	(b) Associations			z		
	(c) Clubs	Coded	*	k		
	(d) Companies		*	c		
	(e) Embassies & Legations	Name	*	e	Blank	Blank
	(f) Hospitals		*	h		
	(g) Hotels		*	t		
	(h) Office Bldgs. [4]		*	b		
	(i) Postal Stations		*	p		
	(j) Schools		*	s		
	(k) Unions		*			
	(l) Miscellaneous		*	m		

[1] The * symbol is here used as the End-of-Building-Number-Field Symbol.

[2] If the Street Type is omitted, the letter s should be entered.

[3] If the Section Designation is omitted, the (nW) designation should be entered.

[4] Letters with room numbers for certain, as yet unspecified, multi-carrier buildings are coded to permit sorting to individual carriers. The room number is entered in the Building Number Field; the coded building name in the Street Name Field, and the letter b in the Street Type Field. The * symbol is used as before, and the Section Designation Field is left blank.

In addition to these, special instruction symbols are necessary. The characters (approximately 70) are summarized below:

(1) Digits 0 thru 9, and 1/2,

(2) Lower case alphabetics a thru z,

(3) Upper case alphabetics A thru Z, except E, H, J, Q, and U,

(4) Section designations,

(5) Street symbol,

(6) Exception symbol,

(7) Break symbol,

(8) Hundreds-unit break symbol,

(9) Odd-even break symbol,

(10) Bin number symbol,

(11) End-of-Building Number Field symbol, (does not appear in memory),

(12) Outgoing residue test instruction,

(13) Character test suppression instruction,

(14) Possibly more special symbols if differential storage is used for outgoing programming.

5.2 Unique Bit Combinations Required

A distinction exists between the unique characters and their binary representations. The characters, to be effectively used by human keyboard operators, must be unique. But some individual binary combinations may be shared among characters with no loss in information content, because of the uniqueness of the fields. The set of incoming characters would bound this sharing since it includes the set of outgoing characters. Some of the bit-sharing alternatives are:

(1) The character 1/2 can be specified by an alphabetic character.

(2) Numbers can be used for the section designations.

(3) Any of the incoming characters not used in the outgoing directory are usable as special order symbols in outgoing programming (as long as local and outgoing mail are not sorted simultaneously).

An allied reduction in the number of unique bit combinations may be realized by using only three distinct special order symbols. Combinations of these could then be interpreted as the remaining symbols. It consequently appears that fewer than 64 unique bit combinations can adequately specify all characters.

5.3 Bits per Character

The question of how many unique characters are required is related to whether or not all characters are specified by the same number of bits. As has been discussed previously, information is read from the drum as a series of pulses after instruction characters are interpreted. This requires all instruction symbols to have an equal number of bits. The orders then control the transfer and interpretation of the alpha-numeric characters following them. Since alphabetic and numeric characters can occur in any field, they must both have an equal number of bits. Furthermore, alpha-numeric and instruction characters are interspersed on the drum. Therefore, all characters must be specified by the same number of bits.

A six information-bit code can specify all characters with a limited amount of bit-configuration sharing, while a seven information-bit code specifies all characters uniquely. The seven bit code also permits characters to have meanings independent of their field position or mode of programming.

5.4 Data-Processing Equipment Considerations

The preceding general analysis has been independent of considerations imposed by the data-processing equipment required for memory preparation. The process of transferring and modifying the presently used directories to the coded and ordered form to be fed into the memory will be an extensive operation requiring standard data processing equipment. The symbols used in this equipment do not include the variety required by the programming. [1] For example, IBM card punches have a vocabulary of only 47 symbols: the numerics 0 thru 9; alphabetics A-Z; and 11 punctuation and operational symbols (no distinction is made between upper and lower case alphabetics). Consequently, unique symbols do not exist for the 60-odd characters required. A reasonable solution using two symbols to specify certain characters is suggested below.

---

[1] Recently announced equipment may obviate this problem for file-directory preparation. But the letter coding process necessitates specifying many characters by two key-strokes if the coding key-board is not to contain individual keys for all the characters required.

5.4.1 Coding Vocabulary

The totality of characters necessitated by the coding are:

(1)	Digits	10
(2)	Lower Case Alphabetics	26
(3)	Upper Case Alphabetics	20
(4)	Section Designations	4
(5)	End-of-Building-Number Field Symbol	1
		61

Upper case alphabetics can be composed of two characters: (1) a shift character; and (2) the desired alphabetic character. Then, the coding keys required for their specification are:

(1)	Digits	10
(2)	Lower Case Alphabetics	26
(3)	Upper Case Alphabetic Shift Key	1
(4)	Section Designations	4
(5)	End-of-Building Number Field Key	1
		42

This technique increases the length of coding fields as follows:

Local Fields

Field	Characters
Building Number	Up to 6
End-of-Building-Number-Field Symbol	1
Street Name	Up to 6
Street Type	1
Section Designation	1
Total:	Up to 15

Outgoing Fields

Field	Characters
City	Up to 6
State	2
Total:	Up to 8

5.4.2 Memory Vocabulary

The characters used in the memory include all the coding characters, except the End-of-Building-Number-Field symbol, together with the following instruction characters:

(1) Street Symbol:	Identifies the entry following it as a street name, type, and section.
(2) Break Symbol:	Identifies the entry following it as the upper limit of the first break on a street.
(3) Hundreds-Unit Break Symbol:	Identifies the entry following it as the hundreds-unit differential between two successive breaks on a street.

(4) Odd-Even Break Symbol:	Identifies the entry following it as a break where odd and even building numbers are serviced by different carriers.
(5) Exception Symbol:	Identifies the entry following it as an exceptional building number.
(6) Bin Number Symbol:	Identifies the entry following it as a bin number.
(7) City Symbol:	Identifies the entry following it as a city.
(8) State Residue Symbol:[2/]	Initiates the performance of the residue tests in the Outgoing translation program.
(9) Character Test - Suppression Symbol:	Suppresses the performance of logical tests on the character position it occupies. This is used only in the memory. For example, it is entered in the Section Designation Field of streets appearing in only one section and in the Street Type Field of street names having only one type.

These instruction symbols (excluding (9)) may be composed of two characters: an Instruction-Prefix symbol common to all; and an Instruction-Suffix symbol (possibly the lower-case alphabetics) specifying the particular instruction.

2/ Residue tests can be performed as equality tests if suitable "cities" are made up including the Character Test Suppression Symbols. Then, both the S tate Residue and City Symbols are unnecessary.

As stated previously, the compound-symbol technique arises from the vocabulary limitations of the coding keyboard and standard punched card-or-tape equipment used for:

(1) Preparing the "source document" directory file used to load the drum, and

(2) Updating the directory file.

However, the character-codes used internally by the special-purpose computer can be arbitrary. Therefore, all characters used by the computer can be specified by single symbols if an input-output translation unit is incorporated into the system. This unit would:

(1) Translate single and compound-symbol codes of the file directory into single-symbol language of the special-purpose computer during drum loading, and

(2) Translate the computer language into file directory language for drum checking and debugging.

This alternative would reduce the memory space needed by instructions and upper-case alphabetics. But it would increase the space used by the remaining characters since it requires a seven information-bit code (the compound symbol technique uses a six information-bit code). Assuming a parity bit is included, the local directory space saving per item equals $(14 + 7X) - (8 + 8X)$, where X is the number of characters in an item. The saving decreases rapidly and is zero at X equals 6 characters. Assuming the average local directory item to be 3 characters, the average saving per item equals 3 bits, or a total memory saving of less than 10%. Assuming the average item to be 4 characters, the total memory saving is only 5%. Greater savings seem possible in the outgoing directory since the compound symbol technique increases the City Field to a possible maximum of 6 characters instead of 4. But if instruction characters are used in the outgoing directory, the average city item should be reducible to approximately 5 characters. Then the saving will equal 3 bits per item, or less than 10%.

It does not appear that a translation unit can be justified economically for full-serial storage. The memory saving is negligible. Furthermore, the use of a non-standard translation unit may significantly lower over-all system reliability.

5.5 Suggested Coding Keyboard

A possible coding keyboard using the compound character technique is shown in Figure 3 (P43) and described below. The keyboard would be used only for coding. File directories would be prepared with standard card-punch machines.

Printing Keys

Number	Key
1 - 10	Digits: 1 - 10
11 - 36	Lower Case Alphabetics: a - z
37 - 40	Section Designations: Ne, sE, nW, and Sw. These automatically actuate the forwarding mechanism when depressed.
41	End-of-Building-Number-Field symbol
42, 43	Upper-Case Shifts: These are printing characters which together with lower case alphabetics define their two-character upper-case equivalents.

Operational Keys[3/]

44	Forward Key: releases coded envelope and feeds in next one for coding. Used only to forward non-standardly addressed Local letters standardly addressed Local letters are forwarded by the Section Designation Symbols).

3/ Another operational key, valuable when successive letters have the same destination, would be a duplicating key. When depressed, it would imprint the stored coded fields of the previous address.

Number		Key
45 [4]		Outgoing Forward Key: Used only to forward outgoing letters. Also separates them from local letters.
46 [4]	(a)	Nixie Key (upper case): separates "nixie" (misaddressed or miscoded) mail.
	(b)	Uncancelled Key: separates uncancelled mail.
47 [4]	(a)	Airmail, Special, and Foreign Key (upper case): separates these types of mail.
	(b)	25 Key: separates Local official mail.

5.6 Directory Preparation

The preparation of the local and outgoing directories may be described as a series of 7 steps:

(1) Adapting and ordering source documents to a usable form,

(2) Coding entries,

(3) Ordering entries alphabetically,

(4) Deriving and inserting differential numbers [5]

(5) Inserting instruction symbols and bin numbers,

---

[4] Keys 45 - 47 are "presort keys", i.e., they separate their associated types of mail, from the remainder of the letters, at the coding station. This concept can be extended to keys 37-40. Then local mail would leave the coding process sorted to section and each section's mail could be individually sorted by the special-purpose computer. This would not materially reduce memory access time under the present memory system. But under a different system, e.g., a high-capacity drum feeding into a large core working storage, significant savings could be effected.

[5] These may not be used in the outgoing directory.

(6) Transcribing and editing entries to a form suitable as input to memory drum,

(7) Preparing sub-directories,

(8) Preparing files and cross-files between original uncoded entries and their alphabetically ordered coded counterparts.

Directory preparation is directly related to the input-output components of the system. These are discussed in the next section of the report.

5.7 Summary

Some salient characteristics of the binary coding problem have been presented. These are discussed further in the next section of the report. It is suggested that a 6 information-bit code be used and that upper-case and instruction characters be specified by two characters. A possible coding keyboard is also suggested.

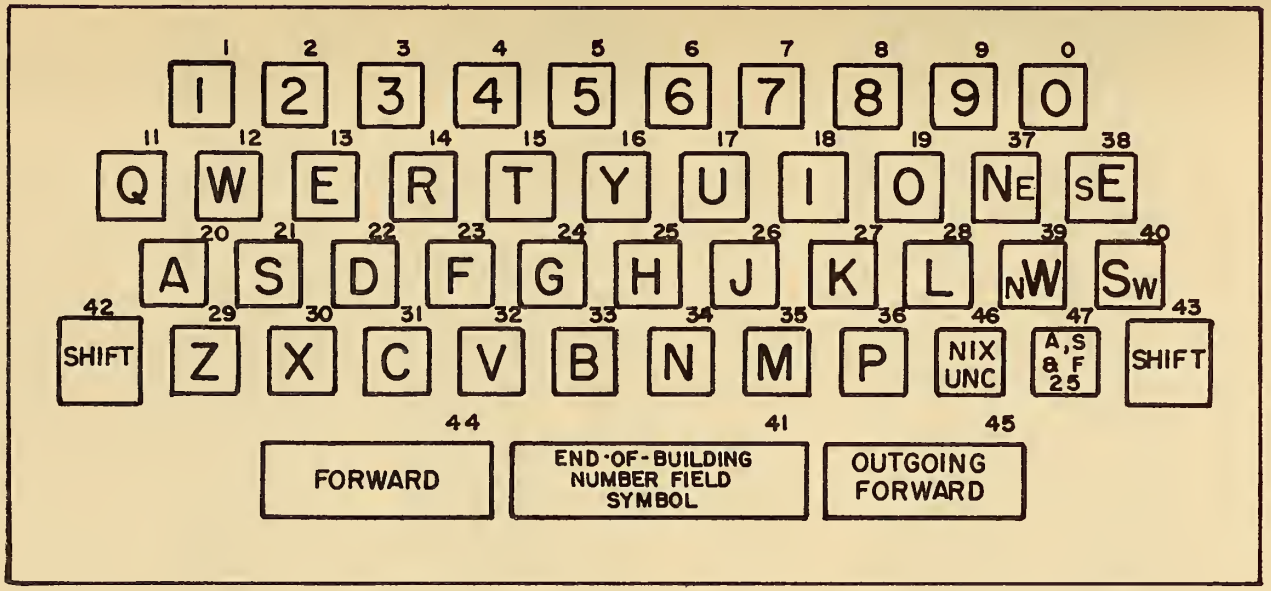


FIGURE 3. CODING KEYBOARD

6. DISCUSSION OF SYSTEM

6.0 Introduction

The report has been primarily concerned with the local and outgoing programs and their adaptability for accommodating special address cases. This area is now discussed further and some characteristics of the computer system presented.

6.1 Programming

The flexibility necessitated by special types of addresses was achieved by introducing special instruction symbols. But, it should be noted that the program can treat all cases as standard breaks and exceptions if a repetition of memory entries can be tolerated. [1/] That is, a tradeoff exists between increased memory space and a less complex order structure. For example, addresses having their section designation omitted were treated as NW by the coding operators. And memory entries for all streets appearing in only one section were entered with the Test Suppression Character as their section. Consequently, error mail was sorted correctly if it were destined for:

(1) A NW street, or

(2) A street appearing in any one of the NE, SE, and SW sections.

Error mail for streets appearing in more than one section including the NW would then be sorted to the NW street (assuming the break limits include the building number). Error mail for streets appearing in more than one section excluding the NW would then be sorted to an error bin. A similar procedure was used for addresses having their street types omitted. Both these cases can be treated with identical results if the entries previously entered with the Test Suppression Instruction are duplicated in the memory and if the coding operator treats the

---

[1/] In fact, all local exceptions can be treated as breaks if they are entered as "sub-breaks" ., e.g., exception 1225 and upper break limit 1399 on I S t., NW can be stored as s_1 i s (Nw)s_3 1224s_3 1225s_3 1399 instead of s_1 i s (Nw)s_2 1225s_3 1399. Consequently, by duplicating the exception entries the s_2 (exception) instruction is no longer necessary.

misaddressed mail as before. For example, Nicholas Ave. appears only in the SE, but it would be entered in the memory as both Nicholas Ave., SE and Nicholas Ave., NW with a corresponding duplication of the break, exception, and bin number entries. The same procedure would be used for street names having only "street" as their type. Thus, incompletely addressed mail can be sorted to the same extent as now done manually by using either a special instruction or duplicating entries.

Odd-even breaks were treated similarly. A special order was entered before these entries to identify them. An alternative method is to enter all buildings in these breaks as exceptions together with their bin numbers. This technique is also applicable to the non-hundreds unit break limits. Again, successive buildings can be treated as exceptions until the hundred-unit building is reached. Then the regular differential break procedure can be resumed.

The programming comparison tests were, for clarification purposes, described as taking place in 4 "local" registers and 2 "outgoing" registers:

(1) Street Register

(2) Temporary Street Register

(3) Building Number Register

(4) Temporary Building Number Register

(5) City Register

(6) Temporary City Register

But this multiplicity of registers is unnecessary. Cities can be stored as "exceptions" and residue cities as "breaks" on their state "streets" thus obviating the need for (5) and (6) [2/]. Furthermore, the local comparisons on street and building number are performed serially.

2/ And also permitting mixed local and outgoing mail to be sorted simultaneously. If this is done, however, some differentiation is required between local and outgoing destinations having identical short codes. This might be accomplished by having the Outgoing Forward Key on the coding keyboard automatically imprint an additional character. The outgoing memory short codes would then be stored including the additional character. This would allow duplication between local and outgoing short codes memorized by the coding operator while yielding operationally distinct codes.

Thus the two temporary registers may be replaceable by a single register holding the entries successively read from the memory.

6.2 Memory

The translation from coded addresses to bin numbers is performed by two testing routines:

(1) Channel selection (predirectory "table look-up")

(2) Channel search (directory search)

Several alternative storage patterns for the drum information are now presented. All assume the predirectory is stored on the same drum as the directory.

6.2.1 Serial Predirectory and Directory

The first method is shown in Figure 4 (P52). An m channel predirectory and an n channel directory are stored on an m + n channel drum. Randomly arriving coded addresses are compared with the first predirectory channel, which stores the greatest entries in the first 1/m channels of the directory. 3/ If the first channel includes the directory channel entry (and its channel number), reading is switched to the appropriate channel. If the first channel did not include the appropriate entry, reading is successively switched to the remaining m-1 channels until the correct one is located. Predirectory access time is therefore 0.5m drum revolutions where m may differ for local and outgoing predirectories. Directory access though, is complicated by the outgoing residue entries stored on certain channels. Since directory entry is random the stored residue test entries will result in appreciable missorting. Therefore, either reading should always commence at one point on all channels (increasing access time by 0.5), or the residue test output should be suppressed until the drum has made a complete revolution without an exact match occurring.

3/ If reading is always begun at one point on the drum (thus increasing access time by 1/2 drum revolution) the correct entry is specified when the comparison test results change from $\leq$ to >. If entry is random, the channel is specified when test outputs change from $\leq$ to > or from > to $\leq$. In either case the channel numbers associated with the entries must be offset by one entry, e.g., channel number for entry i is stored after entry i+1. (An alternative is to store the channel number before its entry and hold it in a temporary storage register while its entry is tested).

Predirectory and directory searches are simultaneously performable on successive addresses. Thus access time per address reduces to 0.5m. But this is unacceptably high, e.g., a million bit 400 channel drum requires approximately 8 predirectory channels or 4 drum revolutions access time.

6.2.2 Serial Directories, Predirectory Channel Selection

Access times are reducible to a single drum revolution if the predirectory concept is extended to the predirectory itself as shown in Figure 5 (P53). The greatest entry in each predirectory channel is listed at the beginning of each channel. These key entries (or the channel reading heads) are successively offset by a constant distance. Entering coded addresses are compared with these key entries until their channel if found. [4] Predirectory access time is then $0.5 + 0.25\ \theta/\pi$ drum revolutions. The latter term should be negligible in comparison to the first. Directory reading can then always begin from line bb' thus precluding any residue test output suppression procedures. Directory access time is 0.5 drum revolutions assuming entries are stored in the shaded area of Figure 5. But if entries are stored there simultaneous searches on two channels will occasionally occur. Therefore a duplication of directory or predirectory comparator circuitry or some operational interchangeability between predirectory and directory comparator circuitry is required. This complication can be bypassed if angle θ is negligible. [5] Then the shaded area of Figure 5 is also negligible and can remain empty. Thus all channels end at line aa' and simultaneous comparisons never occur.

A further extension of the predirectory concept would be to specify not only the directory channel, but also the sector within that channel. Figure 6 is an example of a 4 sector per channel system which decreases directory access time by a factor of 4. A modulo 4 counter is increased by unity every 0.25 revolutions of the drum. Consequently the modulo 4 sum of the channel sector read from the predirectory and the counter value specifies the

---

[4] The key entries could just as well be the lowest entry in each channel and switching performed to the previous channel as soon as the key entry becomes > than the entering address entry.

[5] Angle θ can also be reduced by: (1) Revising the placement of the key entries and increasing predirectory comparator circuitry and/or; (2) Storing the key entries serially, but the characters comprising the entries, in parallel.

reading head to be activated. This would increase the predirectory by a factor of slightly more than 4 and would require four times as many directory reading heads and might also require more complex predirectory comparator circuitry.

6.2.3 Serial Directory, Integral Predirectory

As shown in Figure 7 (P 55) the predirectory channel selector technique can be used within the directory thus making a separate predirectory unnecessary. The greatest entries are offset listed. Then a predirectory search is performed on the helical path traced by the key words. [6] When the correct channel is found, directory reading begins immediately. [7] Average channel selection and reading times are then each 0.5 drum revolutions. But since predirectory channel selection tests are constrained to begin on line aa', operational access time is either 1 or 2 drum revolutions; and average access time is therefore 1.5 drum revolutions. This can be reduced to 1.0 drum revolutions by permitting random entry and duplicating comparator circuits thus allowing simultaneous predirectory and directory searches.

6.2.4 Parallel Storage, Integral Predirectory

As has been seen, the many predirectory tests appreciably complicate the comparator circuitry. This can be mitigated by storing entries in parallel rather than serially, as shown in Figure 8 (P 56). Key words, in parallel, are still used. The number of predirectory tests is then reduced by a factor equal to the product of the number of characters per entry and the number of bits per character. [8] Angle θ is therefore negligible and the shaded area of Figure 8 can be kept as slack space. Access time, assuming reading always begins at line aa' is then fixed at 1.0 drum revolutions.

Parallel storage's simplicity results from its use of fixed-length fields and simultaneous testing on all characters within a field. A possible parallel memory layout is shown

---

[6] As many independent predirectory comparator circuits as there are helical loops are therefore required.

[7] Only one directory comparator circuit is required if initial reading always begins on line aa' after the previous channel search is completed.

[8] Sets of channels are sought rather than individual channels.

in Table VII (P 51). All local and outgoing entries are treated as streets, exceptions, and breaks. [9] While conceptually simpler than serial storage, parallel storage requires much more memory space since the fields are fixed to accommodate the longest entries. [10] It may therefore be feasible to use a code - translation unit and a 7 rather than 6 information bit code. Table VII assumes a 7 information-bit code and fixes each entry at 6 channels. A 6 information-bit code would necessitate storing instructions and upper-case alphabetics as compound characters thus increasing field length and memory size by 33.3%.

The conceptual and operational simplicity of parallel storage and programming suggest that they may be feasible alternatives to the serial methods that have been presented. No separate predirectory is required and all comparison tests are done in a single comparator circuit. Furthermore, access time is fixed at 1.0 drum revolutions thus simplifying the program timing and residue operations. [11]

6.3 Input-Output Considerations

The system's input and output operations are:

(1) File directory loading

(2) Coded Address read-in

(3) Checking and debugging read-out

(4) Bin number read-out.

---

[9] The differential storage procedure of serial programming is no longer applicable since field lengths are fixed.

[10] A local file directory of 3500 streets, 7500 breaks and 7500 exceptions (including special address forms) would require approximately 1.5 million bits (assuming each character to be 8 bits long). A 1.5 million bit drum could store approximately 12,000 cities. A serial storage system having the same 1.0 drum revolutions access time would require only about 3/4 as much memory space to store the same number of entries. But as has been seen it would require duplication of comparator circuitry and a much more complex method of channel selection.

[11] It is assumed that a 12 character entry buffer is used so that an address is always available as soon as line aa' is reached.

Since files will be prepared on punched cards, (1) can be performed by a punched card reader, or a paper tape reader if cards are first converted to tape. If parallel drum storage is used, a translation unit may be required for translating the characters into 7 rather than 6 information-bit codes. If possible, the bit configurations produced by the coding keyboard should be identical to those used internally within the computer. 12/ Thus the translating unit would be used only during directory file loading. 13/ An additional unit may be required for (3) to translate the internal codes to a form acceptable to the output medium, e.g., inquiry typewriter. It appears therefore, that an extended standard computer code should be used since an extra bit position could be readily added and deleted thus simplifying the specification and reading of instructions and upper case alphabetics. A fully arithmetic code might also simplify (4) since bin numbers can be specified by 3 "numbers" yielding all possible combinations of 12 bits.

6.4 Summary

Some major considerations of codesorting have been presented. Both serial and parallel programming are discussed in relation to the memory layout and their flexibility for accommodating standard and special address forms.

12/ This is readily achievable if upper case alphabetics differ from their lower case counterparts by a single constant bit. Then, the shift key could imprint that bit rather than a separate character.

13/ Parallel storage would increase the loading problem since all entries will have to be written linearly across channels rather than serially along a single channel.

TABLE VII

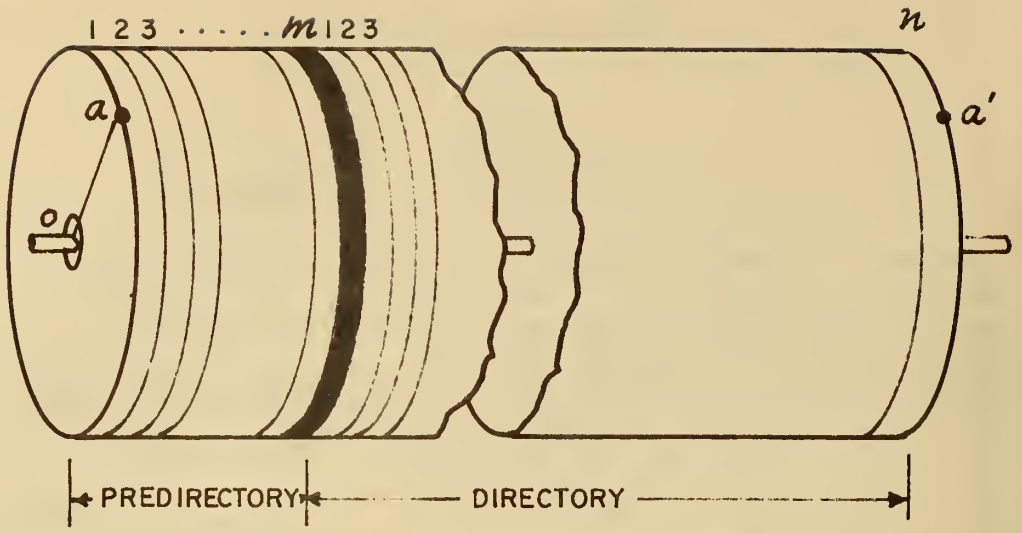
Parallel Storage Entries

Local

x_1	x_2	x_3	x_4	x_5	x_6	Street i
x_1	x_2	x_3	x_4	x_5	s_1	Exception
-	x_2	x_3	x_4	x_5	s_3	Bin Number
x_1	x_2	x_3	x_4	x_5	s_2	Upper Break Limit
-	x_2	x_3	x_4	x_5	s_3	Bin Number
x_1	x_2	x_3	x_4	x_5	s_2	Upper Break Limit
-	x_2	x_3	x_4	x_5	s_3	Bin Number
.	.	.	.	.	.	
.	.	.	.	.	.	
.	.	.	.	.	.	
x_1	x_2	x_3	x_4	x_5	x_6	Street i + 1

Outgoing

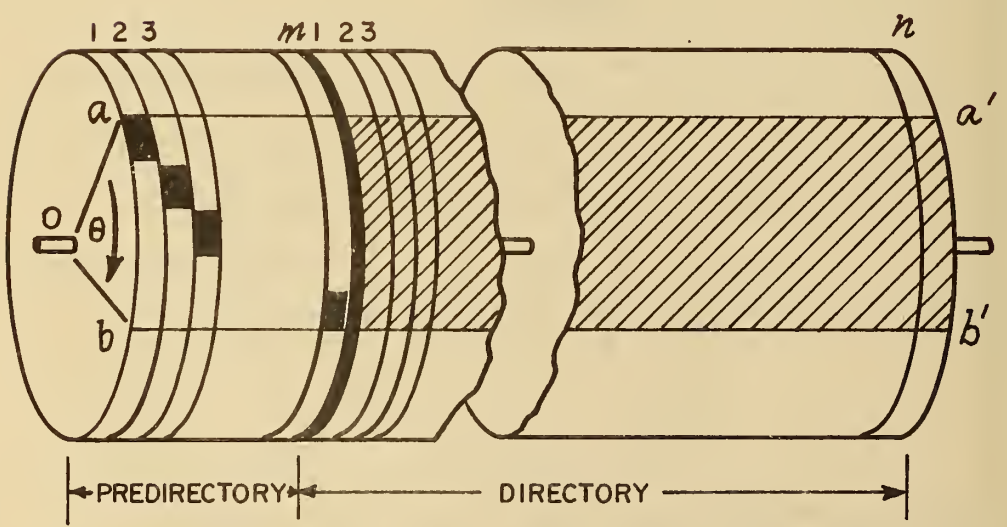
-	-	-	-	x_5	x_6	State j ("Street")
-	x_2	x_3	x_4	x_5	s_1	City ("Exception")
-	x_2	x_3	x_4	x_5	s_3	Bin Number
-	x_2	x_3	x_4	x_5	s_1	City ("Exception")
-	x_2	x_3	x_4	x_5	s_3	Bin Number
.	.	.	.	.	.	
.	.	.	.	.	.	
.	.	.	.	.	.	
-	1	z	z	z	s_2	a to 1 residue ("upper break limit")
	z	z	z	z	s_2	m to z residue ("upper break limit")



AVERAGE ACCESS TIME: 0.5 DRUM REVOLUTIONS

(NOT TO SCALE)

FIGURE 4. SERIAL PREDIRECTORY AND DIRECTORY

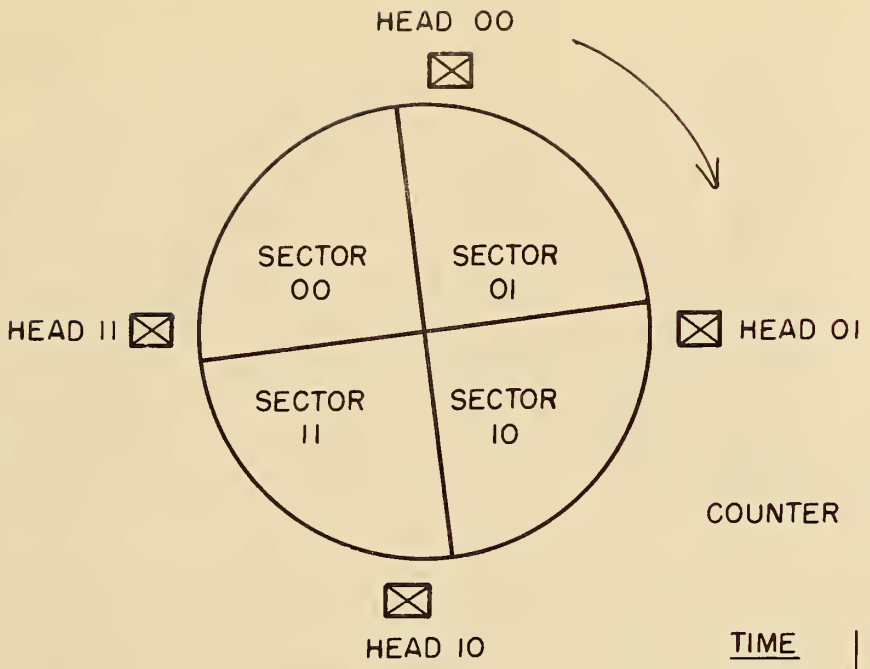


AVERAGE ACCESS TIME: 1.0 DRUM REVOLUTIONS

(ASSUMING θ IS NEGLEGIBLE)

(NOT TO SCALE)

FIGURE 5. SERIAL DIRECTORIES, PREDIRECTORY CHANNEL SELECTION

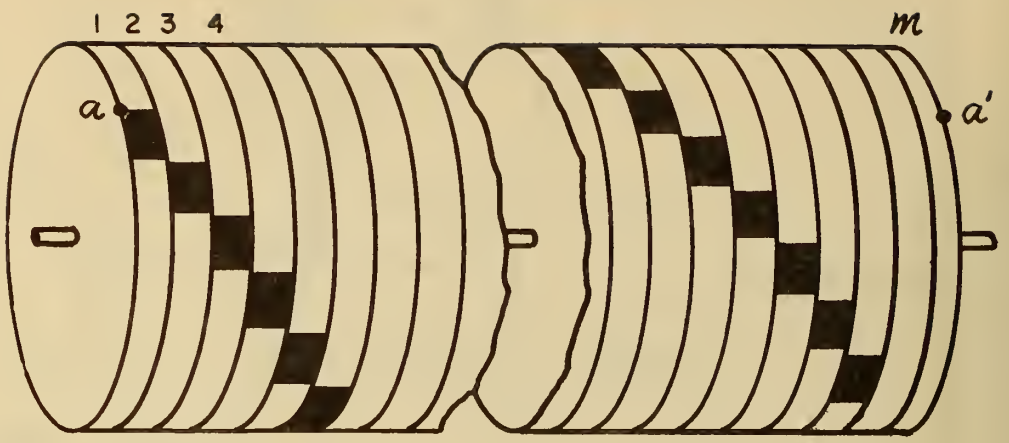


TIME	COUNTER
0	00
$dr/4$	01
$dr/2$	10
$3dr/4$	11

COUNTER VALUE

PREDIRECTORY SECTOR VALUE	00	01	10	11
00	00	01	10	11
01	01	10	11	00
10	10	11	00	01
11	11	00	01	10

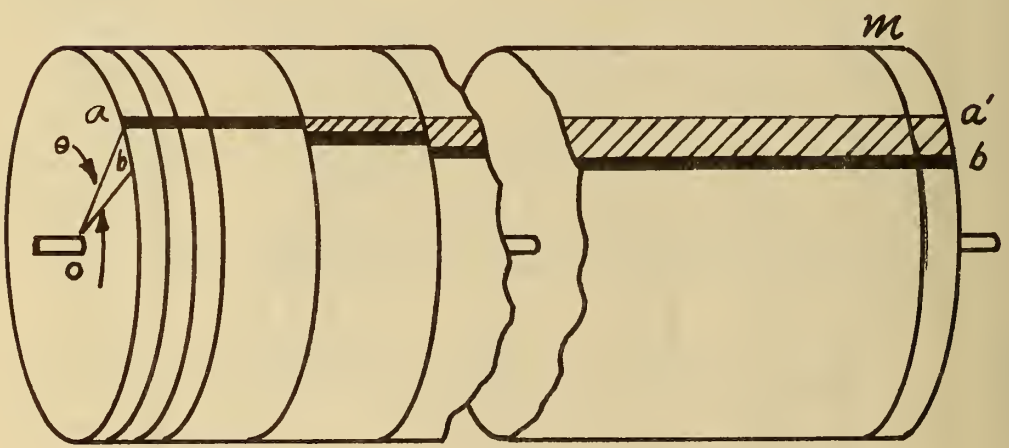
FIGURE 6. SECTOR SPECIFICATIONS



AVERAGE ACCESS TIME: 1.5 DRUM REVOLUTIONS

(NOT TO SCALE)

FIGURE 7. SERIAL DIRECTORY, INTEGRAL PREDIRECTORY



ACCESS TIME: 1.0 DRUM REVOLUTIONS (FIXED)

(NOT TO SCALE)

FIGURE 8. PARALLEL STORAGE, INTEGRAL PREDIRECTORY

U.S. DEPARTMENT OF COMMERCE
Frederick H. Mueller, *Secretary*

NATIONAL BUREAU OF STANDARDS
A. V. Astin, *Director*

THE NATIONAL BUREAU OF STANDARDS

The scope of activities of the National Bureau of Standards at its major laboratories in Washington, D.C., and Boulder, Colorado, is suggested in the following listing of the divisions and sections engaged in technical work. In general, each section carries out specialized research, development, and engineering in the field indicated by its title. A brief description of the activities, and of the resultant publications, appears on the inside of the front cover.

WASHINGTON, D.C.

Electricity and Electronics. Resistance and Reactance. Electron Devices. Electrical Instruments. Magnetic Measurements. Dielectrics. Engineering Electronics. Electronic Instrumentation. Electrochemistry.

Optics and Metrology. Photometry and Colorimetry. Photographic Technology. Length. Engineering Metrology.

Heat. Temperature Physics. Thermodynamics. Cryogenic Physics. Rheology. Molecular Kinetics. Free Radicals Research.

Atomic and Radiation Physics. Spectroscopy. Radiometry. Mass Spectrometry. Solid State Physics. Electron Physics. Atomic Physics. Neutron Physics. Radiation Theory. Radioactivity. X-rays. High Energy Radiation. Nucleonic Instrumentation. Radiological Equipment.

Chemistry. Organic Coatings. Surface Chemistry. Organic Chemistry. Analytical Chemistry. Inorganic Chemistry. Electrodeposition. Molecular Structure and Properties of Gases. Physical Chemistry. Thermochemistry. Spectrochemistry. Pure Substances.

Mechanics. Sound. Mechanical Instruments. Fluid Mechanics. Engineering Mechanics. Mass and Scale. Capacity, Density, and Fluid Meters. Combustion Controls.

Organic and Fibrous Materials. Rubber. Textiles. Paper. Leather. Testing and Specifications. Polymer Structure. Plastics. Dental Research.

Metallurgy. Thermal Metallurgy. Chemical Metallurgy. Mechanical Metallurgy. Corrosion. Metal Physics.

Mineral Products. Engineering Ceramics. Glass. Refractories. Enameled Metals. Constitution and Microstructure.

Building Technology. Structural Engineering. Fire Protection. Air Conditioning, Heating, and Refrigeration. Floor, Roof, and Wall Coverings. Codes and Safety Standards. Heat Transfer. Concreting Materials.

Applied Mathematics. Numerical Analysis. Computation. Statistical Engineering. Mathematical Physics.

Data Processing Systems. SEAC Engineering Group. Components and Techniques. Digital Circuitry. Digital Systems. Analog Systems. Application Engineering.

• Office of Basic Instrumentation.

• Office of Weights and Measures.

BOULDER, COLORADO

Cryogenic Engineering. Cryogenic Equipment. Cryogenic Processes. Properties of Materials. Gas Liquefaction.

Radio Propagation Physics. Upper Atmosphere Research. Ionospheric Research. Regular Propagation Services. Sun-Earth Relationships. VHF Research. Radio Warning Services. Airglow and Aurora. Radio Astronomy and Arctic Propagation.

Radio Propagation Engineering. Data Reduction Instrumentation. Modulation Research. Radio Noise. Tropospheric Measurements. Tropospheric Analysis. Propagation Obstacles Engineering. Radio-Meteorology. Lower Atmosphere Physics.

Radio Standards. High Frequency Electrical Standards. Radio Broadcast Service. High Frequency Impedance Standards. Electronic Calibration Center. Microwave Physics. Microwave Circuit Standards.

Radio Communication and Systems. Low Frequency and Very Low Frequency Research. High Frequency and Very High Frequency Research. Ultra High Frequency and Super High Frequency Research. Modulation Research. Antenna Research. Navigation Systems. Systems Analysis. Field Operations.


NBS