March 11, 1958

Project 7000 File Memo

Subject: Preferred Typewriter Code

1. The standard code for the new typewriter is apparently based on the frequency of use of the characters — the most used characters requiring the least amount of positional displacement from the home positions. The numbers are placed toward the edges of the matrix, since in normal typing these are less used than the letters. However, as a computer I/O device, even in commercial applications the numbers would be used at least as frequently as the alphabetic characters. For this purpose, a revision of the code is in order.

2. The standard code does not allow a binary coded decimal representation of the numerals to be constructed, even by permutations of the columns. There are two examples of character codes which will produce binary coded decimal numerals in memory (Examples 1 and 2). The first preserves the normal frequency distribution assumptions, while the second makes the assumption that the numerals as a class will be used more frequently than the alphabetics. The binary codes for the numerals are to be found in columns 3-6 of the code table appended. It can be proved that this is the only choice of consecutive columns, ordered left to right, for which a set of binary coded decimal numerals can be chosen. (There are other choices possible using permuted or non-consecutive columns — none of the ones investigated show any advantages over using columns 3-6.)

   The disadvantage of the codes of Examples 1 and 2 is that neither have a constant pattern in the four computer unused bits. This means that they cannot be provided for a 4-bit byte decimal field en masse, for the purpose of generating the typewriter code on output. There is, however, a choice of codes (Example 3) which accomplishes this. If the code is transmitted between typewriter and computer in R_5, R_2, S, C, R2A, R_1, T_1, T_2 order, then the numerals will be coded 1101xxxx, where xxxx is the binary coded decimal. Further, this choice of codes places the numerics and special commercial symbols on a par with the most used half of the alphabet. This code has the further property of collecting the numerics within a 3 by 4 area of the head matrix, thereby minimizing indexing for a straight numeric

field. (At least some of the alternate choices based on non-consecutive and permuted columns do not share this property.) Since it can further be proved that no choice can be made which will allow an all-zero high order 4 bits, Example 3 is recommended.

Bruse Moncreiff
Bruse Moncreiff
Product Planning Representative
Project 7000

BM/jcv
cc: Mr. D. W. Sweeney
Dr. W. Buchholz
Mr. J. C. Gibson
Dr. H. G. Kolsky
Mr. E. F. Codd
Miss E. McDonough
Mr. E. W. Coffin

PROPOSED TYPEWRITER CODES

1. Maintains essentially the same frequency distribution as the original, but produces a binary coded decimal number in positions 3-6 of the 8-bit byte for the characters 0-9.

HOME	)	Q	M	P	O	E	D	W	$	*	3/4	0	q	m	p	o	e	d	w	4	8	!
T_1	@	G	F	R	N	T	C	B	¢		=	2	g	f	r	n	t	c	b	6	-	+
T_2	?	J	K	V	Y	A	U	L	%	¯	1/4	/	j	k	v	y	a	u	l	5	9	1/2
T_1, T_2	#	X	H	Z	S	I	,	.	&	"	:	3	x	h	z	s	i	,	.	7	'	;

2. Assumes a frequency distribution more realistic for a Stretch peripheral device, and also produces binary coded decimal representations for the numeric characters, as above.

HOME	M	P	O	E	*	)	$	D	W	?	3/4	m	p	o	e	8	0	4	d	w	/	!
T_1	G	F	R	N	T	@	¢	C	B	Q	=	g	f	r	n	t	2	6	c	b	9	+
T_2	J	K	Y	A	(	L	%	U	V		1/4	j	k	y	a	9	1	5	u	v	-	1/2
T_1, T_2	X	H	Z	S	I	#	&	,	.	¯	:	x	h	z	s	i	3	7	,	.	'	;

3. A compromise of the best frequency distribution for a Stretch device, in order to achieve a code whose four upper bits are constant.

HOME	Q	M	)	$	*	E	D	O	P	W	3/4	q	m	0	4	8	e	d	o	p	w	!
T_1	F	R	@	¢	N	T	C	B	X	G	=	f	r	2	6	n	t	c	b	x	g	+
T_2	J	K	L	%	(	A	U	Y	V	?	1/4	j	k	1	5	9	a	u	y	v	/	1/2
T_1, T_2	H	Z	#	&	S	I	,	.	_	"	:	h	z	3	7	s	i	,	.	-	'	;

1	0	0	0	0	0	1	1	%	1	0	0	0	0	0	0	1	5
1	0	0	1	0	0	1	1	Q	1	0	0	1	0	0	0	1	q
1	1	0	0	0	0	1	1	M	1	1	0	0	0	0	0	1	m
1	1	0	1	0	0	1	1	P	1	1	0	1	0	0	0	1	p
1	1	1	0	0	0	1	1	O	1	1	1	0	0	0	0	1	o
0	0	0	0	0	0	1	1	E	0	0	0	0	0	0	0	1	e
0	0	0	1	0	0	1	1	D	0	0	0	1	0	0	0	1	d
0	1	0	0	0	0	1	1	W	0	1	0	0	0	0	0	1	w
0	1	0	1	0	0	1	1	?	0	1	0	1	0	0	0	1	/
0	1	1	0	0	0	1	1	)	0	1	1	0	0	0	0	1	0
0	1	1	1	0	0	1	1	3/4	0	1	1	1	0	0	0	1	!
1	0	0	0	1	0	1	1	$	1	0	0	0	1	0	0	1	4
1	0	0	1	1	0	1	1	G	1	0	0	1	1	0	0	1	g
1	1	0	0	1	0	1	1	F	1	1	0	0	1	0	0	1	f
1	1	0	1	1	0	1	1	R	1	1	0	1	1	0	0	1	r
1	1	1	0	1	0	1	1	N	1	1	1	0	1	0	0	1	n
0	0	0	0	1	0	1	1	T	0	0	0	0	1	0	0	1	t
0	0	0	1	1	0	1	1	C	0	0	0	1	1	0	0	1	c
0	1	0	0	1	0	1	1	B	0	1	0	0	1	0	0	1	b
0	1	0	1	1	0	1	1	¢	0	1	0	1	1	0	0	1	6
0	1	1	0	1	0	1	1	(	0	1	1	0	1	0	0	1	9
0	1	1	1	1	0	1	1	=	0	1	1	1	1	0	0	1	+
1	0	0	0	0	1	1	1	&	1	0	0	0	0	1	0	1	7
1	0	0	1	0	1	1	1	J	1	0	0	1	0	1	0	1	j
1	1	0	0	0	1	1	1	K	1	1	0	0	0	1	0	1	k
1	1	0	1	0	1	1	1	L	1	1	0	1	0	1	0	1	l
1	1	1	0	0	1	1	1	Y	1	1	1	0	0	1	0	1	y
0	0	0	0	0	1	1	1	A	0	0	0	0	0	1	0	1	a
0	0	0	1	0	1	1	1	U	0	0	0	1	0	1	0	1	u
0	1	0	0	0	1	1	1	V	0	1	0	0	0	1	0	1	v
0	1	0	1	0	1	1	1	#	0	1	0	1	0	1	0	1	3
0	1	1	0	0	1	1	1	@	0	1	1	0	0	1	0	1	2
0	1	1	1	0	1	1	1	1/4	0	1	1	1	0	1	0	1	1/2
1	0	0	0	1	1	1	1	*	1	0	0	0	1	1	0	1	8
1	0	0	1	1	1	1	1	X	1	0	0	1	1	1	0	1	x
1	1	0	0	1	1	1	1	H	1	1	0	0	1	1	0	1	h
1	1	0	1	1	1	1	1	Z	1	1	0	1	1	1	0	1	z
1	1	1	0	1	1	1	1	S	1	1	1	0	1	1	0	1	s
0	0	0	0	1	1	1	1	I	0	0	0	0	1	1	0	1	i
0	0	0	1	1	1	1	1	,	0	0	0	1	1	1	0	1	,
0	1	0	0	1	1	1	1	.	0	1	0	0	1	1	0	1	.
0	1	0	1	1	1	1	1		0	1	0	1	1	1	0	1	-
0	1	1	0	1	1	1	1	¯	0	1	1	0	1	1	0	1	'
0	1	1	1	1	1	1	1	:	0	1	1	1	1	1	0	1	;

1	0	0	0	0	0	0	0	Backspace
0	1	0	0	0	0	0	0	Line feed
0	0	1	0	0	0	0	0	Red ribbon
0	0	0	1	0	0	0	0	Black ribbon
0	0	0	0	1	0	0	0	Tabulate
0	0	0	0	0	1	0	0	Space
0	0	0	0	0	0	1	0	Carriage ret.

$R_5R_2R_{2A}T_1T_2S\ C$

R_5 = twist to left 5 places
R_2 = twist to right 2 places
R_{2A} = another twist to right 2 places
T_1 = attitude (first)
T_2 = attitude (second)
S = shift
C = character vs. function