Code 16k

Code 16K was developed by Ted Williams in 1989 to provide a simple to print and decode multiple row symbology. Williams also developed Code 128, and the structure of 16K is based on Code 128. Not coincidentally, 128 squared happened to equal 16,000 or 16K for short. Code 16K resolved an inherent problem with Code 49. Code 49's structure requires a large amount of memory for encoding and decoding tables and algorithms. 16K is a stacked symbology.

Each Code 16K symbol contains from 2 to 16 rows, with 5 ASCII characters per row. Additionally, up to 107 16-row symbols can be concatenated together to allow encoding of up to 8,025 ASCII characters, or 16,050 numeric digits. In the extended mode, the first three characters in each 16 row symbol defines the mode character, the order of the 16 row symbol in the block, and the total number of symbols in the block.

The code is a continuous, variable-length symbology that can encode the complete ASCII 128-character set.

The minimum value of the x-dimension is 7.5 mils for a symbol to be read by an unknown reader. Minimum bar height is 8 times the x-dimension The maximum data density is 208 alphanumeric characters per square inch or 417 numeric digits per square inch when the symbol is printed at 7.5 mils. In the health care industry for example, a Code 16K symbol printed with a 7.5 mil x-dimension including a flag character, a 10 digit NCD number, a 5 digit expiration date, and a 10 alphanumeric lot code, would fit in a symbol measuring only .35 inches by .61 inches.

Code 16K symbols can be read by modified moving beam laser or CCD scanners. Rows can be scanned in any order. After the last row has been scanned, the bar code reader automatically puts the information in proper sequence. Labels can be printed by standard printing technologies.

Description of the specification:

Code 16k is interesting because it's essentially a lot of reverse-video code128 rows stacked on top of one another, with some bearer bars, start/stop characters, a start symbol, padding, and check symbols thrown in.

Code 16k has 6 modes:

Mode	Starting Code Set	Starting Shift Mode
0	A	none
1	B	none
2	C	none
3	B	FNC1
4	C	FNC1
5	C	1SB
6	C	2SB

FNC1, or "function 1" means that the data following is to be interpreted as a UPC would be. **1SB** and **2SB** are "single-shift B" and "double shift B" meaning that the next or next 2 characters are from character set B, respectively. FNCs and Shifts are also defined in the character set.

A Code 16k block may contain up to 16 rows, separated by horizontal bearer bars. The top and bottom of the symbol have longer bearer bars to include a leading and trailing quiet zone. Each row has its own start and stop character and contains 5 symbols. PAD characters must be inserted to make a row contain 5 symbols if necessary.

Here are the start and stop patterns:

ROW	START	STOP
1	3 -2 1 -1 1	-3 2 -1 1
2	2 -2 2 -1 1	-2 2 -2 1
3	2 -1 2 -2 1	-2 1 -2 2
4	1 -4 1 -1 1	-1 4 -1 1
5	1 -1 3 -2 1	-1 1 -3 2
6	1 -2 3 -1 1	-1 2 -3 1
7	1 -1 1 -4 1	-1 1 -1 4
8	3 -1 1 -2 1	-3 1 -1 2
9	3 -2 1 -1 1	-1 1 -3 2
10	2 -2 2 -1 1	-1 2 -3 1
11	2 -1 2 -2 1	-1 1 -1 4
12	1 -4 1 -1 1	-3 1 -1 2
13	1 -1 3 -2 1	-3 2 -1 1
14	1 -2 3 -1 1	-2 2 -2 1
15	1 -1 1 -4 1	-2 1 -2 2
16	3 -1 1 -2 1	-1 4 -1 1

Symbols may be concatenated by making **FNC2** the first character following the starting symbol. Each subsequent pattern is appended to the preceding pattern. It is also possible to concatenate in a random order by making the value of the first symbol following the start symbol equal to 10 times the symbol number plus the total number of symbols (not to exceed 9). For example, the 4th block out of 8 would encode to 48. In this case, the next symbol should be FNC2 (ie, [48, FNC2]).

The start symbol is calculated to be 7 times the number of rows-2, plus the mode. [7(r-2)+m]. Modes are defined in the table above.

Here is the table of characters and their patterns:

A	B	C	VALUE	BAR/Space Values
		00	0	-2 1 -2 2 -2 2
!	!	01	1	-2 2 -2 1 -2 2
"	"	02	2	-2 2 -2 2 -2 1
#	#	03	3	-1 2 -1 2 -2 3
$	$	04	4	-1 2 -1 3 -2 2
%	%	05	5	-1 3 -1 2 -2 2
&	&	06	6	-1 2 -2 2 -1 3
'	'	07	7	-1 2 -2 3 -1 2
(	(	08	8	-1 3 -2 2 -1 2
)	)	09	9	-2 2 -1 2 -1 3
*	*	10	10	-2 2 -1 3 -1 2
+	+	11	11	-2 3 -1 2 -1 2
,	,	12	12	-1 1 -2 2 -3 2
-	-	13	13	-1 2 -2 1 -3 2
.	.	14	14	-1 2 -2 2 -3 1
/	/	15	15	-1 1 -3 2 -2 2
0	0	16	16	-1 2 -3 1 -2 2
1	1	17	17	-1 2 -3 2 -2 1
2	2	18	18	-2 2 -3 2 -1 1
3	3	19	19	-2 2 -1 1 -3 2
4	4	20	20	-2 2 -1 2 -3 1
5	5	21	21	-2 1 -3 2 -1 2
6	6	22	22	-2 2 -3 1 -1 2
7	7	23	23	-3 1 -2 1 -3 1
8	8	24	24	-3 1 -1 2 -2 2

A	B	C	VALUE	BAR/Space Values
9	9	25	25	-3 2 -1 1 -2 2
:	:	26	26	-3 2 -1 2 -2 1
;	;	27	27	-3 1 -2 2 -1 2
<	<	28	28	-3 2 -2 1 -1 2
=	=	29	29	-3 2 -2 2 -1 1
>	>	30	30	-2 1 -2 1 -2 3
?	?	31	31	-2 1 -2 3 -2 1
@	@	32	32	-2 3 -2 1 -2 1
A	A	33	33	-1 1 -1 3 -2 3
B	B	34	34	-1 3 -1 1 -2 3
C	C	35	35	-1 3 -1 3 -2 1
D	D	36	36	-1 1 -2 3 -1 3
E	E	37	37	-1 3 -2 1 -1 3
F	F	38	38	-1 3 -2 3 -1 1
G	G	39	39	-2 1 -1 3 -1 3
H	H	40	40	-2 3 -1 1 -1 3
I	I	41	41	-2 3 -1 3 -1 1
J	J	42	42	-1 1 -2 1 -3 3
K	K	43	43	-1 1 -2 3 -3 1
L	L	44	44	-1 3 -2 1 -3 1
M	M	45	45	-1 1 -3 1 -2 3
N	N	46	46	-1 1 -3 3 -2 1
O	O	47	47	-1 3 -3 1 -2 1
P	P	48	48	-3 1 -3 1 -2 1
Q	Q	49	49	-2 1 -1 3 -3 1
R	R	50	50	-2 3 -1 1 -3 1
S	S	51	51	-2 1 -3 1 -1 3
T	T	52	52	-2 1 -3 3 -1 1
U	U	53	53	-2 1 -3 1 -3 1
V	V	54	54	-3 1 -1 1 -2 3
W	W	55	55	-3 1 -1 3 -2 1
X	X	56	56	-3 3 -1 1 -2 1
Y	Y	57	57	-3 1 -2 1 -1 3
Z	Z	58	58	-3 1 -2 3 -1 1
[	[	59	59	-3 3 -2 1 -1 1
\	\	60	60	-3 1 -4 1 -1 1
]	]	61	61	-2 2 -1 4 -1 1
^	^	62	62	-4 3 -1 1 -1 1
_	_	63	63	-1 1 -1 2 -2 4
^@~NUL	`	64	64	-1 1 -1 4 -2 2
^A~SOH	a	65	65	-1 2 -1 1 -2 4
^B~STX	b	66	66	-1 2 -1 4 -2 1
^C~ETX	c	67	67	-1 4 -1 1 -2 2
^D~EOT	d	68	68	-1 4 -1 2 -2 1
^E~ENQ	e	69	69	-1 1 -2 2 -1 4
^F~ACK	f	70	70	-1 1 -2 4 -1 2
^G~BEL	g	71	71	-1 2 -2 1 -1 4
^H~BS	h	72	72	-1 2 -2 4 -1 1
^I~HT	i	73	73	-1 4 -2 1 -1 2
^J~LF	j	74	74	-1 4 -2 2 -1 1
^K~VT	k	75	75	-2 4 -1 2 -1 1
^L~FF	l	76	76	-2 2 -1 1 -1 4
^M~CR	m	77	77	-4 1 -3 1 -1 1
^N~SO	n	78	78	-2 4 -1 1 -1 2
^O~SI	o	79	79	-1 3 -4 1 -1 1
^P~DLE	p	80	80	-1 1 -1 2 -4 2

A	B	C	VALUE	BAR/Space Values
^Q~DC1	q	81	81	-1 2 -1 1 -4 2
^R~DC2	r	82	82	-1 2 -1 2 -4 1
^S~DC3	s	83	83	-1 1 -4 2 -1 2
^T~DC4	t	84	84	-1 2 -4 1 -1 2
^U~NAK	u	85	85	-1 2 -4 2 -1 1
^V~SYN	v	86	86	-4 1 -1 2 -1 2
^W~ETB	w	87	87	-4 2 -1 1 -1 2
^X~CAN	x	88	88	-4 2 -1 2 -1 1
^Y~EM	y	89	89	-2 1 -2 1 -4 1
^Z~SUB	z	90	90	-2 1 -4 1 -2 1
^[~ESC	{	91	91	-4 1 -2 1 -2 1
^\~FS	|	92	92	-1 1 -1 1 -4 3
^]~GS	}	93	93	-1 1 -1 3 -4 1
^^~RS	~	94	94	-1 3 -1 1 -4 1
^_~US	^?~DEL	95	95	-1 1 -4 1 -1 3
FNC3	FNC3	96	96	-1 1 -4 3 -1 1
FNC2	FNC2	97	97	-4 1 -1 1 -1 3
1SB	1SA	98	98	-4 1 -1 3 -1 1
CODEC	CODEC	99	99	-1 1 -3 1 -4 1
CODEB	FNC4	CODEB	100	-1 1 -4 1 -3 1
FNC4	CODEA	CODEA	101	-3 1 -1 1 -4 1
FNC1	FNC1	FNC1	102	-4 1 -1 1 -3 1
PAD	PAD	PAD	103	-2 1 -1 4 -1 2
2SB	2SA	1SA	104	-2 1 -1 2 -1 4
2SC	2SC	2SB	105	-2 1 -1 2 -3 2
3SC	3SC	3SB	106	-2 3 -3 3

To read either of the pattern tables here, consider positive numbers to be a bar width and negative numbers to be a space width.

There are 2 checksums at the end of the code, which are calculated by weighting the sum of the values of each character including the start character. The first check symbol starts the weighting at 2, and the second starts weighting at 1. Next, take the modulo 107 of the sum. So if you had the character values 22, 10, 15, 20, the two checksums would be:

$$(2*22 + 3*10 + 4*15 + 5*20) \% 107$$
$$(1*22 + 2*10 + 3*15 + 4*20) \% 107$$