Exhibit 2025
Zynga, Inc. v. Personalized Media Communications, LLC
Case IPR2013-00164 (SCM)

Described are the coding and symbol of the Universal Product Code. The symbol code structure, format, encodation technique, and characteristics with their technical tradeoffs are discussed.

The symbol is analyzed and evaluated. Decodability is shown to depend on the structure of the code and symbol, the size of the symbol, the precision with which the symbol is printed, the technique of scanning employed, the accuracy with which measurements are made, the decoding logic, and the physical operation of scanning. The relationship between the scan pattern of a fixed head scanner and symbol size is shown.

The characteristics and decodability of the Universal Product Code symbol

by D. Savir and G. J. Laurer

The scanning of information from the labels of supermarket merchandise is necessary for a practical supermarket system. This information is encoded in a standard form, the Universal Product Code (UPC). The standard comprises both the code—the representation of decimal characters in binary form—and the symbol—the printed form of the code which can be read by a scanner.

In this paper, we discuss the development of this standard; we define a class of codes suited to optical scanning and investigate some of its properties, and we describe the code belonging to this class that was selected for the UPC.

The code, symbol, appropriate decoding scheme, and scanning scheme are all dependent upon each other. They constitute the structure of the decoding process that is studied to evaluate the decodability, or decoding reliability, of the UPC. (The UPC is standard in the United States and Canada. At the time of writing this paper, proposals for merchandise-identifying symbols for other countries are being advanced.)

Development of the Universal Product Code

The justification for a supermarket system lies both in precise point-of-sale data capture and in the increase of checkout productivity compared to that of a conventional checkstand.[1]

If a code were devised that could both identify each item sold in a store uniquely and enable the item to be checked rapidly, then its use could justify the development of such a supermarket system.

Item identification could be achieved using a number containing sufficient characters; the state of the art dictates that rapid checkout and identification number entry can be achieved only by using a fixed scanner[2] that reads a form of bar code rather than numeric information. Consequently, appropriate item identification should be by a combined code and symbol in which the symbol is the bar-coded representation of the numeric code. Clearly, the code and symbol (which we will henceforth call merely code or symbol, as appropriate) is strongly dependent on a specific supermarket system—the numeric code is filed in the system and constitutes a data base for store decision making and price look-up;[1] the bar representation is attuned to the scanner decoding methodology. It should, therefore, follow that a supermarket system should also include a code that would be placed on each item in the store, replacing the price mark. However, it became clear that the cost to a store of marking its items with scannable code would be much higher than the cost of price-marking, to the extent of negating much of the benefit of the system. The economically acceptable approach, recognizing the *permanence* of the code (the code number is as much part of item identification as its name; price is not part of it), was to have the code printed on the item label by the grocery manufacturer (source-marking). (Note: Subsequently, variants of the code containing price were proposed. These were intended for items of variable weight (meat, produce, cheese) and to be automatically printed by the weighing device.)

The obvious objection is that since the code depends on its supermarket system, provided by a vendor, then one of these events would be necessary for a source-marked code:

a. Grocery manufacturers would mark the symbol of every vendor on every item.
b. Grocery manufacturers would print different sets of labels for each item, each set containing the distinctive vendor symbol; a store using a particular vendor's equipment would receive appropriately labeled merchandise.
c. A standard code would be designed for the community of vendors, supermarkets, and manufacturers; this code would be printed on the labels.

Event a was unacceptable because (1) few labels would be large enough to contain all symbols, (2) as new vendors offered their

products the printers would have to replace their plates, (3) checkout productivity would be impaired substantially, and (4) printing costs would be exorbitant.

Event b was unacceptable because (1) grocery manufacturers would be reluctant to print labels for systems that might not be installed, while supermarkets would be reluctant to install systems without assurance of adequate, appropriately marked labels and (2) printing costs and label inventories would be exorbitant.

Thus, c, the least unattractive choice, was taken—industry groups with conflicting objectives accepted a voluntary standard for the code. Most of the conflicts were about symbol size and print quality. Grocery manufacturers wanted symbols to occupy little label space since the symbol would detract from product identification, whereas larger symbols are more easily scanned. The sloppier the symbol can be printed, the easier it is to print and the harder it is to scan. The UPC code was selected by a committee, Uniform Grocery Product Code Council, Inc., which was composed of representatives of grocery manufacturers and supermarket chains. This committee delegated to its Symbol Standardization Subcommittee the task of soliciting, reviewing, and amending suggestions from vendors, and finally proposing the UPC code to the Council. During its task, the subcommittee engaged a consultant, McKinsey and Co., Inc., and prepared a set of guidelines to enable vendors to offer suggestions for a code that would satisfy the guidelines, offer something more than the guidelines required, and be compatible with scanning equipment that they could be expected to subsequently provide.

Some of the guidelines were:

- The code should contain 10 decimal characters (subsequently, 12 were required).
- The symbol should be scannable omnidirectionally, i.e., regardless of its orientation with respect to a scanning device.
- The symbol should be scannable when in motion at a velocity not exceeding 100 inches per second.
- The scanning reject rate should not exceed 0.01 and the undetected error rate should not exceed 0.0001.
- The depth of field should be at least one inch.
- Normal environmental contamination (abrasion, dirt, etc.) should not affect the scanning process significantly.
- The symbol area should not exceed 1.5 square inches. (The symbol selected is of variable size.)

The symbol selected by the Uniform Grocery Product Code Council, in addition to meeting the guidelines, subject to the

parenthetical comments, also contains parity and a modulus check; it can also be adapted to encode more (and fewer) characters.

Although the principal intent of the UPC code was to provide a data medium for fixed scanners, it was recognized that the coded information should be able to provide input to checkers using hand-held scanners (optical wands) and to checkers using keyboards for decimal code entry. The code was, therefore, designed to be readable by humans, by fixed scanners, and by wands.

Optical codes

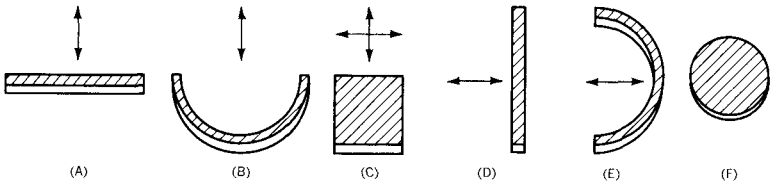
An economically feasible means of printing a symbol is to print dark marks on a light background on a label. (The converse is also true; conceptually both processes yield similar results.) The code is of binary nature—the presence of a mark corresponds to impressed information, represented by a one bit, in a domain of the label (called a *module*), and the absence of a mark corresponds to lack of impressed information, represented by a zero bit in a module. Encoding of a symbol can be done by conventional binary means, i.e., by representing the characters by their binary code, partitioning the space on the label into modules and printing marks on the modules covered by one bits (Figure 6).

Decoding a symbol is more complex because the modules that define the domain of the elements of the code (one and zero bits) are not perceivable by the spot of light that scans the marks.[2] The elements of the code with respect to decoding are the marks (dark bars) and the spaces (light bars) between them, represented by runs of one and zero bits, respectively. Therefore, an integral part of the decoding process is the determination of the length of each run. This determination is confounded by imprecision that can lead to error.

Errors in binary decoding of line signals are due to noise, loss of information, or discrimination error, given that the decoding device is attuned to the signal. The standard error-detecting and error-correcting devices presume that any bit is as likely to be decoded erroneously as any other bit; the probability of erroneous message interpretation is evaluated on this presumption.

The nature of optical binary codes, however, is different. Firstly, the decoding device is not attuned to the signal—the speed at which signals from evenly spaced marks enter the decoding device varies due to variation in scanning-spot velocity (especially when the scanner is hand held), curvature of the surface upon

Figure 1 Effect of smear on marks



which the marked label is placed, variation in depth of field over the label, etc. Secondly, the techniques of error detection must work both with the binary coded input and with the decimal input—the code should contain protection against miskeying on a terminal keyboard. Thirdly, bits are not equiprobably subject to misinterpretation. If the symbol is printed with adequate contrast (which is not difficult), random noise or loss of information will not be significant. The principal source of error is at the edges of marks.

reasons for error

In the scanning process, a spot interprets the location of each edge of each mark and decodes information accordingly. A decoding error occurs if and only if the location of one or more edges is interpreted incorrectly. One combination of code and symbol has a higher decodability than another if the former can tolerate a stochastically greater edge dislocation than the latter while still decoding the information correctly.

The edge of a mark is designed to be at a specific location. However, as the artwork is drawn, for example, an error is introduced. Further errors are introduced in the processes of reduction and platemaking. Additional errors occur in the process of printing. The symbol as seen by the eye contains at each edge of every mark the sum of all these errors. The scanner cannot perceive the edge to be exactly where it is printed; there are errors introduced due to optical effects and to the effects of digital timing and discrete sampling. In addition, there are errors caused by the environmental degradation of the symbol of which dirt, wrinkles, abrasion, and moisture are a few. The total dislocation of the edge of the mark as perceived by the scanner is the sum of all these errors. If this sum of errors on any edge exceeds some value, a decoding error will be made. An appropriate choice of code and decoding scheme will be shown to neutralize certain components of systematic error.

We partition the sources of edge dislocation into print error, $\bar{e}_p$, and system error, $\bar{e}_s$, such that the total dislocation on an edge is $\bar{e} = \bar{e}_p + \bar{e}_s$. Errors in artwork, platemaking, and printing are consolidated into $\bar{e}_p$; errors due to optical effects and the effects of digital timing and discrete sampling and those due to environmental degradation are consolidated into $\bar{e}_s$.

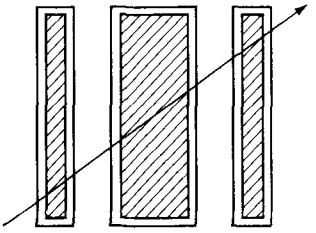
print errors

The errors contained in $\bar{e}_p$ that affect the location of the edge of the mark are due to (1) artwork, (2) platemaking, (3) edge roughness, (4) extraneous ink, (5) voids, (6) smear, (7) inkspread, and (8) expansion and contraction of substrate.

Errors in artwork consist of the *random* error in the line drawing and a *systematic* error in photoreduction. A random error affects an edge or a portion of an edge independently of the rest of the symbol. A systematic error affects all the edges in a similar manner.

Errors in platemaking contain a systematic component, increasing or decreasing the size of all the marks, and a random component. Edge roughness is a random effect whose intensity depends on the printing process and the paper. Extraneous ink and voids affect edges only when sufficiently large to be identified falsely as a mark or when intrusive into the edge of a mark.

Figure 2 Effect of ink-spread on mark sequences



The error of smear is due to a systematic ink deposit in the direction of motion of the paper. The effect of smear on marks of differing shape is shown in Figure 1. Arrows indicate optimal directions of traversal of the scanner spot with respect to the marks, ignoring the effect of smear. The presence of smear on each of the marks affects the optimal trajectory by differing amounts. We observe that mark D of Figure 1 succeeds in neutralizing the smear. The spot does not traverse any edge affected by smear over the effective range of the mark.

The error of inkspread is due to over-inking (or conversely under-inking) which increases (or decreases) the size of all the marks, systematically. The effect of inkspread on sequences of marks of differing width is shown in Figure 2. If the spot of the scanner follows a straight trajectory across the marks, the effect of inkspread is to add a constant increment to the width of each mark.

Other systematic effects result in a change in scale of one dimension with respect to the orthogonal dimension, i.e., an apparent expansion or contraction of length or width of the symbol.

neutralizing systematic errors

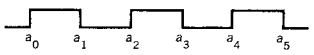
The effects of systematic errors are isolated and controlled by using a rectilinear bar-coded symbol and an appropriate decoding scheme. The symbol is a sequence of long rectangular marks of several widths separated by spaces of several widths (Figure 5). By printing the symbol such that the bars are aligned with the motion of the paper through the printing press, the adverse effects of smear are controlled. The other systematic effects are controlled by the decoding technique.[3]

decoding technique

In this decoding technique, the ratio of the measurement across a mark and an adjacent space to a reference measurement within the character is used. Practically, the measure cannot be one of length; the time to effect the transitions is actually measured. Since the decoding device is not attuned to the signal, the reference measurement is necessary, and since the scanning spot does not necessarily cover the marks at a uniform velocity, the reference measurement should be proximate to the decoding measurement.

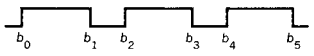
We illustrate the power of this decoding technique by examining the phenomenon of inkspread, separating its error effect from the other errors that contribute to edge dislocation. Schematically, the section across a sequence of marks and spaces through which the scanning spot passes is as shown in Figure 3 where a_0 through a_5 represent the locations of the edges of the marks, with a_0, a_2, and a_4 denoting space-to-mark transitions and a_1, a_3, and a_5 denoting transitions from mark to space. We assume that each of the edges is dislocated due to various errors, but that dislocation due to inkspread is not included. Suppose, in addition, that we have the means of decoding the symbol based on data a_i.

Figure 3 Section through symbol, ignoring effects of systematic errors



Now let us apply an error, ξ, due to inkspread, where ξ is unknown (in fact, will vary from print run to print run) but is consistent throughout any single print run. After the error ξ is applied, the sectioned symbol is like Figure 4 where b_i represent the locations of the edges of the bars corresponding to locations denoted by a_i under the transformation: $b_i = a_i - (-1)^i\xi$. The even-indexed locations are shifted to the left, and the odd-indexed locations are shifted to the right, yielding an increase in the width of each mark of 2ξ, independent of the original width of the mark. The locations b_i are the only edges perceivable to the scanner or to the eye.

Figure 4 Section through symbol, accounting for effects of systematic errors



Now for $i \geq 2$:

$$b_i - b_{i-2} = a_i - (-1)^i\xi - a_{i-2} + (-1)^{i-2}\xi$$
$$= a_i - a_{i-2}$$

which yields sufficient information for the decoding of the symbol. Hence the process of decoding by measuring the distances between the leading edges of adjacent marks and those between the trailing edges of adjacent marks isolates and controls the effect of the error, ξ, due to inkspread. Each such distance spans a mark and an adjacent space. By the same argument, any systematic error that widens or narrows each bar by the same amount is circumvented by this decoding technique.

A class of suitable codes

Desirable properties of a suitable optical code are

1. Each character representation should be of fixed length and stand alone, independent of adjacent characters. This feature enables label makers to print the code simply and facilitates artwork preparation of source-marked labels.
2. Each character representation should contain a fixed reference measurement from the leading (trailing) edge of a mark to the leading (trailing) edge of another known mark. The code should be decodable forwards, in which the reference measurement lies between the leading edges, or backwards, in which the reference measurement lies between trailing edges.

A class of suitable codes is the class of (n, k) *codes*. An (n, k) code represents each character uniquely by n bits containing k runs of one bits and k runs of zero bits, $1 \leq k \leq n/2$. The first bit of a character is a one bit. In the terminology of marks and spaces, we can define an (n, k) code as one in which the representation of each character is by a unique set of k marks and k spaces spanning exactly n modules and beginning with a mark.

The definition is related to forward scanning and decoding. The reference length is n, measured between leading edges k marks apart. An (n, k) code is decodable backwards, in which case the reference measurement is found between the trailing edges of marks.

An (n, k) code represents $\binom{n-1}{2k-1}$ distinct characters. This follows from the fact that there are $\binom{n-1}{r-1}$ distinguishable ways of placing n indistinguishable items into r buckets such that no bucket remains empty.[4]

Table 1 shows the number of distinct characters representable in (n, k) codes for $2 \leq n \leq 10$, $1 \leq k \leq 4$. We note that, for the least necessary number of bits, a (6, 2) code will represent the set of decimal characters adequately, a (7, 2) code will represent the set of decimal characters with additional recognition characters, and a (9, 2) or (9, 3) code will represent a full alphanumeric character set.

If parity (odd or even) is desired, a (7, 2) code will represent the set of decimal characters, an (8, 2) code will represent the set of decimal characters with additional recognition characters, and a (10, 3) code will represent a full alphanumeric character set. In summary, if the code contains six bits, we have decimal

Table 1 Distinct characters in (n, k) codes

$n \downarrow$ / $k \rightarrow$	1	2	3	4
2	1	0	0	0
3	2	0	0	0
4	3	1	0	0
5	4	4	0	0
6	5	10	1	0
7	6	20	6	0
8	7	35	21	1
9	8	56	56	8
10	9	84	126	36

representation, if seven bits, we also have either additional characters or parity, if eight bits, we have both additional characters and parity, if nine bits, we have alphanumeric representation without parity, and if 10 bits, we also have parity.

Given an (n, k) code, we can identify each character represented by the code by an integer $2k$ − tuple $(c_1, c_2, \cdots, c_{2k})$ $c_i \geq 1$, $\sum_{i=1}^{i=2k} c_i = n$. c_i is the length of each run of one or zero bits, or, equivalently, the number of modules contained in each mark and space. We have noted that in the scanning and decoding of the symbol, the determination of the values of c_i may be difficult due to the expansion or contraction of the printed marks. This difficulty is largely overcome by the decoding technique in which the distance encompassed by a bar and adjacent space is measured.

Let

$$t_i = c_i + c_{i+1}, \quad i = 1, 2, \cdots, 2k - 1$$

$$t_{2k} = c_{2k}$$

t is in one-to-one correspondence with c; in fact

$$c_i = \sum_{j=i}^{2k} (-1)^{i+j} t_j$$

We note that t_i is the number of bits covered by two adjacent runs, except for t_{2k} which is the length of the last run of zero bits in a character. The decoding of a character by its t representation rather than its c representation will be less susceptible to the error of mark expansion or contraction. We cannot, however, dispense with t_{2k} because $(t_1, \cdots, t_{2k-1})$ may correspond to more than one value of $(c_1, \cdots, c_{2k})$, hence will not decode uniquely. t_{2k-1} is redundant, since

$$\sum_{i=1}^{k} t_{2i-1} = n$$

The set of $\{t_1, \cdots, t_{2k-2}\}$ is connected at integer points, i.e., given a value of $(t_1, \cdots, t_{2k-2})$ it is always possible to change at least one of the t_i by one integer (up or down) to yield another value of $(t_1, \cdots, t_{2k-2})$ which, of course, corresponds to a different character or characters. Parity on $(t_1, \cdots, t_{2k-2})$ is a protective device; it would be useful if parity on the values of t were equivalent to parity on the one bits of the character, i.e.,

$$\sum_{i=1}^{k} c_{2i-1}$$

$$\left(\sum_{i=1}^{k} c_{2i-1}\right) \bmod k = \left(\sum_{i=1}^{k} \sum_{j=2i-1}^{2k} (-1)^{2i+j-1} t_j\right) \bmod k$$

$$= \left(\sum_{i=1}^{k-1} i(t_{2i-1} - t_{2i})\right) \bmod k$$

Figure 5 UPC symbol

For an $(n, 2)$ code we have

$$(c_1 + c_3) \bmod 2 = (t_1 - t_2) \bmod 2$$
$$= (t_1 + t_2) \bmod 2$$

For an $(n, 3)$ code we have

$$(c_1 + c_3 + c_5) \bmod 3 = ((t_1 - t_2) + 2(t_3 - t_4)) \bmod 3$$
$$= (t_1 - t_2 - t_3 + t_4) \bmod 3$$

For $k > 3$ we cannot express the parity of

$$\left(\sum_{i=1}^{k} c_{2i-1}\right) \bmod k$$

as a mod k linear function of t with unit coefficients.

Therefore, both $(n, 2)$ codes and $(n, 3)$ codes can use parity that is equivalent for both the sum of the one bits and sums (or differences) of the values of $(t_1, \cdots, t_{2k-2})$; for the $(n, 2)$ code the parity is the "traditional" binary parity; for the $(n, 3)$ code a ternary parity must be used. (Note that $(t_1 \pm t_2 \pm t_3 \pm t_4) \bmod 2 = (c_1 + c_5) \bmod 2$; in general, $\Sigma_{i=1}^{2k-2} \pm t_i \bmod 2 = (c_1 + c_{2k-1}) \bmod 2$.)

The decoding process will use the measurements of the variables t_i normalized by the measurement of the reference n. In the next section, we describe the UPC code selected, after which we discuss the details of its decoding.

Figure 6 Binary and bar-coded representation of even-parity characters

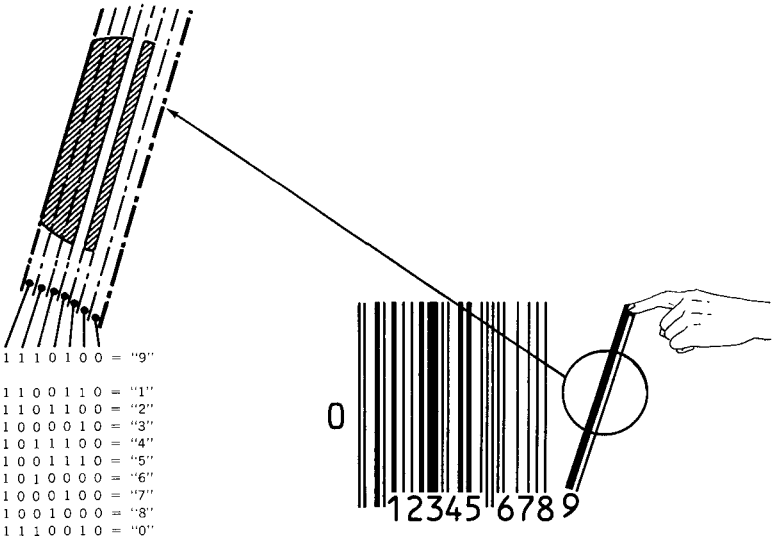


Figure 7 Decomposition into two symbol blocks

Description of UPC code and symbol

Of the (n, k) codes, $(n, 2)$ codes contain parity equivalence between the parity of one bits in the character and the parity of the bits contained in the modules whose measurements are necessary for decoding by the technique previously discussed. Since the (7, 2) code contains sufficient characters for decimal representation with parity, this was the code selected for the UPC symbol[5] in a rectilinear bar-code format.

Each character is represented by two dark and two light bars, of an integral number of modules each, spanning a total of seven modules. The character begins on a dark bar. The symbol of Figure 5 is broken out in Figure 6 to show the representation of a character and the binary representation of the bars in which a dark (light) bar occupying i modules is represented by a run of one bits (zero bits) of length i.

The symbol contains 12 characters (including a modulus check character), broken into two symbol blocks of six characters each which are scanned independently. If the symbol were unbroken, a symbol of comparable decodability and module size would require significantly longer bars than the symbol selected. This point will be discussed with the omnidirectionality of scanning. Each symbol block is delineated by two dark guard bars, separated by light bars, of one module each. The center pattern of guard bars is shared by both symbol blocks (Figure 7). The left block is distinguishable from the right block by the parity of the included characters—odd parity characters fill the left block, even parity characters fill the right block. The beginning of a character (dark bar) is closest to the center pattern. The binary representation of the complete character set is shown in Figure 8. Note that the odd parity characters complement the even parity characters, and characters are unambiguous when reflected, permitting backwards scanning.

Figure 8 Binary representation of character set

ODD PARITY ON THE LEFT

EVEN PARITY ON THE RIGHT

	LEFT CHARACTERS	RIGHT CHARACTERS
0	0001101	1110010
1	0011001	1100110
2	0010011	1101100
3	0111101	1000010
4	0100011	1011100
5	0110001	1001110
6	0101111	1010000
7	0111011	1000100
8	0110111	1001000
9	0001011	1110100

With a rectilinear bar code, omnidirectional scanning is achievable by an X-pattern, comprising one or more Xs (Figure 9). Under omnidirectionality, with a simple X-pattern (one in which all Xs are identical), when the legs of the X subtend an angle (α) of 45° to the normal to the mean direction of item movement, the length of the bars is minimized,[6] at a value $b = a + ut$, where a is the width of the symbol block, u is the item velocity, and t is the period of the scanning cycle[2] (Figure 10).

The value of ut to which the scanner is designed is called the oversquare, for obvious reasons. It should now be clear why the symbol is designed in two blocks; if only one were used, the necessary symbol area would be close to $2a(2a + ut)$ which is significantly larger than the necessary $2a(a + ut)$ of the UPC

Figure 9 Omnidirectional scanning using X-pattern

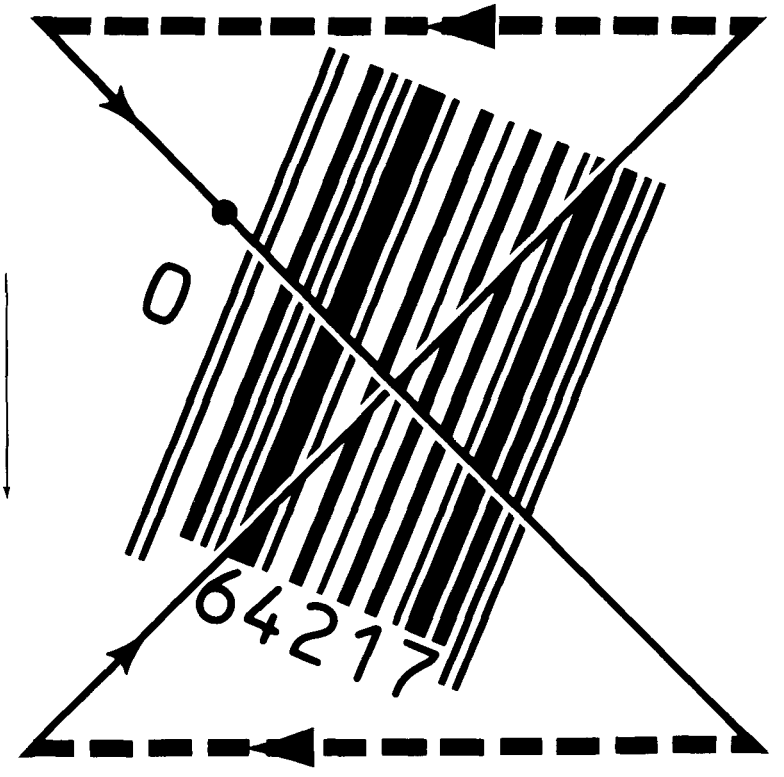
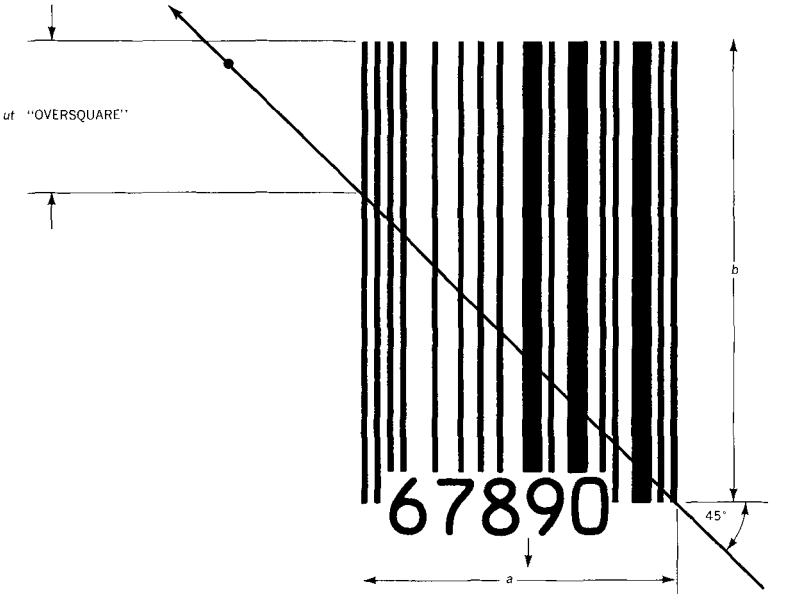


Figure 10 Dimensional parameters for omnidirectional scanning



symbol. It might be inferred that four blocks are superior to two; however, each block requires specific identification within the symbol. The UPC blocks are identified by their parity. If four blocks were used, so many additional characters would be required for block separation and identification as to negate the advantage of the reduced bar length.

The power of parity is slightly weakened, therefore, in the UPC code; parity is used both for block identification and for character validation. Characters are assumed to be of the correct parity if all six characters in a block possess the same parity. The probability of decoding error due to this weakening is vanishingly small. We shall show subsequently that there is a variant of the UPC symbol (Version E) with even weaker parity structure—this symbol contains only one block with three odd and three even parity characters. The error detection and correction methodology is discussed in detail in References 6 and 7; however, in brief, the following interpretations are made:

- If six odd(even) parity characters are scanned, a left (right) block is assumed, to be confirmed if an assumed right (left) block is also found.
- If five odd(even) parity characters are scanned, a left (right) block containing an invalid character is assumed, to be confirmed if an assumed right (left) block is also found.
- If three odd and three even parity characters are scanned, a Version E block is assumed, to be confirmed if no assumed right or left block is found.
- If four odd or even parity characters are scanned, the information is useless and ignored.

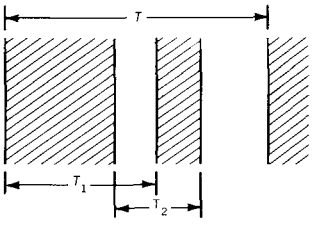
The various versions of the UPC symbol are described in the Appendix.

Figure 11 Decimal characters as functions of t_1 and t_2

t_1 \ t_2	2	3	4	5
2	E 6	O 0	E 4	O 3
3	O 9	E 2 E 8	O 1 O 7	E 5
4	E 9	O 2 O 8	E 1 E 7	O 5
5	O 6	E 0	O 4	E 3

NOTE: ODD (O) AND EVEN (E) PARITY

Figure 12 The T measurements of an even-parity nine



Decoding the UPC symbol

From Figure 8 we can represent the characters in both c and t notation of (n, k) codes as shown in Table 2.

We have previously noted that (t_1, t_2, t_3) are related to measurements of the decoding technique, and also that t_3 is redundant since $t_1 + t_3 = 7$ in all cases. Representing each character by (t_1, t_2) we obtain the matrix of Figure 11.

The decoding process is to first ascertain values of t_1 and t_2 from measurements of the decoding technique, and then to discriminate, where necessary, between one and seven and between two and eight. Measurements T_1, T_2 and T (the reference measurement)[6] are indicated in Figure 12.

Table 2 Decoding tables

Character	c-notation		t-notation	
	odd	even	odd	even
0	1123	3211	2353	5321
1	1222	2221	3442	4431
2	2212	2122	4332	3342
3	1141	1411	2551	5521
4	2311	1132	5421	2452
5	1321	1231	4531	3541
6	4111	1114	5221	2254
7	2131	1312	3441	4432
8	3121	1213	4331	3343
9	2113	3112	3243	4232

We establish the following decision rules with respect to the interpretation of the value of t_i:

$$\frac{T_i}{T} \leq \frac{2\frac{1}{2}}{7} \Rightarrow At_i = 2$$

$$\frac{2\frac{1}{2}}{7} < \frac{T_i}{T} \leq \frac{3\frac{1}{2}}{7} \Rightarrow At_i = 3$$

$$\frac{3\frac{1}{2}}{7} < \frac{T_i}{T} \leq \frac{4\frac{1}{2}}{7} \Rightarrow At_i = 4$$

$$\frac{4\frac{1}{2}}{7} < \frac{T_i}{T} \qquad \Rightarrow At_i = 5$$

where At_i is the value assigned to t_i as a consequence of the decision. The decision is correct when $At_i = t_i$. We note (Figure 11) that an error of one unit in the interpretation of either t_i or t_2 but not both will cause a parity error in the decoded character.

There are two approaches to the resolution between one and seven and between two and eight. One approach, discussed in detail in References 6 and 7, is to extend t_4 to cover the first mark of the subsequent character (or guard bar if no character follows), and to measure an associated T_4. Then, having decoded the subsequent character, its value c_1 is known, which, when subtracted from At_4 will yield the unextended t_4. Another approach is to measure the mark corresponding to c_1 (in the case of the one-seven discrimination) or to c_3 (in the case of the two-eight discrimination) directly, correcting for systematic errors, deciding whether the measured value corresponds to one or two modules and decoding accordingly.

This decoding process is suited to decoding in real time. Since characters can be decoded both forwards and backwards, if a T_4 measurement is required for a character and if the symbol block

is scanned backwards, then the resolution of the character is immediate. Resolution must be delayed if the block is scanned forwards.

Symbol decodability

A probabilistic model of decoding reliability or decodability is presented in Reference 6. In this section, we will qualitatively discuss factors that contribute to the symbol decodability.

When an item is passed over the scanning window, one of the following three consequences will occur, in order of decreasing desirability:

1. The symbol will be correctly decoded and transmitted to the controller.
2. No code will be transmitted, the scan will be rejected, and the system will request another item pass.
3. An incorrect code will be transmitted to the controller.

Symbol decodability is the probability of each of these outcomes. It depends both on how successfully the symbol is decoded and on how well the system recognizes decoding error if it exists. The probability of decoding error depends on the likelihood of making the correct decoding interpretations discussed in the previous section. Three devices are contained in the code and symbol to assist in recognition and possible correction of decoding errors should they occur. They are redundancy, parity, and a modulus check.

As the item is passed over the scanning window, each symbol block will be scanned a variable number of times, depending upon the length of the bars of the symbol, the item velocity, and its orientation with respect to the window. Each scan will collect information from a different "slice" of the label so that it is unlikely for a specific error due to a random print imperfection to occur on more than one of the scans. If an error occurs as a consequence of the incorrect interpretation of a T_1 or a T_2 measurement, but not both, then the parity of the decoded character will be reversed (Figure 11).

A correctly decoded scan will satisfy the modulus check. If all characters except one are correctly decoded and the incorrect character is also of incorrect parity, then the code can be corrected by using the pointer of parity violation in conjunction with the value of the syndrome or checking number.

Simultaneous use of redundancy, parity, and modulus check yields an adequate level of decodability, given sufficient decod-

ing accuracy, which in turn depends on print quality and on the precision of the scanning device.

Summary

Optical scanning components of supermarket systems require machine-readable code. The need for industry-wide cooperation and standardization of a code and symbol was shown. The sources of decoding error in printed optical codes were discussed and a class of codes, the (n, k) codes, relatively resistant to error, were presented. The UPC code is a (7, 2) code.

The UPC code and a method of decoding it were described. Decodability depends on several parameters; their quantitative relationships and effects upon decodability are found in another paper.[6]

ACKNOWLEDGMENT

The material from the UPC Symbol Specification in the Appendix is reprinted by permission of the copyright owner, Distribution Codes, Inc.

CITED REFERENCES

1. P. V. McEnroe, H. T. Huth, E. A. Moore, and W. W. Morris, III, "Overview of the Supermarket System and the Retail Store System," in this issue.
2. L. D. Dickson and R. L. Soderstrom, *The IBM Supermarket Scanner*, Technical Report (in preparation), IBM Corporation, System Development Division, Rochester, Minnesota.
3. P. V. McEnroe and J. E. Jones, *Identification Technology for the Retail Industry*, IBM Corporation, System Development Division, Research Triangle Park, North Carolina (October 1971).
4. W. Feller, *An Introduction to Probability Theory and its Applications*, Vol. 1, 2nd Ed., 37, John Wiley & Sons, Inc., New York, New York (1957).
5. *UPC Symbol Specification*, © Distribution Codes, Inc. (formerly Distribution Number Bank, Inc.), 1725 K Street, N. W., Washington, D. C. (1973).
6. D. Savir, *A Model of the Decodability of the Universal Product Code Symbol*, Technical Report TR29.0123, IBM Corporation, System Development Division, Research Triangle Park, North Carolina.
7. D. Savir, *The Effect of the Design of the IBM Proposed UPC Symbol and Code on Scanner Decoding Reliability*, IBM Corporation, System Development Division, Research Triangle Park, North Carolina (October 1972).

Appendix

There are five versions of the UPC symbol that are reserved for specific uses[5] as listed in Table 3. The format for the five versions varies as shown in Table 4. Of specific interest are Versions A and E. The other versions are modifications of Version A.

Table 3 Five versions of the UPC symbol

Version	*Intended Use and Number of Information Characters*
A	*Basic version*, used to encode the 10-character Grocery Industry UPC as well as the present National Drug Code (NDC) and National Health Related Items Code (NHRIC).
B	*Special version*, reserved for encodation of the National Drug Code and National Health Related Items Code if expansion to 11 characters is required at a later date.
C	*Special version*, with 12 characters, reserved to promote industry-wide product code compatibility, expanding the existing family of compatible codes (Grocery UPC, NDC, NHRIC, Canadian Grocery Code, and the Distribution Code).
D	*The 12 + n character or variable-message length version* was adopted to provide a compatible version of the symbol for possible use in grocery stores that sell general merchandise or in general merchandise or department stores where more information may be needed in the symbol. Note that effective use of this version for source symbol marking implies agreement by general merchandise distributors and their suppliers on a common code numbering system.
E	*The zero-suppression version* of the symbol is included to facilitate source symbol marking on packages that would otherwise be too small to include a symbol. This is achieved by encoding the symbol in six characters in a special way that leaves out some zeros that can occur in the UPC code. For example, code 12300-00045 can be encoded in a symbol as 123453, effectively eliminating half of the area that would otherwise be required for the symbol.

Table 4 Format for UPC symbols

Version	*Title*	*Format*
A	Regular	SXXXXX XXXXXC
B	Drug B	SXXXXX XXXXXX
C	12-character	XSXXXXX XXXXXCX
D	12 + n-character	SXXXXX XXXXXCXX . . .
E	Zero suppression	XXXXXX

X = information character
S = number system character
C = modulo-10 check character

Except for Version E, the number system character identifies both the version and the type of item that is described; for example, both a grocery item whose information characters identify the item, and a weighed item, such as meat or produce, whose information characters contain a variable price, would be encoded in the same version but distinguished by the value of S.

The zero-suppression version (E) is similar to the portion of Version A (regular symbol) to the left of the center except for the following:

Table 5 Parity pattern of the zero-suppressed symbol

Number system	*Modulo check character value*	*Character location number* 1	2	3	4	5	6
0	0	E	E	E	O	O	O
0	1	E	E	O	E	O	O
0	2	E	E	O	O	E	O
0	3	E	E	O	O	O	E
0	4	E	O	E	E	O	O
0	5	E	O	O	E	E	O
0	6	E	O	O	O	E	E
0	7	E	O	E	O	E	O
0	8	E	O	E	O	O	E
0	9	E	O	O	E	O	E
1	0	O	O	O	E	E	E
1	1	O	O	E	O	E	E
1	2	O	O	E	E	O	E
1	3	O	O	E	E	E	O
1	4	O	E	O	O	E	E
1	5	O	E	E	O	O	E
1	6	O	E	E	E	O	O
1	7	O	E	O	E	O	E
1	8	O	E	O	E	E	O
1	9	O	E	E	O	E	O

1. It has a right guard pattern which is coded 010101.
2. Three of the characters are coded in odd parity and three in even, as in Figure 4, except that the even parity characters are reflected.
3. Only two number systems are available: "0," used for regular UPC numbers; and "1", which is currently unassigned.
4. The coding of the zero-supression version is compressed into six characters of varying parity. The determination of whether a character's parity is odd or even is given in Table 5.

There is, therefore, no explicit character encodation of the number system or modulo check characters; their values are derived from the parity permutation of the six encoded characters.

Reprint Form No. G321-5002