EZCode® Specifications

The EZcode [code] layout is pictured below in Figure 1. It consists of the following elements:

- a larger and a smaller guide bar for determining the location and orientation of the code,
- three cornerstones for detecting the distortion, and
- the data area with the actual code bits.

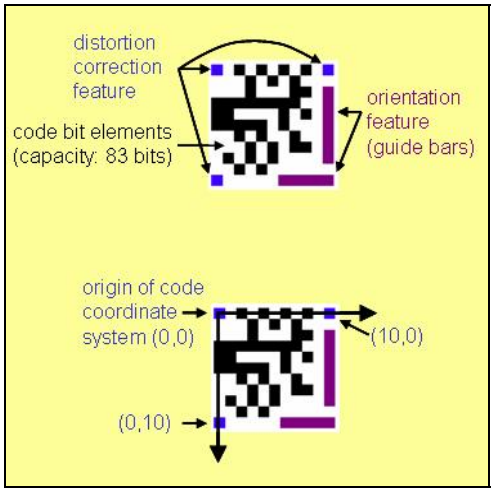

Figure 1: The layout of the EZcode and the code coordinate system

The combination of a large and small guide bar is beneficial for detecting even strongly tilted codes. In the bottom half of Figure 1, the code coordinate system is shown. Each code defines its own local coordinate system with its origin at the upper left edge of the code and one unit corresponding to a single code bit element. Depending on the code size, the mapping between the points in the image plane and the points in the code plane is more precise than a single coordinate unit. The x-axis extends in horizontal direction to the left and to the right beyond the code itself. Correspondingly, the y-axis extends in vertical direction beyond the top and bottom edges of the code. For each code found in a particular input image, the code recognition algorithm establishes an objective mapping between arbitrary points in the code plane and corresponding points in the image plane.

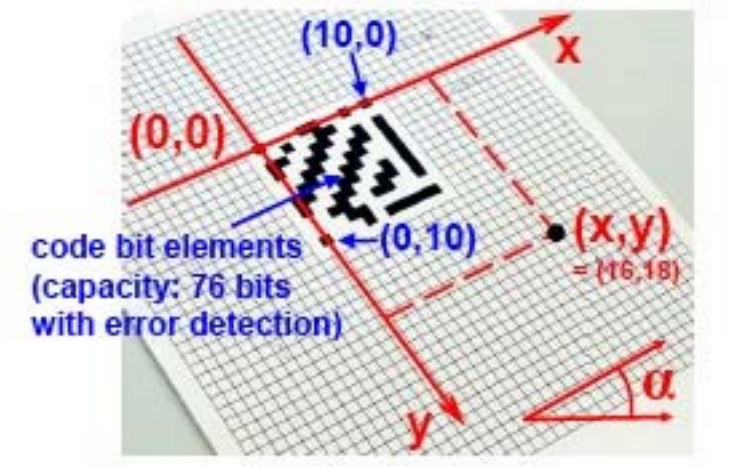

Figure 2: Code coordinate system of rotated and tilted EZcode

The recognition algorithm requires one bit cell of white space around an EZcode. Multiple codes can be laid out in a grid and positioned closely together, as long as one bit cell of white space is left between corner stones and guide bars of neighboring codes. The grid arrangement is depicted in Figure 3 below.

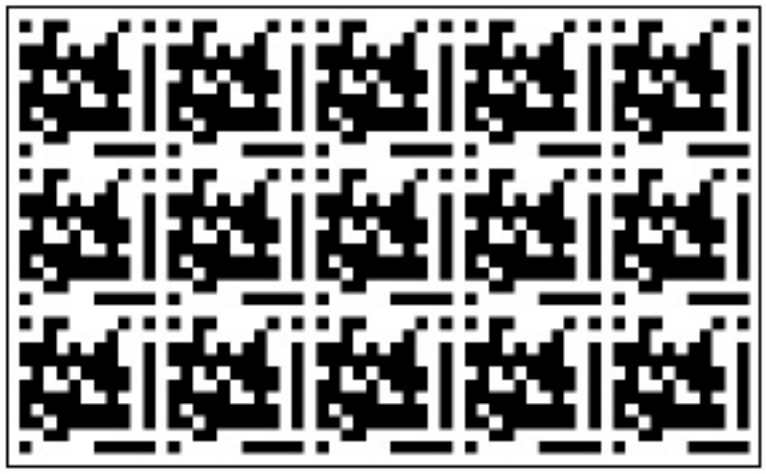

Figure 3: Multiple EZcodes laid out in a grid pattern

The EZcode has a storage capacity of 76 bits. In order to detect pixel errors and false orientation features, the code bits are protected by an (83,76,3) linear code that generates an 83-bit code word from a 76-bit value and has a Hamming distance of 3.

The code word is computed as $x = mG$ with code word $x[1\times 83]$, message $m[1\times 76]$, and generator matrix $G[76\times 83]$. All matrixes have elements of Z_2 and the operations are executed in the field of integers modulo 2 (logical *and* and addition modulo 2). The generator matrix is computed as $G = (I_{76} \mid A)$ with identity matrix $I_{76}[76\times 76]$ and $A[76\times 7]$. The rows of A are taken from the sequence [3, 5, 6, 7, 9, …, 83] (omitting the integers $2^i, i \in \{0,1,...,6\}$).

The algorithm allows correction of a single bit error in an 83-bit code word x using a parity check matrix H in the following way: $Hx^T = s^T$ with code word $x^T[83\times 1]$, syndrome $s^T[7\times 1]$, and parity check matrix $H[7\times 83]$ $H = (A^T \mid I_7)$ with identity matrix $I_7[7\times 7]$ and $A^T[7\times 76]$. If there is no error, syndrome $s = (0000000)$. If s indicates an error, the corresponding code is discarded.

The 83-bit code word is embedded in the data region of an EZcode. As illustrated in Figure 4, all bits inside the data region are marked from 0 to 82. The highest index corresponds to the most significant bit and the lowest index corresponds to the least significant bit.

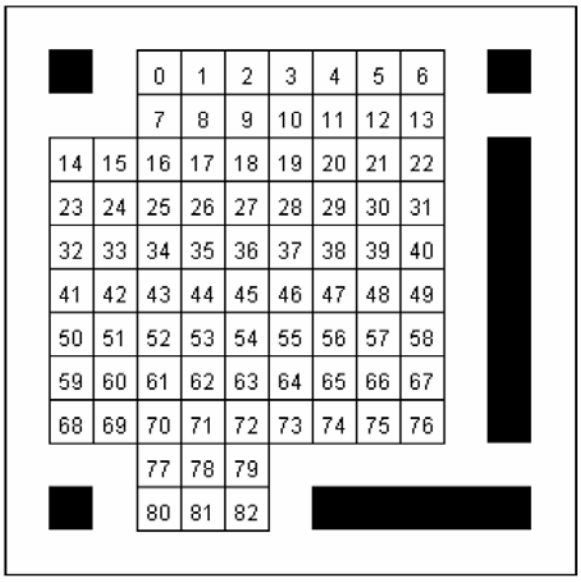

Figure 4: Data region and Encoding order of EZcode. The numbers indicate the bit index within the encoded bit string. The least significant bit has index 0.

For questions, please contact Scanbuy Inc. at info@scanbuy.com.
To create your own EZcodes now, go to www.scanlife.com.