Abstract-Error correcting codes had been an indispensable part of the data communication infrastructure dating all the way back to the advent of vacuum tube computers, for a number of quite obvious reasons. In spite of a good number of powerful codes in existence, this paper proposes a simple systematic block code that deals with burst errors and operates very much like the Hill cipher [I] in its encoding and decoding. It is not meant to outperform any other codes, but might offer a preferable tradeoff between performance and efficiency in some situations.

1. Introduction Any device that gets its input data from an outside source is presumably subject to some types of interference in the real word. For example, an audio CD player inevitably picks up some read errors from any number of obstacles/miscues: dust, scratch, pit, warpage, micro air bubble, displacement, misalignment, etc. Without error correction, the play will be cluttered with clicks and pops as to be unlistenable. By the same token, wireless transmission through the natural medium has to compensate for the unpredictable fluctuation/obstruction of/by the elements, whether earth bound or deep space. The implication is that ECC (error correcting codes) is an integral part of just about any application involving digital data storage/transmission. There are 2 types of ECC's: algebraic such as repetition code, Hamming codes, RS (Reed-Solomon) codes [2], and sparse graph such as Turbo code [3], LDPC (low density parity check) code [4]. Generally speaking, the former is more suited as hard-decision (deterministic) decoder, and the latter softdecision (probabilistic) decoder [5]. They are also classified as block codes in differentiation to the bit or symbol streamed convolutional codes. It suffices to say that these codes had come a long way over the past decades and now routinely come well within 1 dB of the Shannon limit in some cases. But there is no one size fit all code that performs well in all conditions and environments, not to mention the reduction of overall efficiency due to steady increase of size, speed and complexity of data transmission [6]. This paper proposes a code that is in essence just a straight application of the Hill cipher to error detection/correction of burst errors. It is not meant to outperform any other codes, but might offer a preferable tradeoff between performance and efficiency in some situations. Hence, the main strength lies in the simplicity of Hill cipher's encoding/decoding under favorable conditions. Because the input is part of the output, it is a systematic code. For reference, some sample source codes can be found in "An Encoding Kit" [7]. The test/coding platform is shown for all benchmarks in this document, in Table 1 below.

Table 1. Test/coding platform for all benchmarks

Hard- Proces- OS IDE Frameware sor work

Acer AMD E- Win- VISUAL ASP.net

laptop 350 l.60 dows 7 WEB C#

GHZ 4.0 64-bit DEVELOPE

GB R 2010

RAM EXPRESS

II. The general algorithm For convenience, this is named HEC (Hill error correcting) code. Not surprisingly, the redundancy code is simply the equivalence of the ciphertext in the Hill cipher. As an example, let the symbol size be 8 bits (1 byte) and the input A in a block be k symbols. The generator matrix G is a constant m x k matrix. The code word C is made of A + B (B appended to A), where B = G x A and consists of m (redundant) symbols. Thus the code word length is n = k + m, in terms of symbols. The symbol size in B is 1 bit bigger than that in A, to accommodate the prime modular arithmetic. This is necessary because a random matrix is not generally invertible in nonprime modular arithmetic [1]. The prime number in this case is evidently 257 (the 1st one following 256). From the system of linear equations G x A - B = 0, the relevant coefficient matrix for decoding can be seen as M = G - I, where I is the m x m identity matrix. The maximum number of tolerable corrupted symbols is apparently m = n - k. The requirement for successful decoding under correctible conditions is also quite evidently that all applicable sub-matrices of M be invertible, in other words, with non-zero modulo 257 determinants. An applicable sub-matrix means any m x m square matrix in the m x (k + m) M matrix from any m columns in M. In terms of bits, the code word length is 8 x n + m = 8 x (k + m) + m, where m is the number of extra bits needed to accommodate symbol value of 256 (modulo 257 that does not fit into 1 byte) in B. The code rate is thus (8 x k) / (8 x n + m). The only and quite obvious difference between the decoding in this scheme and that of the Hill cipher is that the error locations have to be determined in this scheme, before the unmodified input (plaintext) can be recovered by applying the appropriate inverse matrix to the valid portion of the received code word (ciphertext). Therefore, the error locator constitutes the major difference between HEC and the Hill cipher.