BCH ECC 算法软件仿真分析

本文档主要描述基于 BCH 编码的 8BIT ECC 的仿真及实现过程。经研究和验证后，改善此 ECC 为不需要使用 Galois 的除法运算，这样就可以省去查找表所占的芯片面积；因此可以减少芯片实现的逻辑。本算法采用 BCH 的 CRC104 来进行 ECC 的计算，需要做成并行处理 8bit 的 CRC104；详细的并行 CRC104 实现式子见 C 代码的编码函数。

对于编码过程和解码检错过程可以做到实时性；但解码过程发现数据存在错误后，搜索错误位置需要使用 CHIEN 搜索法进行搜索错误位置；可能出现的错误位置有 4096bits（数据）＋104bits（ECC）＝4200 个位置，因此出现错误位置后纠错需要最坏花费搜索 4200 个位置的时间。如果想做到实时的纠错，可以考虑在芯片上使用在 2 级流水线，一级进行解码检错，一级进行解码搜索纠错。

由于编码过程是计算 CRC104，步骤比较少，这里不详细描述，实现过程参看 CRC104 式子。下面主要描述一下解码的改进过程；另外软件仿真过程中发现 Micron 提供的 bch_encoder_decoder 代码存在一个错误就是如果错误 bit 数目为 8 时，有某几个错误位置的组合其是无法进行正常纠错。后来采用另一 Berlekamp-Massey 算法后可以进行正常的纠错。

1. 解码的计算伴随式方法

   解码使用编码一样的 CRC104 计算，如果结果不为 0，则数据出现错误。否则得到余式，代入余式 $S_j = R(\alpha^j), j = 1,2,\cdots,2t$，可以求得伴随式 $S_1, S_2, \cdots, S_{2t}$。该伴随式在解码计算错误位置中求解 Δ 时使用。

2. 解码的计算错误位置式子

   采用 Berlekamp-Massey,其求解过程如下图：

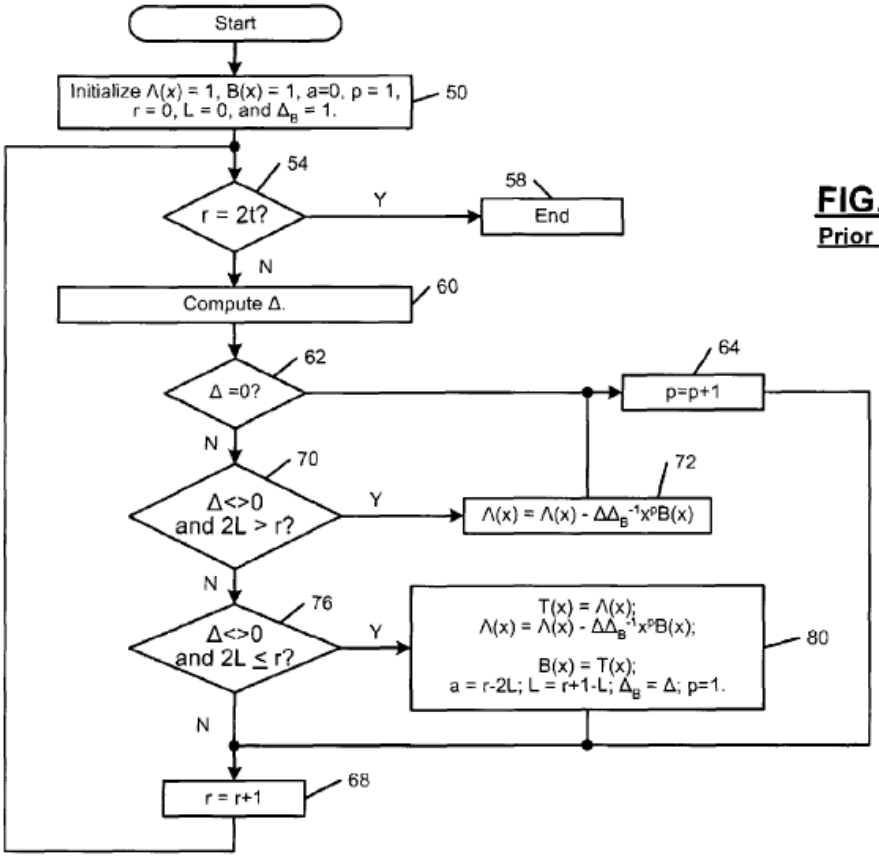

FIG. 4
Prior Art

上图中的Δ计算采用式子$\Delta=\sum_{i=0}^{L}\Lambda_i S_{r-i}$。

软件上对式子$\Lambda(x)=\Lambda(x)-\Delta\Delta_B^{-1}x^pB(x)$进行改进，实现不需要求解$\Delta_B^{-1}$，因为该求解需要做 Galois 的除法，实现逻辑较多。考虑到 CHIEN 搜索法对于等式两边都乘上一个系数，搜索出来的结果也是一样的，因此做法就是对等式两边都乘上Δ_B。

计算错误位置的多项式变为：

$$\Lambda(x)=\Delta_B\Lambda(x)-\Delta x^pB(x)$$

经验证此方法是可行的。

3．解码的 CHIEN 搜索法进行搜索

此处的实现方法是将位置 0 到 4200 逐一带入上面得到的错误位置多项式，如果结果为 0，则该位置就是出错的位置，纠错时，将该位置的数据取反即可。

关于在芯片上的实现，主要的模块有 CRC140 的计算，Galois 的乘法运算，计算伴随式，求错误多项式的实现，CHIEN 搜索法；从 C 实现上来看，逻辑上跟我们以前实现的 REED-SOLOMON 上差不多，但使用的寄存器会增加一些。不过该 BCH ECC 的配置性很强，配置纠错个数时，只需要修改计算错误多项式的模块，其它模块都是一样的；因此对于以后的升级是很有用处的。