APPENDIX Q

Extended Binary Golay Code

This page intentionally left blank.

APPENDIX Q

Extended Binary Golay Code

1.0 Introduction

The extended binary Golay code, G_{24} (sometimes just called the "Golay code" in finite group theory) encodes 12 bits of data in a 24-bit word in such a way that any 3-bit errors can be corrected or any 7-bit errors can be detected. In standard code notation the codes have parameters (24, 12, 8) corresponding to the length of the code words, the dimension of the code, and the minimum Hamming distance between two code words, respectively.[1]

The following sections are C code reference implementation and define the required behavior of encoding and decoding the extended binary Golay code.

2.0 Encoding Golay Code

The extended binary Golay code encoding lookup table can be initialized by the InitGolayEncode() function, and the encoding can be done by the Encode(v) macro of the following C code.

```
#define GOLAY_SIZE      0x1000

// Generator matrix : parity sub-generator matrix P :

uint16_t G_P[12] = {
      0xc75, 0x63b, 0xf68, 0x7b4,
      0x3da, 0xd99, 0x6cd, 0x367,
      0xdc6, 0xa97, 0x93e, 0x8eb
};

/* Binary representation
   1 1 0 0   0 1 1 1   0 1 0 1
   0 1 1 0   0 0 1 1   1 0 1 1
   1 1 1 1   0 1 1 0   1 0 0 0
   0 1 1 1   1 0 1 1   0 1 0 0
   0 0 1 1   1 1 0 1   1 0 1 0
   1 1 0 1   1 0 0 1   1 0 0 1
   0 1 1 0   1 1 0 0   1 1 0 1
   0 0 1 1   0 1 1 0   0 1 1 1
   1 1 0 1   1 1 0 0   0 1 1 0
   1 0 1 0   1 0 0 1   0 1 1 1
   1 0 0 1   0 0 1 1   1 1 1 0
   1 0 0 0   1 1 1 0   1 0 1 1
*/
uint32_t          EncodeTable[ GOLAY_SIZE ];        // Golay encoding table

// encode a 12-bit word to a 24-bit Golay code word
#define Encode(v) EncodeTable[v&0xfff]

// initialize the Golay encoding lookup table
void InitGolayEncode( void )
{
```

[1] Golay, Marcel J. E. *Notes on Digital Coding* in "Proceedings of the IRE," 1949, v.37, p. 657.

```
    for( uint32_t x=0; x < GOLAY_SIZE; x++ ) {
        // generate encode LUT
      EncodeTable[x]=(x<<12);
            for( uint32_t i=0; i<12; i++ ) {
                if( (x>>(11-i)) & 1 )
            EncodeTable[x] ^= G_P[i];
            }
     }
}
```

3.0 Decoding Golay Code

The extended binary Golay code decoding lookup tables can be initialized by the InitGolayDecode() function of the following C code. The 12-bit decoded and corrected word can be calculated by the Decode(v) macro from a 24-bit code word. The number of error bits in a 24-bit code word can be gotten by the Error(v) macro from a 24-bit code word.

```
#define     GOLAY_SIZE              0x1000

uint16_t SyndromeTable[ GOLAY_SIZE ];    // Syndrome table
uint16_t CorrectTable[ GOLAY_SIZE ];     // correction table
uint8_t  ErrorTable[ GOLAY_SIZE ];       // number of error bits table

#define Syndrome2(v1,v2)     (SyndromeTable[v2]^(v1))
#define Syndrome(v)          Syndrome2(((v)>>12)&0xfff,(v)&0xfff)
#define Errors2(v1,v2)       ErrorTable[Syndrome2(v1,v2)]
#define Decode2(v1,v2)       ((v1)^CorrectTable[Syndrome2(v1,v2)])

// get the number of error bits in this 24-bit code word
#define Errors(v)            Errors2(((v)>>12)&0xfff,(v)&0xfff)

// get the 12-bit corrected code from a 24-bit code word
#define Decode(v)            Decode2(((v)>>12)&0xfff,(v)&0xfff)

// Parity Check matrix
uint16_t H_P[12] = {
      0xa4f, 0xf68, 0x7b4, 0x3da,
      0x1ed, 0xab9, 0xf13, 0xdc6,
      0x6e3, 0x93e, 0x49f, 0xc75
};

/* Binary representation
   1 0 1 0   0 1 0 0   1 1 1 1
   1 1 1 1   0 1 1 0   1 0 0 0
   0 1 1 1   1 0 1 1   0 1 0 0
   0 0 1 1   1 1 0 1   1 0 1 0

   0 0 0 1   1 1 1 0   1 1 0 1
   1 0 1 0   1 0 1 1   1 0 0 1
   1 1 1 1   0 0 0 1   0 0 1 1
   1 1 0 1   1 1 0 0   0 1 1 0

   0 1 1 0   1 1 1 0   0 0 1 1
```

```
   1 0 0 1   0 0 1 1   1 1 1 0
   0 1 0 0   1 0 0 1   1 1 1 1
   1 1 0 0   0 1 1 1   0 1 0 1
*/

// calculate the number of 1s in a 24-bit word
uint8_t OnesInCode( uint32_t code, uint32_t size )
{
    uint8_t ret = 0;
    for( uint32_t i=0; i<size; i++ ) {
            if( (code>>i) & 1 )
                ret++;
    }
    return ret;
}


void InitGolayDecode( void )
{
    for( uint32_t x=0; x < GOLAY_SIZE; x++ ) {
        // generate syndrome LUT
        SyndromeTable[x]=0;         // Default value of the Syndrome LUT
        for( uint32_t i=0; i<12; i++ ) {
            if( (x>>(11-i)) & 1 ) SyndromeTable[x] ^= H_P[i];
            ErrorTable[x] = 4;
            CorrectTable[x]=0xfff;
        }
    }

    // no error case
    ErrorTable[0] = 0;
    CorrectTable[0]= 0;
    // generate all error codes up to 3 ones
    for( int i=0; i<24; i++ ) {
        for( int j=0; j<24; j++ ) {
            for( int k=0; k<24; k++ ) {
                uint32_t error = (1<<i) | (1<<j) | (1<<k);
                uint32_t syndrome = Syndrome(error);
                CorrectTable[syndrome] = (error>>12) & 0xfff;
                ErrorTable[syndrome] = OnesInCode(error,24);
            }
        }
    }
}
```

4.0 Decoding the Golay Code (8,1,3)

The one-byte 0x00 or 0xff can also be considered as a binary Golay code (8,1,3). It allows correcting the 0x00 or 0xff transmission of up to 3-bit errors, and detecting 4-bit errors. The (8,1,3) code decoding lookup tables shall be initialized by the InitGolay00FFDecode() function, and the decoding can be done by the Decode00FF(v) macro of the following C code.

```
#define     BYTE_LUT_SIZE           0x100

uint8_t Decode00FFTable[ BYTE_LUT_SIZE ]; // decode 0x00 or 0xff (8,1,3)
```

```
uint8_t Error00FFTable[ BYTE_LUT_SIZE ]; // number of error bits (8,1,3)

#define Decode00FF(v)       Decode00FFTable[v]
#define Error00FF(v)        Error00FFTable[v]

void InitGolay00FFDecode ( void )
{
    // generate (8,1,3) tables
    for( uint32_t i=0; i<BYTE_LUT_SIZE; i++ ) {
        uint32_t j = OnesInCode(i,8);
        Decode00FFTable[i] = j <= 4 ?  0 : 0xff;
        Error00FFTable[i] = j <= 4 ? j : 8-j;
    }
}
```

References

Golay, Marcel J. E. Notes on Digital Coding in "Proceedings of the IRE," 1949, v.37, p. 657.

*** * * END OF APPENDIX Q * * ***