CONSTRUCTION OF QUASI-CYCLIC CODES

by

Thomas Aaron Gulliver

B.Sc., 1982 and M.Sc., 1984
University of New Brunswick

A DISSERTATION SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY
in the Department of
Electrical and Computer Engineering

We accept this dissertation as conforming
to the required standard

Supervisor Dr. V. K. Bhargava

Dr. W.-S. Lu

Dr. P. Agathoklis

Dr. G. C. Shoja

Dr. M. Serra

Dr. N. Dimopoulos

Supervisor: Professor V. K. Bhargava

ABSTRACT

The class of Quasi-Cyclic Error Correcting Codes is investigated. It is shown that they contain many of the best known binary and nonbinary codes. Tables of rate $1/p$ and $(p-1)/p$ Quasi-Cyclic (QC) codes are constructed, which are a compilation of previously best known codes as well as many new codes constructed using exhaustive, and other more sophisticated search techniques. Many of these binary codes attain the known bounds on the maximum possible minimum distance, and 13 improve the bounds. The minimum distances and generator polynomials of all known best codes are given. The search methods are outlined and the weight divisibility of the codes is noted.

The weight distributions of some s-th Power Residue (PR) codes and related rate $1/s$ QC codes are found using the link established between PR codes and QC codes. Subcodes of the PR codes are found by deleting certain circulant matrices in the corresponding QC code. They are used as a starting set of circulants for other techniques. Nonbinary Power Residue codes and related QC codes are constructed over GF(3), GF(4), GF(5), GF(7) and GF(8). Their subcodes are also used to find good nonbinary QC codes.

A simple and efficient algorithm for constructing primitive polynomials with linearly independent roots over the Galois Field of q elements, GF(q), is developed. Tables of these polynomials are presented. These Tables are unknown for polynomials with nonbinary coefficients, and the known binary Tables are incomplete. The polynomials are employed in such diverse areas as construction of error correcting codes, efficient VLSI implementation of multiplication and inverse operations over Galois Fields, and digital testing of integrated circuits.

Using the link established between generalized tail biting convolutional codes and binary QC codes, good QC codes are constructed based on

Optimum Distance Profile (ODP) convolutional codes. Several best rate 2/3 systematic codes up to circulant size 20 are constructed in this manner.

A bound is established for the maximum minimum distance which can be decoded using Weighted Majority Logic. Majority Logic (ML) decodable QC codes are found with the aid of cyclic difference sets and block designs. Others are found using a search of the codewords of the parity check matrix.

Examiners:

Supervisor Dr. V. K. Bhargava

Dr. N. Dimopoulos and Dr. W.-S. Lu

Dr. P. Agathoklis and Dr. G. C. Shoja

Dr. M. Serra and Dr. A. J. Vinck

Table of Contents

List of Tables

Acknowledgements

First and foremost I would like to thank my Supervisor, Professor Vijay K. Bhargava, for his guidance and inspiration during my years at the University of Victoria. I owe much to Dr. Qiang Wang for his invaluable advice and help during our four years of collaboration. Finally, thanks to Professor Micaela Serra for the motivation which spawned Chapter 4 and ultimately Chapter 6. Her diligent reading of this manuscript is also appreciated.

To Karen

Notation

In this dissertation, the following notation is used:

iff	if and only if
$x \mid y$	x divides y without remainder
$x \nmid y$	x does not divides y without remainder
$\forall$	for all
$\sum$	sum
$\prod$	product
$\lfloor x \rfloor$	floor, largest integer less than or equal to x
$\lceil x \rceil$	ceiling, smallest integer greater than or equal to x
$\binom{n}{i}$	$\frac{n!}{i!(n-i)!}$
(x, y)	Greatest Common Divisor of x and y
lcm	Least Common Multiple

Chapter 1

Introduction

Error Correcting Codes were first investigated by R. W. Hamming at Bell Laboratories in 1947. An increasing frustration with relay computers initially motivated this work. Although these machines were capable of error detection, there was no automatic means of rectifying the error. Thus the jobs he submitted were abandoned once an error occurred, and had to be restarted from the beginning. Hamming surmised that if a code could be devised to detect an error, one could also be found to correct it. He set out to find such a code, and in so doing originated the field of Coding Theory. In 1950 Hamming finally published the results of his work [1], introducing many of the fundamental coding concepts used today, such as the Hamming metric and Hamming bound.

In 1948 C.E. Shannon's paper [2], entitled "A Mathematical Theory of Information", established the fundamental concepts of Information Theory. He proved the existence of a coding scheme to ensure an arbitrarily low probability of error, provided that the information rate is less than the channel capacity. Unfortunately, as is the case with most bounds, his proof uses probabilistic methods and provides no insight into how to construct such a scheme. Shannon used as an example in his paper a (7,4) single error correcting code devised by Hamming.

Soon after Shannon's work appeared, Golay [3] published the first paper devoted solely to error correcting codes, an inconspicuous half page in the Proceedings of the I.R.E. He generalized the code mentioned by Shannon and presented some other codes as well. What makes this paper remarkable is that it introduces all but one of the possible classes of perfect linear codes. In fact, it has been called “the best single published page in coding theory” [4].

These early pioneers established error correcting codes and coding theory as an important field of research, and began the difficult and challenging task of finding suitable codes. This difficulty is illustrated with the Gilbert-Varshamov bound, a weaker version of Shannon’s Theorem, which proves the existence of good linear codes. Few coding schemes presently known attain this bound, (and none approach Shannon’s bound). In fact many years passed before a class of error correcting codes was found which reached the Gilbert-Varshamov bound. Since then, the study of Algebraic Curves has led to codes which exceed it.

The main problem of Coding Theory is to find codes with a small redundancy, (or number of parity symbols), and a large minimum distance between codewords. These are conflicting requirements, so it remains to find the largest minimum distance for a given code dimension, and to find the highest code rate, (i.e., the number of information symbols in a codeword), for a given minimum distance.

The best known error correcting codes, those of Hamming, Golay, Bose-Chaudhuri-Hocquenghem[5] and Reed-Solomon[6], are all subclasses of Cyclic codes. Their popularity stems from the existence of an algebraic decoding algorithm, which is made possible by their rich mathematical structure. The Reed-Muller codes [7] are also important because they are Majority Logic Decodable, a scheme which is fast and simple.

Long Goppa codes were the first to meet the Gilbert-Varshamov bound. Goppa [8] and BCH codes are part of the larger class of Alternant codes, thus long Alternant codes also meet the bound. Justesen codes [9] are an infinite class of asymptotically good concatenated codes obtained from

Reed-Solomon codes. The asymptotic lower bound on $\frac{d_{min}}{n}$ for these codes is greater than 0, i.e.,

$$\lim_{n \to \infty} \frac{d_{min}}{n} > 0.$$

d_{min} is the minimum distance of the code, a measure of its error correcting capability. An excellent introduction to these classes of Error-Correcting Codes can be found in [10].

The class of Quasi-Cyclic (QC) codes considered in this dissertation is related to many of these codes. They remain relatively unknown, however, despite the fact that they are good codes, as demonstrated in "Long Quasi-Cyclic Codes are Good" [11]. In fact, it is conjectured that arbitrarily long Quasi-Cyclic codes meet the Gilbert-Varshamov bound[12], (if arbitrarily large primes exist with 2 as a primitive root). On the other hand, it has been shown that the popular BCH codes are not so good, in "Long BCH Codes are Bad" [13]. This in itself is justification for an investigation of Quasi-Cyclic codes.

Quasi-Cyclic codes were introduced by Townsend and Weldon[14]. This was followed shortly thereafter by the works of Karlin[15, 16] and Chen, et al.[17]. Since then extensive research has been done by Bhargava, et. al.[18], [19].

Quasi-Cyclic codes have been shown to be promising codes [17], and their decoding complexity is manageable[16]. As well, many QC codes are majority logic decodable, and subclasses can be found which are so. It has also been shown that many Cyclic codes are equivalent to Quasi-Cyclic codes[20, 21], the most important of these being RS codes. A connection between QC codes and convolutional codes has been discovered by Solomon and van Tilborg [22]. This allows for the convolutional encoding and decoding of many QC codes. Recent attention [23] has centered on this fact because it provides a means of constructing convolutional codes. As well, Quasi-Cyclic codes are equivalent to Rotational Codes, which are used for error correction in computer memories [24].

In this dissertation, the results of a search for good Quasi-Cyclic codes is presented. New construction techniques are developed, some based on the results in [25, 26, 27]. These will be presented in subsequent Chapters. A method of upper bounding the minimum distance of QC codes is developed, and some Majority Logic (ML) decodable QC codes are found.

The next Section introduces the field of error correcting codes. Some fundamental concepts are given, followed by an introduction to the specific class of Quasi-Cyclic codes.

1.1 Error Correcting Code Fundamentals

Error Correcting Codes are divided into two major classes, block codes and convolutional codes. Quasi-Cyclic codes are block codes, but are closely related to convolutional codes, as shown in [22]. This Section provides an introduction to the fundamentals of block codes.

Hamming first conceptualized an error correcting code as containing codewords of length n, partitioned into k symbols of information and $n-k$ symbols of parity. Over GF(2) these symbols are from the set $\{0,1\}$. A set of symbols can be any field, GF(q). The rate of a code is defined as the ratio

$$r = \frac{k}{n},$$

the number of information symbols per codeword.

A code of length n is linear iff it is a subspace of the vector space of dimension n, and so is an additive group. An important consequence of this is that the sum of two codewords in a linear code must also be a codeword. The Hamming weight of a codeword, $\mathbf{wt}[x]$, is the number of nonzero elements contained in it. The Hamming distance between two codewords, $\mathbf{d}(x, y)$, is defined as the number of places in which they differ,

$$\mathbf{d}(x,y) = \mathbf{wt}[x-y].$$

The smallest Hamming distance between all codewords in a code is called the minimum distance,

$$d_{min} = min\ \mathbf{d}(x,y)\ \forall x,y; x \neq y.$$

The minimum distance of a linear code is the weight of the smallest nonzero codeword, since the linear combination of any two codewords is also a codeword,

$$d_{min} = min\ \mathbf{wt}[x]\ \forall x; x \neq 0.$$

The number of errors a code can correct is

$$t = \lfloor \frac{d_{min}-1}{2} \rfloor,$$

where $\lfloor x \rfloor$ is the largest integer less than or equal to x. As well, a code can detect l errors where

$$t + l + 1 \leq d_{min}$$

and $l > t$.

The Generator matrix, G, of an (n,k) linear block code [28] is a $k \times n$ matrix of linearly independent codewords. All codewords can be formed from a combination of the rows of this matrix, thus there are q^k codewords. The parity check matrix of this code is an $n-k \times n$ matrix H such that

$$GH^T = \mathbf{0},$$

where $\mathbf{0}$ is a $k \times n-k$ matrix of zeros. A code is called self-dual if the code generated by G is equivalent to the code generated by H. Every matrix G is equivalent to one whose first k columns are a $k \times k$ identity matrix. In this case

$$G' = [I_k\ P],$$

where P is a $k \times n - k$ Parity matrix. A generator matrix in this form is called Systematic. The parity check matrix is then

$$H' = [-P^T \; I_{n-k}]$$

The inner product of two codewords, x and y, is defined as

$$x * y = \sum_{i=1}^{n} x_i y_i \bmod q.$$

If the inner product of two codewords is zero, they are said to be orthogonal. Thus they are orthogonal if $x * y = 0$. For a binary code, the weight of the sum of two codewords, x and y is

$$\mathbf{wt}[x + y] = \mathbf{wt}[x] + \mathbf{wt}[y] - 2\mathbf{wt}[xy],$$

where xy is the product of x and y, which has 1's only where x and y both do. E.g. if $x = 11101$ and $y = 10011$, then $xy = 10001$.

A linear code is Cyclic if a cyclic shift of any codeword is also a codeword. Elementary row operations (permutations and combinations) on G preserve the cyclic properties, while column operations do not. Thus every Cyclic code can be put in systematic form and still be Cyclic.

The weight enumerator of a code is defined as a polynomial in z [29],

$$A(z) = \sum_{i=0}^{n} A_i z^i$$

where A_i is the number of codewords of weight i. From this definition it is evident that

$$\sum_{i=0}^{n} A_i = q^k,$$

the total number of codewords, with q equal to the symbol size.

MacWilliam's Identities [29] relate the weight distributions of G and H, and thus their error correcting capability. If A_j denotes the number of codewords of weight j in G, and B_i the number of codewords of weight i in H, than we have

$$B_i = q^{-k} \sum_{j=0}^{n} A_j \sum_{s=0}^{n} \left(\begin{array}{c} j \\ s \end{array} \right) \left(\begin{array}{c} n-j \\ i-s \end{array} \right) (-1)^s (q-1)^{i-s}, \tag{1.1}$$

where q is the code symbol size.

Two codes G_1 and G_2 are called equivalent if they differ only in the order of the symbols in the codewords. Thus changing the order of the columns of the Generator matrix does not result in a different code.

In this dissertation a best code is considered to be one which has the largest possible minimum distance for the given code dimensions, n and k, and class of error correcting codes. The term good code defines a code which has the maximum known minimum distance for the class of codes. An optimal code is one which achieves the maximum possible minimum distance for a linear code with the same dimensions.

Note that in general the dual of a best rate k/n code is not a best rate $(n-k)/n$ code. One important exception to this is the class of Maximum Distance Separable (MDS) codes, which includes the well known Reed-Solomon (RS) codes. In this case $d_{min} = n - k + 1$, and both G and H must be MDS codes. More will be said about these codes in Chapter 6.

An excellent treatment of the theory of Error Correcting Codes is given in references [10, 29, 30].

1.2 Quasi-Cyclic Codes

This Section introduces the class of Quasi-Cyclic codes and provides some preliminary results required in subsequent Chapters. It is based on [10] and [14].

Definition 1.1[14] *An (n,k) linear block code of dimensions $n = mn_o$ and k = mk_o, is called Quasi-Cyclic if every cyclic shift of a codeword by n_o symbols* *yields another codeword.*

As an example, consider the following generator matrix of an (8,4) binary linear code over GF(2)

$$G = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 \end{bmatrix}. \tag{1.2}$$

This code is Quasi-Cyclic with $n_o = 2$, since every row of G is the same as the previous with a cyclic shift of two positions. If the columns of G are ordered according to the sequence $1, k+1, 2, k+2, \cdots$, the resulting generator matrix is composed of two 4×4 circulant matrices,

$$G = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \end{bmatrix}. \tag{1.3}$$

An $m \times m$ circulant matrix is defined as

$$C = \begin{bmatrix} c_0 & c_1 & c_2 & \cdots & c_{m-1} \\ c_{m-1} & c_0 & c_1 & \cdots & c_{m-2} \\ c_{m-2} & c_{m-1} & c_0 & \cdots & c_{m-3} \\ \vdots & \vdots & \vdots & & \vdots \\ c_1 & c_2 & c_3 & \cdots & c_0 \end{bmatrix}, \tag{1.4}$$

where c_i is an element of GF(q). This example shows that any (n, k) Quasi-Cyclic code over GF(q) is equivalent to an (mn_o, mk_o) code with an $mk_o \times mn_o$ generator matrix composed of $m \times m$ circulant matrices,

$$G = \begin{bmatrix} C_{1,1} & C_{1,2} & C_{1,3} & \cdots & C_{1,n_o} \\ C_{2,1} & C_{2,2} & C_{2,3} & \cdots & C_{2,n_o} \\ \vdots & \vdots & \vdots & & \vdots \\ C_{k_o,1} & C_{k_o,2} & C_{k_o,3} & \cdots & C_{k_o,n_o} \end{bmatrix}. \tag{1.5}$$

A circulant matrix C is uniquely specified by a polynomial formed of the entries of the first row, $c(x) = c_0 + c_1 x + c_2 x^2 + \cdots + c_{m-1} x^{m-1}$, i.e., there is a one-to-one mapping between the circulant matrices C_i and the polynomials $c_i(x)$. This is formally stated in the following Theorem.

Theorem 1.2[10] *The algebra of $m \times m$ circulant matrices over a Field F is isomorphic to the algebra of polynomials in the ring F[x]/($x^m - 1$).*

The following results for circulant matrices can now be stated.

Theorem 1.3[10] *The sum and product of two circulants is a circulant. In particular, $AB = C$ where $c(x) = a(x)b(x)$ mod $x^m - 1$.*

Thus we can use the more convenient polynomial representation of the circulants.

Theorem 1.4[10] *A circulant matrix C has an inverse C^{-1} iff c(x) is relatively prime to $x^m - 1$. The inverse is then $c^{-1}(x)$ where $c(x)c^{-1}(x) = x^m - 1$.*

Definition 1.5[10] *The transpose of a circulant matrix, C^T, is defined by the polynomial $c_0 + c_{m-1}x + \cdots c_2x^{m-2} + c_1x^{m-1}$.*

Definition 1.6[10] *The reciprocal of c(x) is defined as $c^*(x) = c_{m-1} + c_{m-2}x + \cdots c_1x^{m-2} + c_0x^{m-1}$.*

Definition 1.7 *The complement of c(x) is defined as $\overline{c(x)} = \mathbf{1} - c(x)$, where $\mathbf{1}$ is the polynomial with all one coefficients.*

Therefore $\overline{c(x)} + c(x) = \mathbf{1}$.

From [10] we have the following results on binary double circulant, or rate 1/2 QC codes. Let A and B be defined as,

$$A = [IC_a], \quad B = [IC_b]$$

where C_a and C_b are $m \times m$ circulant matrices. A and B are equivalent codes if

a) $C_b = C_a^T$,

b) $c_b(x) = c_a^*(x)$,

c) $C_b = C_a^{-1}$,

d) $c_b(x) = c_a(x)^2$ and m is odd,

e) $c_b(x) = c_a(x^u)$ and $(u, m) = 1$, i.e., u and m are relatively prime.

These properties are useful for identifying equivalent codes. This allows a reduction in the number of codes which must be examined to determine the best possible.

For a systematic QC code, G has the form,

$$G = \left[I_{km} \begin{array}{ccccc} C'_{1,1} & C'_{1,2} & C'_{1,3} & \cdots & C'_{1,n-k} \\ C'_{2,1} & C'_{2,2} & C'_{2,3} & \cdots & C'_{2,n-k} \\ . & . & . & & . \\ . & . & . & & . \\ C'_{k,1} & C'_{k,2} & C'_{k,3} & \cdots & C'_{k,n-k} \end{array} \right] \tag{1.6}$$

where I_{km} is a $k_o m \times k_o m$ identity matrix, and the C'_i are circulant matrices. Only systematic codes are considered in this dissertation. This is justified by the fact that all linear codes are equivalent to a systematic code, and circulant matrices which are not invertible generally produce poor error correcting codes. This is further explained in the next Section. The dual rate $n - k/n$ QC code is defined by an $(mn_o, m(n_o - k_o))$ generator matrix H,

$$H = \left[I_{m(n-k)} \begin{array}{cccc} C^T_{1,1} & C^T_{2,1} & \cdots & C^T_{k,1} \\ C'_{1,2} & C^T_{2,2} & \cdots & C^T_{k,2} \\ C^T_{1,3} & C^T_{2,3} & \cdots & C'_{k,3} \\ . & . & & . \\ . & . & & . \\ C^T_{1,n-k} & C^T_{2,n-k} & \cdots & C^T_{k,n-k} \end{array} \right] \tag{1.7}$$

1.3 Previous Results

Since the appearance of the first paper on QC codes, several authors have presented construction results. Chen, et. al.[17] provide a Table of best rate 1/2 codes for m up to 21. As well they show that Power Residue codes with one symbol deleted are equivalent to Quasi-Cyclic codes. This method is used to advantage in Chapters 3 and 6. In [18] the equivalence of rate 1/2 codes for m up to 16, and the minimum distance of rate 2/3 codes up to $m = 18$, is given. In [25] the weight distributions of these rate 2/3 codes is presented. The weight distribution of the rate 1/2 QC codes with maximum minimum distance, up to $m = 21$, (from [17]), is given in [31]. In [21, 27] some (mk, k) Cyclic codes are transformed into Quasi-Cyclic codes. In [26] rate $1/p$ QC codes for $m = 7$ and 8 are presented. Several of these codes appear in [32]. Good QC codes are also given in [15, 33]. Other fragmented results exist, but there is no unified collection of all known QC codes. In this dissertation, these codes are compiled along with those found as a result of this work (excluding inferior codes).

By good it is meant the largest known minimum distance, d_{min}, for a QC code of the given dimensions. If the code attains the maximum possible minimum distance for a QC code of the given dimensions, it is a best code. Rate $1/p$ and $(p-1)/p$ binary codes are given for m up to 16, and p up to 18, which is the present practical limit of the construction algorithms. As well, the binary rate 1/2 codes are extended to $m = 31$ and rate 2/3 codes to $m = 25$.

An exhaustive search of all codes is tractable only for the simplest codes. Thus we rely on techniques to reduce the set of candidate polynomials which must be examined to find a good or best code. Only rate $1/p$ and $(p-1)/p$ systematic codes are considered in this dissertation since they are of most interest and practicality, and non-systematic codes are not easily decoded since some circulants have no inverse [16]. As well, a rate $1/p$ code can be put in systematic form if one of the circulant matrices in the generator matrix is invertible. This is not possible only when all of the matrices are singular. Fortunately, codes composed entirely of singular circulant matrices are a small subset of the possible QC codes, and rarely are they contained in the set of best codes.

A simple and obvious method of constructing good codes is to extend (add circulants) or puncture (delete circulants) existing good codes. A punctured QC code is termed a subcode of an existing QC code. Thus codes with a good subset of possible circulants must be found. It is well known that many Cyclic codes have QC equivalents. The Power Residue (PR) codes are Cyclic codes which can be transformed into QC codes using the Normal Basis Theorem[17]. Subcodes can be created from a subset of the generator polynomials of these QC codes. Justification for using PR codes comes from the fact that they are known to be good codes[34]. This set of polynomials reduces considerably the search time. As well, MacWilliams [27] and Solomon and van Tilborg [22] give methods to construct QC codes from other Cyclic codes. Subcodes and extensions of these codes can also be formed.

The area of Spread Spectrum communications has received much attention and found wide applications in solving many important communication problems. In [35] it is shown that an M-ary code, with $M = 4$ or 8, is the best coding scheme to combat worst case interference. Unfortunately, few codes are known with nonbinary symbols beyond the Reed-Solomon codes, and RS codes have a restricted block length. M-ary block codes have not received much attention except for RS codes. This is primarily due to the fact that there are few non-RS M-ary block codes known. As well, the majority of known convolutional codes are binary.

Most types of binary codes can be generalized to Q-ary codes. For instance, nonbinary Hamming codes exist for many lengths, as do nonbinary BCH and Cyclic codes. In this dissertation nonbinary QC codes are constructed.

1.4 Thesis Outline

Subsequent Chapters are organized as follows. In Chapter 2, general construction algorithms are devised for rate $1/p$ and rate $(p-1)/p$ Quasi-

Cyclic codes. The concept for rate $1/p$ codes is based on [26], where integer linear programming is used to find best QC codes for $m = 7$ and 8. The technique for rate $(p-1)/p$ codes is an extension of the method in [18] for rate 2/3 binary codes.

The binary Power Residue (PR) codes are investigated in Chapter 3. These codes were first introduced by Chen, et. al.[17] and later investigated by Bhargava[36]. In this Chapter, PR codes are constructed up to block length 10,000 and circulant size 32. Subcodes of these codes are also given. They are constructed by deleting circulants from the original PR code in QC form.

Chapter 4 presents a construction algorithm for primitive polynomials with linearly independent roots. These are required to form a normal basis over GF(q^m), which is then used to transform a PR code to a QC code.

In Chapter 5 rate 2/3 binary QC codes are constructed from optimum distance profile convolutional codes.

Chapter 6 is an extension of Chapters 2 and 3. The techniques developed for binary codes are extended to nonbinary codes over GF(3), GF(4), GF(5), GF(7) and GF(8), and Maximum Distance Seperable codes are also found over GF(11), GF(13) and GF(16).

A summary of results and suggestions for future research is given in Chapter 7.

Appendix A provides a means of obtaining a quick estimate of the minimum distance. Appendix B presents some Majority Logic (ML) decodable QC codes.

Chapter 2

Some Best Rate 1/p and Rate (p - 1)/p Binary Systematic Quasi-Cyclic Codes

2.1 Introduction

In this Chapter, a computationally efficient search technique is developed for finding good rate $1/p$ QC codes. The results of [18] are extended for rate $(p-1)/p$ QC codes, and of [26] for rate $1/p$ codes. The motivation comes from a recent paper by Verhoeff[32], which presents a Table of bounds on the maximum possible minimum distance of binary linear codes. Many of the codes found using these methods meet or improve the bounds. The best known binary Quasi-Cyclic codes are tabulated, including those found previously by others.

An exhaustive search is intractable for all but the simplest codes. Thus one must rely on techniques to reduce the set of candidate polynomials which must be examined to find a good or best code. Only rate $1/p$ and $(p-1)/p$ systematic codes are considered for reasons given in the previous Chapter. A rate $1/p$ systematic Quasi-Cyclic (QC) code has an $m \times mp$ generator matrix of the form

$$G \;=\; [I_m,\; C_1,\; C_2,\; C_3,\; \cdots\;, C_{p-1}] \tag{2.1}$$

where I_m is an $m \times m$ identity matrix and the C_i are $m \times m$ binary circulant matrices. The dual rate $(p-1)/p$ QC code is defined by the $(p-1)m \times pm$ generator matrix

$$H' = \begin{bmatrix} C_1^T & \\ C_2^T & \\ C_3^T & I_{(p-1)m} \\ \vdots & \\ C_{p-1}^T & \end{bmatrix}. \tag{2.2}$$

This is equivalent to the more common representation [18]

$$H = \begin{bmatrix} & C_1 \\ & C_2 \\ I_{(p-1)m} & C_3 \\ & \vdots \\ & C_{p-1} \end{bmatrix}. \tag{2.3}$$

Since in general the dual of a best rate $1/p$ code is not a best rate $(p-1)/p$ code, these two types of codes are considered separately. The following Theorems present some preliminary results.

Theorem 2.1 *A QC code has only even weight codewords iff every row of G has even weight.*

Proof Every row of G is a codeword, so if any row has odd weight, there exists an odd weight codeword. If one row has even weight, all rows have even weight due to the QC structure of G. Suppose all rows have even weight d. Then the sum of l rows will also have even weight, since the total number of ones is dl, and each mod 2 sum eliminates two out of this total. Thus the codeword weight must be even.$\square$

Corollary 2.2 *A rate 1/p systematic QC code has only even weight codewords iff*

$$\sum_{i=1}^{p-1} \mathbf{wt}[c_i(x)]$$

is odd.

Corollary 2.3 *A rate $(p-1)/p$ QC code has only even weight codewords iff* *all* $c_j(x), j = 1, 2, \cdots, p-1$ *have odd weight.*

Proof Suppose $c_k(x)$ has even weight and consider the codeword formed when $i(x) = x^{m(k-1)}$. This codeword $i(x)G$ will have odd weight since $\mathbf{wt}[i(x)]$ is odd and $\mathbf{wt}[i(x)c_k(x)]$ is even. Conversely, if $\mathbf{wt}[c_k(x)]$ is odd, $\mathbf{wt}[i(x)]$ is odd and $\mathbf{wt}[i(x)c_k(x)]$ is also odd. Thus this codeword has even weight. All codewords can be created by combining rows of G, so they all must have even weight.$\square$

Theorem 2.4 *A QC code has the weights of all codewords divisible by 4 if all rows of* G *are orthogonal.*

Proof Clearly each row must have weight a multiple of 4 since every row of G is a codeword. If any two rows are orthogonal, their inner product is zero, which means that they have a multiple of two locations in common. The weight of two rows is

$$\mathbf{wt}[g_i + g_j] = \mathbf{wt}[g_i] + \mathbf{wt}[g_j] - 2\mathbf{wt}[g_i g_j].$$

Since g_i and g_j are orthogonal, the weight of their product is even. Thus $\mathbf{wt}[g_i + g_j]$ is a multiple of 4.$\square$

The proof of orthogonality is simplified because the code is QC. Only those codewords corresponding to $i(x)$ equal to the distinct cyclic cosets of weight 2 need be checked.

Extending this result to n rows, we have

$$\begin{aligned} \mathbf{wt}[\textstyle\sum_i g_i] &= 2^0(\textstyle\sum_i \mathbf{wt}[g_i]) - 2^1(\sum_i \sum_{i<j} \mathbf{wt}[g_i g_j]) \\ &\quad +2^2(\textstyle\sum_i \sum_{i<j} \sum_{j<k} \mathbf{wt}[g_i g_j g_k]) - 2^3(\sum_i \sum_{i<j} \sum_{j<k} \sum_{k<l} \mathbf{wt}[g_i g_j g_k g_l] \\ &\quad + \cdots + (-1)^{n+1} 2^n (\textstyle\sum_i \cdots \sum_{q<r} \mathbf{wt}[g_i g_j \cdots g_r]). \end{aligned} \tag{2.4}$$

Since G is a QC code, the computation of (2.4) is simpler because every row of G has the same weight. For example, consider the (8,4) QC code shown in Chapter 1. The weight of the codeword composed of all rows of G is from

(2.4),

$$\mathbf{wt}[\sum_i g_i] = 2^0(4\times 4) - 2^1(4\times 2 + 2\times 2) + 2^2(4\times 1) - 2^3(1\times 0) = 8.$$

The first term is the weight of the rows of G, which is 4 times the weight of one row. The second term is the weight of all row pairs. There are two types of pairings possible, corresponding to the unique cyclic cosets of length 4 and weight 2.

This code is equivalent to the first order Reed-Muller code of dimension 8, and the (8,4) extended Hamming code. In Appendix B, it is shown that this code is one-step weighted majority logic decodable.

A more interesting example of weight divisibility is the (128,8) $d_{min} = 64$ rate 1/16 QC code. It is composed of the 16 8×8 distinct circulants with odd weight and has weights divisible by 64. Since

$$\sum_{i=1}^{p-1} \mathbf{wt}[c_i(x)] = 63$$

the rows of G have even weight. As well the rows of G are orthogonal, so the weights are divisible by 4. If $c_j(x)$ is contained in this code, so is $\overline{c_j(x)}$, since m is even. The all 1's codeword, $\mathbf{1}$, is also contained in this code.

The 8 rate 1/2 subcodes formed of the pairs $c_j(x)$ and $\overline{c_j(x)}$ have the following weight distribution:

Weight	Count
0	1
4	28
8	198
12	28
16	1

Thus the minimum distance of the (128,8) code is at least 32. The following proves it is 64.

Consider the codewords corresponding to $\mathbf{wt}[i(x)]$ odd. $i(x)$ essentially selects rows of G to be summed to form the codeword, in this case an odd number of rows. Now if the weight of the codeword bit corresponding to the kth position of $c_j(x)$ is 1, the weight of the bit corresponding to the kth position of $\overline{c_j(x)}$ will be 0. This is due to the fact that if the weight of the codeword bit in position k of $c_j(x)$ is 1, it is the sum of an odd number of ones and an even number of zeros. Then the weight of the codeword bit in position k of $\overline{c_j(x)}$ is the sum of an even number of ones and an odd number of zeros. Since the sum of an even number of ones is zero, for every codeword bit which is one, there is a corresponding bit which is 0. Thus the codeword has weight $n/2 = 64$, and this is true for every odd weight $i(x)$.

For $\mathbf{wt}[i(x)]$ even, proceed as follows. Consider the all ones codeword, $\mathbf{1}$. In this case $\mathbf{wt}[i(x)] = 8$, which is even. $i(x)$ is the sum of a weight 2 polynomial and a weight 6 polynomial, i.e.,

$$i(x) = i_1(x) + i_2(x),$$

where $i_1(x)$ has weight 2 and $i_2(x)$ has weight 6. They divide the Generator matrix into two sections, one with 2 rows and the other with 6. If a column sum of one section is one, the other will have sum 0 since all columns of G have odd weight. Thus one codeword will have weight w, and the other $128 - w$.

A column section of G corresponding to $i_1(x)$ can assume only one of the four possible 2-tuples:

$$\begin{matrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \end{matrix}$$

If the 2-tuple has odd weight, the section of the column of G corresponding to $i_2(x)$ will have even weight, and vice versa. This is because G contains only odd weight columns. Thus these six rows of G will contain two of all possible 6-tuples. Since half of these have odd weight, the codeword $i_2(x)G$

has weight 64, and then so does $i_1(x)G$. This result can be extended to the case where $\mathbf{wt}[i_1(x)] = 4$ and $\mathbf{wt}[i_2(x)] = 4$. Thus all but two codewords have weight 64, and that is the minimum distance.

The weight structure of this code is then,

Weight	Count
0	1
64	254
128	1

The extention of this result to $m = 2^l$ is given in the following Theorem.

Theorem 2.5 *The QC code composed of all circulant matrices, C_i, for which $\mathbf{wt}[c_i(x)]$has odd weight, and $m = 2^l$, is a (2^{m-1},m) code with weight distribution*

Weight	Count
0	1
2^{m-2}	$2^m - 2$
2^{m-1}	1

Proof This follows directly from the above example. When $\mathbf{wt}[i(x)]$ is odd, the weight of the codeword is $n/2$. When $\mathbf{wt}[i(x)]$ is even, the circulants are divided into two sections, as in the above example, which leads to codewords of weight $n/2$.□

For $m = 16$, the corresponding code has weight distribution

Weight	Count
0	1
16384	65534
32768	1

When $m = p$, p an odd prime, the rate $1/p$ QC code composed of all odd weight circulants has the following weight structure,

Weight	Count
0	1
$2^{m-2}-1$	$2^{m-1}-1$
2^{m-2}	$2^{m-1}-1$
$2^{m-1}-1$	1

As in the previous example, these codes contain all possible odd weight columns except for the all 1 column. Thus the blocklength is $(2^m-2)/2$. For example, the (15,5) code has weight distribution,

Weight	Count
0	1
7	15
8	15
15	1

Adding an overall parity bit yields the distribution,

Weight	Count
0	1
8	30
16	1

which is the same form as the (128,8) code. The corresponding (63,7) code has weight distribution

Weight	Count
0	1
31	63
32	63
63	1

Thus for all m prime or a power of 2, a QC code exists with all odd weight circulant matrices, length $n = 2^{m-1}$ and but two codewords with weight $n/2$. This is so because only for these values of m do all odd weight circulants have cycle m.

Theorem 2.6 *The dual of these codes, rate* $(p-1)/p$ *QC codes, has* $d_{min} = 4$.

Proof For the dual code, G represents the parity check matrix. In order for the code to have a weight 3 codeword, three columns of this matrix must be linearly dependent [10]. This requires that the sum of two columns equal a third. However, the sum of two columns will have an even weight, since the total number of ones in them is the sum of two odd numbers. Since all columns of G have odd weight, this sum cannot equal any column of G. Thus $d_{min} > 3$. 4 columns of G are dependent, since a weight 3 column exists, as does the columns of the identity matrix. Therefore $d_{min} = 4$.□

2.2 Rate 1/p Codes

The easiest but most time consuming method of finding good codes is an exhaustive search. It involves examining all possible combinations of generator polynomials, and thus quickly becomes an impractical solution. Consider the following illustration. From [37], the number of circular permutations of length and cycle period m is

$$N_m = \frac{1}{m} \sum_{\substack{d, \\ d \mid m}} \mu(\frac{m}{d}) q^d, \tag{2.5}$$

which is also the number of circulant matrices of dimension m which have unique columns, and eliminating all those $c_h(x)$ which satisfy $c_h(x) = x^n c(x) \ mod \ x^m - 1, 1 < n < m$, for some h. $\mu(m)$ is the Möbius function [37, 38], defined by

$$\mu(m) = \begin{cases} 1 & \text{if } m = 1; \\ 0 & \text{if } m \text{ is divisible by a square;} \\ (-1)^k & \text{if } m \text{ is the product of } k \text{ distinct primes.} \end{cases} \tag{2.6}$$

The total number of circulant matrices of dimension m is then

$$T_m = \sum_{\substack{j, \\ j \mid m}} N_j, \tag{2.7}$$

including the all zero and all one matrices. (Note that in this Chapter we only consider binary matrices.) If $m = 11$, there are $T_m = 188$ possible generator polynomials. For a rate $1/p$ code, one would have to examine $\binom{185}{p-1}$ systematic codes to find the best possible, (excluding the all zero, identity and all ones matrices). Even if equivalent codes are identified, this number increases rapidly with increasing p, so the computational limit is attained quite quickly and very few best codes can be found by this means. A more attractive method [26], uses integer linear programming to search out the best codes, however the computational effort also increases quickly with increasing m and p.

The method presented here uses a simple approach based on an 'ascent algorithm', so called because it attempts to create a new code from the previous one which has a higher minimum distance.

First, the $n \times n$ array of the weights of distinct partial codewords, i.e., of the T_m possible distinct circulants, is formed as in [26],

$$D = \begin{array}{c|cccc} & C_1 & C_2 & \cdots & C_n \\ \hline i_1 & w_{11} & w_{12} & \cdots & w_{1n} \\ i_2 & w_{21} & w_{22} & \cdots & w_{2n} \\ \vdots & \vdots & \vdots & & \vdots \\ i_n & w_{n1} & w_{n2} & \cdots & w_{nn} \end{array} \tag{2.8}$$

where i_j is the jth distinct information vector, C_k is the kth distinct circulant matrix, and w_{jk} is the weight of $i_j(x)c_k(x)$ *mod* $x^m - 1$. By distinct information vector and circulant matrix it is meant to exclude those polynomials equal to $c_n(x) = x^n c(x), 1 < n < m$, i.e., cyclic shifts of $c(x)$ which would have the same weight structure. Note that $i_j(x) = c_k(x)$ when $j = k$, thus D is a symmetric matrix.

To begin the search, an arbitrary code of the desired rate, $1/p$, is formed of the first, say, p circulants, and the row sums of the corresponding columns of D found. Since only systematic codes are considered, $c(x) = 1$ is always contained in the code. Clearly the minimum distance of this code is the minimum of these row sums, since the weights of all distinct codewords are contained in them. To improve the code, a new circulant is found to replace one presently in the code, so that the minimum distance, or row sum, is increased. If one is not found, the new circulant is chosen as the one which minimizes the number of minimum distance codewords. This process is repeated until the required minimum distance is achieved. In every iteration, a new circulant is added, and to avoid cycling, the previously deleted circulant cannot immediately return. As well, there is a limit placed on the number of times a circulant can enter the code. To avoid complete exclusion, the counters are reset to zero after a specified number of iterations to allow all circulants to enter the code again. With this simple algorithm, all presently known best rate $1/p$ QC codes, up to $m = 16$, have been found. This includes several codes which improve the bounds in [32]. A flowchart of the algorithm is given in Table 2.1. Subsets of polynomials found using the methods in [39, 27] (and in the following Chapter), were used as initial conditions to speed up the search.

Another method used to accelerate the search is described in Appendix A. It was employed to obtain an initial estimate of the minimum distance of QC codes. By providing an upperbound on the minimum distance, this technique eliminated most codes which did not attain the target minimum distance.

Note that the complete weight distribution of a QC code can be found

Table 2.1: Flowchart of the Search Algorithm

from the D matrix, since

$$\sum_{d|m} dN(d) = 2^m, \tag{2.9}$$

The following results were used to accelerate the search.

Theorem 2.7 *The number of odd weight generator polynomials in a rate* $1/p$ *code must be a minimum of*

$$\lceil \frac{d_t}{m} \rceil$$

where d_t *is the target minimum distance and* $\lceil x \rceil$ *is the smallest integer greater than or equal to* x.

Proof Consider the information polynomial $i(x)$ and generator polynomial $c(x)$. From [25] we have Table 2.2. If $i(x)$ is the all ones vector, $i(x)c(x) = i(x) = \mathbf{1}$ if $\mathbf{wt}[c(x)]$ is odd, and $i(x)c(x) = \mathbf{0}$, the all zero vector, if $\mathbf{wt}[c(x)]$ is even. Thus there exists a codeword of weight rm, where r is the number of odd weight generator polynomials, and this must be greater than or equal to the target minimum distance, d_t. $\square$

It is obvious that the minimum distance of a rate $1/p$ systematic QC code is no greater than the sum of the weights of the generator polynomials, c_k, which corresponds to $i(x) = 1$.

The minimum distances and generator polynomials of the best rate 1/2 codes are given in Tables 2.3 and 2.4. Table 2.3 extends the results in [18] by dividing the best codes into equivalence classes for m up to 24. This

Table 2.2: Weights in Quasi-Cyclic Codewords

$\mathbf{wt}[i(x)]$	$\mathbf{wt}[c(x)]$	$\mathbf{wt}[i(x)c(x) \ mod \ x^m - 1]$
even	even	even
even	odd	even
odd	even	even
odd	odd	odd

information is useful in reducing the search time for rate $1/p$ codes because higher rate equivalent codes can be identified, as noted in [18]. Table 2.4 gives the generator polynomials for $m = 25$ to 31. These results are used to identify polynomials which have insufficient distance properties so they can be eliminated from the search for rate $(p-1)/p$ codes.

The best rate $1/p$ QC codes for $m = 3$ to 16 are given in Tables 2.5 to 2.27. Generator polynomials are given in octal, with the highest power coefficient on the right, i.e., $713_8 = 1 + x + x^2 + x^5 + x^7 + x^8$. For conciseness, all generator polynomials for a given m are numbered and listed separately, as in Tables 2.6, 2.13, 2.15, etc. The Tables of codes list the corresponding generator polynomial numbers, instead of the polynomials themselves. For example, Table 2.19 lists 84 polynomials for $m = 12$. Table 2.20 gives the best rate $1/p, m = 12$ QC codes for $p = 3$ to 18. For each p is given the code dimensions, the minimum distance, and a list of the generator polynomial numbers for the particular QC code from the previous Table. The minimum distances of these codes are compiled in Table 2.28. A superscript ^{o} denotes a best possible Quasi-Cyclic code. This was determined either by exhaustive search, or meeting a known upper bound.

2.3 Rate (p-1)/p Codes

The construction of these codes follows the method in [25], in that the weight distributions of the dual rate $1/p$ codes are found first, then transformed using MacWilliam's identities. This is more computationally efficient than computing the minimum distance directly, since the original code has $2^{(p-1)m}$ codewords while the dual code has only 2^m codewords. The following can be used to refine the search for good rate $(p-1)/p$ QC codes.

The set of generator polynomials, $c_j(x)$, is the same as in the previous section, all distinct m-tuples (excluding cyclic shifts). Let $i(x)$ denote an information vector, and $p(x)$ a parity vector, i.e., $p(x) = i(x)c_j(x)\ mod\ x^m -$

1.

Theorem 2.8 *For $p < q$, the minimum distance, d_p, of a rate $(p-1)/p$ QC code formed from a subset of the $m \times m$ circulants from a rate $(q-1)/q$ QC code with minimum distance d_q, is lowerbounded by*

$$d_p \geq d_q.$$

Proof Consider the above two QC codes, one having p $m \times m$ circulants and the other q $m \times m$ circulants, $p < q$. Let $i(x)$ be 0 for those circulants of the larger code that are not in the smaller one. Clearly this denotes a codeword of the smaller code if the corresponding all zero sections of the original codeword are deleted. Thus this code is contained in the larger one and so its minimum distance cannot be smaller than the larger code. □

Corollary 2.9 *The minimum distance of a rate $(p-1)/p$ systematic QC code is no greater than the lowest minimum distance of all subcodes formed by deleting circulants.*

Thus to construct a high rate code with the same minimum distance as a lower rate code requires that all subcodes have at least the desired minimum distance. This is a necessary condition.

Theorem 2.10 *There exists, for all m and p, a rate $(p-1)/p$ systematic QC code formed from $m \times m$ circulants with $d_{min} \geq 2$.*

Proof Consider the structure of the code given by (2.3). In order for d_{min} to be 1, a circulant $c_j(x)$ must have weight 0, which is impossible by the definition. □

Furthermore, if the code is composed of distinct circulants with cycle m, the minimum distance must be at least three.

Theorem 2.11 *A rate $(p-1)/p$ systematic QC code has $d_{min} \geq 3$ iff none of the circulants $c_j(x)$ satisfies $(x^n - 1)c_j(x) = 0 \ (mod\ x^m - 1), \forall 1 < n < m$, and $c_j(x) \neq 1$.*

Proof In order for the code to have $d_{min} = 2$, either $i(x) = 1$ and $p(x) = 1$, or $\mathbf{wt}[i(x)] = 2$ and $p(x) = 0$. The first case is impossible since $p(x)$ will equal 1 iff $c_j(x) = 1$. In the second case, $p(x) = 0$ indicates either that $(x^n - 1)c_j(x) = 0$, which is also a violation, or $x^a c_j(x) + x^b c_h(x) = 0$, but then the circulants are not distinct, since $c_j(x)$ is a cyclic shift of $c_h(x)$. Since only distinct circulants are considered, the weight of the code must be at least three.□

From the above Theorems, the construction of QC codes with $d_{min} \leq 3$ is trivial. Therefore we enumerate only those codes which have $d_{min} \geq 4$. In order for a rate $(p-1)/p$ QC code to have $d_{min} \geq 4$ the following must be satisfied:

1. All rate 1/2 subcodes have $d_{min} \geq 4$.
2. $x^a c_j(x) + x^b c_k(x) > 1$, $0 \leq a, b < m$ and $0 < j, k < p$.
3. $(x^a + x^b)c_j(x) + x^d c_k(x) > 0$, $0 \leq a, b, d < m$, $a \neq b$, and $0 < j, k < p$.
4. $x^a c_j(x) + x^b c_k(x) + x^d c_l(x) > 0$, $0 \leq a, b, c < m$ and $0 < j, k, l < p$.

Conditions 1, 2 and 3 result in the requirement that all rate 2/3 subcodes have $d_{min} \geq 4$. In turn, all four conditions are equivalent to requiring that all rate 3/4 subcodes have $d_{min} \geq 4$. Theorem 2.12 generalizes this result.

Theorem 2.12 *A rate $(p-1)/p$ systematic QC code has $d_{min} \geq d$ iff all rate $(k-1)/k$ subcodes have $d_{min} \geq d$, for $2 \leq k \leq d$.*

Proof Since only systematic codes are considered, it is necessary to examine $i(x)$ only up to weight d, because if the weight of $i(x)$ is $\geq d$, the codeword will have weight $\geq d$. Thus only those subcodes containing up to d circulants need be considered. If all these subcodes have $d_{min} \geq d$, then the larger code will have $d_{min} \geq d$ by Corollary 2.9.□

From this Theorem it is clear that there must exist at least $\begin{pmatrix} p-1 \\ k-1 \end{pmatrix}$ rate $(k-1)/k$ QC codes, $2 \leq k \leq d$, with $d_{min} \geq d$ in order for the rate $(p-1)/p$ code to exist. However this is only a necessary condition. Investigation has shown that in many cases this condition is met but no code exists.

Table 2.29 presents the best rate $(p-1)/p$ QC codes for m up to 16, and Table 2.30 the best rate 2/3 codes for $m = 16$ to 26, extending the results of [25]. Table 2.31 gives the maximum minimum distances for these codes up to $m = 16$. Only the highest rate code for a given m and d_{min} is given since all subcodes will have at least the same d_{min}. All generator polynomials are given in octal, with the coefficient of lowest degree on the left, i.e., $713_8 = 1+x+x^2+x^5+x^7+x^8$. For conciseness, all generator polynomials for a given m are numbered and listed separately, as in Figures 2.6, 2.13, 2.15, etc. The Tables of codes list the corresponding generator polynomial numbers, instead of the polynomials themselves. For example, Table 2.19 lists 84 polynomials for $m = 12$. Table 2.20 gives the best rate $1/p, m = 12$ QC codes for $p = 3$ to 18. For each p is given the code dimensions, the minimum distance, and a list of the generator polynomial numbers for the particular QC code.

2.4 Concluding Remarks

Search methods are presented to construct good binary Quasi-Cyclic codes. Many new QC codes have been constructed, including many that are optimal or best possible QC codes. As well, Table 2.32 lists those which improve the bounds on the maximum possible minimum distance for a binary linear code given in [32].

Table 2.3: Equivalent Best Rate 1/2 Systematic QC Codes

$(2m, m)$ *QC Code*	*Generator Polynomial* $c(x)$	d_{min}	d_v	*Number of Codes*	*Number of Distinct Weight Distributions*
$(6, 3)$	3	3	3	1	1
$(8, 4)$	7	4^{d4}	4	1	1
$(10, 5)$	7	4	4	3	2
$(12, 6)$	7	4	4	5	2
$(14, 7)$	7	4	4	12	3
$(16, 8)$	27	5	5	4	1
$(18, 9)$	117	6	6	3	1
$(20, 10)$	57	6	6	17	2
$(22, 11)$	267	7	7	2	1
$(24, 12)$	573	8^{d4}	8	2	1
$(26, 13)$	653	7	7	2	1
$(28, 14)$	727	8	8	6	1
$(30, 15)$	2167	8	8	36	1
$(32, 16)$	1137	8^{d4}	8	396	9
$(34, 17)$	557	8	$8-9$	1344	12
$(36, 18)$	573	8	$8-10$	3276	79
$(38, 19)$	557	8	$8-10$	11684	98
$(40, 20)$	5723	9	$9-10$	120	13
$(42, 21)$	14573	10	$10-11$	138	3
$(44, 22)$	11753	10	$10-12$	1420	9
$(46, 23)$	667657	11	$11-12$	22	1
$(48, 24)$	1666577	12^{d4}	12	8	1

Notes: d_v the bounds given in [32]
n^{dz} the given code has weights divisible by z

Table 2.4: Best Rate 1/2 QC Codes for $m = 25$ to 31

$(2m, m)$ *QC* *Code*	*Generator* *Polynomial* $c(x)$	d_{min}	d_v
$(50, 25)$	11667	10	$10 - 12$
$(52, 26)$	11667	10	$10 - 13$
$(54, 27)$	62573	11	$11 - 14$
$(56, 28)$	546173	12	$12 - 14$
$(58, 29)$	275067	12	$12 - 14$
$(60, 30)$	255707	12	$12 - 15$
$(62, 31)$	131675	12	$12 - 16$

Table 2.5: Generator Polynomials for $m = 3$ to 8

Polynomial Number	m					
	3	4	5	6	7	8
1	1	1	1	1	1	1
2	3	3	3	3	3	7
3	7	5	5	5	5	11
4		7	7	7	7	13
5		17	13	11	11	15
6			17	13	13	17
7			37	15	15	21
8				17	17	23
9				25	23	25
10				27	25	27
11				33	27	31
12				37	33	33
13				77	35	35
14					37	37
15					53	45
16					57	47
17					67	53
18					77	57
19						65
20						67
21						73
22						75
23						77
24						127
25						133
26						137
27						157
28						177

Table 2.6: Rate $1/p,\ m = 3$ Quasi-Cyclic Codes

Code	d_{min}	Generators
(9,3)	4	1,2,3
(12,3)	6	1,2,2,3
(15,3)	8	1,1,2,2,3
(18,3)	10	1,1,1,2,2,3
(21,3)	12	1,1,1,2,2,2,3
(24,3)	13	1,1,1,1,2,2,2,3
(27,3)	15	1,1,1,1,2,2,2,2,3
(30,3)	16	1,1,1,1,2,2,2,2,3,3
(33,3)	18	1,1,1,1,2,2,2,2,2,3,3
(36,3)	20	1,1,1,1,1,2,2,2,2,2,3,3
(39,3)	22	1,1,1,1,1,1,2,2,2,2,2,3,3
(42,3)	24	1,1,1,1,1,1,2,2,2,2,2,2,3,3
(45,3)	25	1,1,1,1,1,1,1,2,2,2,2,2,2,3,3
(48,3)	27	1,1,1,1,1,1,1,2,2,2,2,2,2,2,3,3
(51,3)	28	1,1,1,1,1,1,1,2,2,2,2,2,2,2,3,3,3
(54,3)	30	1,1,1,1,1,1,1,2,2,2,2,2,2,2,2,3,3,3

Table 2.7: Rate $1/p,\ m = 4$ Quasi-Cyclic Codes

Code	d_{min}	Generators
(12,4)	6	1,2,4
(16,4)	8	1,2,3,4
(20,4)	10	1,1,2,4,4
(24,4)	12	1,1,2,2,4,4
(28,4)	14	1,1,2,2,3,4,4
(32,4)	16	1,1,2,2,3,3,4,4
(36,4)	18	1,1,1,2,2,3,4,4,4
(40,4)	20	1,1,1,2,2,2,3,4,4,4
(44,4)	22	1,1,1,2,2,2,3,3,4,4,4
(48,4)	24	1,1,1,2,2,2,2,3,3,4,4,4
(52,4)	26	1,1,1,1,2,2,2,3,3,4,4,4,4
(56,4)	28	1,1,1,1,2,2,2,2,3,3,4,4,4,4
(60,4)	32	1,1,1,1,2,2,2,2,3,3,4,4,4,4,5
(64,4)	33	1,1,1,1,1,2,2,2,2,3,3,4,4,4,4,5
(68,4)	36	1,1,1,1,1,2,2,2,2,3,3,4,4,4,4,4,5
(72,4)	38	1,1,1,1,1,2,2,2,2,2,3,3,4,4,4,4,4,5

Table 2.8: Rate $1/p$, $m = 5$ Quasi-Cyclic Codes

Code	d_{min}	Generators
(15,5)	7	1,4,5
(20,5)	9	1,3,4,5
(25,5)	12	1,3,4,5,5
(30,5)	15	1,2,3,4,5,6
(35,5)	16	1,2,3,4,4,5,6
(40,5)	20	1,2,2,3,4,4,5,6
(45,5)	22	1,1,1,2,2,3,4,4,4
(50,5)	24	1,1,1,2,2,2,3,4,4,4
(55,5)	27	1,1,1,2,2,2,3,3,4,4,4
(60,5)	30	1,1,1,2,2,2,2,3,3,4,4,4
(65,5)	32	1,1,1,1,2,2,2,3,3,4,4,4,4
(70,5)	35	1,1,1,1,2,2,2,2,3,3,4,4,4,4
(75,5)	37	1,1,1,1,2,2,2,2,3,3,4,4,4,4,5
(80,5)	40	1,1,1,1,1,2,2,2,2,3,3,4,4,4,4,5
(85,5)	42	1,1,1,1,1,2,2,2,2,3,3,4,4,4,4,4,5
(90,5)	45	1,1,1,1,1,2,2,2,2,2,3,3,4,4,4,4,4,5

Table 2.9: Rate $1/p$, $m = 6$ Quasi-Cyclic Codes

Code	d_{min}	Generators
(18,6)	8	1,4,10
(24,6)	10	1,3,4,10
(30,6)	14	1,4,5,6,12
(36,6)	16	1,3,4,5,6,12
(42,6)	20	1,2,4,6,6,7,12
(48,6)	24	1,2,3,6,7,8,10,12
(54,6)	26	1,2,3,4,6,7,8,9,12
(60,6)	29	1,2,3,4,5,6,7,8,10,12
(66,6)	32	1,2,3,4,5,6,7,8,9,10,12
(72,6)	34	1,1,2,3,4,4,2,6,7,8,9,10
(78,6)	38	1,2,3,4,5,6,6,7,7,9,10,10,12
(84,6)	40	1,2,2,3,4,5,6,6,7,8,9,10,11,12
(90,6)	44	1,1,2,2,3,4,5,6,7,8,9,10,11,12,12
(96,6)	48	1,1,2,2,3,4,5,6,7,8,8,9,10,11,12,12
(102,6)	50	1,1,2,2,3,4,5,6,6,7,7,8,9,10,10,12,12
(108,6)	53	1,1,2,2,3,3,4,4,5,6,6,7,8,10,10,11,12,12

Table 2.10: Rate $1/p$, $m = 7$ Quasi-Cyclic Codes

Code	d_{min}	Generators
(21,7)	8	1,4,17
(28,7)	12	1,4,9,17
(35,7)	16	1,4,9,10,18
(42,7)	19	1,4,6,9,10,18
(49,7)	22	1,4,6,7,9,10,18
(56,7)	26	1,4,6,7,9,10,14,17
(63,7)	31	1,4,6,7,9,10,14,16,17
(70,7)	33	1,4,5,6,7,9,10,14,16,17
(77,7)	36	1,4,5,6,7,8,9,10,11,12,18
(84,7)	40	1,4,5,6,7,8,9,10,11,12,13,18
(91,7)	44	1,3,4,5,6,7,9,10,11,13,16,17,18
(98,7)	48	1,2,4,5,6,7,8,10,11,12,13,14,16,17
(105,7)	52	1,2,3,4,5,6,8,9,10,12,14,15,16,17,18
(112,7)	56	1,2,3,4,5,7,8,9,10,11,12,14,15,16,17,18
(119,7)	59	1,2,3,4,5,6,7,8,9,10,12,13,14,15,16,17,18
(126,7)	63	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18

Table 2.11: Rate $1/p$, $m = 8$ Quasi-Cyclic Codes

Code	d_{min}	Generators
(24,8)	8	1,4,10
(32,8)	12	1,2,3,26
(40,8)	16	1,2,4,15,22
(48,8)	20	1,2,3,4,17,28
(56,8)	24	1,3,4,5,6,17,28
(64,8)	28	1,4,5,6,8,10,15,28
(72,8)	32	1,2,3,4,5,8,18,24,28
(80,8)	37	1,2,5,10,12,15,17,20,21,22
(88,8)	40	1,2,4,5,7,8,9,18,19,23,28
(96,8)	46	1,2,5,8,9,11,15,16,18,23,26,27
(104,8)	48	1,4,5,6,7,8,9,10,11,13,18,26,28
(112,8)	54	1,4,5,8,9,11,14,15,20,21,22,24,25,28
(120,8)	57	1,4,5,8,9,11,14,15,18,20,21,22,24,25,28
(128,8)	64	1,2,4,5,8,9,11,14,15,18,20,21,22,24,25,28
(136,8)	66	1,2,4,5,8,9,11,14,15,18,20,21,22,24,25,27,28
(144,8)	70	1,2,4,5,8,9,11,14,15,18,20,21,22,24,25,26,27,28

Table 2.12: Generator Polynomials for $m = 9$

1	1	11	25	21	53	31	115	41	155	51	273
2	3	12	27	22	57	32	117	42	157	52	277
3	5	13	31	23	63	33	123	43	165	53	337
4	7	14	33	24	65	34	125	44	167	54	357
5	11	15	35	25	67	35	127	45	173	55	377
6	13	16	37	26	71	36	133	46	175		
7	15	17	43	27	73	37	135	47	177		
8	17	18	45	28	75	38	137	48	253		
9	21	19	47	29	77	39	147	49	257		
10	23	20	51	30	113	40	153	50	267		

Table 2.13: Rate $1/p$, $m = 9$ Quasi-Cyclic Codes

Code	d_{min}	Generators
(27,9)	10	1,18,38
(36,9)	14	1,4,18,54
(45,9)	18	1,5,15,21,54
(54,9)	23	1,4,22,24,33,46
(63,9)	28	1,2,11,22,39,46,51
(72,9)	32	1,2,8,14,30,37,43,52
(81,9)	36	1,6,7,11,25,26,27,36,47
(90,9)	40	1,6,8,10,13,18,40,45,49,55
(99,9)	46	1,7,15,17,22,32,34,35,39,48,51
(108,9)	50	1,4,6,11,12,13,29,30,41,43,47,51
(117,9)	55	1,8,11,24,27,28,30,39,41,47,50,51,55
(126,9)	59	1,2,4,7,10,11,14,15,20,21,22,24,25,28
(135,9)	64	1,8,11,12,19,27,28,32,39,40,47,48,50,51,55
(144,9)	68	1,3,4,8,13,16,22,24,30,32,36,37,41,43,48,53
(153,9)	72	1,2,3,8,9,11,17,19,22,30,37,40,45,46,51,53,55
(162,9)	76	1,4,6,8,11,19,23,24,27,28,30,32,39,40,41,47,51,54

Table 2.14: Generator Polynomials for $m = 10$

1	1	14	45	27	105	40	155	53	235	66	333	79	567
2	7	15	47	28	107	41	157	54	247	67	335	80	573
3	11	16	51	29	111	42	163	55	253	68	337	81	577
4	13	17	53	30	113	43	165	56	255	69	355	82	667
5	15	18	55	31	115	44	167	57	263	70	357	83	677
6	17	19	57	32	123	45	171	58	265	71	365	84	737
7	23	20	61	33	125	46	173	59	267	72	367		
8	25	21	63	34	127	47	175	60	273	73	373		
9	27	22	65	35	131	48	177	61	275	74	375		
10	33	23	67	36	133	49	223	62	277	75	377		
11	35	24	71	37	135	50	225	63	317	76	527		
12	37	25	75	38	145	51	227	64	325	77	537		
13	43	26	77	39	147	52	233	65	327	78	557		

Table 2.15: Rate $1/p$, $m = 10$ Quasi-Cyclic Codes

Code	d_{min}	Generators
(30,10)	10	1,4,41
(40,10)	16	1,2,58,68
(50,10)	20	1,2,31,32,83
(60,10)	24	1,2,4,31,34,83
(70,10)	30	1,9,17,18,23,70,74
(80,10)	34	1,2,4,23,54,58,68,74
(90,10)	40	1,6,7,17,18,23,64,70,74
(100,10)	44	1,36,39,55,61,71,72,74,72,84
(110,10)	49	1,4,14,17,22,24,26,53,54,73,82
(120,10)	54	1,2,19,29,47,48,57,59,67,77,80,84
(130,10)	60	1,15,21,31,38,44,51,52,56,63,65,78,82
(140,10)	64	1,7,8,10,12,16,24,37,38,58,59,69,74,84
(150,10)	68	1,2,4,5,36,39,40,49,59,61,62,71,74,72,84
(160,10)	74	1,3,9,17,20,22,26,35,38,42,52,53,74,75,77,79
(170,10)	80	1,3,4,9,13,19,31,32,39,41,46,50,54,66,75,79,82
(180,10)	84	1,3,9,12,13,19,20,32,33,35,39,44,48,50,71,74,79,82

Table 2.16: Generator Polynomials for $m = 11$

1	1	11	55	21	153	31	257	41	363	51	535	61	765
2	3	12	65	22	165	32	265	42	365	52	555	62	777
3	7	13	67	23	167	33	267	43	447	53	557	63	1253
4	13	14	71	24	171	34	313	44	457	54	563	64	1277
5	15	15	77	25	177	35	315	45	467	55	575	65	1327
6	23	16	105	26	213	36	325	46	473	56	647	66	1367
7	25	17	117	27	231	37	331	47	477	57	657	67	1557
8	31	18	133	28	235	38	333	48	513	58	675	68	1577
9	47	19	135	29	247	39	347	49	517	59	753	69	1727
10	51	20	145	30	251	40	357	50	533	60	757	70	1737

Table 2.17: Rate $1/p$, $m = 11$ Quasi-Cyclic Codes

Code	d_{min}	Generators
(33,11)	11	1,37,40
(44,11)	16	1,3,22,53
(55,11)	21	1,3,18,34,61
(66,11)	28	1,4,15,35,51,58
(77,11)	32	1,5,9,10,31,56,68
(88,11)	39	1,11,13,14,20,53,59,61
(99,11)	43	1,11,13,14,20,36,44,59,61
(110,11)	48	1,11,13,14,20,36,44,53,59,61
(121,11)	53	1,8,12,17,29,38,39,42,45,64,70
(132,11)	58	1,11,13,14,20,36,40,44,53,50,59,61
(143,11)	64	1,11,13,14,20,23,36,44,50,53,59,53,63
(154,11)	68	1,2,11,13,14,20,23,36,37,43,44,53,59,61
(165,11)	74	1,7,19,20,27,28,30,32,46,48,54,55,60,65,67
(176,11)	80	1,2,7,11,13,14,20,23,26,36,41,44,53,59,61,69
(187,11)	84	1,2,7,11,13,14,20,23,26,27,36,41,44,53,59,61,69
(198,11)	90	1,6,7,9,16,17,21,24,25,32,33,34,47,49,50,52,62,66

Table 2.18: Generator Polynomials for $m = 12$

1	1	14	135	27	311	40	477	53	775	66	1373	79	2553
2	15	15	137	28	313	41	511	54	1115	67	1465	80	2667
3	17	16	145	29	331	42	517	55	1117	68	1477	81	2737
4	23	17	157	30	337	43	531	56	1145	69	1537	82	2773
5	25	18	165	31	345	44	535	57	1167	70	1553	83	3357
6	37	19	171	32	347	45	575	58	1175	71	1557	84	3677
7	41	20	223	33	361	46	577	59	1177	72	1565		
8	47	21	225	34	375	47	663	60	1225	73	1573		
9	73	22	235	35	427	48	717	61	1237	74	1637		
10	75	23	245	36	435	49	727	62	1247	75	1675		
11	105	24	251	37	445	50	737	63	1317	76	1753		
12	107	25	263	38	453	51	753	64	1323	77	2533		
13	123	26	275	39	471	52	767	65	1347	78	2537		

Table 2.19: Rate $1/p$, $m = 12$ Quasi-Cyclic Codes

Code	d_{min}	Generators
(36,12)	12	1,52,82
(48,12)	17	1,43,74,82
(60,12)	24	1,10,15,29,72
(72,12)	28	1,14,21,43,74,82
(84,12)	34	1,21,35,43,55,55,82
(96,12)	40	1,21,26,35,43,55,66,82
(108,12)	46	1,6,11,17,25,44,69,75,76
(120,12)	52	1,25,29,32,33,34,37,42,54,68
(132,12)	56	1,7,16,19,46,49,57,58,61,67,84
(144,12)	62	1,3,4,5,18,22,28,45,47,65,67,84
(156,12)	68	1,7,11,16,19,41,46,49,57,58,61,67,84
(168,12)	74	1,6,8,9,14,21,22,35,51,52,53,56,70,71
(180,12)	80	1,6,8,9,14,21,22,35,38,52,53,56,64,70,71
(192,12)	86	1,2,8,22,23,24,30,35,36,39,41,48,53,59,63,81
(204,12)	92	1,13,21,26,27,31,40,43,50,55,60,66,73,74,77,82,83
(216,12)	96	1,3,13,16,21,26,27,31,40,43,55,60,66,73,74,77,82,83

Table 2.20: Generator Polynomials for $m = 13$

1	1	14	231	27	663	40	1433	53	2347	66	3577
2	7	15	253	28	725	41	1451	54	2355	67	3675
3	23	16	255	29	763	42	1535	55	2453	68	3727
4	31	17	273	30	771	43	1573	56	2657	69	3733
5	37	18	313	31	1055	44	1575	57	2757	70	3753
6	43	19	315	32	1057	45	1631	58	2767	71	3757
7	65	20	321	33	1063	46	1667	59	3165	72	5257
8	67	21	335	34	1071	47	1731	60	3265	73	5277
9	73	22	447	35	1077	48	1755	61	3277	74	5327
10	115	23	465	36	1177	49	2227	62	3357	75	5673
11	125	24	517	37	1223	50	2315	63	3477	76	6677
12	153	25	525	38	1271	51	2333	64	3567		
13	207	26	537	39	1431	52	2337	65	3575		

Table 2.21: Rate $1/p$, $m = 13$ Quasi-Cyclic Codes

Code	d_{min}	Generators
(39,13)	12	1,3,70
(52,13)	19	1,3,24,75
(65,13)	25	1,58,61,65,70
(78,13)	30	1,12,58,61,65,70
(91,13)	36	1,6,7,58,61,65,70
(104,13)	43	1,31,46,58,61,63,65,70
(117,13)	48	1,6,7,9,42,58,61,65,70
(130,13)	54	1,30,31,46,58,61,63,65,70,74
(143,13)	60	1,17,30,31,46,58,61,63,65,70,74
(156,13)	66	1,15,30,31,46,53,58,61,63,65,70,74
(169,13)	72	1,15,30,31,46,52,53,58,61,63,65,70,74
(182,13)	78	1,10,15,30,31,46,52,53,58,61,63,65,70,74
(195,13)	84	1,15,30,31,36,46,52,53,58,61,63,65,70,74,76
(208,13)	92	1,14,18,19,20,21,25,29,40,44,49,54,57,68,57,72
(221,13)	98	1,11,13,22,23,28,37,39,47,49,50,55,56,59,64,67,73
(234,13)	104	1,8,15,30,31,36,45,46,52,53,58,61,63,65,69,70,72,76

Table 2.22: Generator Polynomials for $m = 14$

1	1	14	241	27	753	40	1717	53	3135	66	5517	79	16777
2	17	15	257	28	1071	41	1747	54	3171	67	5657		
3	21	16	273	29	1105	42	1753	55	3323	68	5753		
4	23	17	323	30	1107	43	1777	56	3345	69	6747		
5	27	18	355	31	1161	44	2123	57	3725	70	6767		
6	33	19	443	32	1237	45	2317	58	4553	71	7165		
7	75	20	455	33	1243	46	2367	59	4557	72	7365		
8	137	21	513	34	1327	47	2373	60	4733	73	7527		
9	143	22	523	35	1373	48	2507	61	4755	74	10231		
10	153	23	615	36	1375	49	2615	62	4777	75	12105		
11	217	24	717	37	1465	50	2663	63	5173	76	12667		
12	227	25	725	38	1545	51	2667	64	5355	77	12733		
13	237	26	727	39	1667	52	3123	65	5457	78	13573		

Table 2.23: Rate $1/p$, $m = 14$ Quasi-Cyclic Codes

Code	d_{min}	Generators
(42,14)	13	1,26,58
(56,14)	20	1,23,26,58
(70,14)	26	1,4,39,47,64
(84,14)	32	1,4,28,39,46,64
(98,14)	38	1,4,21,28,39,46,64
(112,14)	44	1,2,13,23,26,27,56,58
(126,14)	50	1,2,13,17,23,27,56,57,58
(140,14)	57	1,3,20,22,42,49,50,62,69,78
(154,14)	64	1,14,18,29,43,53,55,63,67,73,79
(168,14)	70	1,5,10,24,32,37,44,60,65,66,70,76
(182,14)	76	1,2,13,15,16,17,23,25,27,56,57,58,71
(196,14)	84	1,5,6,19,34,36,40,48,51,54,61,68,72,77
(210,14)	88	1,2,7,13,15,16,17,23,25,27,38,56,57,58,71
(224,14)	96	1,7,11,13,15,16,17,23,25,27,35,38,56,57,58,71
(238,14)	102	1,7,11,13,15,16,17,23,25,27,35,38,52,56,57,58,71
(252,14)	108	1,7,8,11,13,15,16,17,23,25,27,35,38,52,56,57,58,71

Table 2.24: Generator Polynomials for $m = 15$

1	1	14	537	27	2167	40	4317	53	7275	66	15347
2	25	15	635	28	2243	41	4531	54	7373	67	15773
3	35	16	663	29	2431	42	4571	55	7573	68	16753
4	53	17	677	30	2443	43	4643	56	7757	69	17177
5	121	18	731	31	2475	44	5257	57	11353	70	17573
6	125	19	1027	32	2723	45	5727	58	12265	71	17767
7	247	20	1123	33	2765	46	6233	59	12357	72	33577
8	255	21	1137	34	3045	47	6273	60	12373		
9	273	22	1173	35	3157	48	6507	61	12455		
10	353	23	1343	36	3463	49	6623	62	13637		
11	377	24	1435	37	3513	50	6631	63	14653		
12	433	25	1733	38	3625	51	6755	64	14737		
13	477	26	2135	39	3665	52	7137	65	14767		

Table 2.25: Rate $1/p$, $m = 15$ Quasi-Cyclic Codes

Code	d_{min}	Generators
(45,15)	14	1,10,27
(60,15)	20	1,10,15,27
(75,15)	26	1,10,15,18,27
(90,15)	34	1,41,44,56,60,61
(105,15)	40	1,17,19,30,46,53,69
(120,15)	48	1,11,23,26,27,29,32,33
(135,15)	54	1,23,26,27,29,32,33,40,67
(150,15)	60	1,8,10,14,15,18,27,37,42,65
(165,15)	68	1,2,7,12,21,28,51,52,57,63,71
(180,15)	74	1,3,8,14,15,18,27,37,38,42,65,70
(195,15)	80	1,3,6,8,14,15,18,27,37,38,42,65,70
(210,15)	88	1,3,6,8,14,15,18,27,37,38,42,54,65,70
(225,15)	94	1,3,6,8,14,15,16,18,27,37,38,42,54,65,70
(240,15)	102	1,11,22,23,24,26,27,29,32,33,36,40,48,59,67,68
(255,15)	108	1,3,5,6,8,14,15,16,18,27,31,37,38,42,54,65,70
(270,15)	116	1,11,22,23,24,25,26,27,29,32,33,36,40,47,48,59,67,68

Table 2.26: Generator Polynomials for $m = 16$

1	1	11	14447
2	13	12	25315
3	357	13	31667
4	513	14	32375
5	1705	15	32555
6	2747	16	33755
7	5271	17	37773
8	6531	18	55773
9	7167		
10	13557		

Table 2.27: Rate $1/p$, $m = 16$ Quasi-Cyclic Codes

Code	d_{min}	Generators
(48,16)	14	1,8,14
(64,16)	21	1,6,13,16
(80,16)	28	1,3,8,11,14
(96,16)	34	1,6,7,13,14,15
(112,16)	42	1,3,5,6,7,14,16
(128,16)	50	1,3,4,8,9,11,16,17
(144,16)	57	1,3,4,7,8,9,11,16,17
(160,16)	64	1,6,7,8,10,11,13,14,16,17
(176,16)	72	1,3,5,6,7,8,9,11,12,16,17
(192,16)	80	1,3,4,5,7,8,10,11,12,13,16,17
(208,16)	86	1,3,4,6,7,9,10,11,12,13,14,16,17
(224,16)	94	1,3,4,6,7,8,9,10,11,12,13,14,17,18
(240,16)	103	1,3,4,5,6,7,8,9,10,11,13,14,16,17,18
(256,16)	113	1,3,4,5,6,7,8,9,10,11,12,13,14,16,17,18
(272,16)	118	1,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18
(288,16)	125	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18

Table 2.28: Maximum Minimum Distances for (pm, m) Systematic QC Codes

m \ p	3	4	5	6	7	8	9	10	11	12	13	14	15
3	$+4^{o}$	$+6^{o}$	$+8^{o}$	$+10^{o}$	$+12^{od12}$	$+13^{o}$	$+15^{o}$	$+16^{o}$	$+18^{o}$	$+20^{o}$	$+22^{o}$	$+24^{od24}$	$+25^{o}$
4	$+6^{o}$	$+8^{o}$	$+10^{o}$	$+12^{od4}$	$+14^{o}$	$+16^{od4}$	$+18^{o}$	$+20^{o}$	22	$+24^{od4}$	26	28^{d4}	$+32^{od32}$
5	$+7^{o}$	$+9^{o}$	$+12^{d4}$	$+15^{o}$	$+16^{o}$	$+20^{o}$	$+22^{o}$	$+24^{od4}$	27	$+30^{o}$	$+32^{o}$	$+35^{o}$	37
6	$+8^{od4}$	$+10^{o}$	$+14^{o}$	$+16^{o}$	$+20^{od4}$	$+24^{od8}$	$+26^{o}$	29	$+32^{od4}$	34	$+38^{o}$	40	$+44^{o}$
7	$+8^{o}$	$+12^{od4}$	$+16^{o}$	$+19^{o}$	22^{o}	$+26^{o}$	$+31^{o}$	$+33^{o}$	$+36^{o}$	$+40^{o}$	$+44^{o}$	$+48^{o}$	$+52^{o}$
8	-8^{o}	-12^{o}	-16^{o}	20^{o}	-24^{o}	28^{o}	-32^{od4}	-37^{o}	-40^{o}	-46^{o}	-48^{o}	$+54^{o}$	-57^{o}
9	$+10^{o}$	-14^{o}	-18^{o}	-23^{o}	$+28^{o}$	$+32^{od4}$	-36^{o}	-40	$e46$	$e50$	$e55$	$e59$	64
10	10^{o}	$+16^{o}$	-20^{o}	24^{o}	30	-34	-40^{d4}	$e44$	$e49$	-54	60	64	68
11	11^{o}	-16^{o1}	-21^{o}	$+28^{od4}$	-32^{1}	-39^{1}	$e43$	$e48$	-53	58	64	68	74
12	$+12^{o}$	-17^{o}	$+24^{o}$	28^{d4}	$e34$	-40	$e46$	-52^{d4}	56	62	68	74	80^{d4}
13	-12^{o}	-19^{o}	-25^{2}	30	-36^{2}	$e43$	-48	54	60	66	72	78	84
14	-13^{o1}	-20^{o}	-26	-32	-38	-44	50	57	64^{d4}	70	76	84	89
15	-14^{o}	20	26	$e34$	-40	-48	54	60	68	74	80	88	94
16	-14^{o1}	21^{1}	-28^{1}	34^{1}	42^{1}	50^{1}	57^{1}	64^{1}	72^{1}	80^{1d4}	86^{1}	94^{1}	103^{1}

Notes: n^1 a power residue subcode.
(Since $2^5 - 1$ and $2^7 - 1$ are prime, all possible codes are included in the PR subcode
n^2 a cyclic code decomposition subcode [21].
n^o is a best Quasi-Cyclic code.
$+$ meets the upper bound in [32].
$-$ meets the lower bound in [32].
e exceeds the lower bound in [32].
n^{dz} the given code has weights divisible by z.

Table 2.29: Rate $(p-1)/p$ Quasi-Cyclic Codes

Code	m	p	d_{min}	Generators
(15,10)	5	3	4	4,5
(30,24)	6	5	4	4,6,7,12
(63,54)	7	9	4	4,6,7,9,10,14,16,17
(128,120)	8	16	4	2,4,5,8,9,11,14,15,18,20,21,22,24,25,28
(162,153)	9	18	4	4,6,7,10,11,13,16,17,18,22,25,27,28,32,35,36,37
(30,20)	10	3	5	9,64
(180,170)	10	18	4	2,4,5,7,8,12,13,14,16,19,20,23,25,27,29,34,36
(33,22)	11	3	6	19,47
(44,33)	11	4	5	19,47,66
(198,187)	11	18	4	3,4,5,6,7,10,13,16,17,18,19,21,22,24,25,28,32
(36,24)	12	3	6	10,35
(60,48)	12	5	5	46,74,80,82
(216,204)	12	18	4	3,4,5,14,16,22,25,38,50,51,53,55,62,68,71,78,79
(65,52)	13	5	6	58,61,65,70
(234,221)	13	18	4	2,3,4,5,6,8,9,12,15,16,26,30,31,33,34,35,41
(70,56)	14	5	6	7,12,33,45
(98,84)	14	7	5	7,9,12,33,45,59
(252,238)	14	18	4	2,4,13,15,18,26,28,29,33,36,37,38,41,51,53,57,64
(90,75)	15	6	6	19,30,46,53,69
(150,135)	15	10	5	9,34,35,45,49,62,64,69,72
(270,255)	15	18	4	3,5,8,9,10,11,13,14,16,18,19,20,22,23,25,31,33

Table 2.30: Rate 2/3 Quasi-Cyclic Codes

$(3m, 2m)$ *QC* *Code*	*Generator Polynomials* $c_1(x)$	$c_2(x)$	d_{min}	d_v
$(48, 32)$	57	3733	6	$6-8$
$(51, 34)$	1537	6365	6	$7-8$
$(54, 36)$	355	147527	7	8
$(57, 38)$	2655	317537	8	$8-9$
$(60, 40)$	6323	2757	8	$8-10$
$(63, 42)$	50367	52635	8	$8-10$
$(66, 44)$	6144232	4412177	8	$8-10$
$(69, 46)$	6323	2757	8	$8-10$
$(72, 48)$	57361424	63235074	8	$9-11$
$(75, 50)$	142422547	131657623	8	$8-12$
$(78, 52)$	54557347	240517035	8	$8-12$

Table 2.31: Maximum Minimum Distances for $(pm, (p-1)m)$ Systematic QC Codes

	p																
m	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
3	$+3^o$	$+2^o$	$+2^o$	$+2^o$	$+2^o$	$+2^o$	$+2^o$	$+2^o$	$+2^o$	$+2^o$	$+2^o$	$+2^o$	$+2^o$	$+2^o$	$+2^o$	$+2^o$	$+2^o$
4	$+4^o$	$+3^o$	$+2^o$	$+2^o$	$+2^o$	$+2^o$	$+2^o$	$+2^o$	$+2^o$	$+2^o$	$+2^o$	$+2^o$	$+2^o$	$+2^o$	$+2^o$	$+2^o$	$+2^o$
5	$+4^o$	$+4^o$	$+3^o$	$+3^o$	$+3^o$	$+2^o$	$+2^o$	$+2^o$	$+2^o$	$+2^o$	$+2^o$	$+2^o$	$+2^o$	$+2^o$	$+2^o$	$+2^o$	$+2^o$
6	$+4^o$	$+4^o$	$+4^o$	$+4^o$	$+3^o$	$+3^o$	$+3^o$	$+3^o$	2^o	$+2^o$	$+2^o$	$+2^o$	$+2^o$	$+2^o$	$+2^o$	$+2^o$	$+2^o$
7	$+4^o$	$+4^o$	$+4^o$	$+4^o$	$+4^o$	$+4^o$	$+4^o$	$+4^o$	$+3^o$	$+3^o$	$+3^o$	$+3^o$	$+3^o$	$+3^o$	$+3^o$	$+3^o$	$+3^o$
8	$+5^o$	$+4^o$	$+4^o$	$+4^o$	$+4^o$	$+4^o$	$+4^o$	$+4^o$	$+4^o$	$+4^o$	$+4^o$	$+4^o$	$+4^o$	$+4^o$	4^o	3	3^o
9	$+6^o$	-4^{o1}	$+4^o$	$+4^o$	$+4^o$	$+4^o$	$+4^o$	$+4^o$	$+4^o$	$+4^o$	$+4^o$	$+4^o$	$+4^o$	4^o	4^o	4^o	4^o
10	$+6^o$	$+5^o$	-4^o	$+4^o$	$+4^o$	$+4^o$	$+4^o$	$+4^o$	$+4^o$	$+4^o$	4^o	4^o	4^o	4^o	4^o	4^o	4^o
11	$+7^{o1}$	$+6^o$	$e+5^o$	-4^{o1}	$+4^{o1}$	$+4^{o1}$	$+4^{o1}$	$+4^o$	$+4^o$	$+4^o$	4^o	4^o	4^o	4^o	4^o	4^o	4^o
12	$+8^o$	$+6^o$	-5^o	-5^o	-4^o	-4^o	$+4^o$	$+4^o$	$+4^o$	4^o	4^o	4^o	4^o	4^o	4^o	4^o	4^o
13	$+7^o$	$+6^o$	$+6^o$	$+6^{o2}$	-4	-4	-4	-4	4^o	4^o	4^o	4^o	4^o	4^o	4^o	4^o	4^o
14	$+8^o$	-6^{o1}	$+6^o$	$+6^o$	-5	-5	4	4	4	4	4	4^o	4^o	4^o	4^o	4^o	4^o
15	$+8^o$	-6^{o1}	$+6^{o1}$	$+6^{o1}$	$+6^o$	5^1	5^1	5^1	5^1	4	4	4	4	4	4	4	4
16	$+8^{o1}$	-6^{o1}	-6^{o1}	$+6^{o1}$	$+6^{o1}$	$+6^{o1}$	6^{o1}	6^{o1}	5^1	5^1	5^1	5^1	5^1	5^1	5^1	4	4

Notes: n^1 equals best power residue subcode.
Since 2^5-1 and 2^7-1 are prime, all possible codes are included in the PR subcodes.
n^2 a cyclic code decomposition subcode [21].
n^o is a best Quasi-Cyclic code.
$+$ meets the upper bound in [32].
$-$ meets the lower bound in [32].
e exceeds the lower bound in [32].

Table 2.32: Quasi-Cyclic Codes Which Improve the Bounds on the Maximum Possible Minimum Distance for a Binary Linear Code

QC code	d_{min}	d_v
(44,33)	5	4-5
(99,9)	46	45-47
(108,9)	50	48-51
(117,9)	55	53-56
(126,9)	59	58-60
(100,10)	44	43-47
(110,10)	49	48-52
(99,11)	43	41-46
(110,11)	49	47-50
(84,12)	34	33-37
(108,12)	46	45-48
(104,13)	43	41-47
(90,15)	34	33-38

Chapter 3

The Binary Power Residue Codes and Related Quasi-Cyclic Codes

3.1 Introduction

It is known that the s-th power residue codes are good codes [34]. Chen et. al. [17] have shown that the cyclic s-th power residue (PR) codes with the first digit deleted are equivalent to rate $1/s$ quasi-cyclic (QC) codes. Using this connection, the weight distribution of PR codes can be found by computing the weight distribution of the equivalent QC code. It turns out that subcodes constructed from a subset of the generator polynomials of these QC codes are also good codes. These polynomials can be used to reduce the search time for good QC codes.

The next Section presents the construction method, and this is followed by an example using the $(31, 5)$ sixth PR code. A second example, the $(257, 16)$ 16th PR code, gives the subcode construction method, and shows that these codes are good.

3.2 Code Construction

Let m be the order of 2 mod n, n a prime. Then if m divides $(n-1)/s$, i.e., $n = ems + 1$, a cyclic (n, em), s-th power residue code exists, as does a rate $1/s$, $(n-1, em)$ QC code formed of $m \times m$ circulant matrices. The details of constructing these codes can be found in [17], where the Normal Basis Theorem is used to convert a PR code, with one digit deleted, into a QC code. The roots of a primitive polynomial with linearly independent roots are used to construct this basis. Once the normal basis is found, the generator matrices can be constructed and the weight distributions found. The generator matrix of the related rate 1/s QC code, for $e = 1$, is given by

$$G = [I_m, C_1, C_2, C_3, ..., C_{s-1}], \tag{3.1}$$

where I_m is an $m \times m$ identity matrix and the C_i are $m \times m$ circulant matrices, over $GF(2)$. The representation for G given here is in systematic form. Although in general these rate $1/s$ QC codes are not in this form, in all codes examined at least one circulant matrix was invertible, allowing a transformation to a systematic code. Using MacWilliam's identities we can find the weight distribution of the rate $(s-1)/s$ dual codes. Table 3.1 gives the minimum distances of the binary power residue codes, their duals and related QC codes up to $m = 32$ and $n = 10000$. From this Table it can be seen that all listed PR codes have an even minimum distance, $d_{min} = d$. The related QC codes have an odd $d_{min} = d - 1$. However, both dual codes have the same minimum distance.

3.3 The Maximum Length Sequence Codes

The special case (2) in [17] states that when $n = 2^m - 1$ is prime, the s-th power residue code is a binary maximum length sequence code. With one digit deleted, it is equivalent to a rate $\frac{m}{2^m-2} = 1/s$ QC code. It is well known

Table 3.1: A Table of Binary Power Residue Codes, their Duals and Related Quasi-Cyclic Codes

PR code	d_{min}	dual code	d_{min}	m	rate	QC code	d_{min}	dual code	d_{min}
$(7,3)^{QMo}$	4	$(7,4)^{o}$	3	3	1/2	$(6,3)^{o}$	3		
$(17,8)^{Qo}$	6	$(17,9)^{o}$	5	8	1/2	$(16,8)^{o}$	5		
$(23,11)^{Qo}$	8	$(23,12)^{o}$	7	11	1/2	$(22,11)^{o}$	7		
$(31,5)^{Mo}$	16	$(31,26)^{o}$	3	5	1/6	$(30,5)^{o}$	15	$(30,25)^{o}$	3
(31,10)	10	$(31,21)^{o}$	5	5	1/3	(30,10)	9	$(30,20)^{o}$	5
$(31,15)^{o}$	8	(31,16)	7	5	1/2	(30,15)	7		
$(41,20)^{Qo}$	10	$(41,21)^{o}$	9	20	1/2	$(40,20)^{o}$	9		
$(43,14)^{o}$	14	$(43,29)^{o}$	6	14	1/3	$(42,14)^{o}$	13	$(42,28)^{o}$	6
$(47,23)^{Qo}$	12	$(47,24)^{o}$	11	23	1/2	$(46,23)^{o}$	11		
(73,9)	28	(73,64)	3	9	1/8	(72,9)	27	(72,63)	3
$(73,18)^{o}$	24	$(73,55)^{o}$	6	9	1/4	(72,18)	23	$(72,54)^{o}$	6
$(89,11)^{o}$	40	$(89,78)^{o}$	4	11	1/8	$(88,11)^{o}$	39	$(88,77)^{o}$	4
$(89,22)^{o}$	28	(89,67)	7	11	1/4	(88,22)	27	(88,66)	7
(113,28)	28	(113,85)	8	28	1/4	(112,28)	27	(112,84)	8
$(127,7)^{M}$	64	$(127,120)^{o}$	3	7	1/18	$(126,7)^{o}$	63	$(126,119)^{o}$	3
(127,14)	54	$(127,113)^{o}$	5	7	1/9	(126,14)	53	$(126,112)^{o}$	5
(127,21)	44	(127,106)	6	7	1/6	(126,21)	43	(126,105)	6
$(151,15)^{B}$	60	$(151,136)^{B}$	5	15	1/10	(150,15)	59	(150,135)	5
(151,30)	30	(151,121)	8	15	1/5	(150,30)	29	(150,120)	8
(233,29)	88	(233,204)	7	29	1/8	(232,29)	87	(232,203)	7
(241,24)	94	(241,217)	6	24	1/10	(240,24)	93	(240,216)	6
(257,16)	114	(257,241)	5	16	1/16	(256,16)	113	(256,240)	5
(257,32)	90	(257,225)	8	16	1/8	(256,32)	89	(256,224)	8
(331,30)	124	(331,301)	6	30	1/11	(330,30)	123	(330,300)	6
(337,21)	140	(337,316)	6	21	1/16	(336,21)	139	(336,315)	6
(601,25)	256	(601,576)	5	25	1/24	(600,25)	255	(600,575)	5
(683,22)	306	(683,661)	5	22	1/31	(682,22)	305	(682,660)	5
(1103,29)	488	(1103,1074)	5	29	1/32	(1102,29)	487	(1102,1073)	5
(1801,25)	848	(1801,1776)	5	25	1/72	(1800,25)	847	(1801,1775)	5
(2089,29)	952	(2089,2060)	5	29	1/72	(2088,29)	951	(2088,2059)	5
(2731,26)	1294	(2731,2705)	5	26	1/105	(2730,26)	1293	(2730,2704)	5
$(8191,13)^{M}$	4096	(8191,8178)	3	13	1/630	(8190,13)	4095	(8190,8177)	3

Notes: n^{M} is a Maximum length sequence code
n^{Q} is a Quadratic Residue code
n^{B} given in [36]
n^{o} the code meets the bound in [32], (applicable only to codelengths up to 127)
m is the circulant size.

that when m is prime, there are exactly $\frac{2^m-2}{m}$ distinct generator polynomials for QC codes, excluding cyclic shifts and the all-zero and all-one polynomials, i.e., $T_m - 2$. Thus the QC code derived from a maximum length sequence code contains all possible generator polynomials, and so the subcodes form the complete set of distinct codes created from the polynomials of length m. In this case the construction of PR codes is of little use in finding good QC codes. For the case $m = 7$, the best subcodes are exactly those given in [26].

The dual of the maximum length sequence codes is a Hamming code. It is well known that these codes have minimum distance 3. The dual of the QC code formed with the first digit deleted also has minimum distance 3. That $d_{min} \geq 3$ is a result of Theorem 2.11, since this code contains all distinct circulants. The complete set of distinct circulants contains at least one circulant of weight 2. Thus the systematic rate 1/2 subcode formed of this circulant will have minimum distance 3. Since a code must have a minimum distance less than or equal to that of its subcodes, the weight of the QC dual code can be no more than 3. Therefore the minimum distance is exactly 3.

The following Section presents the (31,5) Maximum Length Sequence code as an example.

3.4 The (31,5) Power Residue Code

Since the order of 2 mod 31 is 5,there exists a $(31, 5)$ Sextic PR code with

$$G = \left[1 \; \beta \; \beta^2 \; \beta^3 \; ... \; \beta^{30}\right]$$

where β is a primitive 31-st root of unity.

Rearranging the columns, we have

$$G = \left[1 \; \beta^{2^0} \; \beta^{2^1} \; \beta^{2^2} \; \beta^{2^3} \; \beta^{2^4} \; ; \; (\beta^3)^{2^0} \; ... \; (\beta^3)^{2^4} \; ; \; ... \; (\beta^{15})^{2^4}\right]$$

Converting the β^k to Normal Basis form, G becomes

$$G' = [1\ C_1\ C_3\ C_5\ C_7\ C_{11}\ C_{15}]$$

with

$$\begin{array}{rclrcl} c_1(x) & = & 20_8, & c_3(x) & = & 22_8, \\ c_5(x) & = & 3_8, & c_7(x) & = & 27_8, \\ c_{11}(x) & = & 15_8, & c_{15}(x) & = & 23_8. \end{array}$$

In binary form,

$$G = \begin{bmatrix} 1\ 10000\ 10010\ 00011\ 10111\ 01101\ 10011 \\ 1\ 01000\ 01001\ 10001\ 11011\ 10110\ 11001 \\ 1\ 00100\ 10100\ 11000\ 11101\ 01011\ 11100 \\ 1\ 00010\ 01010\ 01100\ 11110\ 10101\ 01110 \\ 1\ 00001\ 00101\ 00110\ 01111\ 11010\ 00111 \end{bmatrix}$$

3.5 The (257,16) 16-th Power Residue Code

A (257,16) PR code exists since the multiplicative order of 2 mod 257 is 16, and 16 divides (257-1)/16, thus $e = 1$, $s = 16$ and $m = 16$. This code is an interesting example because k is a multiple of 8, which is useful when encoding digital data, and $257 = 2^{2^3} + 1$ is a Fermat prime, which allows simple computation of Fourier Transforms. It is defined as a cyclic code with a parity check polynomial of the form

$$h(x) \;=\; \prod_{r \epsilon R} (x - \alpha^r) \tag{3.2}$$

where α is a primitive 257-th root of unity, and R is the set of 16-th residues mod 257. The elements of R are solutions of the congruence

$$x^{16} \;\equiv\; r \ mod\ 257. \tag{3.3}$$

Thus

$$R \;=\; \{1, 2, 4, 8, 16, 32, 64, 128, 129, 193, 225, 241, 249, 253, 255, 256\}.$$

The related rate 1/16, (256,16) QC code has a generator matrix of the form

$$G = [I_{16}, C_1, C_2, C_3, \cdots, C_{15}] \tag{3.4}$$

where C_i is a 16×16 circulant matrix. There are only 15 C_i's since the 16-th circulant has been inverted and multiplied through to create a systematic code. The dual code has the generator matrix

$$H = \begin{bmatrix} & C_1^T \\ & C_2^T \\ I_{240} & C_3^T \\ & \vdots \\ & C_{15}^T \end{bmatrix} \tag{3.5}$$

The minimum distance of this dual code, found from the weight distribution of the (256,16) code, is 5. The minimum distance of this code is upper bounded by the minimum distance of the embedded rate 1/2 codes,

$$d_{min}[H] \leq \min_{i=1}^{15} \{d_{min}[I_{16}, C_i]\} \tag{3.6}$$

where d_{min} [.] denotes the minimum distance of the given QC code. Examination of the 15 possible rate 1/2 codes reveals

$$\begin{array}{rll} 1 & \text{with } d_{min} = & 8, \\ 3 & \text{with } d_{min} = & 7, \\ 10 & \text{with } d_{min} = & 6, \\ 1 & \text{with } d_{min} = & 5, \end{array}$$

and so the bound holds with equality.

This method can be extended to all rate $(k-1)/k$ subcodes, of which there are $\binom{15}{k-1}$ rate $(k-1)/k$ systematic QC codes, $2 \leq k \leq 16$. From the previous result all these subcodes must have $d_{min} \geq 5$, but from [25], the best possible minimum distance of a rate 2/3 (48,32) systematic QC code is

6. Thus the minimum distance of the subcodes of rates higher than 1/2 is bounded by $5 \leq d_{min} \leq 6$, so the subcodes must be good. An exhaustive examination of all subcodes revealed 266 best or optimal QC codes. The numbers are as follows:

count	code	rate	d_{min}
1	(32,16)	1/2	8
42	(48,32)	2/3	6
4	(48,16)	1/3	14
56	(64,48)	3/4	6
70	(80,64)	4/5	6
56	(96,80)	5/6	6
28	(112,96)	6/7	6
8	(128,112)	7/8	6
1	(144,128)	8/9	6

This search used only 15 generator polynomials. An exhaustive search would require examining $\approx 2^{12}$ polynomials.

3.6 The Quasi-Cyclic Subcodes

In this section, subcodes of the previous QC codes are enumerated. They were found using the method illustrated in Section 3.5. Only the minimum distances and code dimensions are given for those codes which are the best possible QC codes and/or attain the bounds given in [32]. Tables 3.2 and 3.5 list the generator polynomials, $c(x)$, in octal form, with the least significant digit to the left. Tables 3.3, 3.4, 3.6 and 3.7 give the subcodes; n, k is the code dimension, m is the circulant size, and d_{min} is the minimum distance. Further details on the format can be found in [26]. For the case $m = 7$, the best rate $1/p$ subcodes are exactly those given in [26], and so are not given here.

For the rate $1/p$ codes, the $c(x)$ refer to the C_i in G as in (2.1). For rate $(k-1)/k$ codes, the $c(x)$ refer to the C_i in the dual code of G, which is H as given by (2.3).

Table 3.2: $c(x)$ for $m = 3$ to 20

m	3	5	7	8	9	11^a	11^b	14	15	16	20
i	$c(x)$ (in octal)										
1	4	20	100	200	400	2000	2000	20000	40000	100000	2000000
2	5	22	44	246	114	3342	3406	16236	41542	136340	1447243
3		3	63		676		1031	10637	12620	155631	
4		27	176		737		1317		55010	2454	
5		15	106		137		3440		71654	151764	
6		23	36		123		2517		66365	63225	
7			16		60		1501		10263	134412	
8			140		431		1336		30167	31116	
9			41						76431	35607	
10			166						72432	135570	
11			37							16740	
12			61							126206	
13			144							157664	
14			27							120170	
15			124							154777	
16			127							133766	
17			125								
18			143								

Notes: n^a derived from the (23,11) PR code
n^b derived from the (89,11) PR code.

Table 3.3: The Subcodes for $m = 5$ to 15

n	k	m	d_{min}	$c(x)$ from Table 3.2
10	5	5	$+4^o$	1,5
15	10	5	$+4^o$	1,5,6
20	15	5	$+3^o$	1,4,5,6
25	15	5	$+3^o$	1,3,4,5,6
15	5	5	$+7^o$	1,5,6
20	5	5	$+9^o$	1,2,5,6
25	5	5	11	1,2,3,5,6
27	18	9	-4^o	1,2,3
55	44	11	-4^o	1,2,3,4,5
66	55	11	$+4^o$	1,2,3,4,5,6
77	66	11	$+4^o$	1,2,3,4,5,6,7
44	11	11	-16^o	1,2,4,5
66	11	11	26	1,2,3,4,5,8
77	11	11	-32	1,2,3,4,5,6,7
45	30	15	-6^o	1,3,4
60	45	15	$+6^o$	1,3,4,5
75	60	15	$+6^o$	1,3,4,5,9

Notes: n^o denotes a best Quasi-Cyclic code
+ meets the upper bound in [32]
\- meets the lower bound in [32].

Table 3.4: The Subcodes for $m = 16$

n	k	m	d_{min}	$c(x)$ from Table 3.2
32	16	16	$+8^o$	1,8
48	32	16	-6^o	1,4,8
64	48	16	-6^o	1,4,7,8
80	64	16	$+6^o$	1,4,7,8,9
96	80	16	$+6^o$	1,4,7,8,9,11
112	96	16	$+6^o$	1,4,7,8,9,11,12
128	112	16	$+6^o$	1,4,7,8,9,11,12,13
142	112	16	$+6^o$	1,4,7,8,9,11,12,13,15
48	16	16	-14^o	1,5,8
80	16	16	-28	1,5,8,11,12
96	16	16	34	1,2,3,5,7,13
128	16	16	50	1,4,8,9,11,12,13,15
144	16	16	57	1,4,7,8,9,11,12,13,15
160	16	16	64	1,2,3,5,7,8,10,12,13,15
176	16	16	72	1,2,6,7,8,9,11,12,13,14,15
192	16	16	80	1,3,4,6,7,8,10,11,12,13,14,15
208	16	16	86	1,2,3,4,5,7,8,9,12,13,14,15,16
224	16	16	94	1,2,3,4,5,7,8,9,10,11,12,13,14,15
240	16	16	103	1,2,3,4,5,7,8,9,10,11,12,13,14,15,16

Notes: n^o denotes a best Quasi-Cyclic code
+ meets the upper bound in [32]
- meets the lower bound in [32].

Table 3.5: $c(x)$ for $m = 21$ to 26

m	21	22^d	23	24	25^{cd}	26^d
i	$c(x)$ (in octal)					
1	4000000	10000000	20000000	40000000	100000000	200000000
2	5322626	14565622	7355313	30600255	143107551	255164547
3	5457416	6767355		72301211	72722107	54557347
4	2352036	17641013		51075256	107135603	226477305
5	2026706	15210737		77530342	111266031	123012062
6	2130401	10463320		54344633	175022707	255253206
7	4702724	1025711		32105215	101605134	16310203
8	3133226	2436745		65130346	146453234	111456127
9	4126762	3075155		43356773	73656237	157041127
10	4320445	4135373		43621613	7210245	
11	2573101	3100111				
12	7424663	10025375				
13	4533556	4551230				
14	5117300	10663447				
15	2050277					
16	6215365					

Notes: n^c derived from the (1801,25) PR code
n^d only a partial listing of the $c(x)$ is given.

Table 3.6: The Subcodes for $m = 21$ to 22

n	k	m	d_{min}	$c(x)$ from Table 3.5
105	21	21	33	1,2,10,14,15
126	21	21	42	1,2,3,4,5,14
147	21	21	52	1,5,6,9,12,13,14
168	21	21	60	1,2,3,4,5,6,10,14
189	21	21	70	1,2,5,6,7,11,12,13,15
210	21	21	80	1,2,3,5,6,10,11,12,13,15
231	21	21	88	1,2,3,4,5,6,7,9,12,13,14
252	21	21	98	1,2,3,4,5,6,7,9,11,12,13,14
273	21	21	108	1,2,3,4,5,6,7,9,10,12,13,14,15
294	21	21	118	1,2,3,4,5,6,7,8,11,12,13,14,15,16
315	21	21	127	1,2,3,4,5,6,8,9,10,11,12,13,14,15,16
66	44	22	-8	1,4,6
66	22	22	-18	1,4,8
88	22	22	26	1,4,6,7
110	22	22	34	1,2,3,4,5
132	22	22	44	1,2,3,4,5,7
154	22	22	54	1,2,3,4,6,9,10
176	22	22	64	1,2,3,7,11,12,13,14

Notes: + meets the upper bound in [32]
- meets the lower bound in [32].

Table 3.7: The Subcodes for $m = 24$ to 26

n	k	m	d_{min}	$c(x)$ from Table 3.5
72	48	24	8	1,2,4
96	72	24	7	1,2,3,4
72	24	24	18	1,2,3
96	24	24	27	1,2,9,10
120	24	24	36	1,2,4,6,10
144	24	24	47	1,2,4,7,8,10
168	24	24	57	1,2,3,5,6,7,10
192	24	24	68	1,2,3,5,6,7,9,10
216	24	24	78	1,2,3,4,5,6,7,8,10
50	25	25	-10^o	1,2
75	50	25	-8	1,3,4
100	75	25	-8	1,3,4,5
75	25	25	19	1,6,7
100	25	25	-28	1,2,7,9
125	25	25	38	1,2,8,9,10
52	26	26	-10^o	1,2
78	52	26	8	1,2,3
104	78	26	-8	1,7,8,9
78	26	26	-20	1,2,3
104	26	26	e30	1,4,5,6

Notes: n^o denotes a best Quasi-Cyclic code
+ meets the upper bound in [32]
- meets the lower bound in [32]
e exceeds the lower bound in [32].

3.7 Concluding Remarks

This Chapter presents the construction of binary Quasi-Cyclic codes from Power Residue codes. Several new QC codes with large block lengths have been found, some of which are the best possible QC codes. The extension of this method to nonbinary codes will be addressed in Chapter 6.

Chapter 4

Primitive Polynomials with Linearly Independent Roots

4.1 Introduction

Primitive polynomials with linearly independent roots were first applied to the field of error correcting codes, where they were used via the Normal Basis Theorem[40] to construct Quasi-Cyclic codes from Power Residue codes[17], as in the previous Chapter. A normal basis can also be used to construct QC codes from Cyclic codes with $(n, k) = lm$. In this case the generator matrix can be transformed into one formed of k/m rows and n/m columns of $m \times m$ circulant matrices [20]. Subsequently a normal basis representation has been employed to facilitate multiplication and inversion over GF(2^m)[41]. This representation can easily be used to accelerate the decoding of BCH codes. In practice the roots of a primitive polynomial with linearly independent roots is used to form a normal basis. Another application of these polynomials is found in the area of digital testing of integrated circuits, which uses a Linear Feedback Shift Register (LFSR) implementation as a means of data compaction [42].

Peterson and Weldon [29] provide Tables of primitive polynomials over GF(2), including ones with independent roots for most degrees up to 34. Unfortunately, there exists no similar Tables over fields larger than GF(2), i.e., GF(3), GF(4), GF(5), etc. These are required to construct error correcting codes and LFSRs over nonbinary alphabets. Completion of the Tables in [29] over GF(2) was presented in [39], along with the binary power residue codes constructed with them. Tables of primitive and irreducible polynomials over GF(3) are given in [43]. This Chapter provides Tables of primitive polynomials with independent roots over nonbinary fields, and the completed Table over GF(2) from [39]. The algorithm developed to compile these Tables exploits the properties of Galois Fields, and the polynomial coefficients, to improve the computational complexity by several orders of magnitude over exhaustive methods.

4.2 Polynomial Construction

This section presents the algorithm used to construct polynomials over GF(q), and assumes a rudimentary knowledge of Galois fields. An excellent treatment of the theory of Galois fields can be found in [29] or [44]. The background for the algorithm development follows.

Let $p(x)$ be a monic polynomial over GF(q).

Definition 4.1 A polynomial $p(x)$ over GF(q) is irreducible iff it cannot be expressed as the product of two polynomials, $g(x)h(x)$ of degree less than the degree of $p(x)$.

Definition 4.2 An element α of GF(q^n) is primitive iff $\alpha^m = 1$ for no m less than $q^n - 1$. The order of any element of GF(q^n) divides $q^n - 1$.

Definition 4.3 A polynomial $p(x)$ of degree n over GF(q) is primitive iff it is irreducible and contains a primitive element of GF(q^n) as a root.

Theorem 4.4 [29] *Every polynomial $p(x)$ of degree n irreducible over GF(q) is a factor of $x^{q^n-1} - 1$.*

From Definition 1, it is evident that an easy test for identifying irreducible polynomials is to divide by all polynomials of degree $\lfloor \frac{n}{2} \rfloor$ or less, where n is the degree of $p(x)$. However, this becomes an impractical solution for large m and q. Thus we use Theorem 4.4 as the starting point in the search for primitive polynomials. However, there are other polynomials of degree less than n which are also factors, and when multiplied together, can create a polynomial of degree n which is not irreducible. Thus this condition is only necessary. For sufficiency we require the following.

Theorem 4.5 [44] *An irreducible polynomial $p(x)$ which is a factor of $x^{q^n-1}-1$ will have degree n iff it does not contain a factor of $x^{q^m-1}-1$ for any m a proper divisor of n.*

Thus the product of all irreducible polynomials of degree n over GF(q) is given by

$$\frac{x^{q^n-1}-1}{lcm(x^{q^m-1}-1)\forall m|n}$$

where lcm means least common multiple. A more useful form for implementation is given by the following corollary.

Corollary 4.6 *A polynomial $p(x)$ of degree n which is a factor of $x^{q^n-1}-1$ is irreducible iff it is not a factor of*

$$\prod_i (x^{q^{m_i}-1}-1),$$

where m_i is a proper divisor of n.

Another algorithm for finding irreducible polynomials over GF(q) [45] exploits the relationship between the cyclic classes of sequences of length n and the irreducible polynomials of degree n. This algorithm is more tedious and less obvious than the one given here. Since the computational complexity differences are comparable, the developed algorithm is preferred.

The test for primitivity requires the following,

Theorem 4.7 [29] *An irreducible polynomial of degree n is primitive iff it divides $x^m - 1$ for no m less than $q^n - 1$, m a proper divisor of $q^n - 1$.*

The search is refined with the following Theorems.

Let $f(x) = x^n + a_1 x^{n-1} + a_2 x^{n-2} + \ldots + a_n$ be an irreducible polynomial of degree n over GF(q), (a_i is an element of GF(q)). q is called the characteristic of the field. Define the reciprocal of $f(x)$ as $f^*(x) = x^n f(x^{-1})$.

Theorem 4.8 [29] *The reciprocal of $f(x)$ is irreducible.*

Theorem 4.9 [29] *If $f(x)$ is primitive, $f^*(x)$ is primitive.*

From these two Theorems it is clear that only one of $f(x)$ and $f^*(x)$ need be checked for irreducibility and primitivity. Thus the search time is halved. The polynomial to be tested is arbitrarily chosen to be the one with the largest magnitude when evaluated at $x = q$.

Denote $\rho(a)$ as the companion matrix of $f(x) = x^n + a_1 x^{n-1} + a_2 x^{n-2} + \ldots + a_n$,

$$\rho(a) = \begin{bmatrix} 0 & 1 & 0 & \ldots & 0 \\ 0 & 0 & 1 & \ldots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & \ldots & 1 \\ -a_n & -a_{n-1} & -a_{n-2} & \ldots & -a_1 \end{bmatrix} \tag{4.1}$$

with characteristic polynomial

$$f(x) = \det(xI - \rho(a)) \tag{4.2}$$

If the trace of $\rho(a)$ is defined as

$$T(a) = \sum_{i=1}^{n} \rho_{ii}(a) = -a_1 \tag{4.3}$$

and the norm of $\rho(a)$ as

$$N(a) = \det(\rho(a)) = (-1)^n a_n \tag{4.4}$$

then

$$f(x) = x^n - T(a)x^{n-1} + \ldots + (-1)^n N(a). \tag{4.5}$$

Theorem 4.10 [29] *In a field of characteristic* $q, (a+b)^q = a^q + b^q$

Proof In a field of characteristic $q, q = 0$. Then

$$(a+b)^q = a^q + \binom{q}{1} a^{q-1}b + \binom{q}{2} a^{q-2}b^2 + \binom{q}{3} a^{q-3}b^3 + \cdots + \binom{q}{q-1} ab^{q-1} + b^q$$

and all the binomial coefficients have q as a factor and therefore are 0, leaving only $a^q + b^q$.□

Theorem 4.11 [29] *If* α *denotes a root of* $f(x)$ *then*

$$f(x) = (x-\alpha)(x-\alpha^q)(x-\alpha^{q^2})\ldots(x-\alpha^{q^{n-1}})$$

Proof From Theorem 4.10,

$$\begin{aligned} [f(x)]^q &= (x^n)^q + (a_1x^{n-1})^q + (a_2x^{n-2})^q + \cdots + (a_n)^q \\ &= (x^n)^q + a_1^q(x^{n-1})^q + a_2^q(x^{n-2})^q + \cdots + a_n^q. \end{aligned}$$

From Definition 4.3, $a^{q-1} = 1$, so that $a^q = a$. Therefore,

$$\begin{aligned} [f(x)]^q &= (x^n)^q + a_1(x^{n-1})^q + a_2(x^{n-2})^q + \cdots + a_n \\ &= f(x^q). \end{aligned}$$

Thus if $f(\alpha) = 0$, then $[f(\alpha)]^q = f(\alpha^q) = 0$, and $\alpha, \alpha^q, \alpha^{q^2}, \cdots, \alpha^{q^{n-1}}$ must be roots of $f(x)$, and are all the n roots.□

From this we can now define

$$N(a) = \prod_{i=0}^{n-1} \alpha^{q^i} = \alpha^{\frac{(q^n-1)}{(q-1)}} \tag{4.6}$$

and

$$T(a) = \sum_{i=0}^{n-1} \alpha^{q^i} \tag{4.7}$$

These representations of $N(a)$ and $T(a)$ are used to accelerate the search algorithm via the following Theorems.

Theorem 4.12 *If* ${a_n}^d = (-1)^{dn}$, $d|(q-1)$, $d \neq q-1$, *then* $f(x)$ *is not primitive.*

Proof If ${a_n}^d = (-1)^{dn}$, $N^d(a) = (-1)^{dn}(-1)^{dn} = (-1)^{2dn} = 1$, thus $\alpha^{d\frac{(q^n-1)}{(q-1)}} = 1$ and α has order $d(q^n-1)/(q-1)$. However, for $f(x)$ to be primitive, α must have order $q^n - 1$, thus $f(x)$ is not primitive.□

Example 1

Let $q = 3$, then $d = 1$ and $a_n = (-1)^n$ denotes a nonprimitive polynomial. This condition was shown empirically in the Tables of [43, 46].

Example 2

Let $q = 7$, then $d = 1, 2, 3$ and the a_n for nonprimitive polynomials are as follows:

$$\begin{array}{lll} d=1, & a_n=(-1)^n & \left\{\begin{array}{lcrl} n \textit{ even}, & a_n & = & 1 & \Rightarrow a_n = 1 \\ n \textit{ odd}, & a_n & = & -1 & \Rightarrow a_n = 6 \end{array}\right. \\ d=2, & a_n^2 = 1 & \Rightarrow a_n = 1, 6 \\ d=3, & a_n^3 = (-1)^{3n} & \left\{\begin{array}{lcrl} n \textit{ even}, & a_n^3 & = & 1 & \Rightarrow a_n = 1, 2, 4 \\ n \textit{ odd}, & a_n^3 & = & -1 & \Rightarrow a_n = 3, 5, 6 \end{array}\right. \end{array}$$

Example 3

Let $q = 13$, then $d = 1, 2, 3, 4, 6$ and the a_n for nonprimitive polynomials are as follows:

$$\begin{array}{lll} d=1, & a_n=(-1)^n & \left\{\begin{array}{lcrl} n \textit{ even}, & a_n & = & 1 & \Rightarrow a_n = 1 \\ n \textit{ odd}, & a_n & = & -1 & \Rightarrow a_n = 12 \end{array}\right. \\ d=2, & a_n^2 = 1 & \Rightarrow a_n = 1, 12 \\ d=3, & a_n^3 = (-1)^{3n} & \left\{\begin{array}{lcrl} n \textit{ even}, & a_n^3 & = & 1 & \Rightarrow a_n = 1, 3, 9 \\ n \textit{ odd}, & a_n^3 & = & -1 & \Rightarrow a_n = 4, 10, 12 \end{array}\right. \\ d=4, & a_n^4 = 1 & \Rightarrow a_n = 1, 5, 8, 12 \\ d=6, & a_n^6 = 1 & \Rightarrow a_n = 1, 3, 4, 9, 10, 12. \end{array}$$

Thus only $a_n = 2, 6, 7, 11$ produce primitive polynomials.

Corollary 4.13 *If* q *is a prime of the form* $4m + 1$, *then an irreducible poly-*

nomial $f(x)$ is not primitive if a_n is a quadratic residue of q. If q is a prime of the form $4m-1$, $f(x)$ is not primitive if a_n is a quadratic residue and n is even, or a_n is a nonresidue and n is odd.

Proof $f(x)$ is not primitive if $a_n^d = (-1)^{nd}, d|(q-1)$. If $d = (q-1)/2$, then $a_n^{\frac{q-1}{2}} = (-1)^{\frac{n(q-1)}{2}}$, which equals 1 if $4|(q-1)$, and $(-1)^n$ if $2|(q-1)$.□

Example 4

Let $q = 2^2 = 4$, then $d = 1$ and $a_n = (-1)^n$ denotes a nonprimitive polynomial, as with $q = 3$. However, in this case $-1 = 1$ so that $a_n = 1$ always produces a nonprimitive polynomial.

Corollary 4.14 *Except for $x+1$, any polynomial $f(x)$ with coefficients over $GF(2^m)$, $m > 1$, and $a_n = 1$, is not primitive.*

Corollary 4.15 *The case $d = 1$ is relevant only for $q = 3$ and 2^m, since otherwise $2|(q-1)$, and $d = 2$ results in $a_n^2 = 1$ signifying a nonprimitive polynomial. Thus $a_n = (-1)^n$ is redundant.*

Theorem 4.16 *If $a_1 = 0$, $f(x)$ does not have linearly independent roots.*

Proof If $a_1 = 0$, $T(a) = 0$, so that $\sum_{i=0}^{n-1} \alpha^{q^i} = 0$. However, the α^{q^i} are the roots of $f(x)$, so the roots are linearly dependent.□

4.2.1 The Algorithm

Using the results of the previous section, a computer algorithm was developed to find the polynomials. It was designed to be simple yet efficient. The algorithm outlined below searches through all monic polynomials of degree n and ends when all have been checked. First irreducibility is established, then primitivity, and finally the roots are checked for linear independence.

For all monic polynomials of degree n **do**:

1. Check if $a_n = 0$, if so reject the polynomial, as $f(x)$ has a factor x.

2. Find the reciprocal of $f(x)$, $f^*(x)$. If the value of $f^*(x)$, evaluated at $x = q$, is larger than $f(x)$ evaluated at $x = q$, i.e., the magnitude representation of $f^*(x)$ is greater than $f(x)$, reject the polynomial. This is done so that only one of $f(x)$ and $f^*(x)$ is tested.

3. Form the residue of x^{q^n-1} *mod* $f(x)$. If it is not 1, reject the polynomial.

4. Form the residue of $\prod_i (x^{q^{m_i}-1} - 1)$ *mod* $f(x)$, where m_i a proper divisor of n. If it is 0 at any step in the iterative product, reject the polynomial.
   Note: If the polynomial has passed all the above steps, it is irreducible.

5. The first primitivity check is to examine a_n according to Theorem 4.12. If it fails, reject the polynomial.

6. Form the residue of x^m *mod* $f(x)$ for all m a proper divisor of $q^n - 1$. If any result is 1, reject the polynomial.
   Note: If the polynomial has passed all the above steps, it is primitive. The final two checks are for linearly independent roots, and must be performed on both $f(x)$ and $f^*(x)$.

7. Check if $a_1 = 0$, if so, the roots are dependent so reject the polynomial.

8. Form the $n \times n$ matrix of the roots of the polynomial. If the matrix is singular, the roots are dependent so reject the polynomial.
   Note: If the polynomial has passed all the steps, it is primitive with linearly independent roots.

A flowchart of the algorithm implementation is given in Table 4.1. The above ordering is intended only for clarity, not efficiency. The steps were reordered for implementation.

Implementation was done on a *Sun Microsystems 3-280* computer and the Tables compiled. Program initialization includes forming the residues of x^{2^k} *mod* $f(x)$ for $k = 1, 2, \cdots, \lfloor \log_2(q^n - 1) \rfloor$. This allows for the computation

Table 4.1: Flowchart of the Polynomial Construction Algorithm

of any residue with a minimum of $\lfloor \log_2(q^n - 1) \rfloor + 1$ multiplications. Products in the base field q were computed using log and antilog tables.

4.3 The Number of Primitive and Irreducible Polynomials over GF(q)

Explicit formulas exist for the number of primitive and irreducible polynomials over GF(q). The construction program was tested by enumerating the polynomials to ensure a correct total count, as the number of polynomials grows large quickly with increasing q and degree, thus making manual checks impractical.

4.3.1 Enumeration of Primitive Polynomials

Euler's ϕ (totient) function is required to develop the formula. $\phi(m)$ is defined as the number of positive integers r, smaller than m that are coprime to m, i.e., for which $1 \leq r < m$ and $(r, m) = 1$ holds. For example, if $m = 10$, $r = 1, 3, 7, 9$ are coprime to m. Thus $\phi(10) = 4$.
Note that

$$\phi(1) = 1. \tag{4.8}$$

For m a prime p, each of the numbers $r = 1, 2, \ldots, p-1$ is coprime to m and therefore,

$$\phi(p) = p - 1 \tag{4.9}$$

For prime powers p^α, one obtains

$$\phi(p^\alpha) = (p-1)p^{\alpha-1} = p^\alpha \left(1 - \frac{1}{p}\right). \tag{4.10}$$

From (4.8), (4.9) and (4.10), we have

$$\phi(m) = \begin{cases} 1 & \text{for } m = 1; \\ p - 1 & \text{for } m \text{ prime}; \\ (p-1)p^{(\alpha-1)} & \text{for } m \text{ a prime with multiplicity } \alpha. \end{cases} \tag{4.11}$$

Using ϕ, the number of primitive polynomials of degree m over GF(q) [38] is given by

$$P_m = \frac{\phi(q^m - 1)}{m}. \tag{4.12}$$

4.3.2 Enumeration of Irreducible Polynomials

The enumeration of irreducible polynomials requires the Möbius function, μ, from Chapter 2 (2.6). Recall that $\mu(m)$ is defined as

$$\mu(m) = \begin{cases} 1 & \text{if } m = 1; \\ 0 & \text{if } m \text{ is divisible by a square;} \\ (-1)^k & \text{if } m \text{ is the product of } k \text{ distinct primes.} \end{cases} \tag{4.13}$$

Thus $\mu(m) \neq 0$ only for 1 and squarefree integers.

Using this function, the number of irreducible polynomials of degree m over GF(q) [34], is given by

$$I_m = \frac{1}{m} \sum_{\substack{d, \\ d|m}} \mu\left(\frac{m}{d}\right) q^d \tag{4.14}$$

Note that the number of irreducible polynomials of degree m over GF(q) is equal to the number of distinct circulant matrices of dimension m. This relationship is exploited in [45] to find the irreducible polynomials.

Theorem 4.17 *Over GF(2), the number of irreducible polynomials, I_m, equals the number of primitive polynomials, P_m, when $2^m - 1$ is a Mersenne Prime.*

Proof For m prime, $2^m - 1$ is called a *Mersenne number*, and if $2^m - 1$ is prime, it is called a *Mersenne prime* [38]. Since $2^m - 1$ is prime, $P_m = (2^m - 2)/m$. Since m is prime, $I_m = (1/m)(\mu(1)2^m + \mu(m)2)$. But this is exactly $(2^m - 2)/m$. Thus the number of primitive polynomials equals the number of irreducible polynomials when $q^m - 1$ is a *Mersenne prime.*$\square$

Note that the only value of q for which $q^m - 1$ can be prime is 2, since $q^m - 1$ is even for q an odd prime.

4.4 BCH Error Correcting Code Decoding

Decoding of binary BCH codes requires three steps, syndrome computation, construction of the error locator polynomial, and finding the roots of this polynomial [30]. Construction of the error locator polynomial is often done in software, and direct computation requires a large number of mathematical operations. These operations are over a Galois Field, GF(2^n). They can thus be greatly simplified using a normal basis representation for the field elements.

As an example, to find the coefficients of the error locator polynomial for the four error correcting (127,99) BCH code, the following equations must be solved,

$$\begin{aligned}
\sigma_1 &= S_1 \\
\sigma_2 &= \tfrac{S_1(S_7+S_1^7)+S_3(S_1^5+S_5)}{S_3S_1^3+S_1(S_1^5+S_5)} \\
\sigma_3 &= S_1^3 + S_3 + S_1\sigma_2 \\
\sigma_4 &= \tfrac{S_5+S_1^2S_3+(S_1^3+S_3)\sigma_2}{S_1}
\end{aligned}$$

This requires 14 multiplications and 2 divisions. Using a normal basis representation, we can reformulate the equations to take advantage of the simple squaring operation,

$$\begin{aligned}
\sigma_1 &= S_1 \\
\sigma_2 &= \tfrac{S_1(S_7+S_1^4S_3)+S_1^8+S_3S_5}{S_1S_5+S_1^2(S_1^4+S_3S_1)} \\
\sigma_3 &= S_1^3+S_3+S_1\sigma_2 \\
\sigma_4 &= \tfrac{S_5+S_1^2S_3+(S_1^3+S_3)\sigma_2}{S_1}
\end{aligned}$$

Now only 10 multiplications and 2 divisions are needed, along with 3 shift operations. In this case 30% of the multiplications have been eliminated. For a larger number of equations, similar savings can be achieved.

4.5 Tables of Primitive Polynomials with Independent Roots

This Section contains tables of primitive polynomials with independent roots over GF(2), GF(3), GF(4), GF(5), GF(7), GF(8), GF(11), GF(13), GF(16), GF(17) and GF(19). One monic polynomial is given for each degree, with the coefficient of the highest power on the left. The given polynomial was chosen as one with the lowest number of nonzero coefficients of those found. For example, for degree 22 over GF(2), the given polynomial, in octal, is 30000001_8, which has only 3 nonzero coefficients. In contrast, the polynomial 37771001_8 has 13 nonzero coefficients, but is still primitive with independent roots. A low number of nonzero coefficients results in a simpler implementation.

For polynomials over GF(4), the polynomial used to construct the base field is x^2+x+1, over GF(8), x^3+x^2+1 and over GF(16), x^4+x^3+1. Thus over GF(4), $0=0, 1=1, 2=\omega$ and $3=\omega^2$, where ω is a primitive root of the given polynomial.

Table 4.2: Primitive Polynomials with Linearly Independent Roots over GF(2)

degree	polynomial(in octal)
2	7
3	15
4	31
5	67
6	141
7	301
8	615
9	1461
10	3441
11	6501
12	16401
13	33001
14	65001
15	140001
16	324001
17	740001
18	1620001
19	3440001
20	7400001
21	16200001
22	30000001
23	65000001
24	173000001
25	360000001
26	704000001
27	1406000001
28	3000000001
29	7200000001
30	14000000001
31	32100000001

Table 4.3: Primitive Polynomials with Independent Roots over GF(3)

degree	polynomial
2	112
3	1201
4	11002
5	120001
6	1101002
7	12100001
8	110002202
9	1122000001
10	11000001022
11	121000000001
12	1100000001112
13	12000000000001
14	110000000010122
15	1100000000000221
16	11000000000100212
17	110000000000001011
18	1100000000000100002
19	11000000000000000221

Table 4.4: Primitive Polynomials with Independent Roots over GF(4)

degree	polynomial
2	112
3	1123
4	11012
5	110002
6	1100012
7	11000102
8	110000112
9	1100000102
10	11000012002
11	110000000002
12	1100000002312
13	11000000000012
14	110000000000203
15	1100000000000212

Table 4.5: Primitive Polynomials with Independent Roots over GF(5)

degree	polynomial
2	112
3	1102
4	11042
5	110123
6	1100002
7	11000002
8	110000113
9	1100000043
10	11000000023
11	110000000002
12	1100000000112

Table 4.6: Primitive Polynomials with Independent Roots over GF(7)

degree	polynomial
2	113
3	1112
4	11013
5	110004
6	1100125
7	11000064
8	110000003
9	1100000012
10	11000000013
11	110000000004

Table 4.7: Primitive Polynomials with Independent Roots over GF(8)

degree	polynomial
2	115
3	1104
4	11004
5	110002
6	1100002
7	11000042
8	110000206
9	1100000002
10	11000000025

Table 4.8: Primitive Polynomials with Independent Roots over GF(11)

degree	polynomial
2	117
3	1103
4	11008
5	110014
6	1100017
7	11000004

Table 4.9: Primitive Polynomials with Independent Roots over GF(13)

degree	polynomial
2	112
3	1102
4	11012
5	110016
6	1100026
7	1100004(11)

Table 4.10: Primitive Polynomials with Independent Roots over GF(16)

degree	polynomial
2	112
3	1102
4	11018
5	110025
6	110006(12)
7	11000002

Table 4.11: Primitive Polynomials with Independent Roots over GF(17)

degree	polynomial
2	113
3	1107
4	11017
5	110005
6	1100003
7	11000002

Table 4.12: Primitive Polynomials with Independent Roots over GF(19)

degree	polynomial
2	112
3	1106
4	11002
5	110005
6	110000(15)
7	11000005

4.6 Concluding Remarks

In this Chapter, an algorithm to construct primitive polynomials with independent roots is outlined. They are used to construct QC codes from Power Residue codes via the Normal Basis Theorem. As well, primitive polynomials with independent roots are better for signature analysis in digital testing of VLSI circuits [42], and can be used to simplify Galois Field arithmetic.

Chapter 5

Construction of Best Rate 2/3 Quasi-Cyclic Codes Based on Optimum Distance Profile Convolutional Codes

5.1 Introduction

Tail biting codes were first proposed by Solomon[22] and generalized by Ma and Wolf[47]. One subset of these codes are full tail biting (FTB) codes, which can be encoded in the following way. Suppose we have an (n, k, m) convolutional encoder to encode $L+m$ blocks of k information bits, where L is a positive integer. The last m information blocks are used to initialize the encoder, instead of m all-zero blocks as in conventional convolutional encoding. Then $L+m$ information blocks of k bits are sent into the encoder to get $L+m$ coded blocks of n bits. This FTB code is equivalent to an $((L+m)n, (L+m)k)$ quasi-cyclic (QC) code that has the same code rate, k/n, as the corresponding convolutional code [47]. In fact, the codewords of this QC code can be viewed as paths through the underlying convolutional code trellis which start and end in the same state. Using this relationship, Ma and Wolf[47] constructed some rate 1/3 systematic QC codes and rate 1/2 QC codes from known optimum systematic convolutional codes. Based on the assumption that when L equals $(3\ to\ 4)m$, the corresponding maximum free distance should usually be reached, good QC codes can be constructed. It is observed that QC codes with a good minimum distance can likely be constructed with L less than $(3\ to\ 4)m$. This is believed due to the property of optimum distance profile (ODP), which some convolutional codes possess. This motivates the use of ODP convolutional codes to construct good QC codes.

Another possible choice of convolutional codes from which to construct good QC codes are those with large average distance growth rate, d_o, as defined by Huth and Weber[48]. Unfortunately, d_o is known for very few codes. In contrast, many ODP codes are known. A thorough description of ODP codes can be found in [30].

5.2 Construction of QC Codes From ODP Codes and Some Results

ODP convolutional codes were originally designed for efficient sequential decoding. Due to their property of rapid initial column distance growth, ODP convolutional codes are a good choice for constructing QC codes. By best, it is meant the largest possible minimum distance, d_{min}, for a QC code with the given dimensions. Unfortunately, not all ODP codes will generate best QC codes. The reason being that ODP codes have an optimum distance profile only over the first constraint length m, but in FTB encoding the first m blocks of encoded bits from a conventional convolutional encoder are discarded (truncated). It is difficult to predict what will happen to the column distance profile after such truncation, but the minimum distance of the QC codes constructed can easily be computed, thus identifying the best codes.

To construct the code, increase L of the ODP code by 1 and find the generator matrix of the corresponding QC code, as shown in the proof of Theorem 1 in [47]. The minimum distance is then computed. L may be increased until the free distance of the corresponding ODP convolutional code is reached by the minimum distance of the QC code. Using this method, rate 2/3 systematic codes are constructed from the ODP codes in [49].

Rate 2/3 systematic QC codes (nl, kl) have a generator matrix of the form

$$G = \begin{bmatrix} I_{2l} & \begin{matrix} C_1 \\ C_2 \end{matrix} \end{bmatrix} \tag{5.1}$$

where I_{2l} is a $2l \times 2l$ identity matrix and C_1 and C_2 are $l \times l$ circulant matrices over $GF(2)$. The first rows of C_1 and C_2 correspond to the generator polynomials $c_1(x)$ and $c_2(x)$. The weight distributions of these codes were found by first computing the weight distribution of the dual code [17],

$$H = \begin{bmatrix} I_l C_1^T C_2^T \end{bmatrix} \tag{5.2}$$

then using MacWilliam's identities to find the distribution of the original code. This represents a substantial reduction in the number of computations required (from 2^{2l} to 2^l). In computing the weight distributions, all 2^l codewords were formed and the weights tabulated.

Twenty seven best rate 2/3 QC codes of lengths 18 to 60 were found. Some of these are equivalent to the best codes in [18] and [25]. Generator polynomials of the new best codes are given in Table 5.1 along with the FTB encoder memory length. The memory length m is important if decoding involves the trellis of the code. The complexity of these decoding algorithms is normally at least proportional to 2^m. In this case, codes with shorter memory length are preferred.

Included in Table 5.1 are two best (60,40) $d_{min} = 8$ codes. The proof that no better QC code exists proceeds as follows. It is well known that the best rate 1/2 (40,20) systematic QC code has minimum distance 9 [17]. Through an exhaustive search of all polynomials of length 20, it was found that only 120 distinct generator polynomials, (excluding cyclic shifts), produce this distance. In using all possible combinations of these polynomials in a rate 2/3 systematic QC code, a minimum distance of 9 was not achieved. From Corollary 2.9, the minimum distance of a rate 2/3 systematic QC code cannot be greater than the minimum distance of the two embedded rate 1/2 codes [25], and so the maximum minimum distance of a systematic (60,40) QC code can be at most 8. Thus the minimum distance of the given codes cannot be exceeded, and they are then the best possible.

5.3 Concluding Remarks

A method is described to construct good QC codes from ODP convolutional codes. Several new best codes have been presented, including two with $d_{min} = 8$. These results again demonstrate that good QC codes can be constructed without an exhaustive search of all possible generator polynomials.

The drawback of this method, (which may be applied to any convolutional code), is the lack of suitable convolutional codes to transform to QC codes. Most present methods for constructing convolutional codes rely on search techniques, rather than some definite algorithm. However, good convolutional codes do provide a means of constructing QC codes. In fact if the definition of ODP codes is modified to require a rapid column distance growth after the first constraint length, the resulting codes may be more suitable for the construction of good QC codes.

Table 5.1: A Table of Best Rate 2/3 Systematic QC Codes

Code	Memory Length	$c_1(x)$	$c_2(x)$	d_{min}	$d_{min}B$
(24,16)	6	266	342	4	4
(24,16)	4 *	270	310	4	4
(27,18)	6	554	704	4	4
(27,18)	5 *	570	630	4	4
(39,26)	7 *	15500	17040	6	6
(42,28)	12 *	33206	36156	6	6
(45,30)	13	57372	63226	6	6
(45,30)	12	66414	74334	6	6
(45,30)	11 *	57360	63230	6	6
(48,32)	13	136764	146454	6	6
(48,32)	10	136740	146440	6	6
(48,32)	8 *	155200	170600	6	6
(51,34)	13	275750	315130	6	6
(51,34)	12	332060	361560	6	6
(51,34)	11	275700	315140	6	6
(51,34)	10	275700	315100	6	6
(51,34)	9 *	275600	317400	6	6
(60,40)	18	3320162	3614412	8	-
(60,40)	11 *	2757000	3151400	8	-

Notes:

$c_1(x)$ and $c_2(x)$ are generator polynomials in octal representation, with the highest order term on the right.

d_{min} is the minimum distance of the code (and is the maximum possible minimum distance).

$d_{min}B$ is the minimum distance of the code in [18] and [25].

* means the shortest memory length of all given codes of the same length

Chapter 6

Nonbinary Quasi-Cyclic Codes

As mentioned in Chapter 1, very few nonbinary block codes are known beyond RS codes. An obvious starting point is with Quasi-Cyclic codes, as the construction methods outlined in Chapters 2 and 3 can easily be extended. In particular, Power Residue codes are constructed over GF(q), and the search technique of Chapter 2 is extended to find good nonbinary codes. The method outlined in [22] is used to convert RS codes over GF(q) to Maximum Distance Separable (MDS) QC codes. The compiled Tables provide a measure of the error correcting capability of nonbinary codes. This is important because very few nonbinary codes are known.

To maintain continuity, the Tables for this Chapter have all placed at the end after the text.

6.1 Power Residue Codes

As in Chapter 3, let m be the order of q mod n, ($q^m \equiv 1$ mod n), n a prime. Then if m divides $(n-1)$, i.e., $n = ems + 1$, a cyclic (n, em), es-th power residue (PR) code exists, as does a rate $1/s$, $(n-1, em)$ QC code formed of $m \times m$ circulant matrices. A normal basis can be formed from the roots of a primitive polynomial of degree m with linearly independent roots, as found in Chapter 4.

To illustrate the construction of nonbinary PR codes, consider the following example. Let $n = 11$ and $q = 3$, then $5^3 = 243 \equiv 1 \; mod \; 11$. Thus we have an (11,5) PR code over GF(3) composed of two 5×5 circulant matrices and an all 1's column, (in this case $e = 1, s = 2$ and $m = 5$). By definition [17], this is a cyclic code over GF(3) with generator matrix,

$$G \; = \; \left[1, \; \alpha, \; \alpha^1, \; \alpha^2, \; ... \; , \; \alpha^{10}\right] \tag{6.1}$$

where α is a primitive 11th root of unity over GF(3). To form the circulants, the columns of G must be rearranged according to the cyclic classes mod 11 over GF(3), i.e.,

$$\begin{array}{ccccc} 1, & 3, & 9, & 5, & 4 \\ 2, & 6, & 7, & 10, & 8 \end{array}$$

Thus G becomes

$$G \; = \; \left[1, \; \alpha, \; \alpha^3, \; \alpha^9, \; \alpha^5, \; \alpha^4, \; \alpha^2, \; \alpha^6, \; \alpha^7, \; \alpha^{10}, \; \alpha^8\right] \tag{6.2}$$

Now, if these columns are represented in terms of a Normal Basis, α^3 becomes a cyclic shift of α, α^9 becomes a cyclic shift of α^3, and so on. This resulting Generator matrix is of the form

$$G \; = \; [1, \; C_1, \; C_2] \, . \tag{6.3}$$

This code has minimum distance 6, and is the dual of the (11,6) Golay Code over GF(3), which is a perfect 2 error correcting code.

As mentioned previously, the search for good Quasi-Cyclic codes is restricted by the large number of generator polynomials which can be used for construction. This problem is very acute for codes over nonbinary fields, where an exhaustive search of all codes is tractable for only the most simple codes. Thus we must rely on techniques to reduce the set of candidates polynomials which must be examined to find good codes, such as using the circulants from PR codes in QC form. Codes from this Section are later used to initialize the search for good nonbinary QC codes.

The complete weight distributions of the (11,5) Code (Golay) over GF(3) illustrated above is given in Table 6.1. Two other nonbinary PR codes, the (13,3) code over GF(3) and the (5,2) code over GF(4), are given in Tables 6.2 to 6.3, along with their equivalent QC codes. These three short codes are all optimal QC codes, (the (4,2) code is MDS). They illustrate that nonbinary PR codes also produce good QC codes.

Tables 6.4 to 6.8 present nonbinary PR codes and related QC codes over GF(3), GF(4), GF(5), GF(7) and GF(8).

6.2 Constructing Good Nonbinary QC Codes

In this Section, a search is conducted for small block size good or best QC codes over GF(q). Of particular importance are those codes which are Maximum Distance Separable (MDS), i.e., have $d_{min} = n - k + 1$. In the previous Section, nonbinary QC codes were constructed for PR codes. Here these codes are used to initialize the search routine formulated in Chapter 2, but modified to construct nonbinary codes. In this case, only monic polynomials need be considered, since a polynomial can be divided by the coefficient of highest degree without changing the weight structure of the resulting circulant matrix.

For codes over GF(4), the notation is $2 = \omega$ and $3 = \omega^2$, where ω is a primitive root of the polynomial $x^2 + x + 1$. For codes over GF(8), the same format is used, but 2 is now a root of the polynomial $x^3 + x + 1$. Over GF(16), the polynomial is $x^4 + x^3 + 1$.

The nonbinary QC codes are listed in the following Tables. Tables 6.9 to 6.22 give the codes over GF(3) for $m = 2$ to 9 and rate 1/2 to 1/12. Tables 6.23 to 6.33 give the codes over GF(4) for $m = 2$ to 8 and rate 1/2 to 1/12. Tables 6.34 to 6.40 give the codes over GF(5) for $m = 2$ to 5 and rate 1/2 to 1/12. Tables 6.41 to 6.46 give the codes over GF(7) for $m = 2$ to 4 and rate 1/2 to 1/12. Tables 6.47 to 6.53 give the codes over GF(8) for

$m = 2$ to 4 and rate 1/2 to 1/12. Tables 6.54 to 6.58 provide a compilation of the minimum distances of these codes.

(n, k) codes over GF(q) which have $d_{min} = n - k + 1$ are called Maximum Distance Separable (MDS) codes. If this code is composed of $m \times m$ circulant matrices, it is also a QC code. The following well known facts about C, an (n,k) MDS code over GF(q), are useful.

Theorem 6.1[10] *If C is MDS so is the dual code C^T.*

Theorem 6.2[10] *The number of codewords of weight w in C is*

$$A_w = \binom{n}{w}(q-1)\sum_{j=0}^{w-d}(-1)^j\binom{w-1}{j}q^{w-d-j}. \tag{6.4}$$

Thus the weight structure is known *a priori.*

Corollary 6.3[10] *If $k \geq 2, q \geq n-k+1$, and if $k \leq n-2, q \geq k+1$.*

Corollary 6.4 *In a systematic MDS QC code, the $c_i(x)$ cannot have any zero coefficients. Further, for $m \geq 2$, no $c_i(x)$ can have 3 consecutive identical coefficients.*

These results make the task of finding MDS QC codes simpler.

The simplest method of constructing MDS QC codes is from Reed-Solomon (RS) codes, as shown in [22]. The QC codes which are equivalent to RS codes correspond to those RS codes which have parameters $n = mn'$ and $k = mk'$. However, the class of MDS QC codes contains codes which do not have this form. For instance, the RS codes over GF(2^5) have a blocklength which is prime, thus none of these codes can be converted to QC codes. However, MDS QC codes do exist over GF(32).

As an example, consider the (12,6) RS code over GF(13). The generator polynomial for this code is

$$\begin{aligned} & (x-2)(x-4)(x-8)(x-3)(x-6)(x-12) \\ = \;& x^6 + 4x^5 + 8x^4 + 4x^3 + 10x^2 + 3x + 5 \end{aligned}$$

This can be partitioned into

$$x(x^5 + 8x^3 + 10x) + (4x^5 + 4x^3 + 3x)$$

These two polynomials are the generator polynomials of the two circulant matrices for the equivalent rate 1/2 QC code. Multiplying by the inverse of one matrix, and making the remaining polynomial monic, results in a systematic QC code in the required format. The generator polynomials from the above example are given in Table 6.59 along with those for QC MDS codes over GF(11), GF(13) and GF(16). Those over smaller fields are listed with the codes of the same field.

Table 6.1: The (11,5) Power Residue Code over GF(3) and the Related Quasi-Cyclic Code

(11,5) PR Code Generator Matrix

$$\begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 1 & 2 & 2 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 2 & 2 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 2 & 2 \\ 1 & 0 & 0 & 0 & 1 & 0 & 2 & 1 & 0 & 1 & 2 \\ 1 & 0 & 0 & 0 & 0 & 1 & 2 & 2 & 1 & 0 & 1 \end{bmatrix}$$

Weight Distribution

Weight	Count
0	1
6	132
9	110

(10,5) QC Code Generator Matrix

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 2 & 2 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 2 & 2 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 2 & 2 \\ 0 & 0 & 0 & 1 & 0 & 2 & 1 & 0 & 1 & 2 \\ 0 & 0 & 0 & 0 & 1 & 2 & 2 & 1 & 0 & 1 \end{bmatrix}$$

Weight Distribution

Weight	Count
0	1
5	72
6	60
8	90
9	20

Table 6.2: The (13,3) Power Residue Code Over GF(3) and the Related Quasi-Cyclic Code

(13,3) PR Code Generator Matrix

$$\begin{bmatrix} 1 & 1 & 0 & 0 & 2 & 1 & 2 & 1 & 2 & 0 & 2 & 2 & 0 \\ 1 & 0 & 1 & 0 & 2 & 2 & 1 & 0 & 1 & 2 & 0 & 2 & 2 \\ 1 & 0 & 0 & 1 & 1 & 2 & 2 & 2 & 0 & 1 & 2 & 0 & 2 \end{bmatrix}$$

Weight Distribution

Weight	Count
0	1
9	26

(12,3) QC Code Generator Matrix

$$\begin{bmatrix} 1 & 0 & 0 & 2 & 1 & 2 & 1 & 2 & 0 & 2 & 2 & 0 \\ 0 & 1 & 0 & 2 & 2 & 1 & 0 & 1 & 2 & 0 & 2 & 2 \\ 0 & 0 & 1 & 1 & 2 & 2 & 2 & 0 & 1 & 2 & 0 & 2 \end{bmatrix}$$

Weight Distribution

Weight	Count
0	1
8	18
9	8

Table 6.3: The (5,2) Power Residue Code Over GF(4) and the Related Quasi-Cyclic Code

(5,2) PR Code Generator Matrix

$$\begin{bmatrix} 1 & 1 & 0 & 1 & 2 \\ 1 & 0 & 1 & 2 & 1 \end{bmatrix}$$

Weight Distribution

Weight	Count
0	1
4	15

(4,2) QC Code Generator Matrix

$$\begin{bmatrix} 1 & 0 & 1 & 2 \\ 0 & 1 & 2 & 1 \end{bmatrix}$$

Weight Distribution

Weight	Count
0	1
3	12
4	3

Table 6.4: Power Residue Codes, their Duals and Related Quasi-Cyclic Codes Over GF(3)

PR code	d_{min}	dual code	d_{min}	m	rate	QC code	d_{min}	dual code	d_{min}
$(11,5)^{Qo}$	6^{d3}	$(11,6)^{o}$	5	5	1/2	$(10,5)^{o}$	5		
$(13,3)^{Mo}$	9	$(13,10)^{o}$	3	3	1/4	$(12,3)^{o}$	8	(12,9)	3
$(13,6)^{Qo}$	6	$(13,7)^{o}$	5	3	1/2	$(12,6)^{o}$	5		
$(23,11)^{Qo}$	9^{d3}	$(23,12)^{o}$	8	11	1/2	$(22,11)^{o}$	8		
$(37,18)^{Q}$	10	(37,19)	9	18	1/2	(36,18)	9		
(41,8)	22	(41,33)	5	8	1/5	(40,8)	21	(40,32)	5
(61,10)	31	(61,41)	5	10	1/6	(60,10)	30	(60,50)	5
(73,12)	34	(73,61)	5	12	1/6	(72,12)	33	(72,60)	5
(193,16)	96	(193,177)	5	16	1/12	(192,16)	95	(192,176)	5
(547,14)	336	(547,533)	5	14	1/39	(546,14)	335	(546,532)	5
(757,9)	486^{d9}	(757,748)	3	9	1/84	(756,9)	485	(756,747)	3
$(1093,7)^{M}$	729	(1093,1086)	3	7	1/156	(1092,7)	728	(1092,1085)	3
(3581,11)	2538^{d27}	(3581,3570)	3	11	1/350	(3580,11)	2537	(3580,3569)	3

Notes: n^{o} denotes a best code
n^{M} denotes a Maximum length sequence code
n^{Q} denotes a Quadratic Residue code
n^{dz} weights divisible by z
m is the circulant size.

Table 6.5: Power Residue Codes, their Duals and Related Quasi-Cyclic Codes Over GF(4)

PR code	d_{min}	dual code	d_{min}	m	rate	QC code	d_{min}	dual code	d_{min}
$(3,1)^{Qo}$	3	$(3,2)^{o}$	2	1	1/2	$(2,1)^{o}$	2		
$(5,2)^{Qo}$	4^{d4}	$(5,3)^{o}$	3	2	1/2	$(4,2)^{o}$	3		
$(7,3)^{Qo}$	4	(7,3)	3	3	1/2	$(6,3)^{o}$	3		
$(11,5)^{Qo}$	6	$(11,6)^{o}$	5	5	1/2	$(10,5)^{o}$	5		
$(13,6)^{Qo}$	6	$(13,7)^{o}$	5	6	1/2	$(12,6)^{o}$			
$(17,4)^{o}$	12^{d4}	$(17,13)^{o}$	4	4	1/4	$(16,4)^{o}$	11	$(16,12)^{o}$	4
(17,8)	6	(17,9)	5	4	1/2	(16,8)	5		
$(19,9)^{Q}o$	8	$(19,10)^{o}$	7	9	1/2	$(18,9)^{o}$	7		
$(23,11)^{Q}$	8	(23,12)	7	11	1/2	(22,11)	7		
(31,5)	16^{d8}	(31,26)	3	5	1/6	(30,5)	15	(30,25)	3
(31,10)	10	(31,21)	5	5	1/6	(30,5)	9	(30,25)	5
(41,10)	20^{d4}	(41,31)	6	10	1/4	(40,10)	19	(40,30)	6
(43,7)	27	(43,36)	5	7	1/6	(42,7)	26	(42,35)	5
(73,9)	28	(73,64)	3	7	1/8	(72,9)	27	(72,63)	3
(89,11)	40^{d4}	(89,78)	4	11	1/8	(88,11)	39	(88,77)	4
(127,7)	64^{d32}	(127,120)	3	7	1/26	(126,7)	63	(126,119)	3
(257,8)	180^{d4}	(257,249)	4	8	1/32	(256,8)	179	(256,248)	4
(683,11)	289	(683,672)	3	11	1/62	(682,11)	288	(682,671)	3

Notes: n^{o} denotes a best code
n^{Q} denotes a Quadratic Residue code
n^{dz} weights divisible by z
m is the circulant size.

Table 6.6: Power Residue Codes, their Duals and Related Quasi-Cyclic Codes Over GF(5)

PR code	d_{min}	dual code	d_{min}	m	rate	QC code	d_{min}	dual code	d_{min}
$(11{,}5)^{Qo}$	6	$(11{,}6)^{o}$	5	5	1/2	$(10{,}5)^{o}$	5		
$(13{,}4)^{o}$	8	$(13{,}9)^{o}$	4	4	1/3	$(12{,}4)^{o}$	7	$(12{,}8)^{o}$	4
$(19{,}9)^{Q}$	8	(19,10)	7	9	1/2	$(18{,}9)^{o}$	7		
$(31{,}3)^{Mo}$	25	$(31{,}28)^{o}$	3	3	1/10	$(30{,}3)^{o}$	24	$(30{,}27)^{o}$	3
(31,6)	19	(31,25)	4	3	1/5	(30,6)	18	(30,24)	4
(71,5)	50^{d5}	(71,66)	3	5	1/14	(70,5)	49	(70,65)	3
(71,10)	44	(71,61)	6	10	1/7	(70,10)	43	(60,50)	6
(521,10)	370	(521,511)	4	10	1/52	(520,10)	369	(520,510)	4
(829,9)	635^{d5}	(829,820)	3	9	1/92	(828,9)	634	(828,819)	3
(19531,7)	15625	(19531,19524)	3	7	1/2790	(19530,7)	15624	(19530,7)	3

Notes: n^o denotes a best code
n^M denotes a Maximum length sequence code
n^Q denotes a Quadratic Residue code
n^{dz} weights divisible by z
m is the circulant size.

Table 6.7: Power Residue Codes, their Duals and Related Quasi-Cyclic Codes Over GF(7)

PR code	d_{min}	dual code	d_{min}	m	rate	QC code	d_{min}	dual code	d_{min}
$(3{,}1)^{Qo}$	3	$(3{,}2)^{o}$	2	1	1/3	$(2{,}1)^{o}$	2		
$(19{,}3)^{o}$	15	$(19{,}16)^{o}$	3	3	1/6	$(18{,}3)^{o}$	14	$(18{,}15)^{o}$	3
(19,6)	11	(19,13)	5	3	1/3	(18,6)	10	(18,12)	5
(29,7)	19	(29,22)	6	7	1/4	(28,7)	18	(28,21)	6
(43,6)	30	(43,37)	4	6	1/7	(42,6)	29	(42,36)	4
$(2801{,}5)^{M}$	2401	(2801,2796)	3	5	1/560	(2800,5)	2400	(2800,2795)	3

Notes: n^o denotes a best code
n^M denotes a Maximum length sequence code
n^Q denotes a Quadratic Residue code
m is the circulant size.

Table 6.8: Power Residue Codes, their Duals and Related Quasi-Cyclic Codes Over GF(8)

PR code	d_{min}	dual code	d_{min}	m	rate	QC code	d_{min}	dual code	d_{min}
(7,1)	7	(7,6)	2	1	1/6	(6,1)	6	(6,5)	2
(7,2)	6	(7,5)	3	1	1/3	(6,2)	r	(6,4)	3
$(7,3)^Q$	4	(7,4)	3	1	1/2	(6,3)	3		
(13,4)	9	(13,9)	4	4	1/3	(12,4)	8	(12,8)	4
$(17,8)^Q$	6	(17,9)	5	8	1/2	(16,8)	5		
(19,6)	12	(19,13)	6	6	1/3	(18,6)	11	(18,12)	6
(31,5)	16^{d4}	(31,26)	3	5	1/6	(30,5)	15	(70,65)	3
(73,3)	64^M	(73,70)	3	3	1/24	(72,3)	63	(72,69)	3
(73,6)	56^{d4}	(73,67)	3	3	1/12	(72,6)	63	(72,66)	3
(127,7)	64^{d16}	(127,7)	3	7	1/18	(126,7)	63	(126,119)	3
(151,5)	121	(151,146)	3	5	1/30	(150,5)	120	(150,145)	3
(337,7)	253	(337,330)	3	7	1/48	(336,7)	252	(336,329)	3

Notes: n^o denotes a best code
n^M denotes a Maximum length sequence code
n^Q denotes a Quadratic Residue code
n^{dz} weights divisible by z
m is the circulant size.

Table 6.9: Best Rate 1/2 QC Codes over GF(3) for $m = 2$ to 12

(2m,m) QC code	Generator Polynomial $c(x)$	d_{min}
(4,2)	12	2
(6,3)	112	3
(8,4)	1112	4
(10,5)	1221	5
(12,6)	1112	5
(14,7)	11211	6
(16,8)	11221	6
(18,9)	11121	6
(20,10)	1101121	7
(22,11)	100111212	8
(24,12)	10112112	8

Table 6.10: Generator Polynomials for $m = 2$ to 5 over GF(3)

Polynomial Number	m			
	2	3	4	5
1	1	1	1	1
2	11	11	11	11
3	12	12	12	101
4		111	101	102
5		112	102	111
6			111	112
7			112	121
8			121	122
9			122	1011
10			1112	1012
11			1122	1021
12				1022
13				1111
14				1112
15				1121
16				1202
17				1211
18				1212
19				1221
20				1222
21				11112
22				11122
23				12122

Table 6.11: Rate $1/p$, $m = 2$ Quasi-Cyclic Codes over GF(3)

Code	d_{min}	Generators
(6,2)	4	1,2,3
(8,2)	6	1,1,2,3
(10,2)	7	1,1,1,2,3
(12,2)	8	1,1,2,2,3,3
(14,2)	10	1,1,1,2,2,3,3
(16,2)	12	1,1,1,1,2,2,3,3
(18,2)	13	1,1,1,1,1,2,2,3,3
(20,2)	14	1,1,1,1,2,2,2,3,3,3
(22,2)	16	1,1,1,1,1,2,2,2,3,3,3
(24,2)	18	1,1,1,1,1,1,2,2,2,3,3,3

Table 6.12: Rate $1/p$, $m = 3$ Quasi-Cyclic Codes over GF(3)

Code	d_{min}	Generators
(9,3)	6	1,2,5
(12,3)	8	1,2,2,3
(15,3)	9	1,1,2,2,3
(18,3)	12	1,1,1,2,2,3
(21,3)	14	1,1,1,2,2,2,3
(24,3)	16	1,1,1,1,2,2,2,3
(27,3)	18	1,1,1,1,2,2,2,2,3
(30,3)	20	1,1,1,1,2,2,2,2,3,3
(33,3)	22	1,1,1,1,2,2,2,2,2,3,3
(36,3)	24	1,1,1,1,1,2,2,2,2,2,3,3

Table 6.13: Rate $1/p$, $m = 4$ Quasi-Cyclic Codes over GF(3)

Code	d_{min}	Generators
(12,4)	6	1,3,7
(16,4)	9	1,3,7,8
(20,4)	12	1,3,5,8,10
(24,4)	15	1,6,7,8,9,10
(28,4)	18	1,3,4,6,7,9,10
(32,4)	21	1,2,3,6,7,8,9,10
(36,4)	23	1,2,3,5,6,7,8,9,10
(40,4)	25	1,2,3,4,5,6,7,8,9,10
(44,4)	28	1,2,3,4,5,6,6,7,8,9,10
(48,4)	31	1,2,2,3,4,5,6,7,8,9,10,11

Table 6.14: Rate $1/p$, $m = 5$ Quasi-Cyclic Codes over GF(3)

Code	d_{min}	Generators
(15,5)	8	1,13,19
(20,5)	12	1,7,13,19
(25,5)	15	1,2,9,18,22
(30,5)	18	1,2,10,13,15,20
(35,5)	21	1,4,5,6,15,17,22
(40,5)	24	1,3,5,6,6,7,19,22
(45,5)	28	1,5,6,8,9,12,15,17,22
(50,5)	31	1,5,6,7,8,9,10,11,13,21
(55,5)	35	1,2,6,7,8,10,12,13,16,21,23
(60,5)	38	1,4,7,7,8,9,10,11,14,17,20,23

Table 6.15: Generator Polynomials for $q = 3$, $m = 6$

1	1	11	1101	21	10202	31	11211
2	101	12	1102	22	10121	32	11212
3	111	13	1111	23	10212	33	12121
4	112	14	1112	24	10221	34	12122
5	121	15	1121	25	11012	35	12202
6	122	16	1202	26	11021	36	12221
7	1011	17	1212	27	11111	37	111112
8	1012	18	1222	28	11112	38	112122
9	1021	19	10112	29	11121	39	112222
10	1022	20	10121	30	11122		

Table 6.16: Rate $1/p$, $m = 6$ Quasi-Cyclic Codes over GF(3)

Code	d_{min}	Generators
(18,6)	9	1,8,15
(24,6)	13	1,17,19,25
(30,6)	17	1,6,13,23,29
(36,6)	20	1,7,13,14,16,33
(42,6)	24	1,3,10,22,29,32,39
(48,6)	28	1,4,5,8,18,26,31,37
(54,6)	33	1,3,7,11,15,20,32,34,36
(60,6)	36	1,2,4,5,6,28,30,32,35,36
(66,6)	40	1,6,8,12,14,15,21,24,36,38
(72,6)	44	1,3,4,5,7,9,11,25,27,32,33,39

Table 6.17: Generator Polynomials for $q = 3$, $m = 7$

1	1	11	1222	21	11222	31	102112	41	112121	51	122122
2	111	12	10011	22	12112	32	102122	42	112122	52	122211
3	121	13	10021	23	12122	33	102202	43	120221	53	122212
4	122	14	10112	24	12201	34	102221	44	121021	54	1111112
5	1001	15	10122	25	12202	35	110111	45	121102	55	1122222
6	1002	16	10211	26	101022	36	110121	46	121112	56	1212122
7	1011	17	10221	27	101112	37	110122	47	121122	57	1222222
8	1102	18	11112	28	101212	38	110202	48	121211		
9	1201	19	11122	29	101221	39	111121	49	122021		
10	1202	20	11211	30	101222	40	112102	50	122112		

Table 6.18: Rate $1/p$, $m = 7$ Quasi-Cyclic Codes over GF(3)

Code	d_{min}	Generators
(21,7)	10	1,39,52
(28,7)	15	1,8,20,46
(35,7)	18	1,13,14,18,36
(42,7)	24	1,9,11,34,45,48
(49,7)	27	1,21,23,25,38,44,51
(56,7)	32	1,7,10,16,22,27,37,50
(63,7)	36	1,2,3,4,17,26,31,41,54
(70,7)	41	1,2,5,15,24,30,32,36,53,57
(77,7)	45	1,3,4,6,14,19,29,33,35,42,56
(84,7)	50	1,4,6,7,12,23,28,40,43,49,51,55

Table 6.19: Generator Polynomials for $q = 3$, $m = 8$

1	1	14	11221	27	111112	40	1012011	53	1122202
2	112	15	12011	28	112101	41	1012122	54	1122221
3	121	16	12101	29	112122	42	1022111	55	1202221
4	122	17	12201	30	112201	43	1102111	56	1210222
5	1002	18	12222	31	112221	44	1110122	57	1211021
6	1112	19	101021	32	120201	45	1111212	58	1211111
7	10111	20	101201	33	121112	46	1112012	59	1212112
8	10121	21	102021	34	121122	47	1112021	60	1221121
9	10122	22	102112	35	122021	48	1112111	61	1221211
10	10211	23	102121	36	122102	49	1112121	62	1220222
11	11011	24	102222	37	122211	50	1112122	63	11122222
12	11101	25	110211	38	122222	51	1120122		
13	11202	26	110212	39	1011221	52	1121111		

Table 6.20: Rate $1/p$, $m = 8$ Quasi-Cyclic Codes over GF(3)

Code	d_{min}	Generators
(24,8)	11	1,7,50
(32,8)	16	1,18,36,51
(40,8)	21	1,10,30,32,45
(48,8)	26	1,12,29,41,47,56
(56,8)	30	1,2,15,26,34,44,52
(64,8)	36	1,2,28,49,53,57,62,63
(72,8)	41	1,3,17,21,27,31,43,54,59
(80,8)	46	1,6,14,19,22,24,25,35,55,60
(88,8)	51	1,8,11,13,14,20,37,38,39,58,61
(96,8)	56	1,4,5,9,16,23,33,40,42,46,48,50

Table 6.21: Generator Polynomials for $q = 3$, $m = 9$

1	1	14	101102	27	1011212	40	10111211	53	12121121
2	111	15	101221	28	1012111	41	10201022	54	12210212
3	121	16	102021	29	1022101	42	10210202	55	12211111
4	122	17	102212	30	1101011	43	11012012	56	12221111
5	1011	18	110022	31	1110221	44	11021202	57	111111122
6	1121	19	110111	32	1111101	45	11022021	58	111111212
7	10111	20	110212	33	1111201	46	11111221	59	111121122
8	11011	21	111022	34	1121202	47	11112212	60	111222122
9	11121	22	112101	35	1201211	48	11201221	61	112222212
10	11212	23	112112	36	1210111	49	11222121		
11	12011	24	112201	37	1212221	50	12012121		
12	100102	25	1002112	38	10102102	51	12112022		
13	100222	26	1010201	39	10111111	52	12112102		

Table 6.22: Rate $1/p$, $m = 9$ Quasi-Cyclic Codes over GF(3)

Code	d_{min}	Generators
(27,9)	11	1,6,40
(36,9)	17	1,11,25,46
(45,9)	23	1,30,34,36,49
(54,9)	28	1,2,10,35,44,56
(63,9)	33	1,4,23,37,38,39,44
(72,9)	38	1,3,7,18,22,41,48,58
(81,9)	45	1,3,8,15,31,42,45,59,61
(90,9)	51	1,13,14,19,20,29,33,42,51,55
(99,9)	55	1,12,17,26,27,28,32,52,53,54,60
(108,9)	60	1,4,5,10,14,16,21,24,43,47,50,57

Table 6.23: Best Rate 1/2 QC Codes over GF(4) for $m = 2$ to 12

(2m,m) QC code	Generator Polynomial $c(x)$	d_{min}
(4,2)	12	3
(6,3)	112	4
(8,4)	1112	4
(10,5)	1122	5
(12,6)	1112	5
(14,7)	11121	6
(16,8)	11121	6
(18,9)	1112031	7
(20,10)	12113323	8
(22,11)	1123221	8
(24,12)	1011122323	9

Table 6.24: Generator Polynomials for $m = 2$ to 4 over GF(4)

Polynomial Number	m		
	2	3	4
1	1	1	1
2	11	11	11
3	12	12	12
4	13	13	13
5		111	102
6		112	103
7		113	111
8		121	112
9		122	113
10		123	121
11		132	122
12		133	123
13			131
14			133
15			1112
16			1113
17			1122
18			1123
19			1133
20			1213
21			1222
22			1323
23			1333

Table 6.25: Rate $1/p$, $m = 2$ Quasi-Cyclic Codes over GF(4)

Code	d_{min}	Generators
(6,2)	4	1,2,3
(8,2)	6	1,2,3,4
(10,2)	8	1,1,2,3,4
(12,2)	9	1,1,2,3,3,4
(14,2)	11	1,1,1,2,3,3,4
(16,2)	12	1,1,2,2,3,3,4,4
(18,2)	14	1,1,1,2,2,3,3,4,4
(20,2)	16	1,1,1,1,2,2,3,3,4,4
(22,2)	17	1,1,1,1,2,2,3,3,3,4,4
(24,2)	19	1,1,1,1,1,2,2,3,3,4,4,4

Table 6.26: Rate $1/p$, $m = 3$ Quasi-Cyclic Codes over GF(4)

Code	d_{min}	Generators
(9,3)	6	1,3,6
(12,3)	8	1,3,4,6
(15,3)	11	1,3,4,6,7
(18,3)	13	1,2,3,4,6,9
(21,3)	15	1,2,2,3,4,6,7
(24,3)	17	1,1,2,3,4,6,7,9
(27,3)	20	1,1,2,3,4,6,6,7,9
(30,3)	22	1,1,2,2,3,4,6,6,7,9
(33,3)	24	1,2,3,4,5,6,7,9,10,11,12
(36,3)	26	1,1,2,3,4,5,6,7,9,10,11,12

Table 6.27: Rate $1/p$, $m = 4$ Quasi-Cyclic Codes over GF(4)

Code	d_{min}	Generators
(12,4)	7	1,3,15
(16,4)	11	1,9,11,20
(20,4)	13	1,8,9,14,22
(24,4)	16	1,5,7,8,12,15
(28,4)	19	1,4,8,10,14,19,20
(32,4)	22	1,4,7,8,9,11,12,17
(36,4)	25	1,6,7,8,11,12,13,18,23
(40,4)	28	1,4,6,7,8,9,11,13,15,21
(44,4)	32	1,2,5,8,10,11,12,13,15,16,17
(48,4)	35	1,3,6,7,8,9,10,11,14,18,20,21

Table 6.28: Generator Polynomials for $q = 4$, $m = 5$

1	1	14	1013	27	1223	40	11212
2	101	15	1023	28	1231	41	11232
3	102	16	1031	29	1232	42	11312
4	103	17	1032	30	1233	43	11323
5	111	18	1113	31	1302	44	11333
6	112	19	1121	32	1311	45	12222
7	113	20	1123	33	1313	46	12323
8	121	21	1131	34	1321	47	12333
9	122	22	1133	35	1322	48	13133
10	131	23	1202	36	1333	49	13232
11	132	24	1203	37	11112	50	13233
12	133	25	1211	38	11122	51	13323
13	1011	26	1221	39	11132	52	13333

Table 6.29: Rate $1/p$, $m = 5$ Quasi-Cyclic Codes over GF(4)

Code	d_{min}	Generators
(15,5)	8	1,24,52
(20,5)	12	1,8,10,37
(25,5)	16	1,2,19,25,43
(30,5)	20	1,11,17,27,36,40
(35,5)	23	1,3,12,20,21,32,44
(40,5)	27	1,4,5,19,25,29,35,39
(45,5)	31	1,22,23,27,28,33,38,41,48
(50,5)	34	1,6,7,8,14,16,33,36,42,48
(55,5)	38	1,3,8,9,12,15,18,34,41,47,51
(60,5)	41	1,8,9,12,13,18,26,30,31,35,46,50

Table 6.30: Generator Polynomials for $q = 4$, $m = 6$

1	1	11	1101	21	1322	31	11132	41	12321	51	112133
2	103	12	1102	22	1333	32	11133	42	12323	52	113222
3	111	13	1103	23	10211	33	11221	43	13113	53	113232
4	112	14	1112	24	10212	34	11232	44	13203	54	122222
5	113	15	1133	25	10313	35	12032	45	13221	55	122233
6	121	16	1202	26	10321	36	12103	46	13331	56	123213
7	1012	17	1203	27	11012	37	12211	47	111112	57	123332
8	1013	18	1213	28	11102	38	12222	48	111133	58	123333
9	1032	19	1232	29	11113	39	12231	49	111313	59	131333
10	1033	20	1233	30	11123	40	12303	50	112123	60	132233

Table 6.31: Rate $1/p$, $m = 6$ Quasi-Cyclic Codes over GF(4)

Code	d_{min}	Generators
(18,6)	9	1,6,29
(24,6)	13	1,4,44,54
(30,6)	18	1,10,19,25,47
(36,6)	22	1,10,11,33,46,60
(42,6)	26	1,6,12,16,32,45,56
(48,6)	30	1,2,17,20,22,38,41,57
(54,6)	35	1,3,7,9,24,30,34,47,50
(60,6)	40	1,4,23,26,28,39,42,53,58,59
(66,6)	44	1,5,6,8,18,24,35,48,49,51,55
(72,6)	48	1,13,15,21,27,31,36,37,40,43,46,52

Table 6.32: Generator Polynomials for $q = 4$, $m = 7$

1	1	14	10221	27	112111	40	131213	53	1131213
2	103	15	10321	28	112333	41	132111	54	1132122
3	111	16	11012	29	113023	42	132122	55	1132322
4	112	17	11033	30	113233	43	1111123	56	1132323
5	113	18	11121	31	113321	44	1111223	57	1212132
6	122	19	11213	32	120121	45	1112112	58	1212322
7	1033	20	11231	33	121333	46	1112213	59	1223332
8	1202	21	13233	34	122023	47	1113123	60	1321323
9	1312	22	13322	35	122211	48	1113132	61	1323332
10	10112	23	102222	36	122221	49	1121313		
11	10123	24	103122	37	122223	50	1122113		
12	10132	25	103213	38	123032	51	1122132		
13	10133	26	110221	39	123123	52	1123312		

Table 6.33: Rate $1/p$, $m = 7$ Quasi-Cyclic Codes over GF(4)

Code	d_{min}	Generators
(21,7)	11	1,9,33
(28,7)	15	1,7,16,61
(35,7)	20	1,44,47,55,60
(42,7)	24	1,43,48,50,52,58
(49,7)	30	1,8,15,17,31,45,49
(56,7)	35	1,3,11,14,20,40,41,46
(63,7)	40	1,2,4,21,28,34,35,38,57
(70,7)	44	1,2,4,12,17,23,30,32,48,54
(77,7)	50	1,5,10,11,22,24,25,27,36,39,59
(84,7)	55	1,6,10,13,18,19,26,29,37,42,51,53

Table 6.34: Best Rate 1/2 QC Codes over GF(5) for $m = 2$ to 10

(2m,m) QC code	Generator Polynomial $c(x)$	d_{min}
(4,2)	12	3
(6,3)	112	4
(8,4)	1112	4
(10,5)	1112	5
(12,6)	11124	6
(14,7)	111121	6
(16,8)	111213	7
(18,9)	123144	7
(20,10)	1113123	8

Table 6.35: Generator Polynomials for $m = 2$ to 4 over GF(5)

Polynomial Number	m		
	2	3	4
1	1	1	1
2	11	11	11
3	12	12	12
4	13	13	13
5	14	14	14
6		112	103
7		113	111
8		114	112
9		122	113
10		123	121
11		132	122
12		142	123
13		143	124
14			131
15			133
16			134
17			141
18			144
19			1113
20			1114
21			1122
22			1123
23			1124
24			1132
25			1134
26			1142
27			1143
28			1213
29			1232
30			1333
31			1422
32			1424
33			1432
34			1442

Table 6.36: Rate $1/p$, $m = 2$ Quasi-Cyclic Codes over GF(5)

Code	d_{min}	Generators
(6,2)	4	1,3,4
(8,2)	6	1,2,3,4
(10,2)	8	1,2,3,4,5
(12,2)	10	1,1,2,3,4,5
(14,2)	11	1,1,2,3,3,4,5
(16,2)	13	1,1,1,2,3,3,4,5
(18,2)	14	1,1,2,2,3,3,4,4,5
(20,2)	16	1,1,2,2,3,3,4,4,5,5
(22,2)	18	1,1,1,2,2,3,3,4,4,5,5
(24,2)	20	1,1,1,1,2,2,3,3,4,4,5,5

Table 6.37: Rate $1/p$, $m = 3$ Quasi-Cyclic Codes over GF(5)

Code	d_{min}	Generators
(9,3)	6	1,3,6
(12,3)	8	1,3,4,6
(15,3)	11	1,4,5,6,10
(18,3)	13	1,3,4,5,6,10
(21,3)	16	1,3,4,5,6,12,13
(24,3)	19	1,2,3,4,6,8,10,11
(27,3)	21	1,2,3,4,5,6,7,8,10
(30,3)	24	1,2,3,4,5,6,8,9,10,11
(33,3)	25	1,2,3,4,5,6,7,8,10,11,12
(36,3)	28	1,2,2,3,4,5,6,8,8,9,10,11

Table 6.38: Rate $1/p$, $m = 4$ Quasi-Cyclic Codes over GF(5)

Code	d_{min}	Generators
(12,4)	7	1,2,22
(16,4)	11	1,8,16,19
(20,4)	14	1,7,8,15,28
(24,4)	17	1,7,8,18,32,34
(28,4)	20	1,8,9,11,18,27,31
(32,4)	23	1,5,6,8,14,20,21,25
(36,4)	26	1,4,5,6,8,20,21,23,29
(40,4)	30	1,3,7,15,16,17,19,22,32,33
(44,4)	33	1,2,4,6,7,12,26,27,29,32,33
(48,4)	36	1,2,8,9,10,11,13,16,17,22,24,30

Table 6.39: Generator Polynomials for $q = 5$, $m = 5$

1	1	11	134	21	1123	31	1331	41	11124	51	12134
2	12	12	142	22	1124	32	1333	42	11213	52	13222
3	101	13	1011	23	1132	33	1341	43	11232	53	13233
4	102	14	1013	24	1212	34	1403	44	11242	54	13322
5	111	15	1022	25	1213	35	1413	45	11312	55	14223
6	112	16	1024	26	1214	36	1414	46	11313	56	14322
7	121	17	1043	27	1244	37	11112	47	11332		
8	123	18	1113	28	1302	38	11113	48	11412		
9	124	19	1121	29	1322	39	11114	49	11432		
10	131	20	1122	30	1324	40	11123	50	11433		

Table 6.40: Rate $1/p$, $m = 5$ Quasi-Cyclic Codes over GF(5)

Code	d_{min}	Generators
(15,5)	9	1,6,30
(20,5)	13	1,9,26,39
(25,5)	16	1,7,8,19,37
(30,5)	20	,3,22,30,38,40
(35,5)	24	1,4,11,21,40,44,48
(40,5)	28	1,8,14,16,27,41,42,47
(45,5)	32	1,8,10,12,13,18,43,45,49
(50,5)	36	1,9,21,23,25,28,37,50,51,52
(55,5)	40	1,5,17,24,29,33,34,35,36,53,54
(60,5)	44	1,2,12,15,19,20,31,32,33,46,55,56

Table 6.41: Best Rate 1/2 QC Codes over GF(7) for $m = 2$ to 7

(2m,m) QC code	Generator Polynomial $c(x)$	d_{min}
(4,2)	12	3
(6,3)	112	4
(8,4)	111	4
(10,5)	1112	5
(12,6)	11124	6
(14,7)	111213	7

Table 6.42: Generator Polynomials for $m = 2$ to 3 over GF(7)

Polynomial Number	m	
	2	3
1	1	1
2	11	11
3	12	12
4	13	13
5	14	14
6	15	15
7	16	16
8		112
9		113
10		114
11		115
12		116
13		123
14		125
15		132
16		134
17		143
18		144
19		146
20		155
21		162
22		164
23		165
24		166

Table 6.43: Rate $1/p$, $m = 2$ Quasi-Cyclic Codes over GF(7)

Code	d_{min}	Generators
(6,2)	5	1,3,4
(8,2)	6	1,3,4,5
(10,2)	8	1,2,3,4,5
(12,2)	10	1,2,3,4,5,6
(14,2)	12	1,2,3,4,5,6,7
(16,2)	14	1,1,2,3,4,5,6,7
(18,2)	15	1,1,2,3,3,4,5,6,7
(20,2)	17	1,1,1,2,3,3,4,5,6,7
(22,2)	19	1,1,1,2,3,3,4,4,5,6,7
(24,2)	20	1,1,2,2,3,3,4,4,5,5,6,7

Table 6.44: Rate $1/p$, $m = 3$ Quasi-Cyclic Codes over GF(7)

Code	d_{min}	Generators
(9,3)	6	1,8,9
(12,3)	9	1,4,8,10
(15,3)	12	1,5,9,14,19
(18,3)	14	1,5,6,8,9,10
(21,3)	17	1,3,7,8,9,13,24
(24,3)	19	1,2,5,7,16,18,20,22
(27,3)	22	1,5,6,7,8,13,15,17,24
(30,3)	24	1,4,5,7,9,10,18,22,23,24
(33,3)	27	1,2,3,5,7,8,10,14,18,21,24
(36,3)	30	1,3,5,6,7,8,9,10,11,12,13,14

Table 6.45: Generator Polynomials for $q = 7$, $m = 4$

1	1	11	121	21	1124	31	1252	41	1532
2	14	12	123	22	1125	32	1256	42	1545
3	15	13	125	23	1126	33	1263	43	1553
4	16	14	134	24	1136	34	1333	44	1566
5	102	15	143	25	1146	35	1342	45	1636
6	104	16	152	26	1154	36	1343	46	1645
7	111	17	1112	27	1162	37	1365	47	1653
8	113	18	1113	28	1213	38	1445	48	1665
9	114	19	1116	29	1234	39	1455	49	1666
10	116	20	1123	30	1235	40	1462		

Table 6.46: Rate $1/p$, $m = 4$ Quasi-Cyclic Codes over GF(7)

Code	d_{min}	Generators
(12,4)	8	1,12,17
(16,4)	11	1,11,16,43
(20,4)	14	1,2,13,17,20
(24,4)	18	1,3,15,18,22,28
(28,4)	21	1,3,12,19,20,22,35
(32,4)	25	1,3,7,15,24,36,37,43
(36,4)	28	1,3,5,9,17,22,28,30,33
(40,4)	31	1,14,25,29,38,41,42,43,45,48
(44,4)	34	1,4,6,8,10,11,17,22,26,32,44
(48,4)	37	1,21,22,23,27,31,34,39,40,46,47,49

Table 6.47: Best Rate 1/2 QC Codes over GF(8) for $m = 2$ to 6

(2m,m) QC code	Generator Polynomial $c(x)$	d_{min}
(4,2)	12	3
(6,3)	112	4
(8,4)	111	4
(10,5)	1112	5
(12,6)	11123	6

Table 6.48: Generator Polynomials for $m = 2$ over GF(8)

1	1	5	14
2	11	6	15
3	12	7	16
4	13	8	17

Table 6.49: Rate $1/p$, $m = 2$ Quasi-Cyclic Codes over GF(8)

Code	d_{min}	Generators
(6,2)	5	1,3,4
(8,2)	7	1,3,4,5
(10,2)	8	1,3,4,5,6
(12,2)	10	1,2,3,4,5,6
(14,2)	12	1,2,3,4,5,6,7
(16,2)	14	1,2,3,4,5,6,7,8
(18,2)	16	1,1,2,3,4,5,6,7,8
(20,2)	17	1,1,2,3,3,4,5,6,7,8
(22,2)	19	1,1,1,2,3,3,4,5,6,7,8
(24,2)	21	1,1,1,2,3,3,4,4,5,6,7,8

Table 6.50: Generator Polynomials for $q = 8$, $m = 3$

1	1	11	115	21	145	31	175
2	12	12	116	22	152	32	177
3	13	13	117	23	155		
4	14	14	123	24	156		
5	15	15	124	25	157		
6	16	16	125	26	164		
7	17	17	126	27	165		
8	112	18	133	28	166		
9	113	19	143	29	167		
10	114	20	144	30	174		

Table 6.51: Rate $1/p$, $m = 3$ Quasi-Cyclic Codes over GF(8)

Code	d_{min}	Generators
(9,3)	7	1,17,19
(12,3)	9	1,8,9,15
(15,3)	12	1,8,9,15,19
(18,3)	14	1,9,10,11,14,15
(21,3)	17	1,4,6,8,12,13,16
(24,3)	20	1,4,6,8,11,12,15,19
(27,3)	23	1,3,6,8,17,18,21,30,31
(30,3)	25	1,4,5,6,10,13,19,22,28,29
(33,3)	27	1,2,3,17,19,23,25,26,27,28,32
(36,3)	30	1,4,5,6,7,10,12,19,20,20,24,28

Table 6.52: Generator Polynomials for $q = 8$, $m = 4$

1	1	11	113	21	135	31	1132	41	1263
2	13	12	114	22	152	32	1133	42	1272
3	14	13	117	23	161	33	1142	43	1314
4	16	14	121	24	166	34	1145	44	1335
5	17	15	123	25	172	35	1146	45	1455
6	101	16	124	26	1112	36	1176	46	1542
7	102	17	125	27	1115	37	1233	47	1573
8	103	18	131	28	1122	38	1234	48	1646
9	104	19	132	29	1123	39	1235		
10	111	20	134	30	1124	40	1253		

Table 6.53: Rate $1/p$, $m = 4$ Quasi-Cyclic Codes over GF(8)

Code	d_{min}	Generators
(12,4)	8	1,16,26
(16,4)	11	1,2,26,40
(20,4)	15	1,13,18,29,31
(24,4)	18	1,3,20,30,36,45
(28,4)	21	1,4,15,20,25,29,42
(32,4)	25	1,4,15,25,26,31,35,41
(36,4)	28	1,6,15,21,24,31,37,42,46
(40,4)	31	1,7,11,14,19,21,29,30,33,39
(44,4)	34	1,4,8,12,19,23,29,29,38,47,48
(48,4)	37	1,5,8,9,14,17,28,29,29,32,34,43

Table 6.54: Maximum Minimum Distances for (pm, m) Systematic QC Codes over GF(3)

	p										
m	2	3	4	5	6	7	8	9	10	11	12
2	2^o	4^o	6^o	7^o	8^o	10^o	12^o	13^o	14^o	16^o	18^o
3	3^o	6^o	8^o	9^o	12^o	14^o	16^o	18^o	20^o	22^o	24^o
4	4^o	6^o	9^o	12^o	15^o	18^o	21^o	23^o	25^o	28^o	31^o
5	5^o	8^o	12^o	15^o	18	21	24	28	31	35	38
6	5^o	9^o	13	17	20	24	28	33	36	40	44
7	6^o	10	15	18	24	27	32	36	41	45	50
8	6^o	11	16	21	26	30	36	41	46	51	56
9	6^o	11	17	23	28	33	38	45	51	55	60

Table 6.55: Maximum Minimum Distances for (pm, m) Systematic QC Codes over GF(4)

	p										
m	2	3	4	5	6	7	8	9	10	11	12
2	3^o	4^o	6^o	8^o	9^o	11^o	12^o	14^o	16^o	17^o	19^o
3	4^o	6^o	8^o	11^o	13^o	15^o	17^o	20^o	22^o	24^o	26^o
4	4^o	7^o	11^o	13	16	19	22	25	28	32	35
5	5^o	8^o	12	16	20	23	27	31	34	38	41
6	5^o	9	13	18	22	26	30	35	40	44	48
7	6^o	11	15	20	24	30	35	40	44	50	55

Note: n^o denotes a best code.

Table 6.56: Maximum Minimum Distances for (pm, m) Systematic QC Codes over GF(5)

	p										
m	2	3	4	5	6	7	8	9	10	11	12
2	3^o	4^o	6^o	8^o	10^o	11^o	13^o	14^o	16^o	18^o	20^o
3	4^o	6^o	8^o	11^o	13^o	16^o	19^o	21^o	24^o	25^o	28^o
4	4^o	7^o	11^o	14^o	17^o	20^o	23^o	26^o	30^o	33^o	36^o
5	5^o	9^o	13	16	20	24	28	32	36	40	44

Table 6.57: Maximum Minimum Distances for (pm, m) Systematic QC Codes over GF(7)

	p										
m	2	3	4	5	6	7	8	9	10	11	12
2	3^o	5^o	6^o	8^o	10^o	12^o	14^o	15^o	17^o	19^o	20^o
3	4^o	6^o	9^o	12^o	14^o	17^o	19^o	22^o	24^o	27^o	30^o
4	4^o	8^o	11	14	18	21	25	28	31	34	37

Table 6.58: Maximum Minimum Distances for (pm, m) Systematic QC Codes over GF(8)

	p										
m	2	3	4	5	6	7	8	9	10	11	12
2	3^o	5^o	7^o	8^o	10^o	12^o	14^o	16^o	17^o	19^o	21^o
3	4^o	7^o	9^o	12^o	14^o	17^o	20^o	23^o	25^o	27^o	30^o
4	4^o	8^o	11	15	18	21	25	28	31	34	37

Note: n^o denotes a best code.

Table 6.59: MDS QC Codes over GF(11), GF(13) and GF(16)

GF(11)		
Code	d_{min}	Generator Polynomials
(10,2)	9	1,12,13,15,17
(9,3)	7	1,113,12(10)
(8,4)	5	1,1125
(10,5)	11	1,1(10)375
GF(13)		
Code	d_{min}	Generator Polynomials
(12,2)	11	1,12,13,14,15,16
(12,3)	10	1,112,135,153
(12,4)	9	1,1347,172(11)
(10,5)	6	1,11292
(12,6)	7	1,1(11)(10)482
GF(16)		
Code	d_{min}	Generator Polynomials
(16,2)	15	1,12,13,14,15,18,1(10),1(11)
(15,3)	13	1,185,13(12),178,1(15)(13)
(12,4)	9	1,1247,1776
(15,5)	11	1,13(10)5(11),1(13)623

Chapter 7

Summary of Results and Suggestions for Future Work

In Chapter 1, the historical background and fundamentals of error correcting codes were introduced. The mathematical framework for the specific class of Quasi-Cyclic codes was developed.

Chapter 2 presented the results of a successful search to find good or best rate $1/p$ and $(p-1)/p$ Quasi-Cyclic (QC) codes (they are optimal only if no better linear code exists). Codes up to rates 1/18 and 17/18 were constructed, and for m up to 16. As well, rate 1/2 codes up to $m = 31$ and rate 2/3 codes up to $m = 26$ were found. Of the many new binary codes in this dissertation, 14 extend the known bounds on the minimum distance of binary linear codes. As well, many of the binary QC codes listed attain the bounds as given in [32]. The methods for creating these codes was also described.

In Chapter 3 QC codes were constructed from Power Residue codes. The minimum distances of the binary PR codes, their duals and related QC codes was found for m up to 32 and $n < 10000$. This extends the very limited known results on these codes. Their subcodes were used to construct other systematic QC codes. Search techniques for QC codes with $m > 16$ are extremely difficult because of the large number of available circulant matrices. The PR codes permit a reduced exhaustive search for good codes, and this has produced many best QC codes. The generator polynomials of the PR codes were used to initialize the search algorithm in Chapter 2.

The construction of primitive polynomials with linearly independent roots was the topic of Chapter 4. Tables of polynomials were presented for GF(2), GF(3), GF(4), GF(5), GF(7), GF(8), GF(11), GF(13), GF(16), GF(17) and GF(19). Tables of these polynomials over nonbinary fields are unknown, and over GF(2) they are incomplete. This search was motivated by a requirement for a normal basis over the corresponding fields, which can be formed with the polynomial roots. A normal basis was used in Chapter 3 to construct QC codes from PR codes.

In Chapter 5 QC codes were derived from Optimum Distance Profile convolutional codes. Several best rate 2/3 codes were found in this manner. This extends the known results for this construction, a Table of rate 1/2 QC codes. The (60,40) $d_{min} = 8$ QC codes found using this method constitute the only known QC codes with these dimensions and distance. This warrants further investigation into the connection between QC codes and convolutional codes.

In Chapter 6 the methods of Chapters 2 and 3 were extended to nonbinary QC codes. Codes over GF(3), GF(4), GF(5), GF(7) and GF(8) were tabulated. In addition, Maximum Distance Separable QC codes over GF(11), GF(13) and GF(16) were given. These represent the only known nonbinary QC codes, and establish a basis with which to compare other nonbinary codes, QC or otherwise. Future work in this area will include a comparison with nonbinary Cyclic codes, of which little is presently known.

7.1 Suggestions for Future Work

The paper by Tanner [23] provides a tantalizing look at a class of transforms which may be useful for QC codes. However, the class of Quasi-Cyclic codes requires a mathematical framework designed exclusively for them. The use of frequency domain concepts, while useful for BCH, RS and other algebraically structured codes, is not well suited to QC codes. An approach based on permutation or group theory may be more appropriate.

The analysis of codes of rates other than $1/p$ and $(p-1)/p$, i.e., 2/5, 2/7, 3/5, etc., should yield many new best codes. Convolutional codes have already been constructed for these rates [50].

There exist Hadamard and Conference matrices over complex and other fields. The binary Hadamard matrices have been used successfully to construct Quasi-cyclic codes [51]. As well, Conference matrices have been used to construct ternary QC codes [10, 52]. The use of these types of matrices to construct nonbinary QC codes is therefore quite promising, and well worth further research. As an example, consider the following code over GF(3) with Generator matrix

$$G = [I_8 \; P]\,,$$

where P is defined as

$$P = \begin{bmatrix} 1 & q \\ q^T & C \end{bmatrix}$$

and C is a 7×7 circulant matrix defined by the polynomial $c(x) = 1 + x + 2x^2 + x^3 + 2x^4 + 2x^5 + 2x^6$. This is based on the (7,3,1) cyclic difference set, (see Appendix B), with 0 mapped to -1 (2). q is the all 1's vector. This code has $d_{min} = 6$ and is a Self-Dual code over GF(3) with weights divisible by 3.

7.1.1 Construction of Good Convolutional Codes From Quasi-Cyclic Codes

The construction of good block codes (large minimum distance) has been extensively researched in past work with many algorithms constructed. Most exploit the algebraic structure of the codes. Among these are methods to construct Quasi-Cyclic codes. On the other hand, algorithms to construct good convolutional codes are not as plentiful or powerful, as they are generally based on search techniques. Thus it seems natural to find a method of constructing convolutional codes from good block codes. The reverse has already been shown to be successful in a previous Section. The recent link found between convolutional codes and Quasi-Cyclic codes[23] may be exploited using the Quasi-Cyclic codes already known to find good convolutional codes. However, as experienced with the construction of QC codes from ODP codes in Chapter 5, the criteria for good codes differs between them, and this can affect the quality of the new codes. Thus block code search techniques should be developed (modified) to find QC codes which will yield good convolutional codes.

Bibliography

[1] Hamming, R.W.,"Error Detecting and Error Correcting Codes", *Bell Systems Tech. J.*, vol. 29, pp. 147-160, 1950.

[2] Shannon, C.E., "A Mathematical Theory of Communication", *Bell Systems Tech. J.*, vol. 27, pp. 379-423, 623-656, 1948.

[3] Golay, M.J.E., "Notes on Digital Coding", *Proc. IRE*, vol. 37, pp. 657, 1949.

[4] Thompson, T.M., "From Error-Correcting Codes Through Sphere Packings to Simple Groups", *Carus Mathematical Monographs*, no. 21, 1983.

[5] Hocquenghem, A., "Codes Corecteurs d'Erreurs", *Chiffres*, vol. 2, pp. 147-156, 1959.

[6] Reed, I.S. and Solomon, G., "Polynomial Codes Over Certain Finite Fields", *J. Soc. Ind. Appl. Math.*, vol. 8, pp. 300-304, June 1960.

[7] Reed, I.S., "A Class of Multiple-Error-Correcting Codes and the Decoding Scheme", *IRE Trans. Inf. Theory*, vol. PGIT-4, pp. 38-49, 1954.

[8] Goppa, V.D., "A New Class of Linear Error-Correcting Codes", *Prob. Pered. Inf.*, vol.7, no. 3, pp. 207-212, 1970.

[9] Justesen, J., "A Class of Constructive Asymptotically Good Algebraic Codes", *IEEE Trans. Inf. Theory*, vol. IT-18, pp. 652-656, 1972.

[10] MacWilliams, F.J. and Sloane, N.J.A., *The Theory of Error-Correcting Codes*, North-Holland Publishing Co., 1977.

[11] Weldon, E.J., Jr., "Long Quasi-Cyclic Codes are Good", (abstract), *IEEE Trans. Inf. Theory*, vol. IT-13, pp. 130, 1970.

[12] Kasami, T., "A Gilbert-Varshamov Bound for Quasi-Cyclic Codes of Rate 1/2", *IEEE Trans. Inf. Theory*, vol. IT-20, p. 679, 1974.

[13] Lin, S. and Weldon, E.J., Jr., "Long BCH Codes are Bad", *Inf. and Contr.*, vol. 11, pp. 445-451, 1967.

[14] Townsend, R.L., and Weldon, E.J., Jr., "Self-Orthogonal Quasi-Cyclic Codes", *IEEE Trans. Inf. Theory*, vol. IT-13, pp. 183-195, Apr. 1967.

[15] Karlin, M., "New Binary Coding Results by Circulants", *IEEE Trans. Inf. Theory*, vol. IT-15, pp. 81-92, Jan. 1969.

[16] Karlin, M., "Decoding of Circulant Codes", *IEEE Trans. Inf. Theory*, vol. IT-16, pp. 797-802, Nov. 1970.

[17] Chen, C.L., Peterson, W.W. and Weldon, E.J., Jr., "Some Results on Quasi-Cyclic Codes," *Inf. and Contr.*, vol. 15, pp. 407-423, 1969.

[18] Tavares, S.E., Bhargava, V.K. and Shiva, S.G.S., "Some Rate-p/(p+1) Quasi-Cyclic Codes", *IEEE Trans. Inf. Theory*, vol. IT-20, pp. 133-135, Jan. 1974.

[19] Bhargava, V.K. and Stein, J.M., "(v, k, λ) Configurations and Self-Dual Codes", *Inf. and Contr.*, vol. 28, pp. 352-355, Aug. 1975.

[20] Hoffner, C.W. and Reddy, S.M., "Circulant Bases for Cyclic Codes", *IEEE Trans. Inf. Theory*, vol. IT-16, pp. 511-512, Jul. 1970.

[21] Bhargava, V.K., Seguin, G.E. and Stein, J.M., "Some (mk, k) Cyclic Codes in Quasi-Cyclic Form", *IEEE Trans. Inf. Theory*, vol. IT-25, pp. 112-118, Jan. 1979.

[22] Solomon, G. and van Tilborg, H.C.A., "A Connection Between Block Codes and Convolutional Codes", *J. Soc. Ind. Appl. Math.*, vol. 37, pp. 358-369, Oct. 1979.

[23] Tanner, R.M., "Convolutional Codes from Quasi-Cyclic Codes: A Link Between the Theories of Block and Convolutional Codes", Rep. USC-CRL-87-21, University of California, Santa Cruz, Nov. 1987.

[24] Carter, W.C., Duke, K.A. and Jessep, D.C., Jr., "Lookaside Techniques for Minimum Circuit Memory Translators", *IEEE Trans. Computers*, vol. C-22, pp. 283-289, Mar. 1973.

[25] Stein, J.M., Bhargava, V.K. and Tavares, S.E., "Weight Distribution of Some "Best" $(3m, 2m)$ Binary Quasi-Cyclic Codes", *IEEE Trans. Inf. Theory*, vol. IT-21, pp. 708-711, Nov. 1975.

[26] van Tilborg, H., "On Quasi-Cyclic Codes with Rate $1/m$", *IEEE Trans. Inf. Theory*, vol. IT-24, pp. 628-629, Sept. 1978.

[27] MacWilliams, F.J., "Decomposition of Cyclic Codes of Block Lengths 3p,5p,7p", *IEEE Trans. Inf. Theory*, vol. IT-25, pp. 112-118, Jan. 1979.

[28] Slepian, D., "Some Further Theory of Group Codes", *Bell Systems Tech. J.*, vol. 39, pp. 1219-1252, 1960.

[29] Peterson, W.W. and Weldon, E.J., Jr., *Error-Correcting Codes*, MIT Press, Cambridge, MA, 1972.

[30] Lin, S. and Costello, D.J., Jr., *Error Control Coding: Fundamentals and Applications*, Prentice-Hall, Englewood Cliffs, NJ, 1983.

[31] Stein, J.M. and Bhargava, V.K., "Equivalent Rate 1/2 Quasi-Cyclic Codes", *IEEE Trans. Inf. Theory*, vol. IT-21, pp. 588-589, Sept. 1975.

[32] Verhoeff, T., "An Updated Table of Minimum-Distance Bounds for Binary Linear Codes", *IEEE Trans. Inf. Theory*, vol. IT-33, pp. 665-680, Sept. 1987.

[33] Piret, P., "Good Linear Codes of Length 27 and 28", *IEEE Trans. Inf. Theory*, vol. IT-26, pp. 227, Mar. 1980.

[34] Berlekamp, E.R., *Algebraic Coding Theory*, McGraw Hill, New York, NY, 1969.

[35] Wang, Q., Gulliver, T.A., Bhargava, V.K. and Felstead, E.B., "Error Correcting Codes For Fast Frequency Hopped MFSK Spread Spectrum Satellite Communications Under Worst Case Jamming", to appear in *Int. J. of Satell. Commun.*

[36] Bhargava, V.K., "The (151,136) 10-th Power Residue Code and its Performance", *Proc. IEEE*, vol. 71, pp. 683-685, May 1983.

[37] Hall, M., Jr., *Combinatorial Theory*, Blaisdell Publishing Co., Waltham, MA, 1967.

[38] Schroeder, M.R., *Number Theory in Science and Communication*, Springer-Verlag, New York, NY, 1984.

[39] Gulliver, T.A. and Bhargava, V.K., "The Power Residue Codes and Related Quasi-Cyclic Codes", *IEEE Symp. Inf. Theory*, Kobe, Japan, June 1988.

[40] Albert, A.A., *Fundamental Concepts of Higher Algebra*, The University of Chicago Press, Chicago, IL, 1963.

[41] Wang, C.C., Truong, T.K., Shao, H.M., Deutsch, L.J., Omura, J.K. and Reed, I.S., "VLSI Architectures for Computing Multiplications and Inverses in GF(2^m)", TDA Progress Report 42-75, Jul.-Sep. 1983.

[42] Gulliver, T.A., Serra, M. and Bhargava, V.K., "Primitive Polynomials with Independent Roots and Their Applications", *Proc. Canadian Conf. on Elec. and Comp. Eng.*, pp. 818-822, Nov. 1988.

[43] Serra, M. "Tables of Irreducible and Primitive Polynomials for GF(3)", Technical Report, Dept. of Computer Science, University of Victoria, 1986.

[44] Carmichael, R.D., *Introduction to the Theory of Groups of Finite Order*, Dover Publications, Inc., New York, NY, 1956.

[45] Luneberg, H, "On Dedekind Numbers", in *Combinatorial Theory*, Springer Lectures Notes in Math., no. 969, pp. 251-257, 1982.

[46] Serra, M., "Applications of Multi-Valued Logic to Testing of Binary and MVL Circuits", *Int. J. of Elect.*, vol. 63, no. 2, pp. 197-214, 1987.

[47] Ma, H.H. and Wolf, J.K., "On Tail Biting Convolutional Codes," *IEEE Trans. Commun.*, vol. COM-34, pp. 104-111, Feb. 1986.

[48] Huth, G.J. and Weber, C.L., "Minimum Weight Convolutional Codewords of Finite Length", *IEEE Trans. Inf. Theory*, vol. IT-22, pp. 243-246, Mar. 1976.

[49] Johannesson, R. and Paaske, E., "Further Results on Binary Convolutional Codes with an Optimum Distance Profile," *IEEE Trans. Inf. Theory*, vol. IT-24, pp. 264-268, Mar. 1978.

[50] Ferreira, H.C., Wright, D.A., Shaw, I.S. and Wyman, C.R.," Some New Rate $R = k/n$ $(2 \leq k \leq n-2)$ Systematic Convolutional Codes With Good Distance Profiles", submitted to *IEEE Trans. Inf. Theory*.

[51] Bhargava, V.K., Tavares, S.E. and Shiva, S.G.S., "Difference Sets of the Hadamard Type and Quasi-Cyclic Codes", *Inf. and Contr.*, vol. 26, pp. 341-350, 1974.

[52] Pless, V., "Symmetry Codes over GF(3) and New 5-Designs", *J. Comb. Theory*, vol. 12, pp.119-142, 1972.

[53] Harari, S., "A Polynomial Time Algorithm for Finding Minimum Weight Codewords in a Linear Code," *1985 IEEE Symp. Inf. Theory*, Brighton, England, June, 1985.

[54] Bhargava, V.K. and Gulliver, T.A., "Self-Dual Codes Based on the Twin Prime 35", unpublished manuscript.

[55] Bhargava, V.K., "Codes Form Biquadratic Residues", *Elect. Lett.*, vol. 22 pp. 345-346, Mar. 1986.

[56] Massey, J.L., *Threshold Decoding*, MIT Press, Cambridge, MA, 1963.

[57] Rudolph, L.D., "A Class of Majority Logic Decodable Codes", *IEEE Trans. Inf. Theory*, vol. IT-13, pp. 305-307, May 1967.

[58] Rudolph, L.D., "Threshold Decoding of Cyclic Codes", *IEEE Trans. Inf. Theory*, vol. IT-15, pp. 414-418, May 1969.

[59] Rudolph, L.D. and Robbins, W.E., "One-Step Weighted-Majority Decoding", *IEEE Trans. Inf. Theory*, vol. IT-18, pp. 446-448, May 1972.

[60] Zyablov, V.V., "Piece-wise Cyclic Codes and Their Majority Decoding Schemes", *Prob. Pered. Inf.*, vol.4, no. 2, pp. 31-37 (23-27 transl.), 1968.

[61] Ng, S.W., "On Rudolph's Majority Logic Decoding Algorithm", *IEEE Trans. Inf. Theory*, vol. IT-16, pp. 651-652, Sept. 1969.

[62] Storer, T., *Cyclotomy and Difference Sets*, Markham Press, Chicago, IL, 1967.

Appendix A

Computation of an Upper Bound on the Minimum Distance of Quasi-Cyclic Codes

When searching a large number of rate $1/p$ QC codes, it is important to obtain minimum distance estimates quickly. As well, computation of the true minimum distance of an arbitrary QC code can only be done for moderate sized codes, unless a large amount of resources are expended. It is well known that the computational complexity is exponentially dependent on the size of the code, (i.e., the circulant size m). This is especially true for nonbinary codes, where the number of codewords increases as powers of the alphabet size. A method which efficiently bounds the minimum distance with a complexity which is more proportional to the size of the code is presented. This provides a partial solution to an intractable problem. It has been designed to give a quick upperbound on the minimum distance. The algorithm selectively constructs lower weight codewords in search of a minimum weight codeword. The robustness of this algorithm has been tested against many codes with known minimum distances.

It has been reported that a polynomial time algorithm for finding minimum distance codewords exists[53], but no further results have been forthcoming. The method given here exists as a viable means of bounding the minimum distance. The algorithm was designed to exploit the cyclic nature of these codes.

Consider the (n, k) QC code defined by (2.1).

Theorem A.1 *A minimum distance codeword will contain the first row of G.*

Proof Choose any minimum distance codeword, c, which does not contain the first row of G. For some $i(x)$,

$$c = i(x)G.$$

A cyclic shift of c by k places is equivalent to the codeword

$$c' = (i(x)x^k \; mod \; x^m - 1) \; G$$

and this codeword has the same weight as c. Choose k such that $i_0 = 1$. Then this minimum weight codeword contains the first row of G.□

The search starts by choosing this row of G, so finding a minimum weight codeword involves only the remaining $k - 1$ rows.

To continue the search, an additional row of G is added to this codeword. According to the weight of this new codeword, the following occurs:

- If the weight of the new codeword falls below the target minimum distance d_t, the search ends and the code is rejected.
- If the weight is below a given threshold, d_{th}, the search continues with that codeword.
- If the weight is above the threshold, this codeword is abandoned, with another row of G chosen to create a new codeword.

Once the search from one row of G ends, i.e., the search reaches the last row of G, that row is deleted and the search continues with the next row of G. If l is the index of the last row added, additional rows are added only from rows $l + 1$ to k, (except when a new lowest weight codeword is found).

This algorithm was extensively tested using codes with known minimum distances, (and weight distributions). For a reasonably high threshold

($> d_{min} + \approx 5 - 10$), the true minimum distance was arrived at in almost all instances. In most cases when it was not, the minimum distance was attained when $i(x)c_i(x)\ mod\ x^m - 1 = 0$ for some i. This creates a deep 'hole' in the d_{min} 'surface', i.e., the adjacent codewords have a large minimum distance. The following examples illustrate the capability of this algorithm.

Consider the $(70, 35)$ rate $1/2$ QC code based on the twin prime product 35. This code is specified by a generator matrix of the form $G = [I_{35}, A^*]$, where I_{35} is a 35×35 identity matrix and A^* is the incidence matrix of the $(35, 18, 9)$ cyclic difference set [54]. Although, the circulant nature of A^* was exploited to reduce the computer time necessary to find the weight distribution, it still required over one week on a *SUN Microsystems* $2/120$ computer. Using the developed algorithm, an upperbound of $d_{min} \leq 12$ was found in 18 seconds. Although this is one more than the true minimum distance of 11, there are only 70 codewords with this weight. As well, there are only 315 codewords of weight 12.

For the (70,35) code based on the (35,17,8) cyclic difference set, a bound of $d_{min} \leq 11$ was found in 2.3 seconds. In this case there are only 7 minimum weight ($d_{min} = 10$) codewords, and 315 of weight 11.

These two incidence matrices can be combined to form a (105,35) QC code. The bound on this code is $d_{min} \leq 18$, found in 9.5 seconds.

As a final example, consider the $(123, 41)$ rate $1/3$ QC code based on the biquadratic residues mod 41 [55]. G is composed of three circulant matrices,

$$G = [I_{41} C_1 C_2]$$

where C_1 is the circulant matrix corresponding to $c_1(x)$, with the coefficients of x^k equal to 1 or 0 depending on whether or not k is a biquadratic residue or a biquadratic residue mod 4. $c_2(x)$ is the complement of $c_1(x)$. The bound of $d_{min} \leq 27$ was found in 2.1 seconds, whereas the computation of the weight distribution is intractable with presently available equipment. Comparison with the table of bounds reveals that $28 \leq d_{min} \leq 34$, so this code does not meet the lower bound in [32]. Based on this fact, further investigation is not necessary.

Appendix B

Majority Logic Decodable Quasi-Cyclic Codes

This Appendix presents some results on the Majority Logic (ML) decoding of Quasi-Cyclic codes. This decoding method is important because of its relative speed and simplicity. Thus it is worth investigating which Quasi-Cyclic codes can be decoded in this manner.

Majority logic decoding is well described in [30, 56]. The use of weighted majority logic decoding was first introduced by Rudolph[57, 58, 59]. The advantage of using weighted ML decoding is an improvement in error correcting capability over standard ML decoding. In[59] it is proved that any code can be decoded with one-step majority logic. However, for a general code the complexity of the resulting circuit makes this method impractical. By placing restrictions on the structure of a code, this complexity may be reduced to acceptable levels.

Suppose each error in an (n, k, d_{min}) Weighted Majority Logic (WML) decodable code causes s parity checks to be in error. Thus $2ts+1$ checks are required to decode to the true minimum distance, where

$$t = \lfloor \frac{d_{min}-1}{2} \rfloor.$$

Consider those codewords of the dual $(n, n-k, d_d)$ code, (combinations of the rows of H), which have a 1 in the first position, (d_d is the minimum distance of this dual code). The other $n-1$ positions will have a minimum of $d_d - 1$ 1's. Suppose there are p minimum weight codewords in H with this property. Then a total of $d_d p$ 1's can be distributed amongst $n-1$ columns. In this case

$$\frac{(n-1)s}{d_d - 1}$$

non-orthogonal parity checks are possible, and only if this value is less than p, the number of minimum distance codewords with a leading 1.

Thus for all WML decodable codes, the following equality must be satisfied,

$$(2ts+1) \leq \frac{(n-1)s}{d_d - 1}$$

or,

$$(n-1)s \geq (d_d - 1)(2ts+1).$$

Now if the received bit in the first position is given weight s, so that it contributes equally with the other received bits, the inequality becomes,

$$(n-1)s \geq (d_d - 1)((2t-1)s+1)). \tag{B.1}$$

As an example, consider the $(22,11)$ $d_{min} = 7$ systematic QC code with $t = 3, n = 22$, and $d_{min} = d_d = 7$. Then,

$$21s \geq (7-1)(6s+1) = 30s+6,$$

which cannot hold for any s. Now consider the (16,8) $d_{min} = 5$ systematic QC code with $t = 2, n = 16$ and $d_{min} = d_d = 5$. The inequality then yields,

$$15s \geq (3-1)(4s+1) = 12s+4.$$

For $s = 1, 15 \geq 16$, which is impossible, but for $s > 1$, $s = 2, 30 \geq 28$ and $s = 3, 45 \geq 40$.

Thus it is proven that orthogonal parity checks ($s = 1$) cannot be used, but using non-orthogonal parity checks, WML decoding may be possible.

Since this is only a necessary condition, the codewords of the dual code H must be investigated to determine the actual number of weighted parity check equations for $s > 1$.

The generator matrix for this code is

$$
\begin{aligned}
G &= \begin{bmatrix} I_8 & C \end{bmatrix} \\
&= \begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0
\end{bmatrix}
\end{aligned}
$$

Then $H = \begin{bmatrix} C^T & I \end{bmatrix}$ and every codeword of G is orthogonal to the row space of H. H is given by

$$
H = \begin{bmatrix}
0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}
$$

The following codewords of H with a leading 1 form 10 parity checks on r_0,

$$
\begin{matrix}
1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\
1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\
1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\
1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\
1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\
1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1
\end{matrix}
$$

For $s = 2$, a maximum of 8 parity check equations are possible. However, two errors may cause four checks to be in error, hence WML decoding is not possible. For $s = 3$, (and with r_0 given 3 votes), there are 13 parity checks on r_0. Note that except for the first column, there are at most 3 1's in each column. Thus two errors will result in at most 6 incorrect checks, leaving a minimum of 7 correct checks. r_0 will then be decoded correctly with a majority vote of the 13 parity checks.

In [60] it is stated that this code is not completely orthogonalizable (s = 1), and thus cannot be decoded up to minimum distance by conventional majority logic. However, using a weighted scheme does allow ML decoding.

A second code mentioned in that paper is the (8,4) Q.C. code with $d_{min} = 4$, and this proves to be a most interesting example of WML decoding. This is the same (8,4) QC code used as an example in Chapter 2. For this code (B.1) gives,

$$7s \geq 3(s + 1) = 3s + 3.$$

If $s = 1, 7 \geq 6$, so this code can potentially be 1 step orthogonalized. However, an examination of the possible parity checks shows that this is not the case, as no two of the parity checks with a leading 1 are orthogonal. This same conclusion was reached in [60].

The generator matrix for this code is

$$G = \begin{bmatrix} I_4 & C \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \end{bmatrix}.$$

Then $H = \begin{bmatrix} C^T & I \end{bmatrix}$ and every codeword of G is orthogonal to the row space of H. H is given by

$$H = \begin{bmatrix} 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

The rowspace (codewords) of H are,

$$\begin{array}{cccccccc} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 \end{array} \qquad \begin{array}{cccccccc} 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{array}$$

These are the transpose of the codewords of G, and are orthogonal to them. Now select those codewords of H which have a leading 1 (except for the all 1's word),

$$\begin{array}{cccccccc} 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \end{array}$$

If non-orthogonal checks are used, WML decoding is possible. For $s = 2$, a minimum of 5 checks are required to correctly decode a single error. If r_0 is given weight s, 6 checks are available. The extra parity check may be used for error detection.

If $s = 3$, 7 parity checks are required. However, with r_0 given 3 votes, 10 checks are available. Obviously one error in r_0 will produce 3 incorrect and 7 correct votes. Thus this scheme can correct any single error. As well, double errors can be detected since two errors not including r_0 will produce only 4 checks in error. If r_0 is incorrect along with one other, $3+3 = 6$ checks will be in error, and this can be detected, since the only possible vote inputs to the majority decision device are

r_0	Others	Number of Parity Checks that are Correct	Number of Parity Checks that are Incorrect
correct	correct	10	0
incorrect	correct	7	3
correct	1 incorrect	7	3
correct	2 incorrect	6	4
incorrect	1 incorrect	4	6

So if there are less than 7 votes for one value, a double error can be flagged, and the true minimum distance is attained.

The listed parity check equations form 7 orthogonal parity checks on r_0 (1st bit). Note that except for the first column, there are exactly 3 1's in each column. Careful examination of these 7 equations reveals that, with the first column deleted, they form a (7,7,3,3,1) Symmetric Balanced Incomplete Block Design (SBIBD) [37]. It is because of this that if two errors other than r_0 occur, only 4 checks will be in error (since $\lambda = 1$) and 6 will be correct.

This technique can be extended to other codes. The minimum weight codewords with a leading 1 in the (24,12) $d_{min} = 8$ QC code contain a (23,253,77,7,21) BIBD. However, the bound on the number of correctable errors, given in [61], is only 2. The parameters of the minimum weight codewords of the (48,24) Quadratic Residue code in QC form also correspond to those of a BIBD.

The (22,11) QC code with $d_{min} = 7$ contains an IBD (Incomplete Block Design), but is not balanced. The parameters are, $v = 21, b = 56, k = 6$, and $r = 16$. Some other possible rate $\frac{1}{2}$ WML decodable QC codes are now presented.

1. The (18,9) $d_{min} = 6$ code. In this case, the inequality (B.1), yields,

$$17s \geq 5(3s + 1) = 15s + 5$$

   Only for $s \geq 3$ can this be satisfied. For $s = 2$, 8 checks were found, so two errors can be detected. For $s = 5$, 22 checks (counting r_0), were found. This is the maximum possible since,

$$\frac{s(n-1)}{d_{min} - 1} = \frac{5(17)}{5} = 17$$

   Since two errors will produce only 10 checks in error, two errors are correctable.

   For this code, 3 of the parity checks are orthogonal ($s = 1$), on all but the first bit. This allows the correction of single and some double errors.

2. The (20,10) $d_{min} = 6$ code. From (B.1),

$$\begin{aligned} & 19s \geq 5(3s + 1) \\ = \quad & 19s \geq 15s + 5 \end{aligned}$$

   Thus orthogonalization is possible for $s > 1$. With $s = 4$, 17 parity checks were found, thus double error correction can be done.

3. The (24,12) $d_{min} = 8$ code: $23s \geq 7(5s + 1)$, or $23s \geq 35s + 7$

4. The (26,13) $d_{min} = 7$ code: $25s \geq 6(5s + 1)$, or $25s \geq 30s + 6$.

5. The (28,14) $d_{min} = 8$ code: $27s \geq 7(5s + 1)$

6. The (34,17) $d_{min} = 8$ code: $33s \geq 7(5s + 1)$, or $33s \geq 35s + 7$

7. The (36,18) $d_{min} = 8$ code: $35s \geq 35s + 7$

8. The (38,19) $d_{min} = 8$ code: $37s \geq 7(5s + 1)$, or $37s \geq 35s + 7$
   In this case, $s \geq 4$ for the inequality to hold.

9. The (40,20) $d_{min} = 9$ code: $39s \geq 8(7s + 1)$, or $39s \geq 56s + 8$

Clearly this decoding method is suitable only for a small circulant size.

B.1 Majority Logic Decoding of Quasi-Cyclic Codes Based on (v, k, λ) Difference Sets

A (v, k, λ) difference set [19] provides a simple means of constructing a one-step Majority Logic decodable QC code. Incomplete cyclic difference sets, which have 'don't care' differences not in the set can also be employed. In this case orthogonality still holds, but some of the received bits are not used in any parity checks. An excellent treatment of cyclic difference sets can be found in [62]. When $\lambda = 1$, the code is completely orthogonalizable in one-step, whereas with $\lambda > 1$, weighted majority logic can be used.

As an example, consider the (31,6,1) Cyclic Difference set, (0,1,3,8,12,18). If we use these to form $c(x)$ we have $G = [I_{31}, C]$, where $c(x) = 1 + x + x^3 + x^8 + x^{12} + x^{18}$. The rows of G with a leading 1 are

```
1101000010001000001000000000000
1010000100010000010000000000001
1000010001000001000000000000110
1000100000100000000000011010000
1000001000000000000110100001000
1000000000000110100001000100000
```

From this we can see that all columns have a 1 in the first location and no other column has more than one 1. Thus this code is three error correcting, with seven orthogonal parity checks.

For $\lambda = 1$, the codes listed in Table B.1 are possible. From this table, it is clear that the codes are asymptotically poor. However, they do provide a construction for QC codes that are easily decoded.

Table B.1: (v, k, λ) Difference Sets for QC Codes

(v, k, λ)	t
(7,3,1)	1
(13,4,1)	2
(21,5,1)	2
(31,6,1)	3
(43,7,1)	3
(57,8,1)	4
(73,9,1)	4
(91,10,1)	5
(133,12,1)	6
(157,13,1)	6
(183,14,1)	7
(273,17,1)	8
(307,18,1)	9
(381,20,1)	10
(553,24,1)	12
(757,28,1)	14
(871,30,1)	15
(1057,33,1)	16
(1407,38,1)	17
(1723,42,1)	21
(1893,44,1)	22
(2257,48,1)	24
(2451,50,1)	25
(3541,60,1)	30
(5113,72,1)	36
(6321,80,1)	40
(8011,90,1)	45
(9507,98,1)	49