

# The Mathematics of Barcodes

Cadet Christopher M. Boyls-White

Faculty Mentor: Dr. Troy Siemers

## ABSTRACT

Barcodes have changed the commercial scene by saving time and money due to their expediency. They have provided a medium for automatic rapid data input into a computer. This research project concentrates on the mathematics involved with barcodes. We will briefly review the history of barcodes, investigate features of 1-dimensional barcodes (specifically the UPC and the POSTNET) and 2-dimensional barcodes. Then we will focus on the PDF417 stacked barcode where we will discuss in detail the encoding and decoding algorithms, error detection and correction, image capturing methods for converting barcode graphics into computer data, and our MATLAB® program for coding a user's data. With our ability to reengineer the PDF417 code with MATLAB®, we create our own versions of unique stacked barcodes. Our customizations provide specific visual appeal for a user while ensuring computer readability.

## INTRODUCTION

It is hard to believe that the barcode system has been around for approximately 50 years. Just about every household product has been marked by a barcode of some. They have dramatically shaped the way our industry operates. Barcodes have allowed for speedy and accurate data input. Barcodes are responsible for automatic pricing at your local store, tracking letters and packages around the globe, verifying and validating tickets (cars, airplane, movie, etc.) and storing secure and non-secure information on identification cards. It would be hard to imagine the world without any barcodes.

The "bar code" was invented by Joseph Woodland and Bernard Silver on October 7, 1952 (the date U.S. patent 2,612,994 was issued) to meet demand for developing a system that could automatically read a product's information while a consumer is checking out at the grocery store (Adams, 2007; Sanitate, 2002). It took some time before anyone could adapt the new invention to further practical applications. In 1969 the

NAFC asked Logicon, Inc. to develop a universal bar code system that could meet the needs of the industry (Adams, 2007). In the summer of 1970, the Universal Grocery Products Identification Code (UGPIC) was created in order to start the process for standardization among all grocery stores (Adams, 2007). In 1973, the Universal Product Code (UPC) that was developed by George Laurer and submitted by IBM was adopted by the grocery industry (Adams, 2007). In June 1974, the first UPC scanner was created by NCR Corporation and later that month the first scanned product code appeared on a 10 pack of Wrigley's Juicy Fruit® gum (Adams, 2007). That pack of gum is now on display at the Smithsonian Institution's National Museum of American History. Since the UPC, there have been more complex barcodes developed, such as the composite barcode (ex. PDF417), 2-dimensional barcodes (ex. DataMatrix® and MaxiCode®), Radio Frequency Identification (RFID) tags and even 3-dimensional barcodes. From the initial barcode in the early 1952s to the present,

there have been more than 300 different barcode symbologies (Ashford, 2008).

Concurrent to the development of barcodes has been the development of reader devices.

### READER DEVICES

Having a reading device allows for a fast, easy and accurate way of entering data into a computer interface rather than having to manually enter the number (located usually below the barcode), which is slower, often cumbersome and is prone to more mistakes by the user. By means of a light source of some type, the device reads the barcode and converts the spaces into zeros and bars into ones. This received signal of 0s and 1s is then transmitted to the computer. The computer uses mathematical formulas to convert the array of 0s and 1s into a look up number that directly corresponds to specific data entry in a central computer database system. The process of translating the 0s and 1s into data is equivalent to decoding Morse code into its original data message.

Common reader devices include a pen type reader, a laser scanner, a charge-coupled device (CCD), and a camera-based reader. Figure 1 demonstrates how a laser scanner reader operates.

### 1-DIMENSIONAL BARCODE

The 1-dimensional barcode is the oldest and the most commonly used due to its simplicity and low technology needs. It can be read by all of the barcode readers. There are many variations of one-dimensional (1-D) barcodes, including the length of the barcode, the amount of data that can be stored, the decoding scheme and the height. For a one dimensional barcode, the height is just a redundancy factor used to improve readability (IntelliTech, 2008). The longer the barcode, the taller the bars must be because of the increased possibility for defects or damage. At the beginning and end of every 1-D barcode are start and stop schemes (Inlite, 2008) which are different depending on the type of 1-D barcode (IntelliTech, 2008). To increase reliable reading, each barcode scheme encodes an extra number called a check digit, that allows the reader to detect an error while it was being scanned (Beckwith, 2008). Each code has a specific formula that the computer uses to compute the check digit. If an error is detected, another scan must be done. This process prevents errors such as ringing up the wrong item.

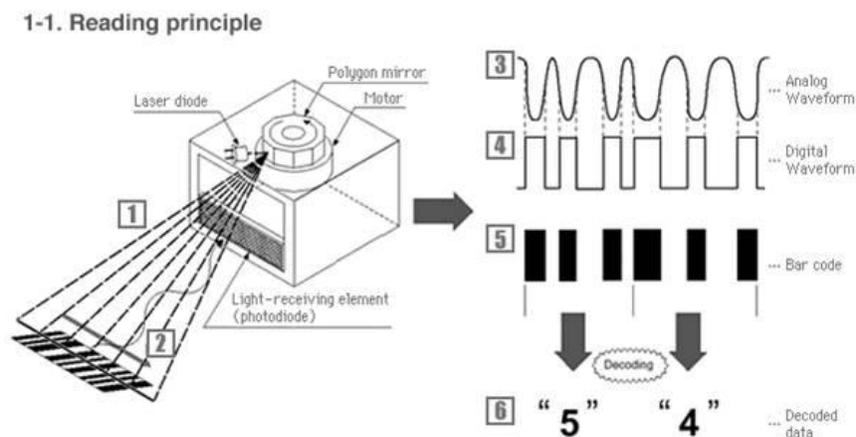


Figure 1. UPC reader device (Wikimedia, 2009).

## Universal Product Code

The Universal Product Code (UPC) is the oldest 1-D barcode. It was developed by George J. Laurer in 1973. It is now necessary for all merchandise to be assigned a UPC code and the quality of the barcode must be universally usable (Craft, 2008).

The UPC barcode consists of 12 blocks of seven modules. Each of the seven modules is a number from zero to nine. Both the white spaces and black bars are read when scanned. There must be at least ¼” on each side of the barcode of quiet space, i.e. white space (Craft, 2008).

The first digit in the UPC sequence determines the kind of product. The next 5 digits in the barcode represent the manufacturer’s specific identification number. The manufacturer prefix is assigned by the GS1 US. These digits are to the left of the middle guard bars and are also referred to as the manufacturer code. This specified number tells the computer where it should look for the product (Barcode Graphics, 2008).

The next 5 digits are used to identify the particular product, based upon the specified manufacturer’s database. The product number may contain, but is not required to

have the size, color, and/or any other important information about the product that may aid in the database lookup. The product number can completely arbitrary, depending on how the manufacture assigns its product numbers. These 5 digits immediately to the right are referred to as the product code (Sanitate, 2002).

The final digit is the check digit. The check digit is used to determine if there were any errors in the reading of the barcode. This number is found using a specific formula (see below).

See figure 2 for a visual of the 7 module spacing, the two groups of 6 digits, the middle guard bars, and the start and stop bars. The black bars represent 1 and white bars/spaces represent 0. The binary scheme for the start and stop guard bars are 101 and the scheme for the middle guard bars are 01010 (Sanitate, 2002).

This barcode, like most, can be read from either direction because the manufacturer and product codes are opposites of each other. This means that the 0’s in the sequence of 0001101 on the left of the middle guard bars will become a 1 on the right side (i.e. 1110010). Figure 3 shows the tables that tell what pattern each digit

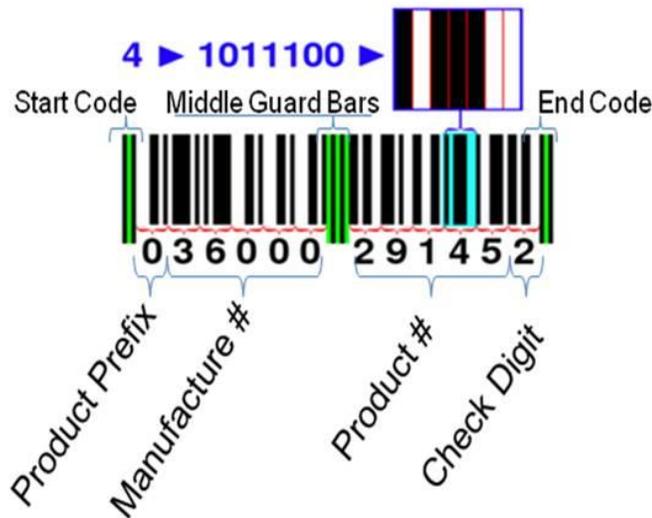


Figure 2. Structure of the UPC barcode.

	Before Middle Guard	After Middle Guard
	Bars	Bars
0	0001101	1110010
1	0011001	1100110
2	0010011	1101100
3	0111101	1000010
4	0100011	1011100
5	0110001	1001110
6	0101111	1010000
7	0111011	1000100
8	0110111	1001000
9	0001011	1110100

Figure 3. UPC digit conversion table (Sanitate, xxxx).

represents on the left and right sides of the middle guard bars.

Before the Middle Guard Bars: The 1s indicate a bar. The 0s indicate a space (Wikimedia, 2008). After the Middle Guard Bars: The 1s and 0s are inverted. See the table in figure 3 below for the patterns that correspond to each digit (0-9).

The following steps are used to calculate the check digit.

1. Sum up the numbers in the odd positions and then multiply by 3.
2. Sum up the numbers in the even positions and add this result to the previous step.
3. Find the next largest number that is evenly divisible by 10.
4. Subtract the result in step 2 from result 3.
5. The result from step 4 is the check digit.

In mathematical form,

$$Y = 3 \times (\sum \text{odd positioned integers}) + (\sum \text{even positioned integers})$$

$$Z = Y \text{ mod } 10$$

The check digit X is:  $X = Z - Y$ .

The check digit is checked for correctness using the following calculation.

X = check digit

$$Y = 3 \times (\sum \text{odd positioned integers}) + (\sum \text{even positioned integers})$$

If  $0 = Y \text{ mod } 10$ , then check digit is ok.

If an error is found in the UPC symbol it can be corrected using the check digit.

Z = (any unreadable single bar value)

$$Y = 3 \times (\sum \text{odd positioned integers}) + (\sum \text{even positioned integers})$$

W = (integer evenly divisible by 10) > Y

X = check digit

$$Z = W - (Y + X)$$

Figure 4 shows the UPC code for Celestial Seasonings® Lemon Zinger® tea bags. This code will be used to demonstrate the procedure just outlined.

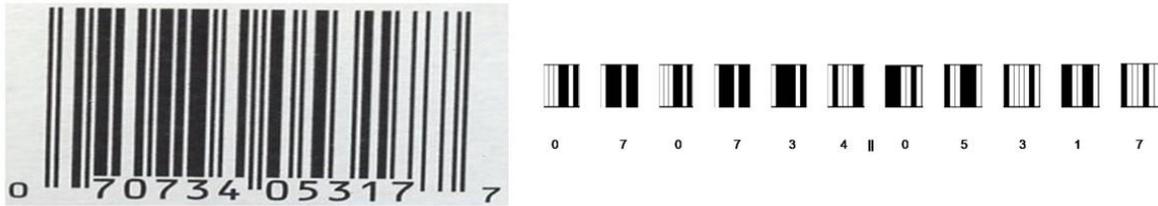


Figure 4. UPC symbol from Celestial Seasoning® Lemon Zinger® tea bags and the breakdown of the various digits.

The check digit in this example is 7. The UPC code can be verified by using other digits to calculate the check digit.

$$X = \text{check digit} = 7$$

$$Y = 3 \times (0+0+3+0+3+7) + (7+7+4+5+1) \\ = 3 \times (13) + (24) = 63$$

$$Z = (\text{integer evenly divisible by } 10) > Y \\ = 70$$

$$X = Z - Y = (70) - (63) = 7$$

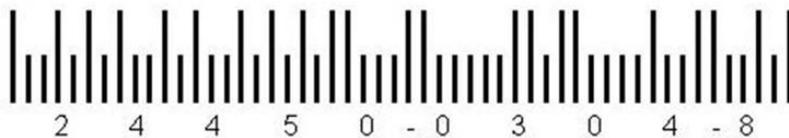
### POSTNET

Postal Numeric Encoding Technique (POSTNET) is the barcode system that the US Postal Service uses to help direct mail to specific ZIP+4 codes with speed and accuracy (**Omniplanar, xxxx**), (**Intellitech, xxxx**). It is read by a scanner, which reads not the thickness of the bars, but the height

of the bars.

Figure 5 shows an example of the POSTNET barcode and the digit conversion table. The barcode for each ZIP + 4 code consists of 10 blocks of five black modules (bars). Every number from 0-9 consists of full height bars and half height bars. There are 3 half bars and 2 full bars for each number (Sanitate, 2002; USPS, 2009). The full bars represent the ‘on’ bits (1s) and the half bars represent the ‘off’ bits (0s) (Ashford, 2008; Beckwith, 2008). There is a full height bar at the beginning and end of the barcode. These bars represent the start and stop feature that is common to most barcodes (Beckwith, 2008; USPS, 2009).

In the POSTNET encoding scheme, the check digit comes after the ZIP + 4 code. The check digit occurs in the 10th block of modules.



Numeric Value	Binary Code Value	Barcode Value
1	00011	
2	00101	
3	00110	
4	01001	
5	01010	
6	01100	
7	10001	
8	10010	
9	10100	
0	11000	

Figure 5. POSTNET barcode for the Virginia military Institute in Lexington, Virginia and the digit conversion table.

The check digit calculated using the following steps.

1. Add ZIP + 4 digits [ $X = \text{sum all digits}$ ].
2. Find remainder when divided by 10 [ $Y = X \text{ mod } 10$ ].
3.  $Z = \text{check digit} = 10 - [X \text{ mod } 10]$

If the check digit is found correctly, the sum of all the numbers (zip+4 and the check digit) should be divisible by 10.

For example, the check digit in figure 5 is calculated in the following manner.

$$X = 2+4+4+5+0+0+3+0+4 = 22$$

$$Y = X \text{ mod } 10 = 2$$

$$Z = 8$$

## 2-DIMENSIONAL BARCODES

The evolution of barcodes has always been towards creating a better barcode that can encode more information in the same or smaller space than the previous version. 2-dimensional (2-D) barcodes come as stacked or matrix-styled barcodes. They were created to hold much more data than the 1-D barcodes in smaller spaces. In fact, some two-dimensional barcodes can contain 500,000,000 characters per square-inch (Beckwith, 2008).

Not only can 2-D barcodes encode text, but some even have the capability to encode images, video, audio and executable files (Beckwith, 2008).

They provide more security-encryption capabilities than the traditional 1-D barcodes. Part of a 2-D barcode contains error correction features that have the ability to recover anywhere from 5-95% of the encoded information if any part of the

barcode is damaged or is misread by the reading device (Omniplanar, 2008; Jetmobile, 2007). These barcodes, however, cannot be manually input into a computer, as is possible for most 1-D ones (Beckwith, 2008).

2-D barcodes are used for tracking manufactured products, ID cards, registrations, medication, patient bracelets, blood banks, package tracking (UPS, USPS, FedEx, etc.), vehicle IDs and recently they have been put on household products (Omniplanar, 2008; Inlite, 2008).

A stacked barcode, like the PDF417 which will be discussed in the next section, is a 2-D barcode that contains a series of 1-D barcodes stacked vertically (Ashford, 2008).

The matrix-style barcode uses various positions of black module dots within a matrix. The position of a dot is what encodes the data (Ashford, 2008).

Although some can be read using a laser scanner, most two-dimensional barcodes require the use of a charged coupled device (CCD) video camera or CCD scanner (Jetmobile, 2007; Wikimedia, 2008; Altek, 2008). They are omnidirectional and depending on the size, they can usually be read up to 36 inches away and some can even be read at distances of up to 125 feet (Beckwith, 2008; IntelliTech, 2008).

### PDF 417

Portable Data File 417 (PDF417) is a two-dimensional machine-readable barcode system. This barcode was patented in September 1993 (Patent Number: 5,243,655) (Wang, 1993) and is composed of several linear rows of stacked 1-D barcodes. It was designed to hold large amounts of text and data, within the same or smaller area than the previously developed 1-D barcodes. In quantitative terms each of these barcodes is capable of containing over

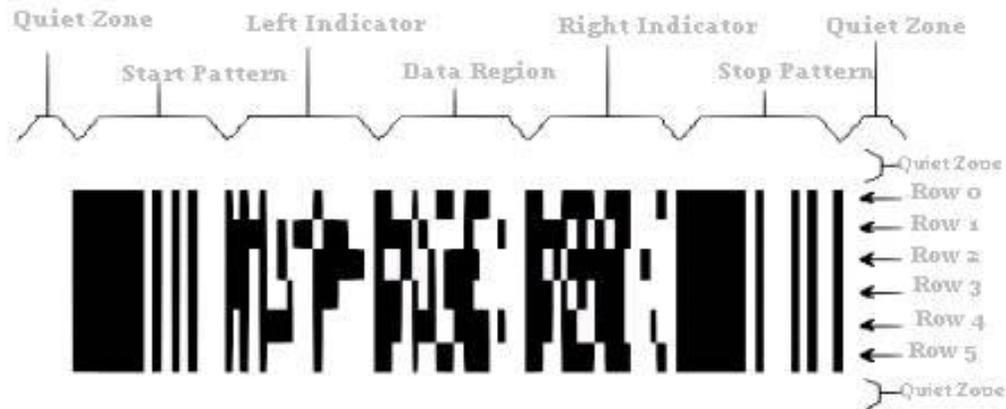


Figure 6. Example of a PDF 417 barcode and its structure (Easesoft, xxxx).

1kb of data (Grand Zebu, 2008; Wang, 1993; Jetmobile, 2007; EaseSoft, 2008).

The PDF417 barcode is primarily used for inventory management, identification cards and transportation (Omniplanar, 2008; Jetmobile, 2007; Inlite, 2008). Anyone who possesses a Virginia State driver's license has a 1-D barcode and a PDF417 barcode on the back of that card. The 1-D barcode contains the driver's name and your registration number. The PDF417 barcode contains all the information from the front side of the driver's license.

Figure 6 shows an example of a PDF 417 barcode and outlines its structure. The barcode has a standard start and stop pattern that tells the reading device where the information begins and ends. Between the start and stop patterns, there are left and right indicators. These indicators identify the row and how the code should be read (i.e. which tables to use in the decoding process of each row). Between the left and right indicators are the columns that contain the data and correction information (Grand Zebu, 2008; Wang, 1993; Jetmobile, 2007).

The number of columns that the barcode can contain between the left and right indicators ranges from 1 to 30, although the maximum suggested is 28 to ensure that the reading device can read the barcode properly. The number of rows this barcode

can contain ranges from 3 to 90 (IDAutomation, 2008; PDF417, 2008).

There are three unique look-up tables that encode and decode codewords into a particular scheme. Within each table there are 929 codewords. The tables are cycled through in order (1-3) for each row.

A module is the width of the narrowest bar in a barcode. A codeword is composed of 17 modules. Within each codeword, there must be four blocks of black and four blocks of white modules (this is where the "417" of PDF417 came from). The height of each row within PDF417 must satisfy a width-to-height ratio between 1:2 and 1:5. This ratio will depend on the amount of data and how big the barcode needs to be. This aspect ratio is important because if the standard module is not within this range, the device will not be able to decode any part of the PDF417 barcode. The best ratio to use is 1:4 (Symbol Technologies, 2004). Figure 7 shows the structure of a codeword.

The data column(s) contain the length descriptor, the data itself, padding (if needed), and security/error correction. The length descriptor tells the reading device how much data is contained within the barcode, to include padding. The data is the encoded information. There are a formula and a character table (both appear later) that are used to encode the information.

## PDF417 SYMBOL

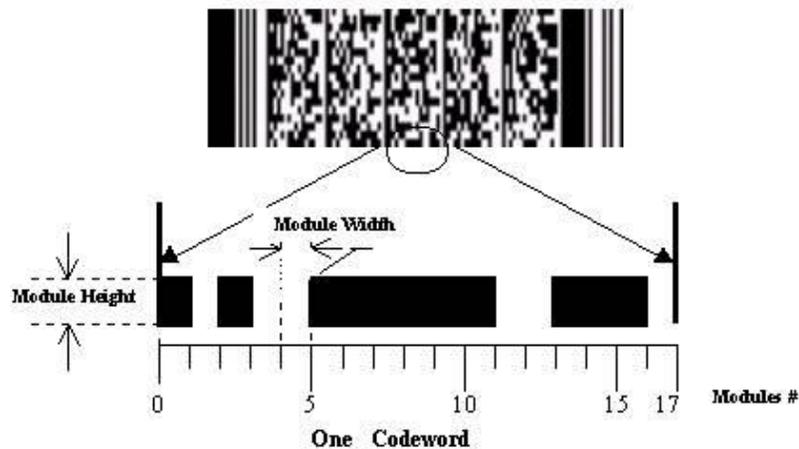


Figure 7. Codewords are composed of 17 modules (EaseSoft, xxxx).

The padding is used to fill in the extra codeword spaces between the final data codeword and the start of the security/ error correction code within the same row. In a three row barcode, if the data is contained in the first line only and the security/ error correction codewords only occupy the 3rd row, then padding will occupy any empty spaces on the 1st and 3rd rows and the entire 2nd row (Wang, 1993).

The security/error correction codewords are created using a Reed-Solomon algorithm. Unlike the 1-D barcodes, the PDF417 barcode uses security/error correction codewords instead of check digits. These codewords allow for some damage to the barcode without losing the ability to decode the data. The security/error correction level can range between 0 (2 error correction codewords) and 8 (512 error correction codewords).

The process to take a user's desired text/data and convert it into a PDF417 barcode will be discussed with a specific example in the following section . We do not layout every step of the process, but rather provide the MATLAB® code and other appendices for reference. In addition we refer the reader to <http://grandzebu.net/index.php> for more details.

1. Select your desired text/data.
2. Convert data into numbers using the table found in Table 1 and the following instructions:

SP: space

CR: carriage return

HT: horizontal tab

LF: line feed

UPP: switch to "Uppercase"

LOW : switch to "Lowercase"

IX: switch to "Mixed"

PUN : switch to "Punctuation"

T\_UPP: switch to "Uppercase" only for next character

T\_PUN: switch to "Punctuation" only for next character

3. Use the following formula to transform the numbers into codewords:

For each pair of consecutive numbers, C1, C2:

$$(C1 \times 30) + C2 = \text{encoded codeword value,}$$

where C1, C2 are the values of each character

4. Calculate the number of codewords.

**Table 1. PDF417 text to number conversion table.**

Value	Uppercase	Lowercase	Mixed	Punctuation
0	A	a	0	;
1	B	b	1	<
2	C	c	2	>
3	D	d	3	@
4	E	e	4	[
5	F	f	5	\
6	G	g	6	]
7	H	h	7	_
8	I	i	8	` (Quote)
9	J	j	9	~
10	K	k	&	!
11	L	l	CR	CR
12	M	m	HT	HT
13	N	n	,	,
14	O	o	:	:
15	P	p	#	LF
16	Q	q	-	-
17	R	r	.	.
18	S	s	\$	\$
19	T	t	/	/
20	U	u	+	“
21	V	v	%	
22	W	w	*	*
23	X	x	=	(
24	Y	y	^	)
25	Z	z	PUN	?
26	SP	SP	SP	{
27	LOW	T_UPP	LOW	}
28	MIX	MIX	UPP	‘
29	T_PUN	T_PUN	T_PUN	UPP

5. Select the number of columns (between 1-30, suggested max 28).
6. Select the level of security/error correction capabilities. This is a number between 0 and 8 where 0 is the lowest level and 8 is the highest.
7. Using the desired security level, compute how many error correction codewords (ECC) are required using the formula below:

$$ECC = 2^{(n+1)},$$

where  $n$  is the security level, for  $0 \leq n \leq 8$

Use the Reed-Solomon Algorithm (MacWilliams, 1988) to obtain the codewords. Note that the details of this particular algorithm are beyond the scope of this paper.

8. After obtaining the quantity of security/ error correction codewords and the amount of data codewords, determine how much padding will be needed in order to fill in empty space between the end of the data and the beginning of the security/ error correction. Use the 900 codeword for padding.
9. Use table 1 to calculate the left and right codeword indicators.

10. Convert the codeword values into PDF 417 font. Use the three lookup tables to convert the codeword values into the binary representation. The 1s become black modules and the 0s become the white modules.

Here we refer you to <http://grandzebu.net/index.php> (under the PDF417 link) for the specific lookup tables. They are very large and thus unreasonable to include them in a paper of this length.

11. Create the image. We have written MATLAB® program that takes the desired text/data, number of columns, and a level of security to create the image output.

The following provides and specific example of the encoding procedure. Color is used to help identify the various parts of the table of codewords.

Text to Encode: *The Mathematics of Barcodes*

Enter number of columns: 7

Security level: 2;  $S = 2^{(2+1)} = 8$  ECC

Convert the text into numbers:

597 214 807 360 577 132 19 242 566

425 807 30 512 423 138

Security/Error correction codeswords:

Table 2. Formulae used to calculate left and right codeword indicators.

<i>Table used to encode the CWs of this row</i>	<i>X for the left side CW</i>	<i>X for the right side CW</i>
1	(Number of rows -1) \ 3	Number of data columns - 1
2	(Security level x 3) + (Number of rows -1) MOD 3	(Number of rows -1) \ 3
3	Number of data columns - 1	(Security level x 3) + (Number of rows -1) MOD 3

526 582 143 602 126 876 621 74

Add padding:

Needs four 900 padding codewords

Add left and right indicators:

Row1 (1,6), Row2 (6,1) Row3 (6,6), Row4 (6,6)

Create table of codewords:

1	20	597	214	807	360	577	132	6
6	19	242	566	425	807	30	512	1
6	423	138	900	900	900	900	526	6
31	582	143	602	126	876	621	74	36

Convert table of codewords into PDF417 font:

```
+*xfs*fBk*seb*htk*keg*sow*aDB*cEE*pDA*-
+*yog*usk*mwE*itz*gl*sFl*ptA*wdB*zew*-
+*pDw*vIj*mkz*lyd*lyd*lyd*lyd*roa*pDw*-
+*ftw*xsf*tfk*Bhk*uBi*wyo*loc*odk*xFw*-
```

Create the PDF417 image using the MATLAB code. Figure 8 shows the final result.

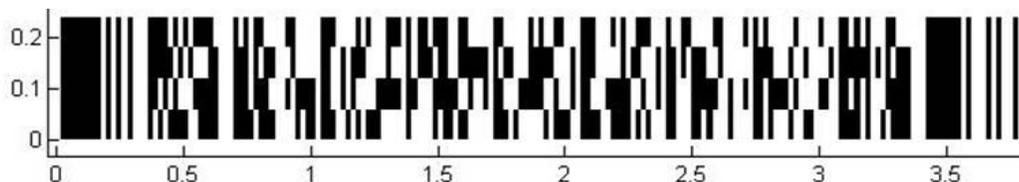


Figure 8. PDF 417 barcode containing the text *The Mathematics of Barcodes*.

## NOVEL CODES

In the final part of our project we created four variations of PDF417. These variations are to give the PDF417 barcode enhanced visual aesthetics. Each of these modifications can be customizable as well.

In the first variation, shown in figure 9, we decided to create the letters V, M, and I to the right of the PDF417 barcode. We positioned them on the right so that it would not disrupt the quiet zone. The letters were created using the same modules that are used in constructing the actual barcode so it would look like the code itself. In order for this variation to work, the number of rows must be less than 12.

In the second variation, shown in figure 10, we embedded the letters V, M and I in the start code. The start sequence tolerates the disturbance because of the height redundancy. As in our first variation, the letters were created using the same modules that are used in constructing the actual barcode. This is so it would look like the code itself. In order for this variation to work, the number of rows must be greater than or equal to 12.

In the third, shown in figure 11, we decided to create a red VMI spider in the background of the PDF417 barcode. We programmed the spider in red to ensure readability (because the red is interpreted as white). The red is interpreted as white. The spider is not made up of modules like the previous two variations, but it is just a picture set behind the barcode. Any company logo or design could be put in the background, but it must be red to ensure readability.

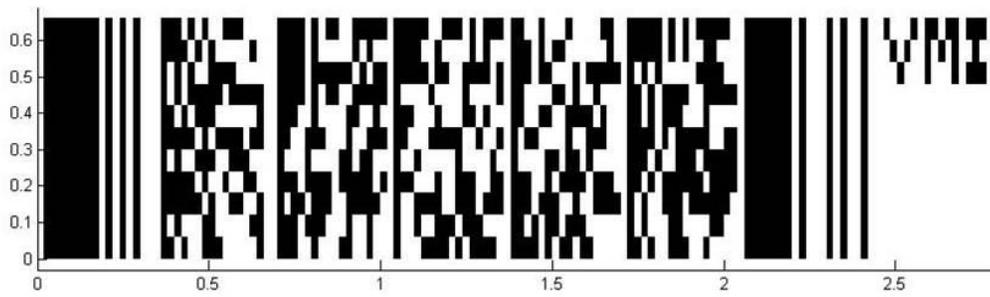


Figure 9. PDF 417 barcode with VMI encoded into the upper right hand side.

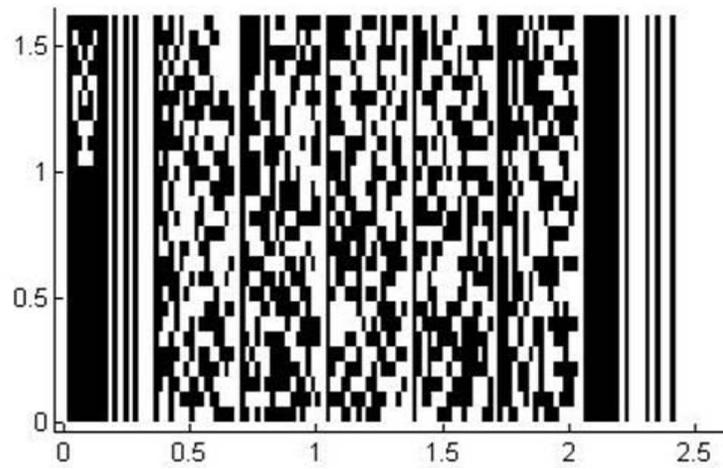


Figure 10. PDF 417 barcode with VMI embedded in the start code.

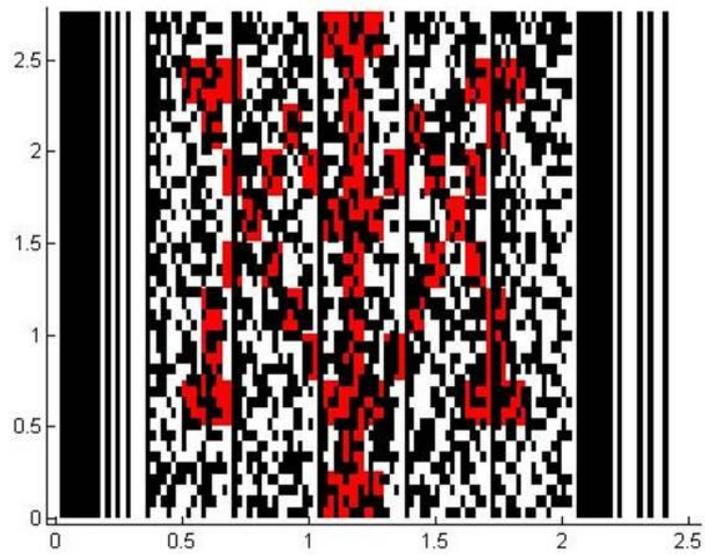


Figure 11. PDF 417 barcode with the VMI spider integrated.

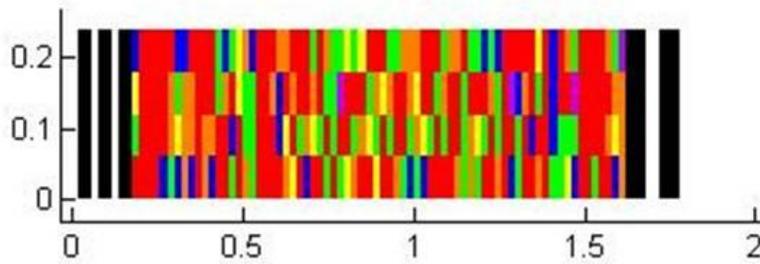


Figure 10. Color coded PDF 417 barcode.

In the fourth variation, shown in figure 12, we decided to color code the PDF417 barcode. In following the evolution of barcodes (i.e. creating a barcode that can hold more information in the same space) we decided on a color coding scheme that turns runs of 0s and 1s into defined colors. The usual 17 module long code words are converted into 8 colored blocks. The following colors define the corresponding run length:

Red = 1  
 Green = 2  
 Orange = 3  
 Blue = 4  
 Yellow = 5  
 Purple = 6

If this variation were used the reader would need a color-coded calibration strip to interpret each color. The reader device must also be color sensitive.

### CONCLUSION

Throughout this project on the mathematics of barcodes, we have covered a brief history of barcodes, discussed the basics of the 1-D barcode, broke down the UPC and the POSTNET barcodes and then discussed the PDF 417 barcode, a 2-D barcode. Our work in developing novel barcodes should have help the reader to understand how existing barcodes are used to create new ones that save space and incorporate more visual aspects. In the future, there is potential to create new codes

that save space and incorporate more visual aspects.

### REFERENCES

- Adams, R. (2007) Bar Code History Page. <http://www.adams1.com/pub/russadam/history.html>.
- Sanitate, R.L. (2002) Bar Code History. <http://www.missioncollege.org/depts/math/sanitate/lynn.htm>.
- Ashford, T.L. (2008) Bar Code Basics: Symbolologies. [http://tlashford.com/frames/Basics\\_sym/basics\\_symbol.htm](http://tlashford.com/frames/Basics_sym/basics_symbol.htm).
- Intellitech International, Inc. (2008) Bar Codes 101. <http://www.intellitech-intl.com/portasp/U-barcodes101.asp>.
- Inlite Research Corporation (2008) Barcode Standards. [http://www.inliteresearch.com/homepage/technology/b\\_standards.html](http://www.inliteresearch.com/homepage/technology/b_standards.html).
- Beckwith, A. (2008) POSTNET Barcodes. <http://mail.colonial.net/~abeckwith/barcodes.html>.
- Wikimedia Foundation (2008) <http://www.wikipedia.com>.
- Craft and Hobby Association (2008) U.P.C. Codes. <http://www.insightu.org/hobby/matrix/matrix1.htm>.

Barcode Graphics (2008) UPC Resellers.  
[http://www.barcode-us.com/info\\_center/upcreseller.htm](http://www.barcode-us.com/info_center/upcreseller.htm).

OmniPlanar, Inc. (2008) *Barcode Definition – 1D, 2D and Postal Symbologies*.  
<http://www.omniplanar.com/barcode-definition.php>.

United States Postal Service (USPS) (2009) *Intelligent Mail® Barcode Technical Resource Guide*.

Jetmobile S.A.S. (2007) 1D Barcodes, Postal Barcodes and 2D Barcode Symbologies supported by Jetmobile.  
[http://www.jetmobile.com/products/BarDIMM/barcode\\_symbologies\\_BarDIMM.htm](http://www.jetmobile.com/products/BarDIMM/barcode_symbologies_BarDIMM.htm).

Altek Instruments Ltd. (2008) 2D Barcodes Explained.  
<http://www.barcodeman.com/faq/2d.php>.

Wang, Y.P. (1993) System for Encoding and Decoding Data in Machine Readable Graphic Form. United States Patent Number 5,243,655.

Grand Zebu (2008) The PDF417 Code.  
<http://www.grandzebu.net/infomatique/codbar-en/pdf417.htm>.

EaseSoft Inc. (2008) The PDF417 Code.  
<http://www.easesoft.net/pdf417.html>.

IDAutomation.com, Inc. (2008) MaxiCode Barcode FAQ and Tutorial.  
<http://www.idautomation.com/maxicodefaq.html>.

PDF417 Online (2008) PDF417 Online – Info  
<http://pdf417-online.com/en/info.jsp>.

Symbol Technologies (2004) Bar Code Basics & PDF417 Symbology Overview.  
[http://web.archive.org/web/20421463354/http://www.symbol.com/products/barcode\\_sc](http://web.archive.org/web/20421463354/http://www.symbol.com/products/barcode_sc)

MacWilliams, F.J, Sloane, N.J.A. (1988) *The Theory of Error-Correcting Codes*. Elsevier Science, Oxford.